Gilbert-Multiplier-Based Parallel 1-D and 2-D Analog FIR Filters for Medical Diagnostics

Rafał Długosz[1,2,3,*], Vincent Gaudet[2], and Ryszard Wojtyna[3,4]

[1] University of Neuchâtel, Institute of Microtechnology,
Rue A.-L. Breguet 2, CH-2000, Neuchâtel, Switzerland
[2] University of Alberta, Department of Electrical and Computer Engineering
114 St – 89 Ave, Edmonton Alberta, T6G 2V4, Canada
[3] The College of Computer Science in Łódź, Poland, Department in Bydgoszcz,
ul. Fordońska 246, 85-766, Bydgoszcz, Poland
[4] University of Technology and Life Sciences, Faculty of Telecommunication and Electrical Engineering, ul. Kaliskiego 7, 85-791 Bydgoszcz, Poland
rdlugosz@ualberta.ca, vgaudet@ece.ualberta.ca,
woj@mail.utp.edu.pl

Abstract. A novel idea and CMOS implementation of power-efficient 1-D and 2-D finite-impulse-response (FIR) filters based on a current-mode Gilbert-vector-multiplier (GVM) have been proposed. MOS transistors included in the GVM multiplier work in a weak inversion resulting in an ultra low power consumption, with currents as low as several hundred nanoAmps to several microAmps. Due to the proposed multiplier, all output samples of the FIR filters can be calculated in parallel, making the filtration operation very fast. Moreover, the GVM enables a simple normalization of the output samples. As a consequence, not current values but current ratio values are significant here. As an example, a time-domain 1-D 3rd order filter has been presented which dissipates 8-µW of power enabling a 2-MHz sampling frequency (f_S). Another example is a 2-D filter with the frame resolution of 6x1 pixels that dissipates 7-µW at the data rate of 1 Mframe/s.

Keywords: FIR filters, Gilbert vector multipliers, parallel data processing.

1 Introduction

Modern medical devices often require high speed and ultra-low power integrated circuits (IC), which allow for increasing a system reliability and for device miniaturization. One of examples of diagnostic systems where the mentioned features are essential is so called camera endoscopic capsule that can be used in gastroscopy. This autonomous device can be swallowed by a patient, enabling an observation of the alimentary canal by means of a built-in low-power vision system with a micro-camera. The collected data are transmitted to an external computer using an embedded RF transceiver. In this paper, we propose ultra-low power analog filters suitable

[*] Fellow of the EU Marie Curie International Outgoing Fellowship.

for low-energy signal preprocessing inside the device. In this way, we can reduce the amount of data that must be sent via the RF and save the consumed energy.

In typical vision systems with charge coupled devices (CCD's), matrix analog data registered by a camera are directly converted into digital signals, using analog-to-digital converters (ADC's), and next processed by means of digital signal processors (DSP's) or field programmable gate arrays (FPGA's). This is sufficient in applications, where power dissipation is of secondary importance and a relatively low data rate is acceptable. Serial data processing in microprocessors is a bottleneck especially visible in processing high resolution images. For example, when an input image would have the resolution of 1000x1000 pixels, then to realize a simple (3x3) image filtering, a DSP would have to perform even 10 million operations per each image frame. This means that even a very fast DSP is able to calculate not more than several dozen of frames in a real time. The other problem are I/O circuitries in a DSP system, which contains relatively slow ADC's. For example, to calculate images with a resolution of 1000x1000 pixels with a data rate of 100 frames per second and an 8-bits color depth, the ADC must enable data rate that is equal to about 1 Gbit/s. Converters designed for such data rates dissipate at least several to several dozen mW of power [1]. In the other approach, a matrix of low-power and low-rate ADC's operating in parallel may be used [2], but the problem with a large amount of data is the same. The described constrains draw trade-offs that are between the data rate, the image resolution and the power dissipation.

One also observes various efforts to implement analog image processing. Such analog processors are very often realized using Cellular Neural Networks (CNN) [3]. An example circuit of this type has been described in [3]. This circuit designed in CMOS 0.5μm process enables a parallel data processing, calculating images with a 64x64 resolution for data rate of 1 MSamples/s and dissipating power of 1.5 W from the 3.3 V supply.

In this paper, anther approach to build data processing systems is proposed. Our circuits incorporate ultra low power analog circuitry, such as current mode Gilbert vector multipliers, which use transistors working in a weak inversion region. The Gilbert multipliers have been used in various applications for many years. FIR filtering is one of possible applications but so far only scalar Gilbert multiplication cells (SGM's) have been used in an implementation of FIR filter coefficients [4-6], mostly in adaptive equalization systems, where a high selectivity is not required. On the other hand, current-mode Gilbert vector multipliers (GVM's) working in a weak inversion have recently been successfully used to implement ultra low power analog decoders [7]. Here, we propose a novel application of these circuits – the current-mode FIR filters, which enable a parallel data processing with a mach better efficiency.

Our filters have been implemented in two versions, i.e. for filtering 1-D and 2-D signals. In both cases the core circuit is the Gilbert multiplier. The difference is in the domain of input data. In the first case, the GVM circuit receives samples of only one signal sampled in time domain. In this case a delay line is needed. In the second case, all inputs are independent data streams coming, for example, from particular pixels in a CCD matrix. In this case the image coordinates form the signal domain. Both cases are closely related and are discussed in this paper.

