Configurable 3D integrated focal-plane cellular sensor-processor array architecture

Péter Földesy, Ákos Zarándy, and Csaba Rekeczky

Abstract

A mixed-signal Cellular Visual Microprocessor architecture with digital processors is described. An ASIC implementation is also demonstrated. The architecture is composed of a regular sensor readout circuit array, prepared for 3D face-to-face type integration, and one or several cascaded array of mainly identical (SIMD) processing elements. The individual array elements derived from the same general HDL description and could be of different in size, aspect ratio, and computing resources.

Keywords:
Focal plane, sensor-processor, parallel processing, SIMD, 3D integration

I. Introduction

In this paper, we introduce in details our configurable, scalable, and clusterable cellular focal-plane sensor processor architecture and an operational ASIC validating the approach.

The architecture is derived from the concept introduced in [4]. Its functionality, near-sensor processing capability, and topographic architecture are inspired by the Cellular Neural Network Universal Machines [1]-[3].

In a typical standalone vision system, we can find sensor, AD converters, processor and memory, and the embedding and communication circuits [5]-[6]. It is obvious, that it is difficult to implement efficiently all these components in a single chip. On the other hand, the tight integration of such a system would be favourable due to compactness, performance, price, power consumption, etc. Our approach is to integrate as much functionality as possible from sensing through data conversion, reaching low and high level image processing and possible decision making into this architecture.

The 3D or vertical integration technologies for sensor integration are extensively researched nowadays [21]-[24], and become commercially available [26]-[27]. Following this trend, we inserted a separate sensor layer above the processor layer. In this way, the electrical interface and the processing kernel can be implemented with the help of mainstream semiconductor methodology, while the sensory technology and materials become arbitrary. The interconnection between the sensors and the processors are produced by 3D bump bonding technology.

We describe the concept of the system architecture in Chapter II, in Chapter III the processor array design is summarized; in Chapter IV general considerations are given about the sensory interfacing. The ASIC implementation called, Xenon V3, is described in Chapter V, and finally the conclusions are given.

II. System Architecture

A. Motivations

In the design of focal-plane sensor processor arrays, there are several tight trade-offs [4], [8]-[9], [32]-[33]. One of them is the antagonism of the required sensor resolution and the attached processing power. Another compromise is the sensor size versus the processor size. One must also take care of the application point of view of such a systems: a typical vision task could be mapped to a generic operation sequence, which contains sensing, grayscale filtering, segmentation, binary morphological feature extractor, classification, object tracking, decision making.

With the rise of 3D integration technologies, the possibility is given to separate the sensing technology from the rest of the system. This opportunity leads to another question, namely which type

Cellular Sensory and Wave Computing Research Laboratory, in the Computer and Automation Research Institute of the Hungarian Academy of Sciences, also with Eutecus Inc Berkeley, California. E-mail: [foldesy,zarandy,rcsaba]@sztaki.hu; web: www.sztaki.hu, www.eutecus.com

of sensor would be connected to the system. Our approach is to create a generalized host interface for many different sensor arrays. As the application demands, the signal conditioning could be altered without affecting the processing system and vice-versa. In order to achieve it, the analog signal conditioning and the AD conversion is tightly coupled to the physical sensor grid. The digital processor array takes place next to the mixed-signal block (possibly in different ICs may be quilt packaged). The connectivity in between is maintained through a well-defined protocol (the reader is referred in the topic of near pixel AD conversion to [35]-[38]).

Regarding the processor architecture, we have developed a scalable array, where the complexity – number and type of the embedded resources – of a single processor can vary in a wide assortment. The programming environment (such as the assembly compiler) is also prepared to accommodate the reduced or extended instruction set and size.

B. Configurability and flexibility

The architecture is highly scalable, and this property allows us to build customized processing chains from dedicated processor arrays. The parametric space enables to change not only the number of processors and the aspect ratio of a processor array (e.g. in multi-cluster case, the series of wide processor vectors), but to customize the local memory size of the processors, the ALU (arithmetic and logic unit) resources, the number of sensors and the involved ADCs attached to a processor.

