

# Current-Mode FPAA with CMRR Elimination and Low Sensitivity to Mismatch

Szymon Szczęsny<sup>1</sup> 

Received: 23 October 2015 / Revised: 21 October 2016 / Accepted: 24 October 2016 /  
Published online: 4 November 2016  
© The Author(s) 2016. This article is published with open access at Springerlink.com

**Abstract** This article introduces a current-mode field-programmable analog array (FPAA) architecture with its programming methods. The biggest benefit of the proposed approach is solving the problem of implementing reconfigurable analog circuits in modern nanometre technologies. It is achieved thanks to adopting a switched-current (SI) technique which allows to implement the array using transistors based only on the standard digital CMOS technology. The work describes an implementation of a reconfigurable current mirror basing on using a digital-to-analog converter. The article addresses a routing problem of current-mode modules working in a balanced mode. Author proposes methods for CMRR compensation in a huge array architecture. The array was programmed taking into consideration parasitic elements of the layout with the emphasis on topography mismatch. Examples of implementing a 10-bit digital–analog converter, an elliptic filter with SNR = 40.42 dB, 2D-DCT processor with PSNR = 53.05 dB and RGB-to-YCrCb converter with PSNR = 46.95 dB are presented. The elaborated array can be used as IPcore in a larger mixed-signal system or can act as a dedicated circuit.

**Keywords** Field-programmable analog array · Common mode rejection ratio · Signal processing · Image processing · Current mirror · Mismatch

---

✉ Szymon Szczęsny  
Szymon.Szczesny@put.poznan.pl

<sup>1</sup> Faculty of Computing, Chair of Computer Engineering, Poznań University of Technology, Piotrowo 3A Street, 61-138 Poznan, Poland

## 1 Introduction

The development of reconfigurable analog circuits, especially field-programmable analog arrays (FPAA) is nowadays the most current challenge in the VLSI<sup>1</sup> branch of science which follows the trends of miniaturisation and automation [11, 14, 16, 26, 47]. Solutions for implementing large-scale programmable circuits prototypes with parameters comparable to these of dedicated analog circuits have been appearing in the last few years [1, 3, 22, 24, 30]. Noteworthy examples of conferring the reconfigurable feature for hybrid techniques with a memristor using were introduced [23]. Other works take into consideration the problem of simulating and performing a synthesis of analog structures into array resources [37]. The literature presents new possibilities for adopting analog reconfigurable architectures in the field of neural computing [29]. FPAAs are also subjects of industrial implementations and patents [45]. All of these examples were designed as voltage-mode arrays.

Implementation of reprogrammable circuits working in the voltage mode using amplifiers is becoming impossible in modern nanometre technologies, which are characterised by low supply voltages. Therefore, the literature contains FPAA implementations working in the current mode. One of the first propositions was using a current conveyor working in the continuous time mode [8, 32]. The main benefit of such an approach is the high working frequency of such circuit. However, still, data processing accuracy and the nonlinear relation between the resistance of a tunable resistor, capacitances of tunable capacitors and input frequency remain the biggest disadvantages. Limited functionality and high dependency on the transistor parameters dispersion create additional limitations. The continuous time mode prevents implementing advanced structures. Another FPAA proposition bases on a digitally controlled balanced output transconductor [25]. However, it does not fully work in the current mode, and its implementation requires implementing a gain amplifier. Implementing filters using such structures requires using capacitors; therefore, FPAA contains an array of keyed capacitances. Limited performance of current FPAA implementations is the very reason for their low popularity. In fact, difficulties in implementing analog structures (because of many parasitic effects which also require expertise to deal with), high sensitivity for mismatch and a lack of the possibility of full debug (design for testing strategy [19]) are reasons for low popularity of analog circuits.

Taking into account the above implementations and their limitations, the author decided to propose a fully analogue solution, with a digital interface, which can be implemented using the standard digital CMOS technology and providing the possibility of implementing advanced structures with high data processing accuracy. The author decided to use the current mode, due to the possibility of implementing the solution in modern nanometre CMOS technologies. The presented reconfigurable architecture can be implemented as a standalone chip or IPcore<sup>2</sup> of a larger mixed system. The proposed FPAA structure features flexibility in selecting resources and a

---

<sup>1</sup> Very large scale integration.

<sup>2</sup> Intellectual property—core.

high versatility of applications. The solution is especially dedicated for applications in the sensor technology, in which it is worth performing the initial data preprocessing in an analogue circuit [46]. IoT<sup>3</sup> sensing devices require a reconfigurable structure for the variety of sensing with a feature of low power consumption. Analogue preprocessors and accelerators are used in vision systems for compressing analogue signals coming from image sensors [39]. The reconfigurability of a preprocessor is particularly important because of the possibility of implementing different compression algorithms and selecting the right degree of compression. The work analyses the accuracy of processing of the designed FPAA structure concerning its parameter dispersion. A method for eliminating the concurrent component, present in balanced structures working in the current mode, has also been proposed.

Due to the mentioned potential areas of application, the developed FPAA architecture has been equipped with modules making it possible to perform compression and image processing tasks. The whole current-mode FPAA architecture, reconfigurable modules and their routing are presented in Sect. 2. Few words about the programming method basing on the redundancy feature are presented in Sect. 3. Section 4 presents examples of circuits structures implemented with FPAA. Results of the implementation are discussed in the conclusion section.

## 2 FPAA Architecture

This section takes up a current-mode field-programmable analog array architecture. The architecture described in this work was inspired by an idea explained in [9]. Authors created an analogue circuit simulator working in a balanced structure. Admittedly, on the interface level, the proposed structure behaves like an analogue circuit. However, it is a digital circuit, and the proposed modification bases only on designing a DAC<sup>4</sup> and ADC<sup>5</sup> converters interface. Such implementation lacks any of the benefits brought by analogue solutions, such as low power consumption and small area of such integrated circuitry. Existing modules were used in the project for implementing the example circuits. These modules included an FPGA<sup>6</sup> circuit and 16-bit converters (ADS8412, AD5546). Implementing converters with such high resolution in modern technologies is non-trivial. Thus, the solution is not very practical. The author of the current work decided to propose a fully analogue, balanced structure which, on one hand, provides high data processing accuracy and, on the other, is easier to implement. At the end of the Sect. 2.1, an analysis of the influence of dispersion was conducted.

The structure of the proposed FPAA is based on the CPLD<sup>7</sup> [7] concept in which dedicated modules are attached to the routing core. Next, subsections describe analog current-mode modules used in the FPAA and the proposed routing implementation.

---

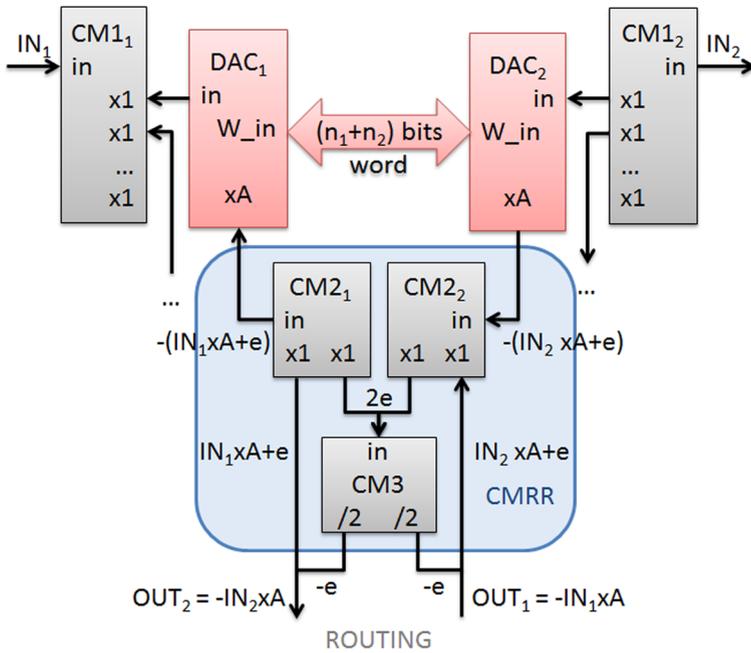
<sup>3</sup> Internet of things.

<sup>4</sup> Digital-to-analog converter.

<sup>5</sup> Analog-to-digital converter.

<sup>6</sup> Field-programmable gate array.

<sup>7</sup> Complex programmable logic device.



**Fig. 1** Reconfigurable current mirror (RCM) output stage implementation

### 2.1 Configurable Current Mirror

One of the most common cells used in circuits working in the current-mode is a current mirror implementing the multiplication operation. In the traditional ASIC<sup>8</sup> implementation, scaling factors in mirrors are determined by choosing the relation between transistor sizes at their input and output stages [12]. With a programmable array, such an implementation has to be configurable with digital words. The structure of the proposed reconfigurable current mirror (RCM) is shown in Fig. 1.