The proposed circuits enable initial low-power data preprocessing at the analog side. After that, some amount of data can usually be dropped off. As a result, the ADC that is used in the system as a next block can be simplified. One of examples is a decimation process, which allows for significant reduction of data amount in case, when a signal temporarily does not contain important information at high frequencies. When a decimation preceded by an antialiasing filtering is performed at the analog side, which can be easily performed using the proposed filters, this technique allows for a significant reduction of the power dissipation in the following ADC.

The text is organized as follows. In the next section, a short necessary information about 1-D and 2-D FIR filters is provided. The idea of using GVM circuits in implementation of parallel FIR filters has been presented in the following section. Then, in two sections we present implementation of both the 1-D and 2-D filters. Application of the proposed filters to realization of multirate systems is shown in the successive section. Finally, conclusions are drawn in the last section.

2 Basics of 1-D and 2-D FIR Filtration

As FIR filters are widely described in the literature, here we only present key aspects, which are important to explain some analogies between FIR filters and the GVM circuits. A general structure for a 1-D N^{th} order FIR filter is shown in Fig. 1. When an input signal, $X=\{x_0, x_1,\ldots, x_N\}$, is passing through a filter, particular samples, x_i, are stored in delay elements T. After multiplication by coefficients h_i , they are summed at the output. Thus, the output filter samples, y_i , are calculated using the following equation:

$$y_n = \sum_{i=0}^{N} h_i x_i \tag{1}$$

Various implementation techniques of FIR filters, both digital and analog, have been widely described in the literature. In existing implementations, usually only one output sample is calculated after each new sample enters the filter. In this paper, we propose a different approach where several output samples are calculated in parallel after receiving a certain input data block.

In 2-D filtering, an image frame is a 2-D signal, where pixels are samples of this signal in (x, y) image coordinates. In case of 2-D FIR filtering, particular input pixels (brightness) are multiplied by filter coefficients given by a 2-D mask and summed to produce output image samples. The principle mentioned is illustrated in Fig. 2 for an example (3x3) filter mask H. In this case, pixels of the output image, B, are calculated on the basis of the input image, A, using the following equation:

$$B(x, y) = \sum_{n=1}^{3}\sum_{m=1}^{3} A(x+n-2, y+m-2)h(n,m) \tag{2}$$

The lowpass filter mask, H, is often used in multirate systems to realize anti-aliasing or anti-imaging filtering. A highpass filtering, that is useful for example in detection of edges, requires a mechanism that enables realizing negative coefficients.

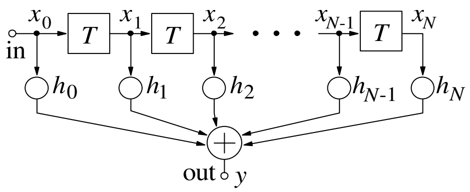


Fig. 1. Block diagram of a finite impulse response (FIR) filter

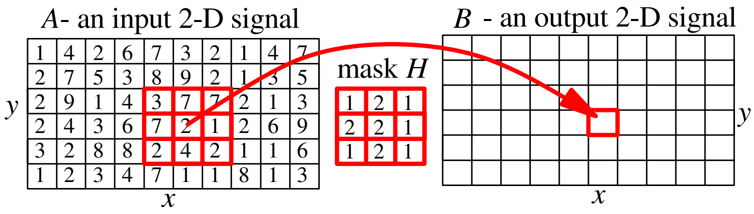


Fig. 2. Block Two-dimensional FIR image filtration [9]

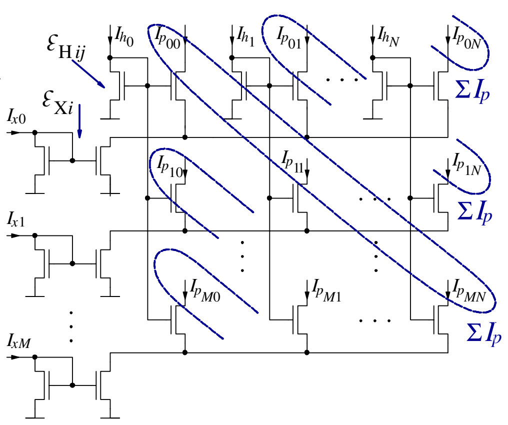
3 Implementation of FIR Filters Using Gilbert Vector Multipliers

To realize the multiplication operations required in the filtration process, we propose to utilize so called Gilbert multiplier adapted to multiply two vectors [7]. Scheme of the adapted vector-by-vector Gilbert multiplier, denoted by GVM, is shown in Fig. 3 (a). One of the multiplied vectors (denoted by *X*) includes the currents I_{xi} while the other (denoted by *H)* the currents I_{hi}. In Fig. 3 (b) we have the Gilbert multiplier adapted for multiplying the vector *H* by the scalar I_x.

