An SIMD Array of Analogue Microprocessors for Early Vision

Piotr Dudek and Peter J. Hicks, *Member, IEEE*

Abstract--An idea of using analogue microprocessors as processing elements in an SIMD array is presented. The resulting system retains some of the advantages of the analogue signal processing paradigm, while being a fully programmable general-purpose massively parallel processor array. The implementation of an analogue processing element as a switched-current circuit is discussed. Low-level image processing application examples are presented.

Index terms-- SIMD, massively parallel processors, image processing, analogue signal processing, switched-current circuits

I. INTRODUCTION

The inherent fine-grain parallelism of low-level image processing tasks makes SIMD (Single Instruction Multiple Data) arrays of processors very efficient in executing early vision algorithms [1]. A processing element (PE) is often built as a simple bit-serial processor, which promises the capability of integrating hundreds of PEs onto a single chip [2].

It is possible to find applications that would benefit from integrating an even larger number of PEs, although such high levels of integration can lead to an I/O bottleneck, which restricts the rate at which data can be transferred in and out of the array. One of the ways to overcome this problem is to integrate an image sensor and a processor onto a single chip [3]. A small processing element would allow smart sensors to be built with a processor at every pixel.

Sometimes the A/D conversion of an input image can also be avoided, when the preliminary processing is done in an analogue fashion. Indeed, for specific early vision tasks analogue solutions are often superior in terms of speed, area and power, although they lack the versatility of digital processors, offering at best template-based programmability [4,5] or more or less sophisticated forms of reconfigurability [6].

We propose an architecture, which takes advantage of the benefits of an analogue solution and enables the integration of thousands of PEs onto a single chip, while retaining the flexibility of software programmable digital processors.

II. ARRAY OF ANALOGUE MICROPROCESSORS

Consider the well-known mesh-connected array of PEs, as shown in Fig.1. Instructions are issued to the array from an external controller unit, common to all PEs, and the array executes the program in an SIMD fashion.
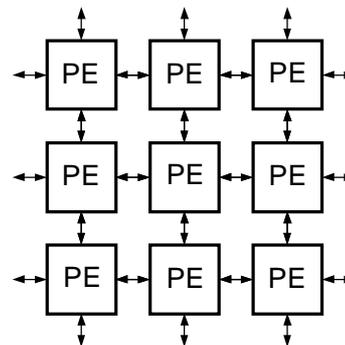

Fig.1. Mesh-connected array of processing elements

As a PE, we introduce the analogue microprocessor, a block diagram of which is presented in Fig. 2. The analogue microprocessor operates on analogue samples of data and consists of a set of analogue data storage elements or 'registers' and an analogue arithmetic unit which is analogous to the Arithmetic Logic Unit (ALU) found in digital microprocessors. The processors communicate with their nearest neighbours, and with the outside world, through analogue data lines. Since the entire processing is performed on analogue values we will refer to the processors as Analogue Processing Elements (APEs)
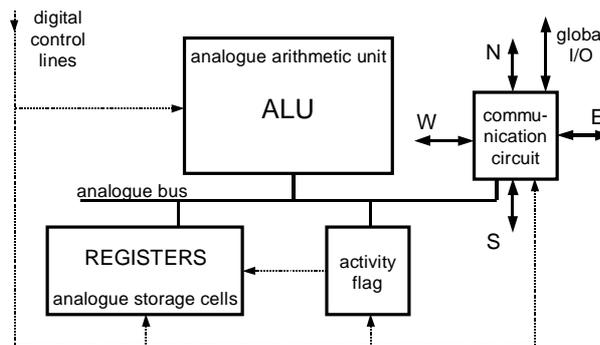

Fig.2. The block diagram of an APE.

The authors are with the Department of Electrical Engineering and Electronics, University of Manchester Institute of Science and Technology, P.O. Box 88, Manchester M60 1QD, United Kingdom

An APE executes a software program, performing consecutive instructions issued by a digital controller, in a way similar to the digital microprocessor. These instructions may include register transfer operations, which move the analogue data between registers of the APE and/or its neighbours, and arithmetic operations, which modify the analogue data. Also, some of the APEs may be disabled for a certain period, as a result of a conditional instruction.

As a result, we obtain a general-purpose analogue system, whose functionality is determined only by the application software. Undoubtedly, the success of such a system will depend on the efficient implementation of the APEs.