In fact, the figure presents a single output stage implementation of a multi-output RCM. Directions of currents are marked with arrows in the figure. The proposed FPAA works in the balanced mode, and therefore, input signals in the reconfigurable mirror fulfil Eq. 1.

$$IN_1 = -IN_2 \tag{1}$$

CM1<sub>1</sub> and CM1<sub>2</sub> cells are multi-output mirrors with scaling factor equal to 1. They play roles of separate modules and duplicators of input signals. Number of their outputs is equal to number of outputs in the RCM. The outputs of CM1 mirrors are attached to the inputs of digital-to-analog converters (DAC). Their structure is described at the end of the current subsection. At this stage, let us just notice that both converters are

<sup>8</sup> Application-specific integrated circuit.

controlled with a common  $n_1 + n_2$  length word which defines factor  $A$ . Input DAC signals have values from Eq. 2.

$$CA_1[in] = -IN_1, CA_2[in] = -IN_2 \quad (2)$$

Output DAC signals are obtained by multiplying input signals by factor  $A$ . In fact, this factor implements a scaling factor of a single output stage in the RCM. DAC converters may be sources of errors in the processing path. Moreover, as it was discussed in [38] and [33] a common mode rejection ratio (CMRR) [10] may appear in a balanced current-mode structure. Compensation of the concurrent component by optimising the current structure is impossible because its value depends only to a certain extent on choosing MOS transistors parameters. Both of the mentioned phenomena are sources of a nonlinear error  $e$  in current signals. Therefore, a CMRR elimination module [41] was added to the circuit in Fig. 1. Input signals in the CMRR module have values from Eq. 3.

$$\begin{aligned} CM2_1[in] &= -(IN_1 \cdot A + e), \\ CM2_2[in] &= -(IN_2 \cdot A + e) \end{aligned} \quad (3)$$

$CM2$  mirrors are used to duplicate signals. One of the pairs was added in the input node  $CM3$ . Current at this input can be described using Eq. 4 and its simplified version in Eq. 5.

$$\begin{aligned} CM3[in] &= -[-(IN_1 \cdot A + e)] - [-(IN_2 \cdot A + e)] \\ CM3[in] &= A(IN_1 + IN_2) + 2e = 2e \end{aligned} \quad (4)$$

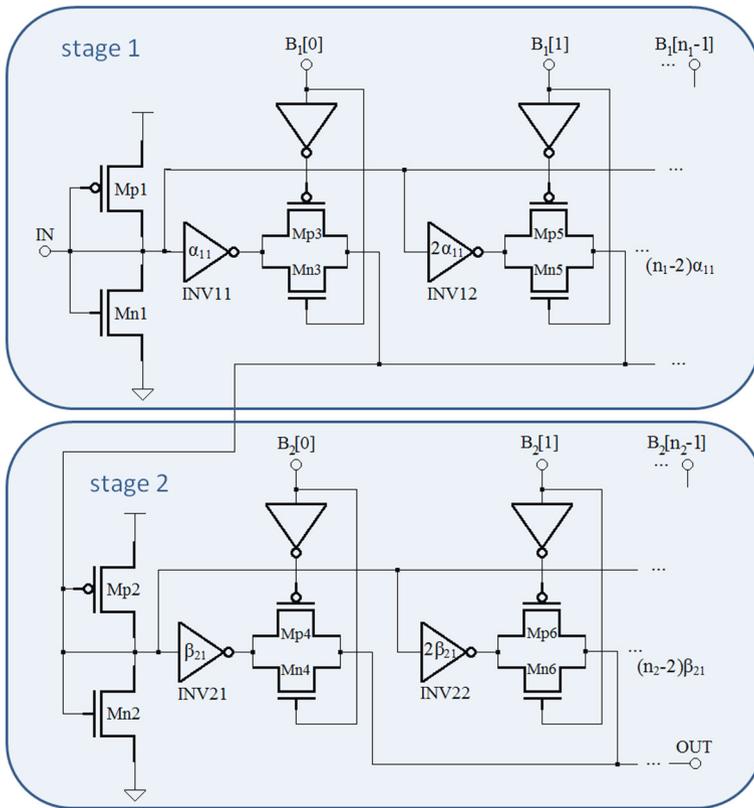
$CM3$  mirror produces the  $e$  error signal by dividing input current by 2. Output from CMRR module is obtained by subtracting  $e$  from  $CM2$  output signals (Eq. 6).

$$OUT_1 = IN_2 \cdot A, OUT_2 = IN_1 \cdot A \quad (6)$$

Taking Eq. 1, output RCM signals can be written in the form of Eq. 7.

$$OUT_1 = -IN_1 \cdot A, OUT_2 = -IN_2 \cdot A \quad (7)$$

Removing the  $e$  component of the signal processing track increases the accuracy of calculations and makes it possible to implement more complex structures. Let us notice that interface of the analysed circuit has the functionality of a current mirror configured by a digital word. Structure of the circuit is fully symmetrical, which minimises the current offset and guarantees a common delay on both paths. The whole structure of the RCM stage is based on current mirrors; therefore, DAC538 converters were as well implemented with the current mirrors concept. In this case, a single mirror implementation is insufficient because of the low resolution and the necessity of using extremely long channels for implementing small factors. Because of the above, a structure in Fig. 2 was proposed.



**Fig. 2** DAC converter data flow

It consists of two stages. The first one is controlled with the  $B_1$  word and implements scaling factor  $\alpha$ . Its output signal is driven to the second stage, which is controlled with  $B_2$  word and implements factor  $\beta$ . It means that the  $A$  factor of the whole converter can be written in the form of Eq. 8.

$$A = \alpha \cdot \beta \tag{8}$$

Each stage is composed of a diode-connected transistor pair  $(Mp1, Mn1), (Mp2, Mn2)$  and inverter-connected transistor pairs  $(INV11, INV12, \dots, INV1n_1), (INV21, INV22, \dots, INV2n_2)$  controlled with CMOS switches. A single diode, in combination with an inverter, works as a current mirror with a scaling factor dependent on choosing transistors sizes. Transistors sizes in diodes and inverters are chosen to achieve (at each  $i$  inverter output) the scaling factor two times bigger than at output  $i - 1$ . Assuming the smallest factors in both stages as  $\alpha_{11}$  and  $\beta_{21}$ , the factors of stages can be represented by Eq. 9 and the final DAC converter factor by Eq. 10.

$$\alpha = \alpha_{11} \sum_{i=0}^{n_1-1} 2^i, \beta = \beta_{21} \sum_{j=0}^{n_2-1} 2^j \quad (9)$$

$$A = \alpha_{11} \cdot \beta_{21} \cdot \sum_{i=0}^{n_1-1} \sum_{j=0}^{n_2-1} 2^{i+j} B_1[i] \cdot B_2[j] \quad (10)$$

Sizes of bit words ( $n_1$  and  $n_2$ ) are marked in Fig. 1. The whole converter has a  $n_1 + n_2$  length input bit word.

Taking into consideration physical parameters of the proposed DAC structure, special attention should be put to choosing channel lengths in transistors which are parts of the input stage and inverters. Power consumption of the converter depends on  $I_{Dp}$  currents flowing through PMOS transistors of the mentioned circuits. Let us consider power consumption of a single stage of a converter. The value of power consumption can be defined with:

$$P = VDD \cdot \sum I_{Dp} \quad (11)$$

Taking into account that each output stage consists of a pair of inverters, the equation can be written as:

$$P = VDD \cdot \left[ I_{Mp1} + \sum_{i=1}^{n_1} (I_{DINV1i} + I_{DI}) \right] \quad (12)$$

This article assumes that PMOS and NMOS transistors in diodes or inverters have a common length. Moreover, all NMOS transistors in diodes and inverters have a common width. Therefore, scaling factors depending on relations between transistors lengths and PMOS transistors widths are established to ensure a symmetrical answer for positive and negative currents and the input of the selected stage. Assuming that transistors work in the saturation region—Eq. 12 can be written in a form:

$$P \approx \varphi \cdot W_{avg} \cdot \left[ \sum_{i=0}^{n_1} \frac{(V_{SG} + V_{Tp})^2}{L_i} + \sum_{i=1}^{n_1} I_{DI} \right], \quad (13)$$

where

$$\varphi = \frac{\mu C_{ox} VDD}{2} \quad (14)$$

$W_{avg}$  parameter determines the average width of transistor channels. In fact, in such strategy it is only slightly different for all PMOS transistors and its choice depends on the common width of NMOS transistors. The above equation therefore proves that parameters which make it possible to decrease power consumption are  $L_i$  lengths of the input and output stages, as well as transistor sizes used for implementing the remaining inverters. From a functional point of view of the FPAA array, sizes of inverters which programme switches are insignificant. However, choosing  $L_i$  lengths

is limited by the time constant of the circuit and influences its work speed. The time constant of the whole circuit from Fig. 2 can be written as Eq. 15, assuming that bit 0 is the least important bit. This means that INV11 implements the lowest scaling factor and consists of the longest transistors, according to Eq. 16 ([12]):