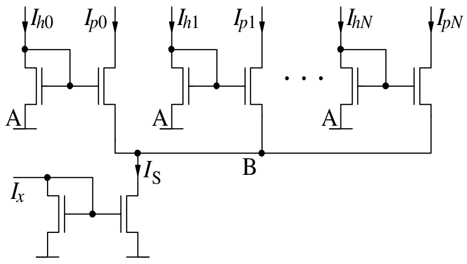
The vector-by-vector GVM block calculates components of a matrix denoted by *P*, which presents I_p currents resulting from multiplication of *X* and H^T, according to (3):

$$P = \frac{XH^{\mathrm{T}}}{\sum_{i=0}^{N} h_i} = X|H|^{\mathrm{T}} \qquad \text{or} \qquad P = \begin{bmatrix} x_0|h_0| & x_0|h_1| & x_0|h_2| & \cdots & x_0|h_N| \\ x_1|h_0| & x_1|h_1| & x_1|h_2| & \cdots & x_1|h_N| \\ x_2|h_0| & x_2|h_1| & x_2|h_2| & \cdots & x_2|h_N| \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ x_M|h_0| & x_M|h_1| & x_M|h_2| & \cdots & x_M|h_N| \end{bmatrix} \tag{3}$$

It is worth noting that particular elements of the *P* matrix are divided by the sum of all components of the vector *H*, which enables an automatic normalization of *H*. The normalized vector *H* in (3) is denoted as |*H*|. Comparing equations 1 and 3 we see that all elements of the *P* matrix needed in equation 1 are situated along the matrix diagonal (provided that *X* represents input signals and *H* coefficients of the filter). The *H* normalization possibility is a very important superiority of our FIR filter realization over the ones reported earlier, based on scalar multipliers. This is because in our approach not absolute values but ratios of the currents forming the *H* vector are important, which simplifies the filter controlling significantly.



(a)



(b)

Fig. 3. Gilbert multipliers: (a) vector-by-vector multiplier (GVM) and (b) scalar-by-vector multiplier (GSVM)

One of the GVM multiplier features is that it also calculates other elements, which in the *P* matrix are placed off the diagonal. In a typical implementation of FIR filters, these currents would be treated as parasitic currents, which only increase the filter power consumption and are never used in calculations of the output samples. Assuming that both vectors, *X* and *H*, include the same number of components, i.e. $M = N$, a number of these off-diagonal currents is equal to $N{\cdot}(N\text{-}1)$. Then, the filter efficiency decreases hyperbolically with the filter order *N*. As the number of parasitic currents increases with a square of the filter order, *N*, the number of the currents equals about $1/N$.

Originality of our approach is that the *P*-matrix off-diagonal elements have been utilized in parallel calculations of other than the diagonal output samples, *y*, in order to increase the filtration speed and reduce a power consumption associated with one sample (improve power per sample efficiency). To explain the principle of our method, consider an example GVM block where N=4 and M=4. In a given k^{th} cycle, where the cycle starts with introducing a new set of *M* input samples, the GVM block calculates $N{\cdot}M$ of new elements $p_{n,m}$. These elements are used in calculations of the following signals:

$$
\begin{aligned}
&y_{0,1}(k)=x_0h_0+x_1h_1+x_2h_2+x_3h_3 &&\mid y_{0,2}(k)=0\\
&y_{1,1}(k)= \quad x_0h_1+x_1h_2+x_2h_3 &&\mid y_{1,2}(k)=x_3h_0\\
&y_{2,1}(k)= \quad x_0h_2+x_1h_3 &&\mid y_{2,2}(k)= x_2h_0+x_3h_1\\
&y_{3,1}(k)= \quad x_0h_3 &&\mid y_{3,2}(k)=x_1h_0+x_2h_1+x_3h_2
\end{aligned}
\tag{4}
$$

Note, that the first sum, $y_{0,1}(k)$, is a complete output sample from equation 1. The GVM block calculates also partial sums $y_{m,1}(k)$, which will be used in calculations of other output samples in the next $(k + 1)$ cycle. The block calculates also partial sums $y_{m,2}(k)$, which in a k^{th} cycle supplement the partial sums $y_{m,1}(k-1)$ calculated in the previous cycle, creating in the same time next complete output samples. In general, in a k^{th} cycle, the following samples are generated concurrently:

$$y_m(k)=y_{m,1}(k-1)+y_{m,2}(k),\ m = 0, .., M \tag{5}$$

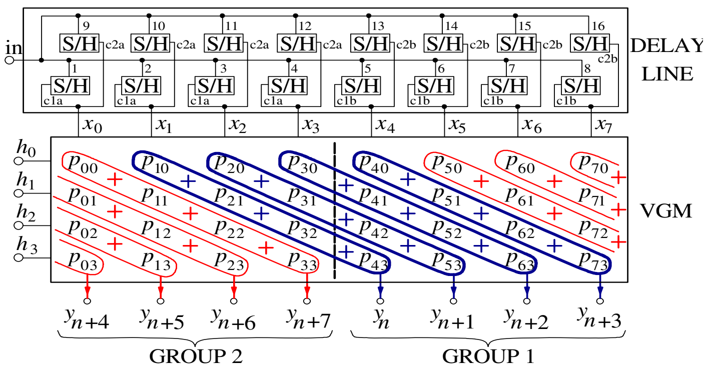
In our approach, all elements from the P matrix are utilized in the calculations, i.e. there are no parasitic currents which are useless. It is worth noting that the case when N>M makes no sense from the practical point of view because the number of samples must be not lower than the number of filter coefficients. The main advantage of our technique is that in one cycle, which lasts M / f_S [s], we calculate M signal samples concurrently. As a result, the GVM block has sufficient time (M times more than the sampling period duration) for calculating the new P matrix. As a result, the GVM block can work with significantly smaller currents, which means power saving. Another question is, how to solve in practice the problem of introducing the parallel data in a cyclical pattern. This problem is discussed in the following sections.

