Low power image processing: analog versus digital comparison

Jacques-Olivier Klein, Lionel Lacassagne, Herve Mathias, Sebastien Moutault, Antoine Dupret

Low power Image Processing: Analog versus Digital comparison

Jacques-Olivier Klein, Lionel Lacassagne, Hervé Mathias, Sébastien Moutault and Antoine Dupret
Institut d'Electronique Fondamentale
Bât. 220,
Univ. Paris-Sud, France
Email: jok@ief.u-psud.fr

***Abstract*— In this paper, a programmable analog retina is presented and compared with state of the art MPU for embedded imaging applications. The comparison is based on the energy requirement to implement the same image processing task. Results showed that analog processing requires lower power consumption than digital processing. In addition, the execution time is shorter since the size of the retina is reasonably large.**

I. Introduction

Smart sensors, vision chips [3]–[6] have potential to take an increasing part in navigation or surveillance systems: toys or industrial robots, car driving assistance... For this class of applications, one has to provide vision systems which feature high processing capabilities, low cost, compactness and reduced power consumption. In a previous paper [10] we introduced the architecture of the X-Cell, a universal analog computation cell. Compared to its digital counterpart, lower power consomption and reduced silicium area are expected. Such statement has to be proven with fairly quantitative study. Consequently, we propose a comparison between a vector of X-Cell dedicated to image processing called PARIS and a similar digital architecture comprising SIMD units: PowerPC G4 Altivec. This comparison is performed using a well-known algorithm, representative of image processing task: edge detector. We present a detailed implementation on both architectures and focus on the hot spots for an optimized implementation. Two benchmarks are provided, the first one is about the execution time only to estimated the efficiency of general purpose processor as a challenger to dedicated architectures, the second deals with the most embedded constraining criterion: power consumption.

II. PARIS ARCHITECTURE

In most vision chips, photodetectors form an array.

With our programmable approach, photodetectors are associated to memory elements, them also organized in array. These arrays are bordered on one of their side by a column of analog/digital processors (see Fig. 1). Operations are performed sequentially on columns while snapshot mode image acquisition is concurrently achieved. A decoder selects then the column reached by processors. Furthermore, each processor access to a set of rows by the way of a mux (MUX3). Finally, fully random addressing can be convenient for reading and writing images.

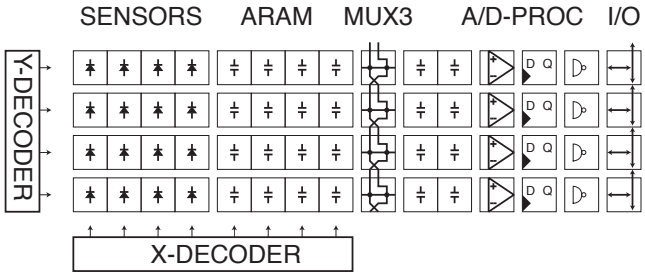

Fig. 1. Array and decoder architecture.

A. Architecture of rows

Each row of the retina is organized around two mixed analog-digital buses used to connect various functional units (see Figure 2). The functional units that can compose the row of a vision chip are: the rows of photosensors, the row of analog memory map, the set of analog registers, the Analog Processing Unit (X-Cell), the Boolean Processing Unit and few special registers. These last are notably required for I/O and global operators. In each processor, linear processings are handle by the Analog Processing Unit. Boolean units associated to the condition register allows to achieve different operations according to locally stored values. Binary data stemming from a comparator are combined by the Boolean Processing Unit and can be written in a condition register. Mixed registers will then be modified wherever this condition is true. Such architecture paves the way to numerous linear, isotropic or not algorithms [8].