III. IMPLEMENTATION OF THE ANALOGUE MICROPROCESSOR

The dominant consideration, if we want to integrate a large number of processing elements onto a single chip, is the silicon area consumed by a single processor cell. A sampled analogue-data circuit can be implemented in CMOS using switched-capacitor (SC) or switched-current (SI) techniques. The latter one is particularly suitable for designing a massively parallel array of APEs, because it results in very compact implementation of both registers and ALU.
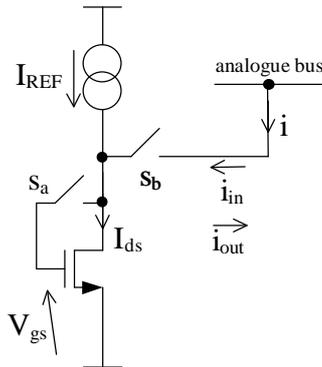

Fig.3. Basic SI memory cell.

A. Analogue register

As an analogue register we can use the SI memory cell [7] presented in Fig. 3. In a basic configuration both switch and current source can be implemented by single transistors, and therefore a complete data storage element consists of only three transistors.

When an MOS transistor works in saturation its drain current can be described by the equation:

$$I_{ds} = K(V_{gs} - V_t)^2 \qquad (1)$$

When writing to the register both of the switches s_a and s_b are closed, and current i_{in} from the analogue bus forces the gate-source voltage of the transistor to the value:

$$V_{gs} = \sqrt{\frac{I_{ds}}{K}} + V_t = \sqrt{\frac{I_{REF} + i_{in}}{K}} + V_t \qquad (2)$$

At the end of the write phase the switch s_a is opened and the gate of the memory transistor is put into a high impedance state. The register remembers therefore the value of the input current by storing charge on the gate capacitance of an MOS transistor.

When reading from the register s_b is closed and s_a remains opened. Now, the transistor acts as a current source, and the output current to the analogue bus is equal to:

$$i_{out} = I_{REF} - I_{ds} = I_{REF} - K(V_{gs} - V_t)^2 \qquad (3)$$

In this way the register works as a current-memory cell, since:

$$i_{out} = -i_{in} \qquad (4)$$

B. Analogue ALU

The analogue ALU is required to provide the basic arithmetic operations of addition, inversion, multiplication and division. However, as can be seen from (4), the inversion operation is inherent in the basic SI memory cell. Moreover, since data values are represented by currents, the addition operation can be performed directly on the analogue bus, by simultaneously connecting the outputs of two or more registers to the bus. From Kirchhoff's Law we get:

$$i_{bus} = \sum_n i_{out_n} \qquad (5)$$

The only operation which needs to be explicitly implemented is multiplication/division. The analysis of low-level image processing algorithms shows, that in a vast majority of cases only the multiplication by a constant factor is required. Therefore a digitally controlled multiplier/divider can be built as a set of scaled current mirrors, as depicted in Fig.4. During the write phase only the first transistor is connected (i.e. switches s_a and s_0 are closed), and the gate-source voltage of all transistors is set to:

$$V_{gs} = \sqrt{\frac{I_{ds}}{1 \times K}} + V_t = \sqrt{\frac{1 \times I_{REF} + i_{in}}{1 \times K}} + V_t \qquad (6)$$

During the read phase, any number of transistors may be connected to the output port of the multiplier. If $k_n = 1$ signifies switch s_n turned on and $k_n = 0$ signifies switch s_n turned off then:

$$i_{out} = \sum_n k_n \frac{1}{2^n} I_{REF} - \sum_n k_n \frac{1}{2^n} K(V_{gs} - V_t)^2 \qquad (7)$$

And therefore the multiplication factor is equal to:

$$N = \sum_n k_n \frac{1}{2^n} \qquad (8)$$

Additional features, like the ability to conditionally disable certain processors in the array during program execution, are also easy to implement. To disable a processor it is only necessary to disable the registers during

the write phase. It can be simply done by introducing another switch, serial to s_a, controlled by an activity flag register which is set during a conditional instruction.

The number of registers available in each processor is a trade-off between the ability to handle complex algorithms and the area of the processor. Our analysis shows that six to eight registers is a satisfactory compromise.
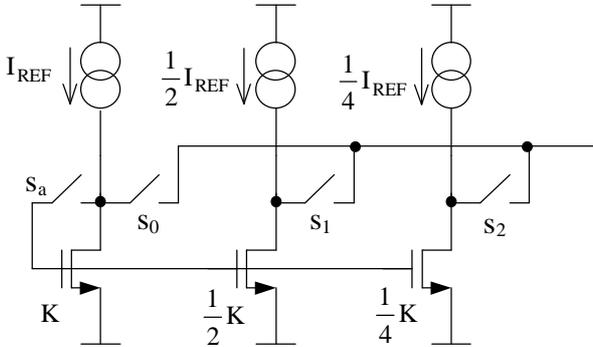