3.1 Implementation Issues of the Proposed 1-D Filter

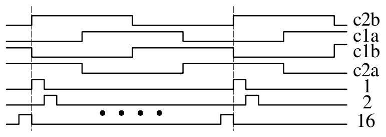
In the previous section, a conception of cyclic delivering parallel data has been outlined. In practical implementations, however, some problems occur. The most important of them is storing partial sums created in the previous cycle. Using intermediate analog memory cells to store partial sums is not a proper solution. This is because values of particular sums may differ significantly when particular sums contain a different number of the P-matrix elements.

Nonlinear properties of analog memories cause the final output samples to have different DC levels. In addition, the DC levels depend on temperature, which makes an easy level correction impossible. We have decided to solve this problem in a different way. Instead of using an N x M dimension GVM multiplier block, a block with an N x $2M$ resolution was applied. The proposed circuit is shown in Fig. 4 together with the required clock phases. In this solution, the number of sample-and-hold (S&H) elements is doubled, but all output samples are calculated under equal conditions and their DC levels are the same for each output sample. As a consequence, there are two groups of outputs that are read-out alternately.

The filter has been implemented in a TSMC CMOS 180-nm technology and verified using HSPICE post-layout simulations. Transfer properties of the filter can be controlled by means of currents collected in the H vector and different transfer functions can be realized depending on the current values. Results of simulations in time


in
S/H
c1a
c1b
c2a
c2b
DELAY LINE
x_0 x_1 x_2 x_3 x_4 x_5 x_6 x_7
h_0 h_1 h_2 h_3
VGM
y_{n+4} y_{n+5} y_{n+6} y_{n+7} y_n y_{n+1} y_{n+2} y_{n+3}
GROUP 2
GROUP 1

(a)



(b)

Fig. 4. Example of a 8x4 1-D GVM FIR filter: (a) general structure (b) clock generator

domain, concerning passband and stopband of an example filter, are shown in Fig. 5. Frequency-domain amplitude-characteristics of the filter are shown in Fig. 6 for two cases, i.e. for *H*={1/4, 1/4, 1/4, 1/4} (currents: 200, 200, 200, 200 nA) and *H*={1/8, 3/8, 3/8 1/8} (currents: 200, 600, 600, 200 nA).

As can be seen from Fig. 6, for some frequencies the simulated attenuation is about 55 dB. This value, however, is achievable provided that transistors used to build the current mirrorshave equal dimensions. In real applications, various mismatch components affect the filter performance. Variations in the transistor threshold voltage, V_{TH}, are usually regarded as the main reason for the mismatch problem. As a result, relation between input and output currents in the mirror is given by [7]:

$$I_{\mathrm{out}} = I_{in}(1+\varepsilon) \tag{6}$$

where: ε is a zero-mean Gaussian random variable that results from the threshold voltage mismatch [7]. Taking (6) into account, particular elements in the *P*-matrix can be described as:

$$p_{ij} = \frac{x_i(1+\varepsilon_{\mathrm{X}i})\cdot h_j(1+\varepsilon_{\mathrm{H}ij})}{\sum_{k=0}^{N} h_k(1+\varepsilon_{\mathrm{H}ik})} \tag{7}$$

For a properly designed layout, with sufficiently large transistor dimensions, the mirror gain variation is kept below 2 % [8]. To perform analysis for the worst case scenario, values of particular ε variables in equation 7 have been selected to be equal to

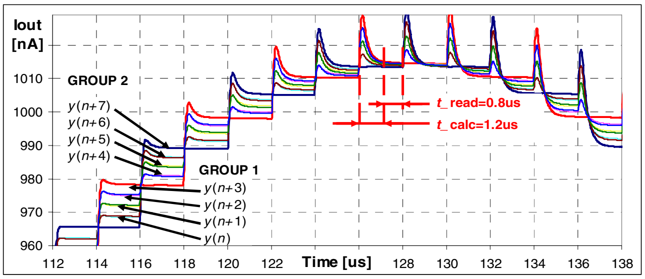
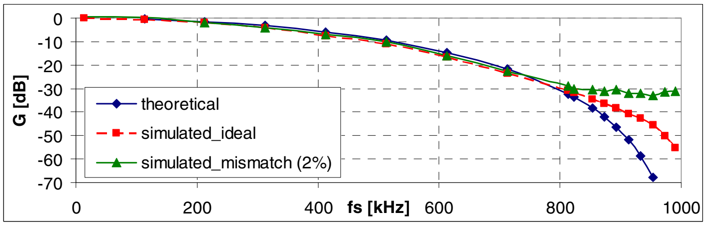
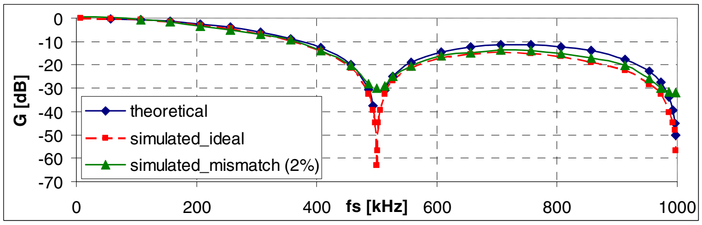