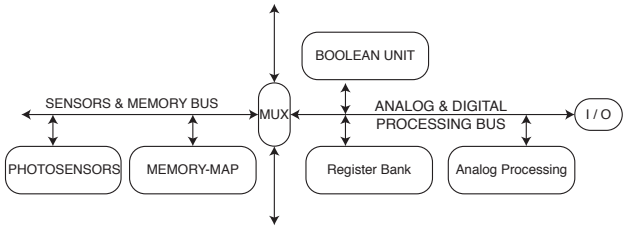

Fig. 2. Architecture of rows

B. Generic functional units

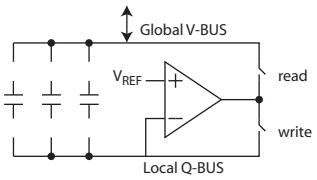

Fig. 3. Generic Functional Unit

Derived from [10], each functional unit is organized around one OTA, a set of capacitors associated to switches and of two buses: a global one, and a local one (see Figure 3). The global bus, which is dedicated to inputs/outputs, is named V-BUS. It is intended to distribute a value represented by a voltage, therefore allowing to realize non-destructive copies. The voltage is forced by the output of one OTA or by the output of a digital cell. A voltage mode operating drastically reduce its sensitivity to parasitic capacitors. The local Q-BUS, is intended to realize charge transfers and balancing. The charge transfer is used to perform accumulations while division is based on charge balancing. The voltage of the $Q-BUS$ is set to V_{REF} by the output of one OTA thanks to a feedback loop. So, its parasitic capacitor keeps its charge and thus has little impact during the transfer of charges [10].

C. Operating with switched capacitors

All the functional units are based on switched capacitors structures. Four different operations are used. They are illustrated by an example on the scheme given figure 4. At the

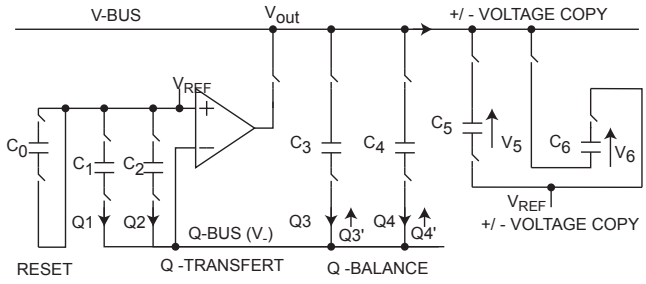

Fig. 4. Operation of the switched capacitors

instant all the switches close, the charge of all the capacitors are modified:

1) The capacitor C_0, is shorten, thus reset.
2) The capacitors C_1 and C_2 are also emptied of their charges, Q_1 and Q_2, which flow by way of the Q-BUS to capactors C_3 and C_4. It is a cumulative transfer of charges.
3) The total charge $Q_1+Q_2+Q_3+Q_4$ divides up between two parallel capacitors, C_3 and C_4, in proportion to their respective capacitance. It achieves charge balancing.
4) The resulting voltage on capacitors C_3 and C_4 is copied onto capacitors C_5 and C_6 by way of the V-BUS. It performs a copy in voltage mode. The configuration of switches allows to do or not a change of sign by reversal of the target capacitor during the copy.

D. Analog processing unit

The analog processor is constituted by a set of capacitors associated to switches allowing various configurations. It includes a set T of processing capacitors associated to registers-capacitors (cf. Fig. 5). To improve accuracy, each capacitor is an instance of a unitary capacitor C_u. Let define the *weight* of a set S of capacitors, the dimensionless quantity: $\frac{1}{C_u} \times \sum_{i \in S} C_i$, where C_i is the capacitance of the ith capacitor of S. Typicaly, four capacitors of weights 1, 1, 2 and 4 compose the T set, allowing 3-bit processing.

More general operation of the analog processor, multiplication-accumulation can be decomposed into three steps: *Load*, *Distribute*, *Accumulate*. For each of these 3 steps, a set of the implied capacitor (respectively L, B, A) is considered.

- During the first step (Load), the set $L \subset T$ (of weight l) is charged by a positive or negative voltage-copied. The input voltage V_in is copied (positively or negatively) in a subset $L \subset T$ of capacitors (of weight l). After loading, the charge Q_L, stored in set L, is $Q_L = \pm l \times V_in \times C_u$.
- During the second step (Balancing), the charge Q_L is distributed on the set $B \supset L$ (of weight b), so that each capacitor belonging to B has a voltage $V_B = \frac{1}{b} \times \pm l \times V_in$.
- Finally, the last step (Accumulation) consists in adding charges stored in a set of capacitors $A \subset B$, of weight a, on a register-capacitor C_R of capacitance C_u. So: $V_{C_R}(t+1) = V_{C_R}(t) \pm \frac{a \times l}{b} \times V_in$.

a) : Hence, the realized operation is a fractional multiplication/accumulation (FMAC) with a coefficient $\frac{a}{b} \times l$. Obviously, if $B = L$, step 2 can be omitted and the realized operation is a integer multiplication/accumulation (IMAC) with a coefficient a. As a consequence, the MAC instruction duration is 2 (IMAC) or 3 cycles (FMAC). Table I describes a subset of the X-Cell instructions. MR represents any mixt register and BR the boolean register. LC stands for *Local Condition*.