An important observation is that a single function or operator should be executable independently from the position of the particular pixel within an image. Similarly, the functionality should not depend on the number or data sharing method of the processors. These facts motivate that the processor arrays operates in single instruction multiple data (SIMD) type concurrency. Naturally, the function changes as the data flows through the chain. Consequently, each processor array has its own repetitive task, but these tasks differ from array to array. With this generalization, the architecture becomes so-called multi-SIMD type. The practical form of this architecture is to insert instruction decoders per processor array, where each of the array elements within an array shares the decoded microinstructions.

At the processor level, there are typical building blocks such as registers, crossbars, memory banks, and arbiters. On the top of this general skeleton, we have included arithmetical and mathematical morphology oriented modules as well. These modules are optionally included and parameterized during array formation. Furthermore, considering the large area requirement of on-chip mass storage, the processors' memory banks could be also excluded.

Figure 1 shows a typical mapping of a general vision task to the described architecture. Note that the individual processor arrays are equipped with different operation capabilities. The reader is referred to the following chapters about the implementation details that are also shown here.

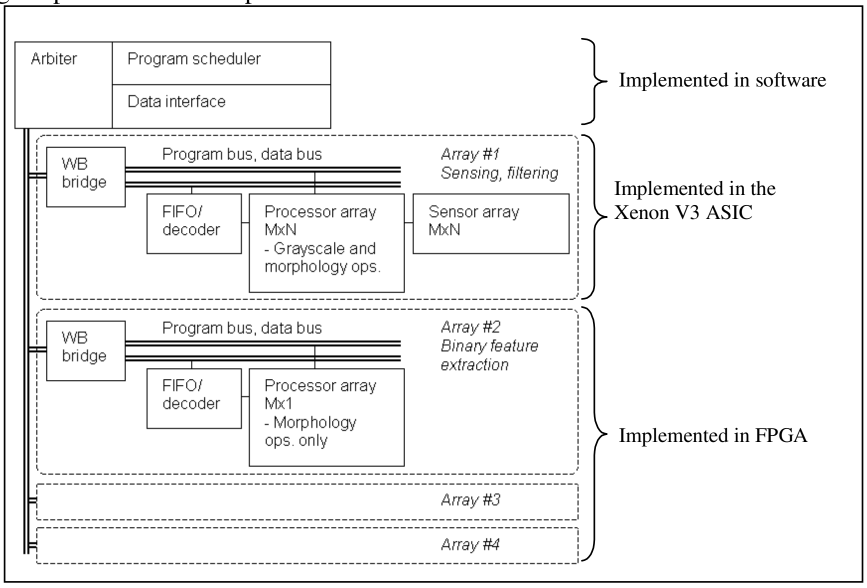


Figure 1. The figure shows an example of a cluster of task specialized processor arrays. Such a cluster can be easily derived from the described architecture on different platforms.

C. Functional organization

The high level organization of the integrated sensor interface processor array [4], [34] is shown in Figure 2. As it can be seen in the figure, each array is embedded into a relatively simple shell. This shell contains the programming and data communication modules that are described in details in the next Chapter as well.

The close sensor-processor organization must be emphasized, since as we mentioned the physical topology is different due to implementation considerations. The interactivity between the sensor and processor modules can provide the proactive or adaptive sensing [11]-[15] feature that is not available in the typical separated sensor processor architectures.