Fig. 5. Filter's output signal in the passband for f_S=2 MHz, and input data frequency 10 kHz



(a)



(b)

Fig. 6. Simulated and theoretical filter transmittances: (a) H=[1/4 1/4 1/4 1/4] (b) H=[1/8 3/8 3/8 1/8]

+/- 2%. As sum of all filter coefficients is equal to one, the maximum (worst case) error level introduced by the mismatch is not greater than approximately (MAX(ε))·2, i.e. not greater than about 4% in our case. This has been verified for several transfer functions and for different distributions of mismatch components in the P matrix. In most of the tested cases (roughly 70%), the attenuation was larger than 30-dB, which in case of color picture filtering means a possible color depth of 4-5 bits. Such a value is sufficient in many diagnostic tasks and other applications of the filters [10]. This is worth noting that values of ε parameters are for a given chip constant, resulting in systematic errors that load particular p components, which can be corrected at the digital side, after device calibration.

4 Implementation of 2-D Programmable Image Filter

General idea as well as initial results concerning an image filter considered in this section have been presented in [9]. In some respects, the filter is similar to the 1-D FIR one described in the previous sections. Its basic block, like in the 1-D filter, is a GSVM circuit shown in Fig. 3, for which the following expression holds:

$$P_{xy} = \frac{I_{xy} H^{\mathrm{T}}}{\sum_{i=0}^{N} h_i} = I_{xy} \left| H \right|^{\mathrm{T}} \tag{8}$$

Each input signal (pixel) in a vision system is associated with one GSVM circuit, which calculates currents, I_p, that are products of a given input sample, $A(x, y)$, at the I_{xy} input, and a H vector. Length of the output vector, P_{xy}, is determined by the number of coefficients, whose values are unique. For example, for the lowpass filter mask shown in Fig. 2, length of the P_{xy} vector equals 2 only, as this mask contains only two distinguished coefficients, namely 1 and 2. To simplify the filter analysis, it is convenient to present particular vectors, P_{xy}, as if they were columns of a 3-D P-matrix with p_{xyz} elements, which is shown in Fig. 7. In such a case, the x and y coordinates indicate a given input pixel, $A(x,y)$, while the z coordinate indicates a given product of this pixel and a given filter coefficient. To calculate the output image, B, particular elements of the P matrix must be summed in a proper way.

If the mask size is N x M, in calculations of $N \cdot M$ output samples, $B(x, y)$, elements of each P_{xy} vector are used. In our image programmable filter, to enable a realization of different filter masks, each p_{xyz} product is copied into $N \cdot M$ independent output branches, using PMOS current mirrors. This is shown in Fig. 8. The block presented in Fig.8 is associated with each input signal. As a result, the I_{xy} signal is multiplied by the I_h coefficients, using the GSVM circuit. There are two types of the circuit outputs denoted in Fig. 8 by (-) and (+). Currents in the (-) branches are used to create negative coefficients by means of the “b” circuit of Fig. 8. In this circuit, all products representing the samples to be multiplied by negative coefficients are summed in one node and then are subtracted from the sum of the products representing samples multiplied by positive coefficients. A problem may occur when the first sum becomes larger than the second one. Then, an additional biasing I_{DC} current is added to the output signal. Our approach is a very simple solution to this problem as only one additional NMOS current mirror per each output pixel has to be applied. The required map of connections is programmed using a 4-dimensional matrix D. The first two dimensions (x and y) in D indicate a given output pixel, while the 3rd one indicates the p element to be used in calculating a given sample $B(x, y)$. The 4th dimension of D indicates, whether the product must be added of subtracted at the output. Note that only particular elements, d, of this matrix control given switches in each circuit of Fig. 8. This means that to program a 2-D image filter with any resolution, only one D matrix is necessary.

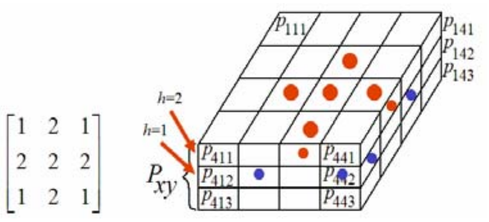
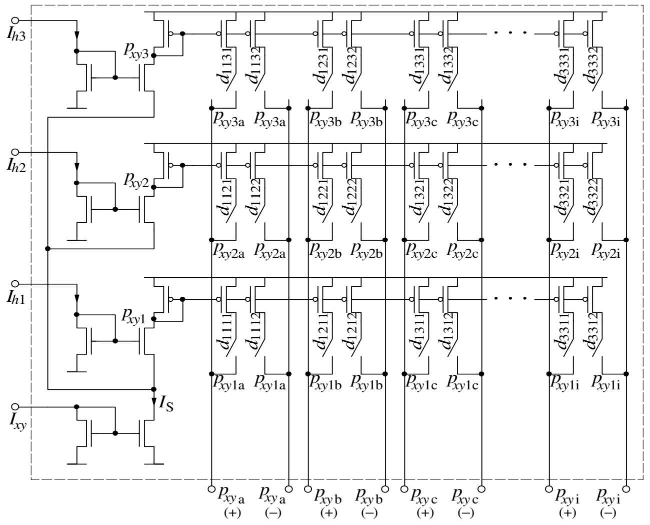
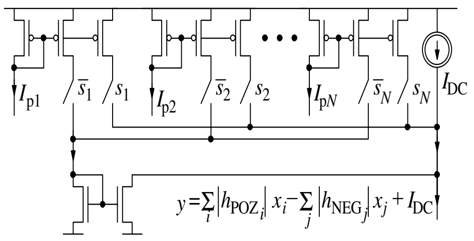