Instruction	Description	Cyc.
`IMAC MR,`V_in ± `a`	$MR \leftarrow MR \pm a \times V_in$	2
`FMAC MR,`V_in ± `al/b`	$MR \leftarrow MR \pm \frac{a \times l}{b} \times V_in$	3
`ASTR MR,`V_out	$V_out \leftarrow MR$	1
`CLR MR`	$MR \leftarrow 0$	1
`CMP MR`	$V_out \leftarrow (MR > 0)$	1
`WHERE`	$LC \leftarrow BR$	1
`WHERN`	$LC \leftarrow notBR$	1
`ENDWH`	$LC \leftarrow TRUE$	1
`<OP> MR`	$BR \leftarrow BR$ op MR	1
`BSTR MR`	$MR \leftarrow BR$	1

TABLE I
X-CELL INSTRUCTION SUBSET

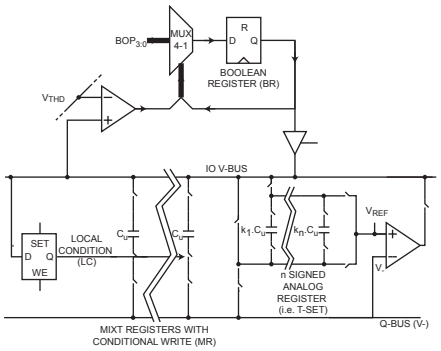

Fig. 5. Boolean and Analog Unit.

III. PHYSICAL IMPLEMENTATION

Two retinas prototypes were designed. Although the first, *PARIS I*, is based on a slightly different structure from the universal structure described here, its functioning is somewhat identical. It consists in a 16×16 pixel array - each including a photosensors and 3 analog memory elements - associated to a minimal analog processor including only four capacitors: three for processing and one for register [8]. Its main characteristics are presented in the table II.

Parameter	PARIS I	PARIS II
Resolution	16×16	256×256
Processor	16	256
Pixel Size	$50 \times 50 \mu m^2$	$25 \times 25 \mu m^2$
Max Frequency	$10MHz$	$40MHz$
Power cons.	$30mW$	$800mW$
$MixtRegisters$	2	3
$Resolution$	7-bits	10-bits
$Processing$ $Capacitors$	2	4
$Boolean$ $Processor$	No	Yes
I/O	1 analog	1 analog and 8 digital
$Reduction$ $Operator$	No	1 global-OR and 1 Mean Op

TABLE II
PARIS I AND PARIS II PARAMETERS

This circuit has been successfully tested and operates properly [13]. It is currently being evaluated for applications in mobile robotics.The second circuit, *PARIS II*, was designed according to the principle described in this paper. It brings improvements with regard to *PARIS I*, notably on readout circuits of analog memory and photosensors [12]. Its main characteristics are presented in the table II.

IV. DERICHE BENCHMARK

In order to estimate the performance of the X-Cell architecture, we have decided to compare it to another SIMD vector architecture and to implement a de facto image processing algorithm like edge detection. The closest "software" architecture are the General Purpose Processor with multimedia SIMD extension (also called SWAR for SIMD Within A Register). The most embedded GPP are the PowerPC Altivec and Intel Centrino. PowerPC has a more extensive SIMD ISA for image processing (crossbar capabilities, reductions and 8-bit multiplier) Centrino implements SSE2 but with only 16 multipliers, Pentium4 Prescott extends SSE2 instructions with reduction capabilities with SSE3, but can not be considered as an "embedded" processor. Note that an SoC version of the PowerPC G4 has been released by Motorola/Freescale Other embedded processors might be chosen for their SIMD architecture: the ARM11 (SIMD in 32-bit registers: four 8-bit computations in parallel) or the latest Intel Xcale/PCA which includes a multimedia extension called Wireless MMX (64-bit registers for 8/16/32-bit integer and 32-bit FP).