Fig.4. Digitally-controlled SI multiplier/divider

C. Accuracy issues

Due to the non-ideal properties of the elements the accuracy of SI circuits is limited by several factors [7]. These include charge injection in the analogue switches, the finite output conductance of the transistors and the capacitive coupling through the parasitic gate-source capacitance of the transistors. As a result of these effects, the value read from the register will differ from the value that has been written to it and the expression (5) has to be modified, to include an error Δi. In general, this error will depend on various factors, but in particular it will depend on the value of the signal:

$$i_{out} = -i_{in} + \Delta i(i_{in}) \qquad (9)$$

Many methods have been proposed to overcome these problems. They include introduction of regulated cascode output stages [8], utilisation of feedback loops [7], usage of dummy devices [9], and various differential and algorithmic error cancellation techniques [10-12]. However, more sophisticated methods of increasing the accuracy of SI circuits are more costly in terms of area, power and speed. These trade-offs have to be weighted up carefully when designing an analogue microprocessor.

It also has to be remembered, that the majority of low-level image processing tasks do not require very high accuracy. It has been shown [6], that in many cases a modest accuracy equivalent to 6 or 7 bits is sufficient. Therefore a simple yet area-efficient error-cancellation technique, such as S^2I [13], may be the best solution to the problem of designing a massively parallel array of APEs.

D. Physical implementation.

To evaluate the trade-offs between accuracy, area, speed and power we have designed an integrated circuit in

$0.8\,\mu m$ CMOS technology and fabricated it through EUROPRACTICE. The chip contains various analogue microprocessor designs, using different cancellation techniques, and proves that accuracy of the order of 6 bits can be achieved using APEs operating with clock frequencies of a few MHz, while being small enough to enable the integration of a 128×128 array of processors onto a single chip.

The small processor area is of particular importance. Firstly, if we consider processor/sensor integration, we must ensure that the array can be made large enough to provide reasonable resolution. Secondly, the large number of processing elements working in parallel will yield a high performance for the overall system, even if the speed of an individual processor is not very high. A 128×128 array of processors working with 2MHz clock frequency is capable of performing 32 GOPS (Giga Operations Per Second).

IV. IMAGE PROCESSING EXAMPLES.

To evaluate the effects of errors in APEs on image processing algorithms, and to illustrate the capability of our architecture to perform a variety of image processing tasks, we have developed a software simulation of a complete SIMD analogue microprocessor array. The simulator includes error modelling which is based on actual experimental results.

An important observation has to be made, which is that the algorithms may always be constructed in such a way that signal-independent errors cancel out. Let's for example consider moving certain analogue data i_X from register A to register B. Because the signals are inverted we use auxiliary register C and move the data in two steps. The first step reads data from A and writes it to C:

$$i_{outA} = -i_x + \Delta i(i_x) \qquad (10)$$

In second step, the value is read from C and written to B:

$$i_{outC} = -i_{outA} + \Delta i(i_{outA}) \qquad (11)$$

Therefore the value written to B is equal to:

$$i_{inB} = i_{outC} = i_x - \Delta i(i_x) + \Delta i(i_{outA}) \qquad (12)$$

The last two terms have opposite signs and therefore lead to the cancellation of the signal-independent component of the error. Similar equations can be written for addition and multiplication operations, and in each case an appropriate sequence of instructions leads to signal-independent error cancellation. Therefore, the accuracy of the system will be determined by the magnitude of the signal-dependent error associated with the APEs.

As an example let's consider the Sobel edge-detection algorithm [14] which involves convoluting the original image with the following kernels for horizontal (k_{hor}) and vertical (k_{ver}) edges detection:

$$k_{hor} = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}; \; k_{ver} = \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix} \qquad (13)$$

The program realising this on an SIMD array of APEs contains 49 instructions. The results of executing this program, on an array of processors having 2% accuracy are presented in Fig. 5. The image resolution is equal to 128x128 pixels. The execution time, on a 128×128 array clocked with 2MHz clock would be equal to $24.5\,\mu s$.

Fig.5. Sobel edge detection. Original image and edge map.