Fig. 7. Simulated and theoretical filter transmittances: (a) H=[1/4 1/4 1/4 1/4] (b) H=[1/8 3/8 3/8 1/8]


I_{h3}
I_{h2}
I_{h1}
I_{xy}
I_S
P_{xy3}
P_{xy2}
P_{xy1}
d_{1131} d_{1132} d_{1231} d_{1232} d_{1331} d_{1332} d_{3331} d_{3332}
d_{1121} d_{1122} d_{1221} d_{1222} d_{1321} d_{1322} d_{3321} d_{3322}
d_{1111} d_{1112} d_{1211} d_{1212} d_{1311} d_{1312} d_{3311} d_{3312}
P_{xy3a} P_{xy3a} P_{xy3b} P_{xy3b} P_{xy3c} P_{xy3c} P_{xy3i} P_{xy3i}
P_{xy2a} P_{xy2a} P_{xy2b} P_{xy2b} P_{xy2c} P_{xy2c} P_{xy2i} P_{xy2i}
P_{xy1a} P_{xy1a} P_{xy1b} P_{xy1b} P_{xy1c} P_{xy1c} P_{xy1i} P_{xy1i}
P_{xy}a (+) P_{xy}a (−) P_{xy}b (+) P_{xy}b (−) P_{xy}c (+) P_{xy}c (−) P_{xy}i (+) P_{xy}i (−)

(a)



(b)

Fig. 8. GVM multiplier suited for parallel image filtration

The connection map is constructed in a way that enables fully parallel data processing, where all output pixels are calculated in the same time. This means an increase in the data processing rate (important advantage). Furthermore, the D matrix is setting up only ones (at the beginning). Then, the filter works asynchronously without the clock generator (another advantage).

Table 1. Input and output signals in the designed image filter for small input currents

In and out No.	A1/B1	A2/B2	A3/B3	A4/B4	A5/B5	A6/B6
Input [nA]	200	200	200/250	200/270	200/300	200
Theoretical out [nA]	400	650	720	820	770	500
Post-layout simulations [nA]	***296***	354.2	365.9	382	373.4	***312***
Normalized output [nA]	***285***	649	722	822	768	***385***
Error [%]	***28.87***	0.20	-0.21	-0.22	0.23	***23.05***

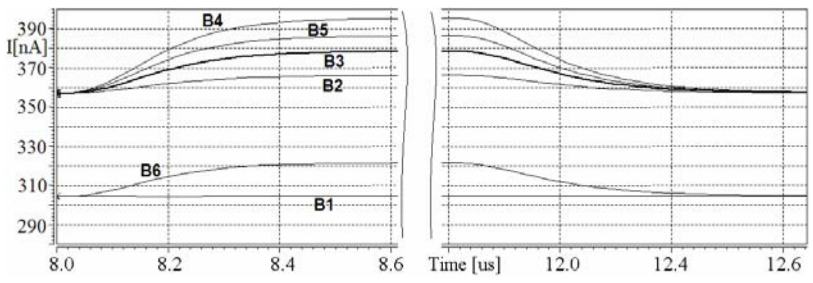
Table 2. Input and output signals in the designed image filter for larger input currents

In and out No.	A1/B1	A2/B2	A3/B3	A4/B4	A5/B5	A6/B6
Input [nA]	2000	2000	2000/2500	2000/2700	2000/3000	2000
Theoretical out [nA]	4000	6500	7200	8200	7700	5000
Post-layout simulations [nA]	***2770***	3283	3372	3492	3426	***2893***
Normalized output [nA]	***2202***	6481	7224	8225	7674	***3228***
Error [%]	***44.95***	0.29	-0.33	-0.30	0.34	***35.44***

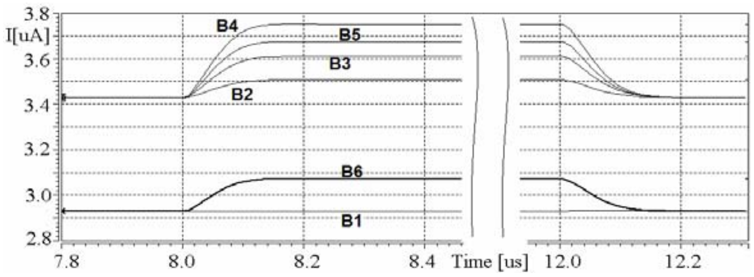
To experimentally test the proposed idea, a 2-D FIR image filter has been implemented in a 0.18 μm CMOS process. The image resolution in the designed filter was 6 x 1, while a resolution of the filter masks was 3 x 1, which is sufficient to verify the proposed filter concept. In this paper, post-layout simulation results are presented. In our simulations, presence of parasitic elements, such as unwanted node capacitances and signal path resistances, was taken into account. The obtained results confirm the observation made in [9] that the proposed circuit exhibits a low sensitivity to parasitic elements.