Classical edges detector operators implemented in artificial retinas FIR filters like Sobel, Prewitt or Roberts filters. Canny-Deriche filters have assert themselves for their robustness. These filters can be expressed as a non recursive filter like Canny's filter or a recursive filter like Deriche's one. Each have drawback and advantage : Deriche have a fixed complexity that does no depend on the smoother coefficient, but requires large memory to hold a complete image, Canny is more adapted to "data-flow" because the image must not be store in memory, only the current row, but the filter size depends on the smoother coefficient.

X-Cell is well-adapted to Deriche filter: it has three memory plans to store 3 images, and the performances of the processor vector array are not limited by Deriche's filter structure, if the vector displacement is orthogonal to the filter. The Deriche's filter complexity has been reduced by a factor two by Garcia Lorca [16]. That is this filter that will be implemented.

The second order filter is:

$y(n) = b_0 x(n) + a_1 y(n-1) + a_2 y(n-2)$

with:

$\gamma = e^{-\alpha} \quad b_0 = (1-\gamma)^2 \quad a_1 = 2\gamma \quad a_2 = -\gamma^2$

A. 2D filter implementation

The Q8 fixed-radix code Deriche H & V smoothers are:

```
for(i=0; i<n; i++)
  for(j=0; j<n; j++)
    x0 = X[i][j]
    y1 = Y[i][j-1]
    y2 = Y[i][j-2]
    y0 = (b0.x0+a1.y1+a2.y2) >> 8
    Y[i][j] = y0
```

Deriche H

```
for(j=0; j<n; j++)
  for(i=0; i<n; i++)
    x0 = X[i][j]
    y1 = Y[i-1][j]
    y2 = Y[i-2][j]
    y0 = (b0.x0+a1.y1+a2.y2) >> 8
    Y[i][j] = y0
```

Deriche V

b0=$256 \times b_0$ a1= $256 \times a_1$ a2= $256 \times a_2$

B. PowerPC Altivec implementation

The three main problems to address for SIMD implementation are:

- cache impact
- recursive filter structure
- underflow

The horizontal filter does not generate cache miss whereas the vertical filter does. The solution is to permute the internal loop with the external loop of the filter to obtain an horizontal-like scan with a vertical filter. Such a permutation correspond to a cache blocking optimization [15].

```
for(i=0; i<n; i++)
  for(j=0; j<n; j++)
    x0 = X[i][j]
    y1 = Y[i-1][j]
    y2 = Y[i-2][j]
    y0 = (b0.x0+a1.y1+a2.y2) >> 8
    Y[i][j] = y0
```

Deriche VH

Between two iterations of the filter there is a loop-carried dependency. The solution proposed (figure 6) is to perform a block-transposition of pixel into a band, to process the band and then to perform a second block-transposition into the source image.

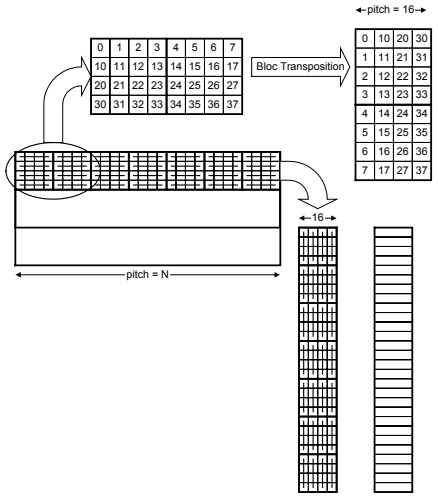

Fig. 6. Deriche Band transposition

The last problem is about underflow: since the coef a_2 is negative, for a long set of zero input values, one can have $x_0 = 0$, $y_1 = 0$ but $y_2 \neq 0$, so an underflow can happen.