$$\tau = \frac{4 \cdot VDD \cdot C'_{ox} \cdot L_{11} \cdot scale}{I_{on}} \quad (15)$$

$$\alpha_{11} = \frac{L_{Mp1, Mn1}}{L_{11}}, \quad (16)$$

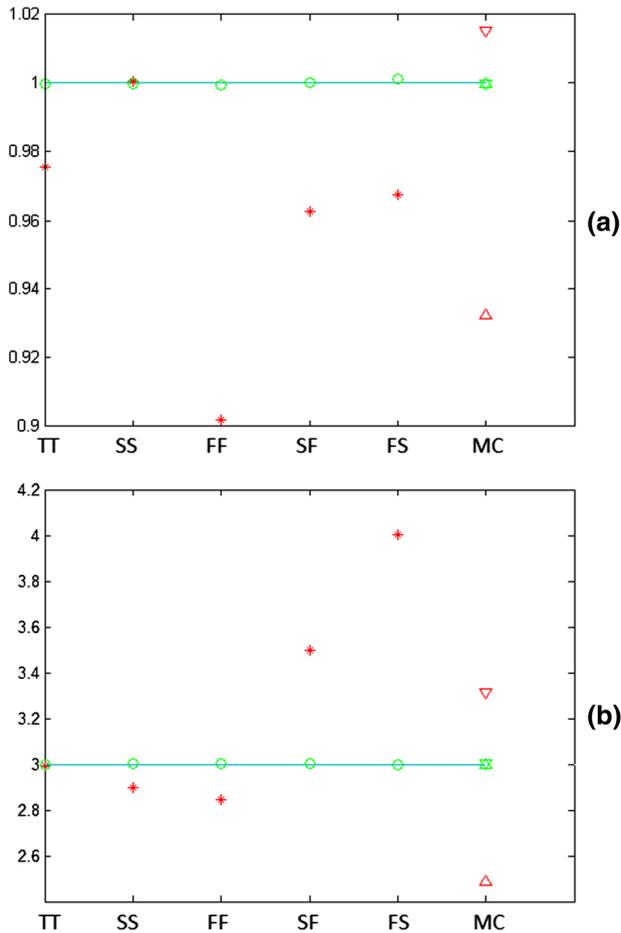
where  $C'_{ox}$  is the capacitance per unit gate area,  $VDD$  is the supply voltage, and  $I_{on}$  is a so-called ON current when  $VDS = VGS = VDD \cdot Scale$  (in  $\mu\text{m}$ ) is a generic scale factor used by the GDS<sup>9</sup> (Calma stream format) layout file provided the topography is drawn with reference to the minimum device dimension [20].

Taking into consideration the problem of choosing transistors sizes in diodes and inverters, two example methods can be suggested. The first one was described in [35] and is based on choosing solutions from a previously generated technology grid. The second method is optimisation using the Hooke–Jeeves algorithm [18] described in [28, 42]. The first approach seems to be faster when having a technology grid. However, the process of calculating the grid is time-consuming. The optimisation method gives solutions with a smaller factor reflection error.

The accuracy of mapping scaling factors also depends on their susceptibility to parameter dispersion of the silicone structure. Figure 3 presents the results of an analysis of the influence of the transistor parameters dispersion on mapping the scaling factor. The research was conducted using threshold voltage mismatch modelling and the Monte Carlo analysis for the circuit in Fig. 1, programmed to implement the functionality of a scaling circuit with coefficient equal to 1.0 and 3.0 and for a classic mirror (composed of a transistor in a diode connection and a transistor with a common gate) designed in the same technology (using piecewise cubic Hermite's [34] and cubic spline [27] Interpolation) and implementing the same scaling factor. The analysis was done with 20 trials. The classic mirror was designed for transistor sizes comparable with the ones in RCM. It is worth noticing that using longer channels makes it possible to design mirrors with lower dispersion (during the MC analysis) [5]. However, according to Eq. 15, the maximal working frequency is then also lowered. As presented in Fig. 3, in the dedicated circuit, the discrepancy between the expected and the actual current mirror multiplier coefficient, assuming a fixed sampling time of circuits with switched currents, may range up to dozens of %. Applying the approach described in Sect. 3 of this article makes it possible to compensate these phenomena in the full range of changes in input signals and maintaining a full symmetry of operation of the RCM module.

In the end, it is worth mentioning a few words about the sizes of transistors in the DAC circuit from Fig. 2, designed in the 180-nm technology. While maintaining the above-described strategy of selecting transistor sizes, in a diode connection, the transistor length equals 0.5  $\mu\text{m}$ . However, in output stages it varies from 0.42 to 15.26  $\mu\text{m}$ .

<sup>9</sup> Graphic database system.

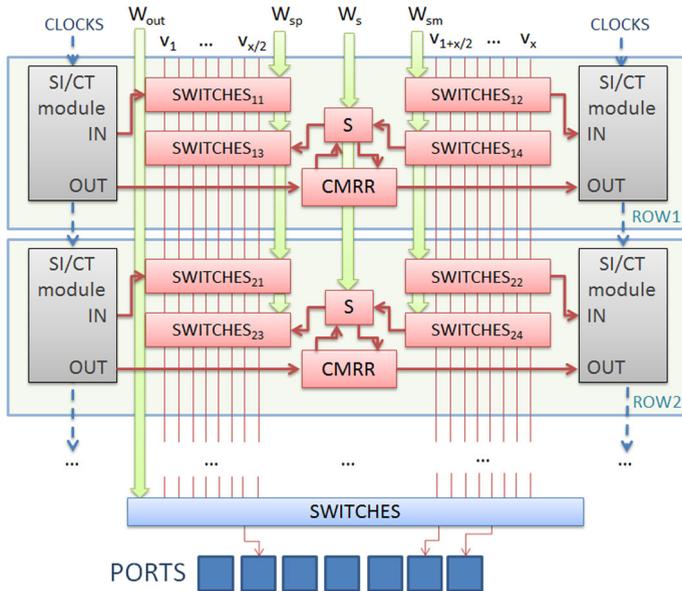


**Fig. 3** Current mirrors parameter dispersion **a** scaling factor 1, **b** scaling factor 3. *Asterisk* Classic mirror (a transistor in diode connection and a transistor with a common gate), *circle* RCM, *solid line* ideal value, *triangle inverted triangle* Monte Carlo analysis. *TT* typical transistors; *SS* slow NMOS, slow PMOS, *FF* fast NMOS, fast PMOS; *SF* slow NMOS, fast PMOS; *FS* fast NMOS, fast PMOS; *MC* Monte Carlo. The analysis was performed using models provided by the Taiwan Semiconductor Manufacturing Company

The lengths were chosen as a result of a consensus between power consumption and the maximum work frequency.

## 2.2 Routing of Array

This subsection presents the problem of a configurable routing of current-mode modules. The RCM described in the previous section is one of the possible modules working in the continuous time (CT) domain. Let us notice that in programmable systems, sequenced circuits are usually preferred; therefore, a switched-current (SI) modules implementation can be suggested. Unfortunately, the SI technique is characterised by

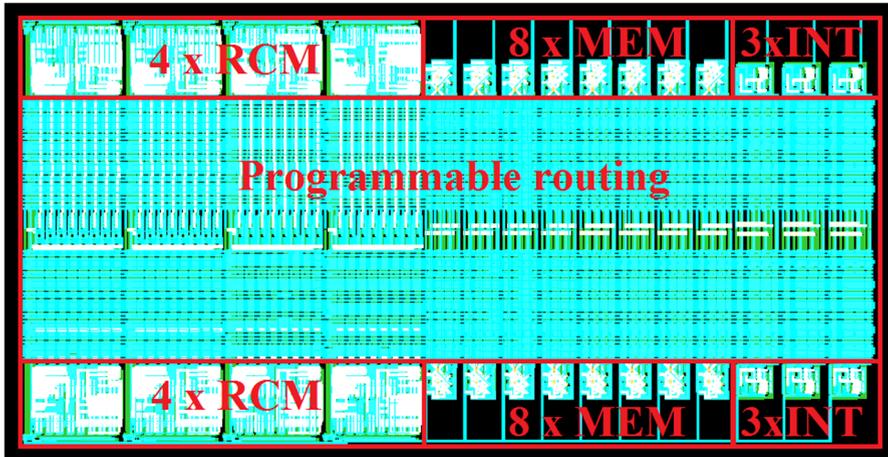


**Fig. 4** Configurable routing of modules

low data processing precision, which has its source especially in CMRR, unmatched between modules and unmatched between transistors used to build modules which work in the balanced mode.

Taking into account the mentioned phenomena, reconfigurable tracking of modules was designed to ensure the symmetry of the final array. Its structure is shown in Fig. 4.