From our studies it follows that the filter operates correctly for a wide range of the input current values. Experimental results obtained for $H = [1, 1, 1]$ (filter mask) and for different average values of the input currents are presented in Tables 1 and 2. To enable investigating dynamic features of our filter, the input image, A, was varied in time. The A1, A2 and A6 input samples were constant during the whole test while the others (A3, A4 and A5) oscillated not exciding values 200 and 300 nA in the case of Table 1, and between 2000 and 3000 nA in the case of Table 2 (second row in both Tables). Theoretical values of the output pixels are gathered in the 3rd row while real values of them in the 4th one.

To make a comparison between the theoretical and experimental results to be more clear, the latter have been normalized in such a way to obtain comparable values of the theoretical and experimental currents (rows 5 in both tables). Differences (errors) between these signals are given in the last row of each table. Notice that average errors in the columns 2, 3, 4, and 5 and are much below 1%. This is a regularity observed for different input images and filter masks as well. On the other hand, errors concerning the B1 and B6 outputs (columns 1 and 6) are much larger than 1%, due to a border effect, which is associated with the fact that these samples are calculated using only two input signals.



(a)



(b)

Fig. 9. Time domain post-layout simulations for input signals from (a) Table 1 and (b) Table 2

Dynamic parameters, like a maximum data rate, can be assessed on the basis of time-domain results shown in Fig. 9 (for currents given in Tables 1 and 2). All output samples are calculated in parallel. In the case of higher currents (bottom), time required to calculate the output image is about 7 times shorter than in the low-current case (upper). Power consumption equals to about 7μW in the first case and 70 μW in the second one. The power increase gives, fortunately, a shorter calculation time. Since our filter works in an asynchronous way, energy per one pixel is the right measure characterizing the power aspect. In both cases the energy per pixel is almost the same and equals approximately 1 pJ.

5 Multirate Systems Realized Using the GVM FIR Filters

The proposed GVM filters can be used in power-efficient multirate signal processing. The idea of decimation and interpolation systems has already been presented for a 1-D FIR filter case. Decimation consists in down-sampling in the input signal preceded by an antialiasing lowpass filtering. Our GVM filter calculates N+1 output samples in parallel. Decimation in this case is simply realized by selecting one or more output samples y. Superiority of such an approach over typical solutions is that samples at the filter output are available within a time period equal to $(M+1) / f_S$ [s], while in typical multirate systems within a time period equal to $1 / f_S$ [s] only. As a result, the subsequent block in the system, that gets the GVM filter output signal, does not need to be as fast as in typical solutions.

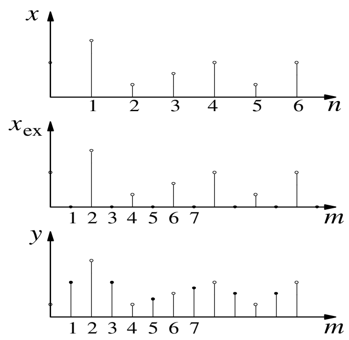


Fig. 10. Interpolation principle: x - input signal, x_{ex} - signal beyond expander, y - output signal

Interpolation by a non-fractional up-sampling factor, K, consists of two stages, as illustrated in Fig. 10. In the first stage, K-1 uniformly spaced zero-valued samples are placed between each pair of samples of the input signal x, shown in the upper diagram, producing the signal x_{ex} – the expander output – shown in the middle diagram. In the next stage, the anti-imaging lowpass filter is used to eliminate undesired spectrum components, which occur in the signal after the up-sampling operation, producing the signal, y, that is shown in the bottom diagram.

In a typical approach, the anti-imaging filter operates with the sampling frequency of the output signal, which is K-times higher than the input sampling frequency [11]. As the proposed GVM filter calculates the p products in parallel, the interpolation is simply realized by summing only selected products p. For example, consider the example 8x4 GVM filter shown in Fig. 4 and assume that anti-imaging filter has the coefficients $h = \{1,\ 2,\ 1\}$ and an up-sampling factor $K = 2$. The output signal (y) shown in Fig. 10 can be calculated using the expander output signal, x_{ex}, like in the equation 9(a) or using the expander input signal, x, like in equation 9(b), which can be simplified to the form 9(c).

$$\text{(a)}\quad \begin{aligned} y_1 &= x_{\text{ex}2}h_0 + 0\cdot h_1 + x_{\text{ex}0}h_2 \\ y_2 &= 0\cdot h_0 + x_{\text{ex}2}\cdot h_1 + 0\cdot h_2 \\ y_3 &= x_{\text{ex}4}h_0 + 0\cdot h_1 + x_{\text{ex}2}h_2 \\ y_4 &= 0\cdot h_0 + x_{\text{ex}4}\cdot h_1 + 0\cdot h_2 \\ &\dots \end{aligned} \qquad \text{(b)}\quad \begin{aligned} y_1 &= x_1h_0 + 0\cdot h_1 + x_0h_2 = x_1h_0 + x_0h_2 \\ y_2 &= 0\cdot h_0 + x_1\cdot h_1 + 0\cdot h_2 = x_1\cdot h_1 \\ y_3 &= x_2h_0 + 0\cdot h_1 + x_1h_2 = x_2h_0 + x_1h_2 \\ y_4 &= 0\cdot h_0 + x_2\cdot h_1 + 0\cdot h_2 = x_2\cdot h_1 \\ &\dots \end{aligned}$$

$$\text{(c)}\quad \begin{aligned} y_1 &= x_1h_0 + x_0h_2 \\ y_2 &= x_1\cdot h_1 \\ y_3 &= x_2h_0 + x_1h_2 \\ y_4 &= x_2\cdot h_1 \\ &\dots \end{aligned} \tag{9}$$

P matrix of the GVM shown in Fig. 4 contains all necessary products required to calculate output samples, y, in parallel. In this case, at least eight samples can be calculated on the basis of this matrix. An advantage of this approach is that such an anti-imaging filter still works with the sampling frequency of the *x* input signal and not with the *K*-times higher sampling frequency of the expander, like in typical systems of this type. Thus, our solution is power efficient (important advantage).