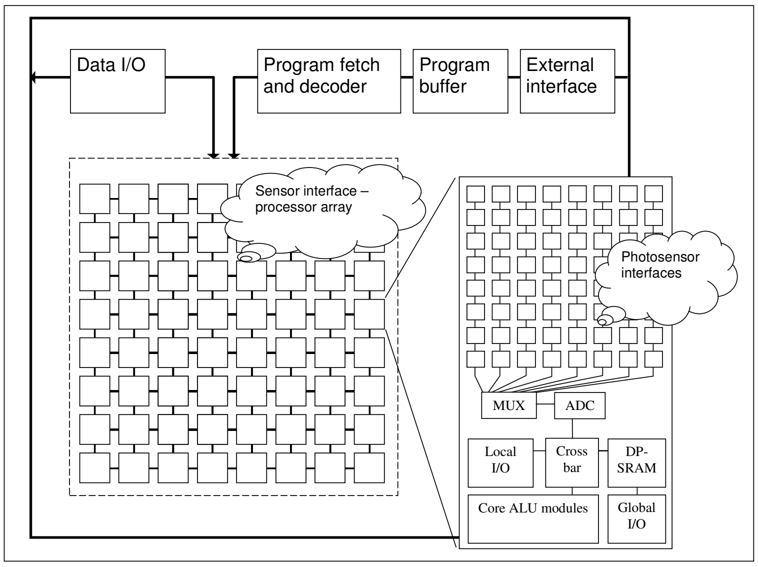


Figure 2. Architecture of the generalized processor array.

III. Processor Array Design

The complete system is formulated in Verilog RTL (register transfer level) code. With careful programming the code is synthesizable not only for ASIC target, but for FPGAs as well (or in case of clusters, for a mix of targets).

The source code is prepared for advanced testability, synthesis, and implementation design flows. The involved features are the coding style to minimize the power consumption (e.g. extensive clock gating); CRC checking that is employed at critical buses; and well-defined portion of the design is accessible through scan chain.

For the shake of generality and keeping in mind that different implementation platform offers different core speed and I/O bandwidth, the processor array is divided into four main clock domains and several minor test clock domains. The main clock domains are the data transfer, program transfer, processor core, and additionally the sensor interface's ADC control and data buffering. The separation of operational speed enables to run each block at the highest possible clock rate without being restrained by others. It is typical in a system that the clock of the data bus is lower than the main processor clock.

In addition, each domain is designed to be able to operate in parallel. That is, the image acquisition, conversion, processing, and data transfer can be done at the same time, resulting in a pipelined high throughput procedure.

The case of the program transfer is special. The instruction flow for the individual arrays comes from the same central program scheduler. In order to mitigate this possible transfer bottleneck, two mechanisms have been built in. First, the instruction streams are compacted, and secondly the communication shell around the processor arrays employs a buffer mechanism. This buffer allowing the instruction decoder to act independently without being affected by the fluctuating transfer speed. At the edges of the different clock domains dual-port memories, FIFO and dual sampled registers maintains the signal integrity.

At the points, where the buses forks, joins, or changes the data representation or bus speed, bus bridges are inserted. Both external and most of the internal connectivity is maintained by a standard bus protocol, called Wishbone [39]. This protocol is a public domain System-on-Chip (SoC) interconnect architecture for portable IP cores and interfaces. It supports, among others, point-to-point interfaces, shared bus architectures, crossbar switches, and off-chip routing. These properties are widely used in the design. In the external communication, additional handshaking signals are integrated too. The bidirectional signals force the system to be synchronized to external devices.

The architecture of a processor array can be seen in Figure 3.

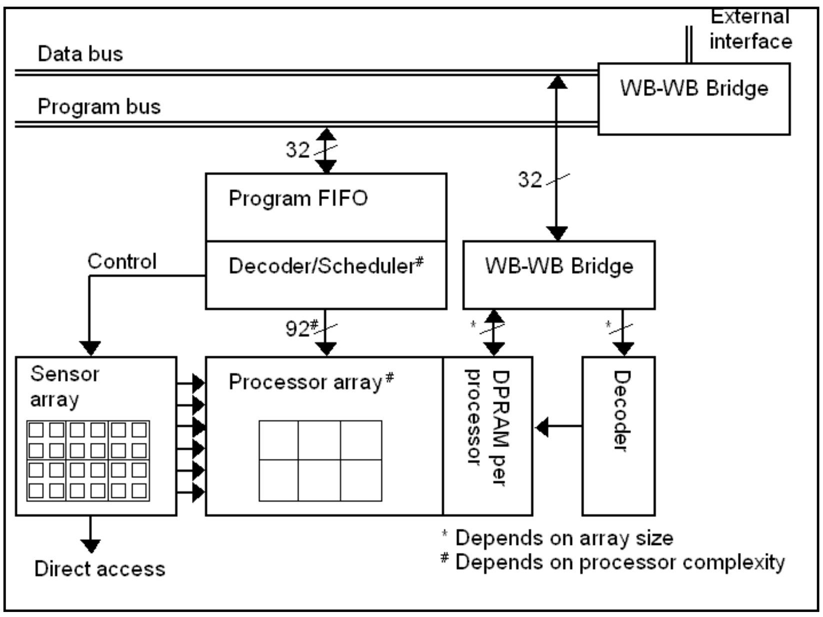


Figure 3. Organization of the sensor interface, the processor array, and the external interfacing showing typical bus width as well.