C. X-Cell implementation

The following pseudo-code sources describe the primitives used for edge detection. The program iterates on all this routines for each column.

```
CLR MR0
FMAC MR0, I1(i, 0) 0.375
FMAC MR0, I2(i-1, 0) 0.875
FMAC MR0, I2(i-2, 0) 0.25
ASTR MR0, I2(i, 0)
```

PARIS horizontal FGL smoother

```
CLR MR0
IMAC MR0, I2(i, 0) -1
IMAC MR0, I2(i, 1) -1
IMAC MR0, I2(i+1, 0) 1
IMAC MR0, I2(i+1, 1) 1
CMP MR0
```

PARIS horizontal gradient

```
CLR MR1
IMACC MR1, MR0 -1
WHRN
ASTR MR1,MR0
ENDWH
```

Horizontal absolute value

```
MACC AR1 , AR0 1
ASTR AR1, I2(i, 0)
```

Addition of the two previous results

Applying the four FGL filters (forward/backward, horizontaly/verticaly) requires $4 \times 11 = 44$ cycle/column. In addition, $2 \times 10 = 20$ cycles/col and $2 \times 6 = 12$ cycle/col are required respectively to compute the gradient and absolute value, horizontaly and verticaly. Finaly, 3 cycles/ col perfom the addition of horizontal and vertical gradient. All things considered, the execution time of this algorihm is 79 cycles / col i.e. 0.51 ms with a 256×256 array running at 40 MHz.

V. RESULTS & ANALYSIS

To observe the impact of cache behavior we use the *cpp* (Cycle Per Pixel):

$$cpp = \frac{t \times F}{n^2}$$

n	128	256	512	1024
cpp Deriche H	2.95	2.85	3.31	3.87
cpp Deriche VH	4.86	4.88	5.24	6.19
cpp gradient	2.69	2.88	3.17	3.65
cpp total	10.5	10.61	11.72	13.71

TABLE III

cpp FOR 128, 256, 512 AND 1024 IMAGE SIZE FOR POWERPC

n	128	256	512	1024
t(ms) Deriche H	0.048	0.187	0.868	4.058
t(ms) Deriche VH	0.080	0.320	1.374	6.491
t(ms) gradient	0.044	0.189	0.831	3.827
t(ms) total	0.172	0.696	3.073	14.376

TABLE IV

EXECUTION TIME (MS) FOR 128, 256, 512 AND 1024 IMAGE SIZE FOR POWERPC

The execution time on the Xcell does not suffer from cache misses: *cpp* is still constant: 11 cycles per Deriche filter, for a total of 44 for the four filters and 39 cycles for the gradient.

If we only compare the execution time, PowerPC and Xcell run at same speed (the G4 is even faster), for small images (128 and 256), when they fit the cache. Such a comparison is biased since it does not take into account the required energy for these architectures.

n	128	256	512	1024
time(ms) total	0.253	0.506	1.011	2.022

TABLE V
EXECUTION TIME (MS) FOR 128, 256, 512 AND 1024 IMAGE SIZE FOR XCELL

The classical metric used to compare embedded processor is *Mips/Watt*. We do no believe that Mips or Mops is still an up-to-date metric since the latency of instructions may vary a lot, and so, counting the number of instructions could lead to erronous conclusion except if you want your system to run the Dhrystone benchmark. Not very useful indeed. We prefer the $t \times Watt$ (in Joule) which is the amount of energy required to apply an algorithm on an image. The idea is that if a processor is by far real-time for an application, it's SoC version will use a *downclocked* version of the classic processor version, the energy remains constant but the power is smaller. For 256×256 images the classic G4 is 78 times faster that the realtime constraint (40 ms). Dividing its clock frequency by 10 will also reduce its power consumption by approximately 10, for a still realtime 5.1 ms execution.

$$E = t \times Watt$$

The technology used for the current XCell processor is a 0.60 μm CMOS. Switching from 0.60 to 0.25 μm will decrease the capacitor surface, that is the leaking capacitor, the required curent and finally the consumption. A scale factor can be applied to estimate not a faster XCell but *smaller* XCell. The factor is $(0.60/0.25)^{1.5}$. The exponent is 1.5 and not 2 since it appears in the Literature that such a switch provides a factor that is smaller than the gain in surface, and closer to 1.5 than 2. For XCell we estimated the consumption of the micro-controler to 200 mW and 800mW for a 256 XCell vector. With such an assumption, the result for the new criterion is:

n	128	256	512	1024
PowerPC (mJ)	1.72	6.96	30.73	143.76
XCell (mJ)	0.15	0.51	1.82	6.88
scaled XCell (mJ)	0.07	0.23	0.81	3.06
gain	11.3	13.8	16.9	20.7
scaled gain	25.5	30.9	37.9	46.9

TABLE VI
COMPARISON OF REQUIRED ENERGY FOR POWERPC AND XCELL

With such a criterion, the difference of performances for *extreme* embedded applications is more realistic from our point of view.