Another example is median filtering. The value of each pixel is replaced by the median value of 9 pixels from the neighbourhood centred on this pixel [14]. The algorithm involves a sorting operation, which due to the limited number of registers in a processor has to be performed in a somewhat unusual way. Nevertheless, a solution can be found, and a program realising median filtering, with 2% accuracy, after loop unrolling contains 4400 instructions. Even if APEs work with a low 200 kHz clock frequency this algorithm can be executed in real time, with 25 frames per second. The simulation result, showing the recovery of the original image from the image distorted by salt and pepper noise is shown in Fig.6.
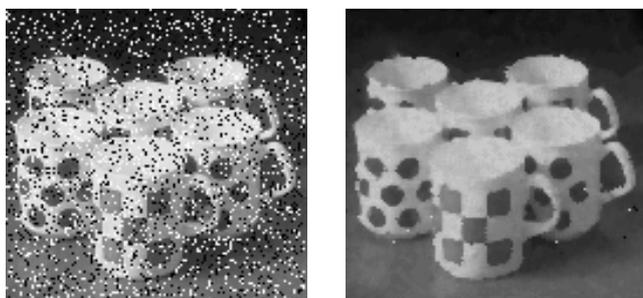

Fig.6. Median filtering. Image with 20% salt and pepper noise and filtering result.

V. CONCLUSION

An SIMD array of analogue microprocessors is a general-purpose system, which executes a software program while operating on analogue values of data. Switched-current techniques allow for a very compact implementation of APEs and therefore thousands of processors can be integrated onto a single chip. Experimental and simulation results show that in this way a low-cost, high performance, versatile image processing system may be built.

REFERENCES

[1]   M. Gokhale, B. Holmes and K. Iobst, "Processing in Memory: The Terasys Massively Parallel PIM Array", IEEE Computer, April 1995, pp.23-31.

[2]   J. C. Gealow, F. P. Herrmann, L. T. Hsu and C. G. Sodini, "System Design for Pixel-Parallel Image Processing", IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 4, no. 1, pp. 32-41, March 1996.

[3]   K. Chen, P. E. Danielsson, A. Aström, "PASIC: A Sensor/Processor Array for Computer Vision", Proceedings of IEEE Conference on Application Specific Array Processors (ASAP'90), California, September 1990, pp. 352-366.

[4]   T. Roska and L. O. Chua, "The CNN Universal Machine: An Analogic Array Computer", IEEE Transactions on Circuits and Systems-II:Analog and Digital Signal Processing, vol. 40, no. 3, pp. 163-173, March 1993

[5]   T. Spirig, P. Seitz, O. Vietze and F. Heitger, "A Smart CCD Image Sensor with Real-Time Programmable Parallel Convolution Capabilities", IEEE Transactions on Circuits and Systems-I: Fundamental Theory and Applications, vol. 44, no. 5, pp. 465-468, May 1997.

[6]   D. A. Martin, H. S. Lee, I. Masaki, "A Mixed-Signal Array Processor with Early Vision Applications", IEEE Journal of Solid-State Circuits, vol. 33, no. 3, pp. 497-502, March 1998.

[7]   C. Toumazou, J. B. Hughes and N. C. Battersby (Eds.), "Switched-Currents: An Analogue Technique for Digital Technology", Peter Peregrinus Ltd., London, 1993.

[8]   C. Toumazou, J. B. Hughes and D. M. Pattullo, "Regulated Cascode Switched-Current Memory Cell", Electronics Letters, vol. 26, no. 5, pp. 303-305, March 1990.

[9]   C. Eichenberger and W. Guggenbühl, "Dummy Transistor Compensation of Analog MOS Switches", IEEE Journal of Solid-State Circuits, vol. 24, no. 4, pp. 1143-1146, August 1989.

[10]   C. Toumazou and S. Xiao, "N-step Charge Injection Cancellation Scheme for Very Accurate Switched Current Circuits", Electronics Letters, vol. 30, no. 9, pp. 680-681, April 1994.

[11]   J. B. Hughes, K. W. Moulding, J. Richardson, J. Bennett, W. Redman-White, M. Bracey, R. S. Soin, "Automated Design of Switched-Current Filters", IEEE Journal of Solid-State Circuits, vol. 31, no. 7, pp. 898-907, July 1996.

[12]   M. Helfenstein, G. S. Moschytz, "Improved Two-Step Clock-Feedthrough Compensation Technique for Switched-Current Circuits", IEEE Transactions on Circuits and Systems-II:Analog and Digital Signal Processing, vol. 45, no. 6, pp. 739-743, June 1998

[13]   J. B. Hughes and K. W. Moulding, "S2I: A Switched-Current Technique for High Performance", in Electronics Letters, vol.29, no.16, pp. 1400-1401, Aug. 1993

[14]   E. R. Davies, "Machine Vision: Theory, Algorithms, Practicalities", Academic Press Limited, London, 1990