A. *Individual processors*

The basic constructing element of our digital sensor-processor architecture is the parametric processors. The processors are a composition of a core and connected I/O and data memory units. The processor does not have local program memory accordingly to the SIMD operation mode.

1) *Processor core*

The processor core is arranged around a crossbar switch. The design of the crossbar switch makes easier the inclusion or exclusion of the different operation modules of the core.

The maximum configuration processor core contains an arithmetic unit, a morphologic unit, register bank, standby logic, and flags (Figure 4). Since basic image processing operators are defined over the single byte (8 bit) precision, we accommodated this data representation throughout the processor. However, the arithmetical unit is an exception, as it can handle 16 or 24 bit data as well.

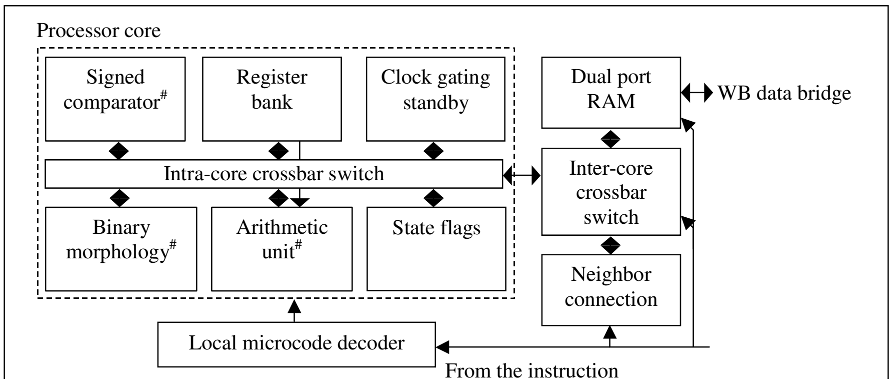


Figure 4. The architecture of a single processor. The modules marked by hashmark are optionally included during the processor instantiation.

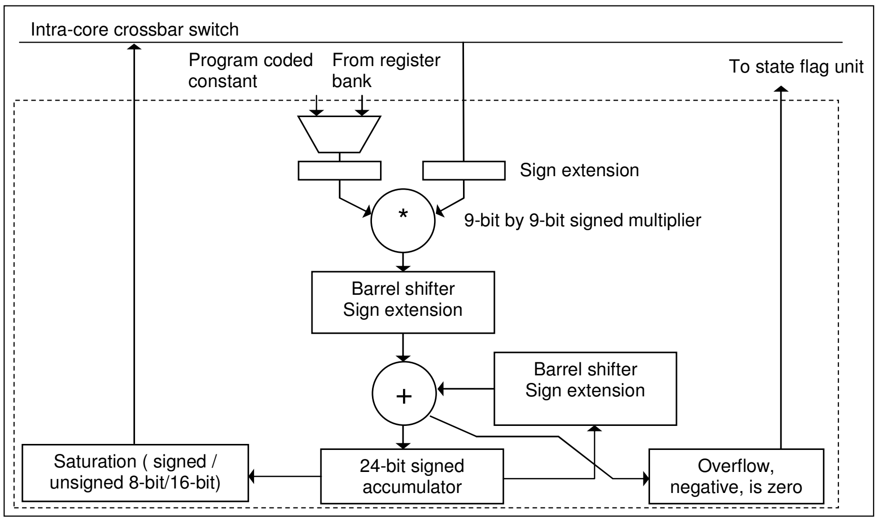
2) *Arithmetical and comparator unit*

The arithmetic unit contains an 8 bit multiply-add datapath logic with a 24 bit accumulator. The data path enables either 8 or 16 bit precision calculations. The arithmetic unit can calculate multiplication, multiple-add, addition, subtraction, and saturation operations (Figure 5a). Adopted from the common practice of handling both signed and unsigned data by the same unit, the hardware multiplier is of signed 9 by 9 bit precision, and the barrel shifter and the accumulator logic supports sign extension as well. The saturation mechanism also has a great importance in image processing, allowing the user to avoid the time consuming overflow and underflow management. Beside the arithmetical operations, this unit encompasses bit-field access as well.