6 Conclusions

New 1-D and 2-D analog current-mode FIR filters have been proposed in this paper. The main block in our filters is the Gilbert vector multiplier (GVM), which enables an automatic normalization of filter coefficients represented by vector of currents. The filter transfer function depends only on relationships between particular currents (current ratios) and not on the current values. Normalization performed in the GVM block allows for keeping the filter gain stable and equal to one. This is of prime importance in the filtration tasks. Our filers are characterized by concurrent data processing and lead to power economic implementations of multirate systems.

To verify the proposed idea, two filters have been design in a CMOS 0.18 μm technology The first circuit is a 1-D 3rd-order filter. Operating with a sampling frequency of 2 MHz it dissipates power of 8 μW. Attenuation observed in post-layout HSICE simulations reaches the level of about 55 dB. Theoretical analysis concerning an influence of the transistor-threshold-voltage mismatch on the filter properties shows that even in the worst case scenario, attenuation higher than 30-dB can be achieved. This makes the 1-D filter to be attractive in many low-precision applications.

The second circuit presented in this paper is an asynchronous image processor. The processor calculates all output pixels in a parallel way and requires no clock generator. This is an important advantage because clock generators are a typical source of feedthrough noises. An image resolution in this experimental circuit is 6 x 1, while a filter mask has resolution of 3 x 1. An effective data rate depends on values of input signals. For example, for a power dissipation of 7 μW and an average level of input currents equal to about 300 nA, the possible data rate equals 1 Mframes/s, while for a power dissipation of 70 μW, the data rate is 7 Mframes/s. Energy consumption per a single pixel is as low as 1 pJ. Another significant advantage of the processor is that the number of frames per a given time unit does not depend on the image resolution. This feature allows for implementing very large and low power image processors operating with a very high data rate. It is possibly, for instance, to build a circuit with the resolution of 100x100 pixels (10000 pixels in parallel every 1 μs), which will dissipate a power of about 10-20 mW only. The effective data rate in this case will be equal to about 60-70 Gbit/s. The filter is easily programmable by means of only several bits and several DC currents. Both lowpass and high-pass filters can be realized using our approach.

References

1. Długosz, R., Iniewski, K.: Flexible Architecture of Ultra-Low-Power Current-Mode Interleaved Successive Approximation Analog-To-Digital Converter for Wireless Sensor Networks. Hindavi VLSI design (2007)

2. Tanner, S., Heubi, A., Ansorge, M., Pellandini, F.: An 8-bit low-power ADC array for CMOS image sensors. In: 1998 IEEE International Conference on Electronics, Circuits and Systems, September 1998, vol. 1, pp. 147–150 (1998)
3. Linan, G., Foldesy, P., Espejo, S., Dominguez-Castro, R., Rodriguez-Vazquez, A.: A 0.5 μm CMOS 106 transistors analog programmable array processor for real–time image processing. In: 25th European Solid-State Circuits Conference (ESSCIRC), pp. 358–361 (1999)
4. Alini, R., et al.: A 200-MSample/S Trellis-Coded PRML Read/Write Channel With Analog Adaptive Equalizer and Digital Servo. IEEE Journal of Solid-State Circuits 32(11), 1824–1838 (1997)
5. Brown, J.E.C., Hurst, P.J., Rothenberg, B.C., Lewis, S.H.: A CMOS Adaptive Continuous-Time Forward Equalizer, LPF, and RAM-DFE for Magnetic Recording. IEEE Journal of Solid-State Circuits 34(2), 162–169 (1999)
6. Vahidfarl, M.B., Shoaei, O., Fardis, M.: A Low Power, Transverse Analog FIR Filter for Feed Forward Equalization of Gigabit Ethernet. In: IEEE International Symposium on Circuits and Systems (ISCAS), Kos Greece (2006)
7. Winstead, C.: Analog Iterative Error Control Decoders, Ph.D dissertation. University of Alberta, ECE Department, Edmonton, Alberta (2004)
8. Croon, J.A., Rosmeulen, M., Decoutere, S., Sansen, W., Maes, H.E.: An easy-to-use mismatch model for the MOS transistor. IEEE Journal of Solid-State Circuits 37(8), 1056–1064 (2002)
9. Długosz, R.: Analog, Continuous Time, Fully Parallel, Programmable Image Processor Based in Vector Gilbert Multiplier. Paper submitted to International Conference Mixed Design of Integrated Circuits and Systems (MIXDES), Poland (June 2007)
10. Hoffmann, H.: Perception through visuomotor anticipation in a mobile robot. Neural Networks 20(1), 22–33 (2007)
11. Vaidyanathan, P.P.: Multirate systems and filter banks. Prentice Hall, Englewood Cliffs (1993)