VI. CONCLUSION

A programmable analog retina has been presented and compared with state of the art MPU for embedded imaging applications. The comparison is based on the energy requirement to implement the same image processing task. Each version has been independently optimized to fit the considered architecture. To complete the performance evaluation, an evaluation of 1GHz DSP C64x is planned. Right now, the valididty of such a analog design has been demonstrated. Even when obsolete process are used for the retina, results showed that analog processing requires lower power consumption than digital processing. In addition, the execution time is shorter since the size of the retina is reasonably large.

REFERENCES

[1] Moutault S. Klein J.-O., Dupret A. "A universal switched capacitors operator for the automatic synthesis of analog computation circuits.", CAMP'03 - IEEE Sixth International Workshop on Computer Architecture for Machine Perception, New Orleans (USA), May 12-14, 2003, ISBN 0-7803-7971-3.

[2] B. Granado, L. Lacassagne, P. Garda, "Cab general purpose microprocessors simulate neural network in real time", IWAN (2), 1999, pp 21-29.

[3] C. A. Mead, "Analog VLSI and neural systems", Addison Wesley, 1989.

[4] A. Moini, "Vision Chips", Kluwer Academic, 1999.

[5] K. Kyuma, E. Lange, J. Ohta, A. Hermanns, B. Banish, and M. Oita, "Artificial Retinas-Fast, Versatile Image Processors", Nature, vol. 372, 1994.

[6] T. Bernard, B. Zavidovique, and F. Devos, "A Programmable Artificial Retina", IEEE Journal of Solid State Circuits, vol. 28, pp. 789-797, 1993.

[7] R. Etienne-Cummings, Z. Kalayijan and D. Cai, "A programmable focal-plane MIMD image processor chip", IEEE J. Solid State Circuit, vol. 36, N1, pp 64-73, January 2001.

[8] A. Dupret, J.-O. Klein , A. Nshare ,"A programmable vision chip for CNN based algorithms", in Proc of 6th IEEE Int Workshop on Cellular Neural Networks and their Applications, pp.207-212 , 23-25 may 2000, Catania, Italy

[9] A. Abo, "Design for reliability of low-voltage switched-capcitor circuits", PhD Thesis, University of California, Berkley, May 1999.

[10] S. Moutault, J.O. Klein, A. Dupret, "A universal switched capacitors operator for the automatic synthesis of analog computation circuits.", IEEE Sixth International Workshop on Computer Architecture for Machine Perception, May 11-14, 2003, New Orleans, LA.

[11] R. Carmona, S. Espejo, R. Dominguez-Castro, A. Rodriguez-Vazquez, T. Roska ,T. Kozek, L.O. Chua, "A 0.5 μm CMOS CNN Analog random access memory chip for massive image processing", Proc of 5th IEEE Int Workshop on Cellular Neural Networks and their applications, pp 243-248, London, UK, April 1998.

[12] Nshare A., Klein J.O., Dupret A. "An improved ARAM for PARIS, an original vision chip", CNNA2002, Frankreset/Main, 2224 July 2002, World Publishing 2002, pp 347-354.

[13] A. Dupret,J.O. Klein,A. Nshare, "A DSP-like analog processing unit for smart image sensors" International Journal of Circuit Theory and Applications, 2002, vol. 30, pp.595-609.

[14] R. Deriche. "Using Canny's criteria to derive a recursively implemented optimal edge detector". The International Journal of Computer Vision, 1(2):167-187, May 1987.

[15] D. Demigny *et al* "Méthodes et architectures pour le traitement du signal en temps rel. Trait'e Information, Commande et Communication. Hermès. 2001.

[16] F. Garcia Lorca "Filtres récursifs temps réel pour la détection de contours : optimisations algorithmiques et architecturales" PhD Thesis 1996.

[17] L. Lacassagne, F. Lohier, P. Garda, "Realtime execution of optimal edge detectors on RISC and DSP processors", ICASSP 1998.