Let us analyse the tracking method basing on ROW1 from the figure. CMRR cells were moved to the centre of routing. Outputs current signals from SI/CT modules are attached directly to the CMRR modules. A set of switches  $S$  controlled with a  $W_s$  word is used to choose the next way of the current flow. Current can flow to the  $v_1 \dots v_{x/2}$  or  $v_{1+x/2} \dots v_x$  nodes and is connected to the suitable node with block of switches  $SWITCHES_{13}$  or  $SWITCHES_{14}$ , respectively. The blocks are controlled using  $W_{sp}$  and  $W_{sm}$  words. Subsequent blocks  $SWITCHES_{11}$  and  $SWITCHES_{12}$  are used to connect inputs to nodes. A single switch is composed of a CMOS pair of transistors with short channels (to minimise switch resistance) and a less than twice minimal width (to minimise parasitic capacitance in the routing node). Let us notice that red arrows in the figure are in fact representations of buses of currents and node lines correspond to single currents. Nodes are common for all of the switches blocks. Any of the nodes can be used as an input or an output port controlled with the  $W_{out}$  word. The whole structure is fully symmetrical, and the symmetry of implemented circuits depends on the method of assigning nodes. Finally let us mention that such an architecture can be easily divided into subcircuits marked in the figure with ROW rectangles. Thanks to such feature, the proposed FPAA concept can be used for ASIC or IPcore devices assembled with smaller modules, depending on required resources.



**Fig. 5** Layout of an example FPAA IPcore with 4 pairs of balanced mode CT mirrors, 6 SI integrators and 16 SI memories

### 2.3 Layout of IPcore

This subsection presents a layout of an example FPAA IPcore designed in the 180-nm technology. The proposed architecture consists of rows shown in Fig. 4. It features large versatility because it can be easily modified with respect to requirements, by adding additional rows including the necessary CT/SI cells. An example of IPcore shown in Fig. 5 consists of 15 ROWs: 4 with an 8-output RCM pair, 3 with an SI integrator [36] pair and 8 with SI memory cells [17] with delay elements. Let us notice that routing is a part of ROWs. It means that there is now routing between ROWs. Moreover, metal1 layer is used to draw signal nets in routing regions. Thanks to the above, using of vias has been largely reduced, similarly to parasitic effects coming from routing. The topography sizes are:  $748 \times 1492 \mu\text{m}$ , and it was used to implement the example filter described in Sect. 4.2 and image processors described in Sect. 4.3.

## 3 Programming

The proposed FPAA architecture has a very beneficial property, which can be seen in the modular structure, the flexibility of choosing scaling factors in RCMs or in routing methods.  $B_1$ ,  $B_2$  words controlling the DAC converter shown in Fig. 2 do not correspond directly to the converted current value. Scaling factor  $A$  from Eq. 10 depends on the multiplication product of words. Hardware calculation of the scaling factor would force the usage of a large digital decoder. The author proposes a quite different programming method based on choosing solutions from the previously generated grid. The method for its generation is described in Sect. 4.1. There are many benefits of such an approach:

1. No need to use a hardware decoder or a ROM memory for storing  $B$  words.
2. The possibility to generate grid of solutions at any design stage, the schematic stage or the layout stage (with parasitics) and even on the stage of the physical chip. It is a way to achieve higher data processing precision with thanks to taking into account the actual properties of circuits.
3. In contrary to the analytic method here, there are no discrepancies between given factors and the obtained ones, caused by parasitics. The only differences have their sources in the resolution of the grid.
4. In the literature, many methods for modelling a mismatch phenomenon with a specific probability were proposed [4]. The approach proposed in the current work makes it possible to synthesise analog circuits taking into account an actual topography mismatch.

Another benefit of such an approach is that transistors parameters in DAC or in the whole RCM module do not have to be calculated with high precision, which means that restriction in Eq. 9 is not crucial. As a proof for this thesis, the worst scenario is analysed in the next section: stage1 and stage2 of DAC from Fig. 2 have their factors  $\alpha_{11}$  and  $\beta_{11}$  equalled and have equalled  $B$  word length (Eq. 17).

$$\alpha_{11} = \beta_{11}, n_1 = n_2 = n_{1,2} \quad (17)$$

In such a case, using an analytic method the number of unique solutions  $N_{US}$  would be strongly reduced and could be expressed using Eq. 18 in comparison with the best scenario where it can have its maximum number  $N_{MAX}$  in Eq. 19.

$$N_{US} = 2^{2n_{1,2}} - \sum_{i=1}^{2^{n_{1,2}}-1} i \quad (18)$$

$$N_{MAX} = 2^{n_1+n_2} \quad (19)$$

Practically, the worst case is the easiest one, as far as the design complexity is concerned, because it means that the DAC is build of the two same stages. Next section proves that no optimisation effort is necessary with FPAA ROWs implementation. This means it is an easy-to-use solution. Section 4.1 presents the efficiency of the redundancy feature in RCM programming, and Sect. 4.2 shows the precision of the proposed FPAA implementation. Both are the answer to the uncertainty of how many resources are needed to ensure the proper data processing accuracy.

Few words about memory size for configuring an FPAA must be said concerning required resources to sum up the current section. It depends on dimensions of the RCM: the number of RCM outputs ( $k_1$ ) and the digital word size ( $n_1, n_2$ ) of the DAC. Next, dimensions of routing are important: the routing verse height ( $k_2$ ) which depends on routed modules interfaces, the number of nodes ( $x$ ), the size of  $W_S$  in each ROW, the number of rows ( $R$ ) and finally the size of port switches, which can be equalled to the number of nodes ( $x$ ). Memory size can be calculated using Eq. 20.

$$M_{size} = R_{RCM}[(n_1 + n_2 k_1 + (1 + k_1)(x + W_S)] + R_{INT, MEM}(N_{in} + N_{out})(x + W_S) + N_p \quad (20)$$

Layout shown in Fig. 5 is build of an 8-output RCM with a 12-bit DAC, integrators with 2 inputs and 2 outputs, and has 32 nodes. All 4 rows with RCMs are programmed with 1572 bits, 3 rows with integrators are routed with 396 bits, 8 rows with memories are routed with 1056 bits, and ports are controlled with 32 bits, which gives 3056 bits of resources needed for configuring the IPcore. The author, using C++, developed computer tools for the design process, integrated with the presented architecture, which automate the process of generating a grid of solutions and configure an FPAA memory based on a description of a synthesised analogue circuit architecture, with its description in VHDL-AMS. The next chapter presents details concerning implementation of circuits of different classes, which were included in the developed system.

## 4 Example Implementations

This section describes four examples of circuits implemented with the FPAA. The first example, which shows a potential of an RCM, is a DAC converter. The second example is an elliptic filter implemented using RCMs and SI integrators modules, and the third and fourth examples are image processors implemented using RCMs and SI memories.

### 4.1 10-Bit DAC Converter

As the first example, a 10-bit digital-to-analog converter is analysed to show the efficiency of the redundancy feature. The converter was implemented using only one RCM module. As mentioned in the previous section, the RCM was designed for the worst case (Eq. 17) to demonstrate the low sensitivity of the implemented examples according to array parameters. The grid of solutions obtained from post-layout simulations of the single RCM module gives a set of assignments of scaling factors  $A$  (Eq. 8) to concatenations  $B_1 B_2$ . Let us notice that the number of possible solutions  $N$  which allow to implement a searching factor  $S$  with an acceptable mismatch  $e$  varies in the whole set. Moreover, this number is inversely proportional to the value represented by multiplication  $B_1 \cdot B_2$  (Eq. 21).

$$N_{S+e} \sim \frac{1}{B_1 \cdot B_2} \quad (21)$$

In other words, having the DAC from Fig. 2 designed for the set of possible factors  $\langle S_{min}, S_{max} \rangle$  there are many solutions which implement the factor approximately equalled to  $S_{min}$  and only one solution which implements the factor approximately equalled to  $S_{max}$ . This only one solution corresponds to concatenation  $B_{1MAX} B_{2MAX}$ . Because of these properties, the mismatch  $e$  depends directly on the multiplication of  $B$  words (Eq. 22).

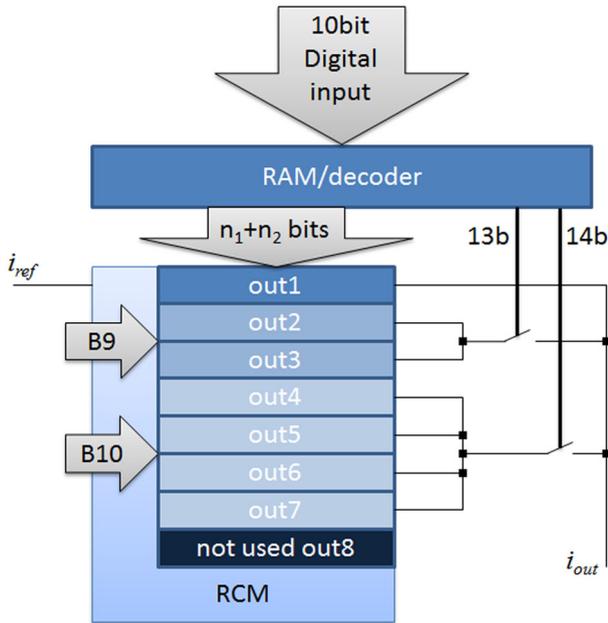
$$e \sim B_1 \cdot B_2 \quad (22)$$

It means that mismatch may depend on the searching factor value  $S$  (Eq. 23).

$$e \sim S \quad (23)$$

**Table 1** RCM and grid parameters from post-layout simulation

180-nm Technology		
Parameter	Symbol	Value
$B_1$ word size	$n_1$	6
$B_2$ word size	$n_2$	6
Minimal scaling factor	$S_{MIN}$	0.02
Maximal scaling factor	$S_{MAX}$	4.2
Average $S$ mismatch	$e_{avg}$	0.0657%
Maximal $S$ mismatch	$e_{MAX}$	2.24%



**Fig. 6** 10-bit DAC converter realised using a single RCM

The above dependencies raise a question about the necessary resources which allow to obtain satisfactory parameters of circuits implemented with the FPAA. Table 1 shows parameters of a grid of solutions obtained for the RCM designed with specifications from the previous section. DAC stages were designed for given limits of the set of scaling factors. The maximum error of the factor reflection is equal to 2.24% of its value.