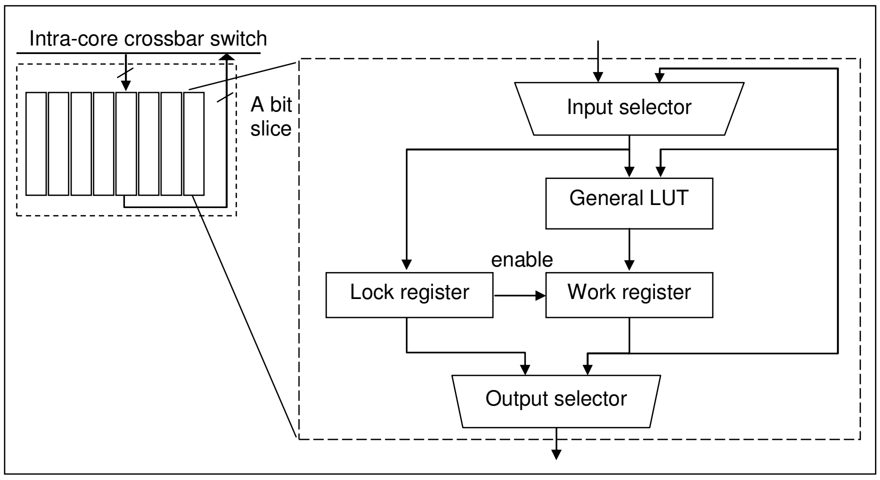
The comparator unit is capable to evaluate the relation between signed or unsigned data of the modules. Depending on the outcome of the relation it sets several flags that are used in later operations.

3) Morphology unit

The morphology unit supports the processing of black-and-white images (i.e. the pixel representation is one bit per pixel). It contains eight pieces of identical single bit morphology processor (Figure 5b). Hence, it accelerates greatly the parallel calculation of local or spatial logic operations, like erosion, dilation, opening, closing, hit and miss operations [29]-[30].


Intra-core crossbar switch
Program coded constant
From register bank
To state flag unit
Sign extension
*
9-bit by 9-bit signed multiplier
Barrel shifter Sign extension
+
Barrel shifter Sign extension
Saturation (signed / unsigned 8-bit/16-bit)
24-bit signed accumulator
Overflow, negative, is zero

(a)



(b)

Figure 5. The architecture of the arithmetical (a) and the binary morphology (b) units.

4) Memory and neighboring connectivity

The evergreen challenge of self and off-processor data access is solved by mapping the neighboring memories into the processors' memory map. This has been obtained simply by inserting an arbiter between the processor cores and their main memory. The neighboring connectivity works concurrently all over the array, which excludes data congestion.

The arbiter is capable to access the neighboring memories and substitute the required data by the neighbor's one. In this way, the processor needs no distinguished instructions to operate on image pixels that are actually stored in a nearby processor. The arbiter has a special task as well in case of binary image processing. Specifically, when the pixel representation is 1 bit per pixel, the near neighbors of a given pixel could fall in only in a few pixel radius. To get most out of the morphology unit, the arbiter is capable to align the data access to 1 bit resolution.

As regarding to the neighborhood operators, the neighboring access is not limited to support processing of large kernels (e.g in Xenon V3 this kernel size is 15 by 15).

Furthermore, there are special boundary modules that are straightforward extensions of the arbiter mechanism. These modules provide the boundary condition for the operations.

5) Other elements

Most of the processor's operations can be set to be conditionally executed (i.e. masked) depending on the state of the flags. This means, that content dependent masks may enable or disable the execution of a certain image processing operation in arbitrary pixel locations. This is extensively used for example in many nonlinear operators such as the rank order filters [31].