The architecture of the proposed 10-bit DAC converter is shown in Fig. 6. It was built using seven outputs of a single RCM. The first output is controlled using 12 bits. The next two outputs are switched together with a single bit, and the next four outputs with another bit. Outputs 2 and 3 act as sources of current which corresponds to the 9th DAC bit and are configured by word  $B_9$ . Outputs 4–7 act as sources of current which corresponds to the 10th DAC bit and are configured by word  $B_{10}$ . A digital

**Table 2** Parameters of a converter in comparison with other current-mode implementations

Parameter	[43]	[44]	[6]	Current work
Technology	0.35 $\mu\text{m}$	90 nm	180 nm	180 nm
Supply (V)	3.3	1.2/2.5	1.8	1.8
Input current	(0.2–3) mA	16 mA	15 mA	(0–10) $\mu\text{A}$
Speed (Ms/s)	31.25	1250	500	5
DNL (LSB)	–	$\pm 0.5$	$\pm 0.6$	$\pm 0.65$
INL (LSB)	$\pm 1$	$\pm 1.2$	$\pm 1$	$\pm 0.56$
FoM (fJ/step)	407.8	12.5	52.7	223.6
Power (mW)	26.1	128	216	2.29

RAM or a digital decoder is used to decode the 10-bit word of the converter, out of the 14 bits which control the circuit.

Decoding is done using the previously generated grids of solutions. The main problem is to find  $B9$  and  $B10$  words with the minimal mismatch in the output current. Using just one grid (same as for configuring output 1) does not provide a possibility to obtain satisfactory parameters of the converter because factors in the RCM could in this case be chosen only from solutions approximately equalled to  $i_{ref} \cdot S_{MAX}$  with the maximal mismatch. Equations (24) present ranges of possible solutions to implement the first output current  $i_{out1}$  and the next output currents  $i_{outj}$ .

$$\begin{aligned} i_{out1} &\in \langle 0, MAX_1 \rangle, \\ i_{outj} &\in \langle 0, j \cdot MAX_1 \rangle, MAX_1 = i_{ref} \cdot S_{MAX} \end{aligned} \quad (24)$$

Parameters of the converter can be improved provided the next  $j$  output is programmed using a new grid, generated in the case where its scaling factor  $S_j$  is chosen from a different range than factor  $S_{j-1}$ . It can be proved that the distribution of redundancy is more uniform in the whole range used to design the converter if Eq. (25) is fulfilled.

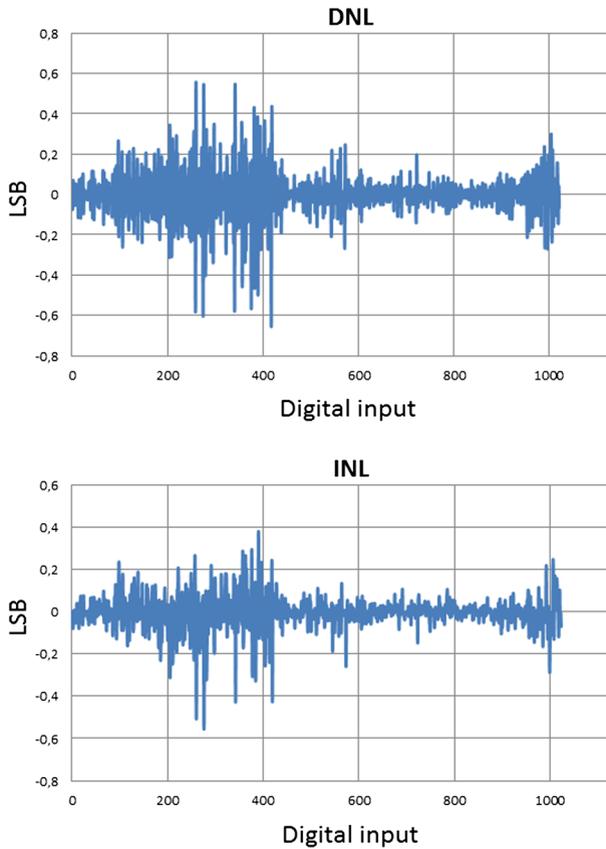
$$\begin{aligned} i_{out1} &\in \langle 0, MAX_1 \rangle, \\ i_{outj} &\in \langle (1 - \frac{\sqrt{2}}{2}) \cdot MAX_{j-1}, (2 - \frac{\sqrt{2}}{2}) \cdot MAX_{j-1} \rangle \end{aligned} \quad (25)$$

Figure 7 presents INL<sup>10</sup> and DNL<sup>11</sup> parameters of the designed converter. Let us notice that the figures reflect the distribution of redundancy in the whole range of the converter factor. INL and DNL deviations increase from 0 to 256 bits, according to redundancy decreasing in the first grid. Next, at bit 427, after crossing the limit from eq. (25) an increasing trend is hampered because of broadcasting a possibility of choosing a factor by adding another grid. A small downgrade of converter parameters is observed at the end of the range in which factors in outputs 3–7 have values which influence mismatch, according to eq. (23).

Converter parameters are compared in Table 2 with corresponding current-mode ASIC implementations in suitable technologies taken from the literature.

<sup>10</sup> Integral nonlinearity.

<sup>11</sup> Differential nonlinearity.

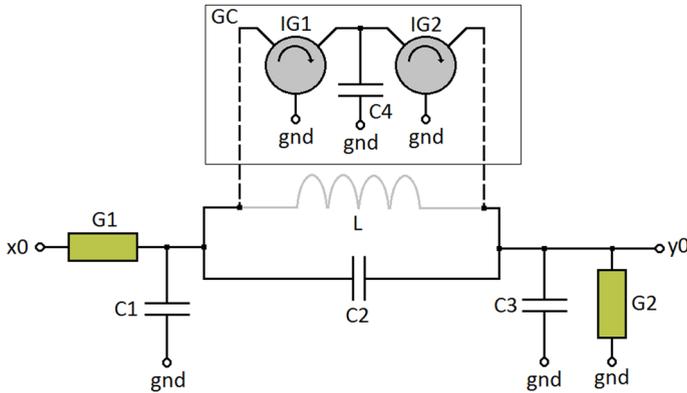


**Fig. 7** Differential nonlinearity and integral nonlinearity of a 10-bit DAC converter

## 4.2 Elliptic Filter

This section presents an example of an elliptic filter implementation. Example is taken from [35]. The filter has the following parameters: third order, 20-dB attenuation in the stop band and a 0.6-dB ripple in the pass band. The prototype of a ladder filter is presented in Fig. 8.

The inductor  $L$  was replaced by a gyrator–capacitor circuit ( $IG1 - C4 - IG2$ ) [13, 21]. Parameters of the gyrator–capacitor prototype were calculated using the method proposed in [15]. Choosing the calculation method depends on hardware resources of the FPAA matrix, especially concerning the size of the grid calculated for the RCM circuit. Note that the maximum parameter dispersion cannot be higher than the one defined in Eq. 26, basing on Eq. 10. Table 3 presents parameter dispersion concerning gyrator, capacitance and conductance calculated for the analysed example with two methods: the Hooke–Jeeves algorithm [18] and the Powell’s method.



**Fig. 8** Prototype of a third-order ladder filter

**Table 3** Parameter dispersion of the elliptic filter

Dispersion	Hooke–Jeeves algorithm	Powell’s method
AVG	0.921	1.142
MAX	1.181	1.437

$$DISP = \langle \alpha_{11} \cdot \beta_{21}, \alpha_{11} \cdot \beta_{21} \sum_{i=0}^{n_1-1} \sum_{j=0}^{n_2-1} 2^{i+j} \rangle \tag{26}$$

Current-mode implementation of the filter can be obtained by solving node voltage equations; therefore, parameter dispersion of the SI circuit depends on parameter dispersion of the GC model. The dispersion should be confronted with parameters  $S_{MIN}$  and  $S_{MAX}$  from Table 1, as well as Eq. 23 which defines the accuracy of mapping parameters of the circuit depending on the sizes of these parameters. Both proposed methods make it possible to calculate parameters of the model with values which have counterparts in the lower area of the solution grid, which, in turn, ensures mapping with high accuracy. Calculations can be made automatically using the environment proposed in another work [16]. Below, a snippet of a VHDL-AMS schematic description was added to present the spread of scaling factors and the whole architecture of the filter with a balanced structure. The description is readable by the EDA system, developed by the author, which parses a filter architecture into a bit stream programming an FPAA memory. As shown, 4 pairs of current mirrors and 3 integrator cells are needed during the placement. Mirrors are numerated according to ROWs from Fig. 4, and they are placed in pairs  $([CMXXp, CMXXm], ROW_{XX+1})$  in the array. In the filter, there are 14 nodes which have to be assigned to proper nodes in the array routing. They are numerated with symbols  $sXp$  and  $sXm$  which corresponds to  $v_1 \dots v_{x/2}$  and  $v_{1+x/2} \dots v_x$ , respectively, in Fig. 4.  $W_S$  word manages the change in polarity of nodes according to data from the netlist.

```
entity filter is
port( terminal input x0p, x0m : electrical;
```

```

    terminal output y0p, y0m : electrical );
end entity filter;
architecture SCHEMATIC of filter is
    constant const01 : real := 0.8957; constant const02 : real := 0.7092;
    constant const03 : real := 0.3777; constant const04 : real := 1.499;
    constant const05 : real := 1.265; constant const06 : real := 0.5431;
    constant const07 : real := 0.107; constant const08 : real := 0.3426;
    ...
terminal s01p,s01m,s1p,s1m,s2p ... s7p,s7m : electrical;
begin
    -- input and output signals
    s01p <= x0p; s01m <= x0m; y0p <= s7p; y0m <= s7m;
    -- Integrators:
int0 : INT port map( xp=>s1p, xm=>s1m, yp=>s2p, ym=>s2m, clkint=>clk );
int1 : INT port map( xp=>s3p, xm=>s3m, yp=>s4p, ym=>s4m, clkint=>clk );
int2 : INT port map( xp=>s5p, xm=>s5m, yp=>s6p, ym=>s6m, clkint=>clk );
    -- Current Mirrors:
CM00p : CM generic map (coeff1=>const01) port map( x=>s01p, y1=>s1m );
CM00m : CM generic map (coeff1=>const01) port map( x=>s01m, y1=>s1p );
CM01p : CM generic map (coeff1=>const01,coeff2=>const04,...) port
    map(x=>s2p,y1=>s1p,y2=>s3m,y3=>s6m);
CM01m : CM generic map (coeff1=>const01,coeff2=>const04,...) port
    map(x=>s2m,y1=>s1m,y2=>s3p,y3=>s6p);
CM02p : CM generic map (...) port map(x=>s4p,y1=>s1p,y2=>s5m);
CM02m : CM generic map (...) port map(x=>s4m,y1=>s1m,y2=>s5p);
CM03p : CM generic map (...) port map(x=>s6p,y1=>s2m,y2=>s3p,y3=>s5p,y4=>s7m);
CM03m : CM generic map (...) port map(x=>s6m,y1=>s2p,y2=>s3m,y3=>s5m,y4=>s7p);
end architecture SCHEMATIC;

```

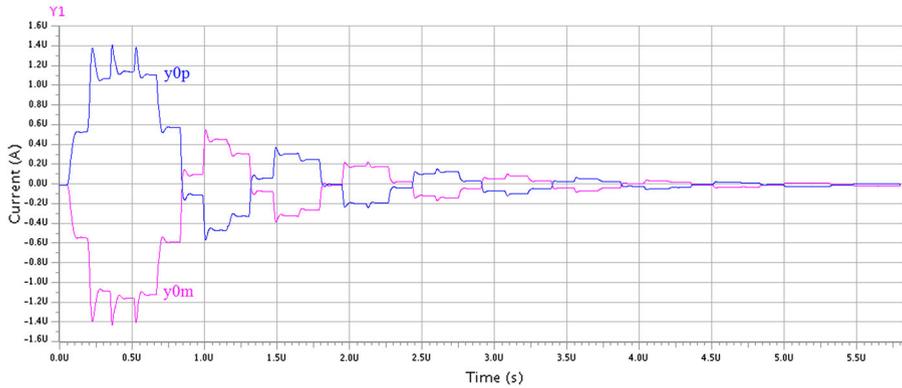
Figure 9 presents a filter pulse response in the time domain, obtained in post-layout simulations with a parasitics extraction, and Fig. 10 shows its answer in the frequency domain, obtained using the FFT. The SNR<sup>12</sup> coefficient of the simulated filter is 40.42 dB. The power consumption of an IPcore is equal to 22.97 mW with a 1.8 V power supply. The maximum clock frequency in integrators is 3.6 MHz.

### 4.3 Image Processors

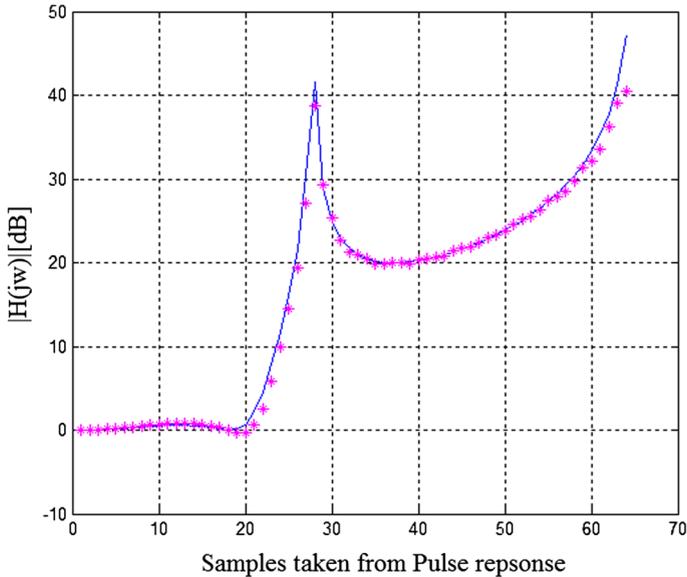
In vision systems, FPAA accelerators perform preprocessing of analogue signals coming from a sensor. Colour space conversion is often carried out in order to eliminate the chrominance component of the image, followed by a compression, using a 2-dimensional discrete Fourier transform (2D-DCT<sup>13</sup>). This section presents results of

<sup>12</sup> Signal-to-noise ratio.

<sup>13</sup> Discrete cosine transform.



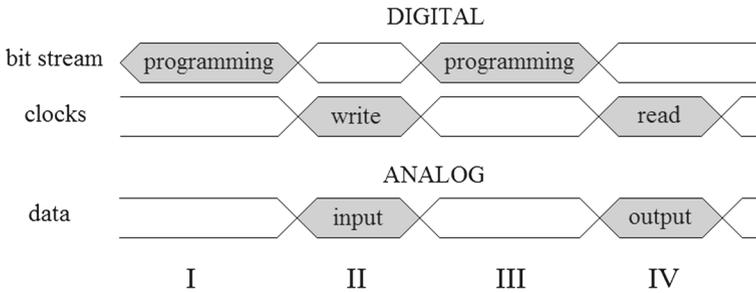
**Fig. 9** Pulse filter response obtained for a balanced structure in post-layout simulations



**Fig. 10** The frequency characteristic of a filter implemented with an FPAA: *solid line* ideal characteristic, *asterisk* post-layout simulations of the implemented circuit

the implementation of both circuits on the sensor. Both the DCT and the colour space conversion are transformations used in many image standards, e.g. JPEG and MPEG.

The idea of the 2D-DCT has been repeatedly discussed in the literature [2]. The hardware implementation of calculating the transform comes down to implementing the following equation 27:



**Fig. 11** 2D-DCT calculation with FPAA reprogramming

$$Y = Z^T C^T, Z = X^T C^T \tag{27}$$

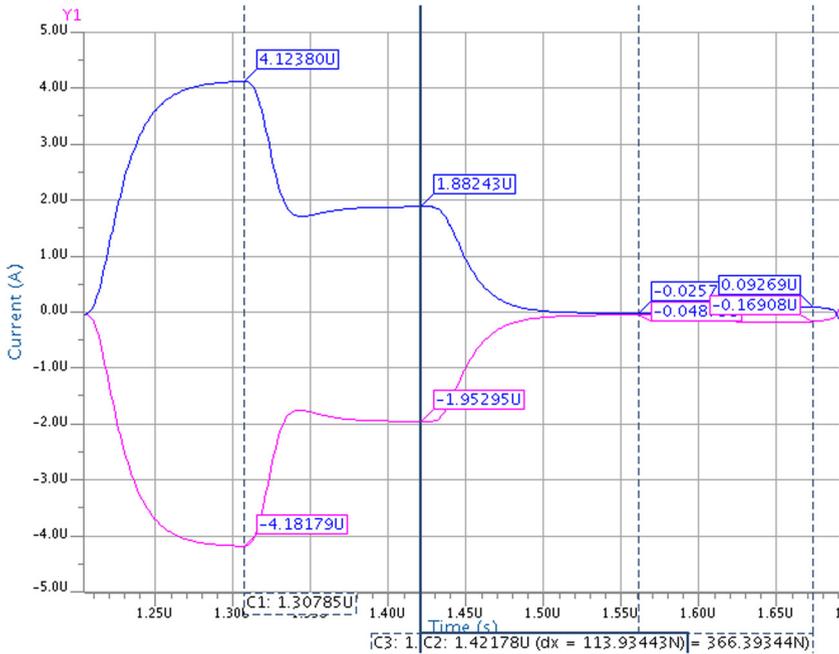
$X$  is the frame of the processed image and the  $C$  matrix, in case of the  $4 \times 4$  DCT, has the following form 28:

$$C = \frac{1}{2} \cdot \begin{bmatrix} a & a & a & a \\ b & c & -c & -b \\ a & -a & -a & a \\ c & -b & b & -c \end{bmatrix} \tag{28}$$

where  $a = \frac{1}{2}$ ,  $b = \sqrt{\frac{1}{2}} \cos(\frac{\pi}{8})$ ,  $c = \sqrt{\frac{1}{2}} \cos(\frac{3\pi}{8})$ . The hardware implementation of the above transformation comes down to implementing multiplication operations by coefficients  $a$ ,  $b$ ,  $c$  and additions resulting from matrix multiplication operations. Implementing a two-dimensional transformation requires two computing blocks and a memory block for intermediate results. Constants in the  $C$  matrix were implemented as scaling factors in RCM mirrors. The sign of the multiplier depends on connecting the output of the mirror to a proper inverting or non-inverting node in the balanced structure. The discussed FPAA structure makes an on-the-run reconfiguration possible, which means that the second matrix multiplication operation can be implemented in the same multiplying block, on condition that the matrix routing is changed. Figure 11 shows subsequent sequences of the operation.

In the I sequence, the bit stream programmes scaling factors of the multiplying block, as well as the routing for implementing the calculation of a one-dimensional DCT. In the II sequence, in the configured processor, multiplying blocks calculate the first matrix operation, and clocks control loading all the calculated data into the SI memory. Once the data have been loaded, the array routing is reprogrammed during the III sequence, so that memory outputs and inputs of multiplying blocks are on common nodes, so that data stored in the memory will be again a subject of a matrix operation. Additionally, matrix output nodes for the calculated signals are selected. During the IV sequence, the second calculation of the one-dimensional transform takes place, resulting in a two-dimensional transform of the original input data.

Figure 12 presents a sample output waveform of a two-dimensional transform, calculated from INPUT data with Eq. 29. Equation 30 shows the result of a perfect



**Fig. 12** 2D-DCT processor’s response for input signals from Eq. 29

2D-DCT, while Eq. 31 presents the array response calculated and scaled to the dynamic range. The values are given in  $\mu\text{A}$ .

$$DCTIN = \begin{bmatrix} 2 & 2 & 2 & 2 \\ 1.5 & 1.5 & 1.5 & 1.5 \\ 1 & 1 & 1 & 1 \\ 0.5 & 0.5 & 0.5 & 0.5 \end{bmatrix} \tag{29}$$

$$IDEAL = \begin{bmatrix} 5 & 0 & 0 & 0 \\ 2.231 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0.159 & 0 & 0 & 0 \end{bmatrix} \tag{30}$$

$$CALC = \begin{bmatrix} 4.942 & 0 & 0 & 0 \\ 2.282 & 0 & 0 & 0 \\ 0.043 & 0 & 0 & 0 \\ 0.162 & 0 & 0 & 0 \end{bmatrix} \tag{31}$$

Preprocessor parameters in Table 4 are compared with implementations of 2D-DCT  $4 \times 4$  analogue processors, as classic dedicated circuits, in the same technology. A greater processing accuracy is mainly the effect of programming the architecture, taking into account routing parasites and the loads.

**Table 4** DCT results comparison

Design	[31]	[40]	Current work
Technology	0.18 $\mu\text{m}$	0.18 $\mu\text{m}$	0.18 $\mu\text{m}$
Transform size	$4 \times 4$	$4 \times 4$	$4 \times 4$
PSNR (dB)	47	35.5	53.05
Transform speed ( $\mu\text{s}$ )	<2.5	>0.1	1.7

As a second example of an analogue preprocessor, an RGB-to-YCrCb<sup>14</sup> colour space converter [12] was implemented. The transformation makes it possible to eliminate redundant information about the colour and to transmit, to the subsequent signal processing track, only information about the brightness of a pixel. It comes down to calculating a simple matrix multiplication described with Eq. 32.

$$\begin{bmatrix} \pm Y \\ \pm Cr \\ \pm Cb \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ 0.5 & -0.419 & 0.081 \\ -0.169 & -0.331 & 0.5 \end{bmatrix} \begin{bmatrix} \mp R \\ \mp G \\ \mp B \end{bmatrix} \quad (32)$$

The preprocessor implementation algorithm is similar to the previous case and is based on the implementation of multiplying matrix elements as scaling factors in RCM blocks. In this case 6 three-output mirrors are required. The work of the circuit was verified in post-layout simulations, in which the following PSNR<sup>15</sup> coefficient was obtained: 46.95 dB. Implementing the same structure as a dedicated circuit in the same technology after optimisation gives the PSNR factor equal to 49.16 dB, and in the 90-nm technology, 43.54 dB [28].

## 5 Conclusion

The article presents the architecture of a current-mode field-programmable analog array, which is an answer for problems occurring while implementing a classic FPAA in modern technologies. Details about the proposed structures were explained, and few words about its properties were said. The proposed solution provides many advantages, compared to existing solutions. Firstly, it makes it possible to implement analogue reprogrammable circuits using a standard, digital CMOS technology. Furthermore, using the current mode makes it possible to implement it in submicron technologies, which are difficult for implementing FPAA circuits working in the voltage mode. The proposed solution features a high processing accuracy and resistance to the mismatch phenomenon, thanks to the proposed software methodology, which takes into account load parasites and routing of modules in the process of their programming. A method for synthesising analog structures from their VHDL-AMS descriptions corresponding to the existing digital solutions. The solution has versatile applications, which has been proven by examples of implementation of different classes of circuits, like a converter,

<sup>14</sup> Y-luma component, Cr and Cb—red-difference and blue-difference chroma components.

<sup>15</sup> Peak signal-to-noise ratio.

a filter and image processors. Comparing the processing accuracy with dedicated circuits, designed in the same technology, proves that the proposed architecture and programming methods can compete with the existing solutions and should contribute to popularisation of reconfigurable analog circuits.

**Open Access** This article is distributed under the terms of the Creative Commons Attribution 4.0 International License (<http://creativecommons.org/licenses/by/4.0/>), which permits unrestricted use, distribution, and reproduction in any medium, provided you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license, and indicate if changes were made.

## References

1. J. Becker, F. Henrici, S. Trendelenburg, M. Ortmanns, Y. Manoli, A field-programmable analog array of 55 digitally tunable OTAs in a hexagonal lattice. *IEEE J. Solid State Circuits* **43**, 2759–2768 (2008)
2. V. Bhaskaran, K. Konstantinidies, *Image and Video Compression Standards* (Kluwer, Boston, 1995)
3. S. Brink, J. Hasler, R. Wunderlich, Adaptive floating-gate circuit enabled large-scale FPAA. *IEEE Trans. VLSI Syst.* **22**(11), 2307–2315 (2014)
4. M. Conti, G. Dalla Betta, S. Orcioni, G. Soncini, C. Turchetti, C., N. Zorzi, Test structure for mismatch characterization of MOS transistors in subthreshold regime, in *Proceedings of the IEEE International Conference on Microelectronic Test Structure*, vol. 10 (1997)
5. R. Dlugosz, T. Talaska, W. Pedrycz, Fellow IEEE, Current-mode analog adaptive mechanism for ultra-low-power neural networks. *IEEE Trans. Circuits Syst. II Express Briefs* **58**(1), 31–35 (2011)
6. K. Doris, J. Briaire, D. Leenaerts, M. Vertregt, A. van Roermund, A 12 b 500 MS/s dac with > 70 dB SFDR up to 120 MHz in 0.18 μm CMOS. *IEEE ISSCC Dig. Tech. Papers*, pp. 116–117 (2005)
7. R. Dueck, *Digital Design with CPLD Applications and VHDL*, Cengage Learning, 2nd ed. (2011)
8. V.C. Gaudet, P.G. Gulak, CMOS implementation of a current conveyor-based field-programmable analog array, in *Conference of Signals, Systems and Computers*, vol. 2 (1997), p. 1156
9. C. Gianni, S. Pennisi, G. Scotti, A. Trifiletti, The universal circuit simulator: a mixed-signal approach to-port network and impedance synthesis. *IEEE Trans. Circuits Syst. I Reg. Pap.* **54**(10), 2178–2183 (2007)
10. G. Giustolisi, G. Palmisano, G. Palumbo, CMRR frequency response of CMOS operational transconductance amplifiers. *IEEE Trans. Instrum. Meas.* **49**(1), 137–143 (2000)
11. H. Graeb, F. Balasa, R. Castro-Lopez, Y.-W. Chang, F.V. Fernandez, P.-H. Lin, M. Strasser, Analog layout synthesis - Recent advances in topological approaches, in *Design, Automation & Test in Europe Conference & Exhibition* (2009), pp. 274–279
12. A. Handkiewicz, *Mixed-Signal Systems: A Guide to CMOS Circuit Design* (Wiley, New York, 2002)
13. A. Handkiewicz, P. Katarzyński, S. Szczęsny, M. Naumowicz, M. Melosik, P. Śniatała, VHDL-AMS in switched-current analog filter pair design based on a gyrator-capacitor prototype circuit. *Int. J. Numer. Model. Electron. Netw. Devices Fields* **27**(2), 268–281 (2014)
14. A. Handkiewicz, P. Katarzyński, S. Szczęsny, M. Naumowicz, M. Melosik, P. Śniatała, M. Kropidłowski, Design automation of a lossless multiport network and its application to image filtering. *Expert Syst. Appl.* **41**(5), 2211–2221 (2014)
15. A. Handkiewicz, P. Katarzyński, S. Szczęsny, J. Wencel, P. Śniatała, Analog filter pair design on the basis of a gyrator-capacitor prototype circuit. *Int. J. Circuit Theory Appl.* **40**(6), 539–550 (2012)
16. A. Handkiewicz, S. Szczęsny, M. Naumowicz, P. Katarzyński, M. Melosik, P. Śniatała, M. Kropidłowski, SI-Studio, a layout generator of current mode circuits. *Expert Syst. Appl.* **42**(6), 3205–3218 (2015)
17. A. Handkiewicz, P. Śniatała, M. Łukowiak, Low-voltage high-performance switched current memory cell, in *Proceedings of the Ninth Annual IEEE International ASIC Conference and Exhibit, ASIC 97*, Portland, Oregon, 7–10 Sept. 1997, pp. 12–16
18. R. Hooke, T.A. Jeeves, 'Direct search' solution of numerical and statistical problems. *J. Assoc. Comp.* **8**(2), 212–229 (1961)
19. J.L. Huertas, *Test nad Design-for-Testability in Mixed-Signal Integrated Circuits* (Kluwer, Boston, 2004)

20. R. Jacob Baker, CMOS—Circuit Design, Layout and Simulation, Rev. 2nd edn., IEEE Solid-State Circuits Society, Sponsor; IEEE SSCS Liaison to the IEEE Press, Stuart K. Tewksbury, Wiley-Interscience (2008)
21. P. Katarzyński, M. Melosik, M. Naumowicz, S. Szczęsny, Symbolic analysis in gyrator–capacitor filters, in *Proceedings of the 19th International Conference IEEE Mixed Design of Integrated Circuits and Systems (MIXDES)* (2012), pp. 392–397
22. H. Kutuk, S.M. Kang, A field-programmable analog array (FPAA) using switched-capacitor technique, in *Proceedings of the IEEE International Symposium on Circuits and Systems*, vol. 4 (1996), pp. 41–43
23. M. Laiho, J. Hasler, J. Zhou, D. Chao, L. Wei, E. Lehtonen, J. Poikonen, FPAA/Memristor hybrid computing infrastructure. *IEEE Trans. Circuits Syst. I Regul. Pap.* **62**(3), 906–915 (2015)
24. C.A. Looby, C. Lyden, Op-amp based CMOS field-programmable analogue array. *Proc. IEE Circuits Devices Syst.* **147**, 93–95 (2000)
25. S.A. Mahmoud, Digitally controlled cmos balanced output transconductor and application to variable gain amplifier and Gm-C filter on field programmable analog array. *J. Circuits Syst. Comput.* **14**(4), 667–684 (2005)
26. R. Martins, N. Lourenco, N. Horta, LAYGEN II-automatic layout generation of analog integrated circuits. *IEEE Trans. Comput Aided Design Integr. Circuits Syst.* **32**(11), 1641–1654 (2013)
27. J. Mathews, K. Fink, *Numerical Methods Using Matlab* (Prentice-Hall, Upper Saddle River, 1999)
28. M. Naumowicz, M. Melosik, P. Katarzyński, A. Handkiewicz, Automation of CMOS technology migration illustrated by RGB to YCrCb analogue converter. *Opto-Electron. Rev.* **21**(3), 326–331 (2013)
29. S. Nease, S. George, P. Hasler, S. Koziol, S. Brink, Modeling and implementation of voltage-mode CMOS dendrites on a reconfigurable analog platform. *IEEE Trans. Biomed. Circuits Syst.* **6**(1), 76–84 (2012)
30. B. Pankiewicz, M. Wojcikowski, S. Szczepanski, S. Yichuang, A field programmable analog array for CMOS continuous-time OTA-C filter applications. *J. Solid State Circuits* **37**(2), 125–136 (2002)
31. M. Pänkäälä, K. Virtanen, A. Paasio, An analog 2-D DCT processor. *IEEE Trans. Circuits Syst. Video Technol.* **16**(10), 1209–1216 (2006)
32. C. Premont, R. Grisel, N. Abouchi, J.P. Chante, Current-conveyor based field programmable analog array, in *Proceedings of the IEEE MWSCAS*, Ames, IA (1996)
33. X. Quan, S.H.K. Embabi, E. Sanchez-Sinencio, Improved fully-balanced current-mode integrator. *Electron. Lett.* **34**(1), 1–3 (1998)
34. G. Recktenwald, *Numerical Methods with MATLAB—Implementations and Applications* (Prentice-Hall, Upper Saddle River, 2000)
35. R. Rudnicki, Chosen tools for automatic design of switched-current circuits, (in Polish), Ph.D. dissertation, accessible via (Poznań University of Technology, Poznań, Poland, 2006)
36. R. Rudnicki, M. Kropidłowski, A. Handkiewicz, Low power switched-current circuits with low sensitivity to the rise/fall time of the clock. *Int. J. Circuit Theory Appl.* **38**(5), 471–486 (2010)
37. C.R. Schlottman, C. Petre, P.E. Hasler, A high-level simulink-based tool for FPAA configuration. *IEEE Trans. VLSI Syst.* **20**(1), 10–18 (2012)
38. S. Szczęsny, Computer Tools for Layout Generation of Switched-Current Circuits, Ph.D. dissertation, accessible via (Poznań University of Technology, Poznań, Poland, 2013)
39. S. Szczęsny, *FPAA Accelerator for Machine Vision systems*. *Przegląd Elektrotechniczny*, ISSN 0033-2097, R. 91 NR 9/2015
40. S. Szczęsny, M. Naumowicz, A. Handkiewicz, SI-Studio—environment for SI circuits design automation. *Bull. Pol. Acad. Sci. Tech. Sci.* **60**(4), 757–762 (2012)
41. P. Śniatała, A. Handkiewicz, M. Naumowicz, S. Szczęsny, M. Melosik, P. Katarzyński, M. Kropidłowski, Automated design of switched current sigma-delta modulator with a new comparator structure, in *MIXDES 2013, 20th International Conference* (2013), pp. 198–203
42. P. Śniatała, M. Naumowicz, S. Szczęsny, J.L.A. de Melo, N. Paulino, J. Goes, Current mode sigma-delta modulator designed with the help of transistor’s size optimization tool. *Bull. Pol. Acad. Sci.* **63**(4), 919–922 (2015)
43. C.-C. Tsai, C.-H. Lai, W.-T. Lee, J.-O. Wu, 10-bit switched-current digital-to-analogue converter. *IEE Proc. Circuits Devices Syst.* **152**(3), 287–290 (2005)
44. W.-H. Tseng, C.-W. Fan, W. Jieh-Tsornng, Senior Member, A 12-Bit 1.25-GS/s DAC in 90 nm CMOS with >70 dB SFDR up to 500 MHz. *IEEE J. Solid State Circuits* **46**(12), 2845–2856 (2011)
45. C.M. Twigg, P.E. Hasler, J.D. Gray, R. Chawla, Systems and Methods for Programming Large-Scale Field-Programmable Analog Arrays, United States Patent, U.S. Patent No 7,439,764 B2, 21 Oct 2008

46. E. Vittoz, Analogue VLSI signal processing: why, where and how. *Analog Integr. Circuits Signal Process.* **6**, 27–44 (2007)
47. E. Yilmaz, G. Dündar, Analog layout generator for CMOS circuits. *IEEE Trans. Comput. Aided Design Integr. Circuits Syst.* **28**(1), 32–45 (2009)