There is another way to block the processor operation, namely using the standby mechanism. The processors can individually enter standby state and wake up depending on their own data. Whenever it is set by the result of a comparison or another single bit operation, the whole processor idles (except from the memory and the inter-core data arbitration). In other words, it enables program alteration in this way.

Finally, there is an array wide logic value evaluation operation (global OR). Its sources are the flags of the processors, and it provides external feedback through the handshaking mechanism about the existence of an active or inactive processors.

B. Instruction set

A processor array can be considered from the programmer's point-of-view as a 8-bit CISC (complex instruction set computer) microprocessor with a number of replicated arithmetical unit and distributed memory. Since the same program controls all the processors, they execute always the same instructions on their own data (SIMD operation model).

Using the rich instruction set (112 in total without the conditional counterparts) one can efficiently implement a basic image processing function library (convolution, statistical filters, gradient, grayscale and binary mathematical morphology, etc). During processor specialization, several instructions may fall out of the range of the simplified processor array. This situation is handled by the compilers simply rejecting the unavailable instructions.

The general set of the instructions can be divided into five main groups with close relationship to the architecture:

- Initialization instructions
- Data transfer instructions
- Arithmetic instructions
- Logic instructions
- Comparison instructions

The initialization instructions are needed to clear or set the accumulator, the boundary condition registers, the flags, and other registers of the processor. These set also contains special instructions to setup and operate the sensor interface. Its importance is clear, as the synchronicity between the sensing and processing should be maintained continuously.

The data transfer instructions are used to transfer data between the internal registers and the memory. The cells can also access the memory of their direct neighbors through the neighborhood crossbar. From programming point of view, there are no distinguished instructions to handle the neighboring pixels. This dramatically improves the efficiency of programming e.g. larger neighborhood operators.

The arithmetic operation set contains addition, subtraction, multiplication, multiply-add operation, and left/right shift. All operators can be signed, unsigned, or mixed. These operators certainly set the state flags. These flags can be used later in the next instructions as conditions, carry propagation, or even block the processor by the standby module.

The collection of logic operations supports the execution of the binary mathematical morphology operations, like erosion and dilation, hit-or/and-miss type operators [29].

The comparison instructions are introduced to calculate the relation between two scalars. These operators can be used for statistical filter implementations.

The Figure 6 shows a piece of the Sobel operator's assembly code.

```
// vertical sobel operator
mul.nbr.const       Mem[ input ][-1,-1], 1;
macc.nbr.const      Mem[ input ][0,-1], 2;
macc.nbr.const      Mem[ input ][1,-1], 1;
macc.nbr.const      Mem[ input ][-1,1], -1;
macc.nbr.const      Mem[ input ][0,1], -2;
macc.nbr.const      Mem[ input ][1,1], -1;
// division by eight
acc.shr;
acc.shr;
acc.shr;
// saturated storage
sat8.reg        reg[1];
// square
mul.reg0.reg    reg[0] reg0_signed ;
mov.reg.treg    reg[0], reg[1];     // source
macc.reg0.reg   reg[1] reg0_signed ;
// normalization and saturated storage
acc.shl;
acc.shl;
sat16.mem       Mem[ output ];
return;
```

Figure 6. The assembly code of the vertical Sobel operator.

IV. SENSOR INTERFACE

In this chapter we describe the proposed interface architecture in general. To cope with the sensor processor resolution, technology trade-offs, we found a balanced solution by introducing relatively complex digital processors – and processor array clusters – serving a small array of sensor pixels. As to the technology and area problem, we have separated the sensors from the processors by adding and extra sensor layer above the processor layer.

In order to make the architecture technology independent as much as possible, we have formulated a simple synchronization protocol for the communication between the processors and the mixed-signal sensor interfaces. Each interface tile is connected directly to a single processor in the array through data buffer modules. The interface and the buffer modules are controlled by the associated processors and in return, the modules give feedback about the operation state to them.

Each of the sensor interfaces contains M by N external sensor mechanical and electrical interfaces, multiplexers, and an AD converter. These tiles work as simple high-speed imagers (Figure 7). Note that all the tiles are connected to the processors in parallel. The output of the sensor and ADC array is buffered, so the ADC can work at full speed without the need for waiting until their data processed.

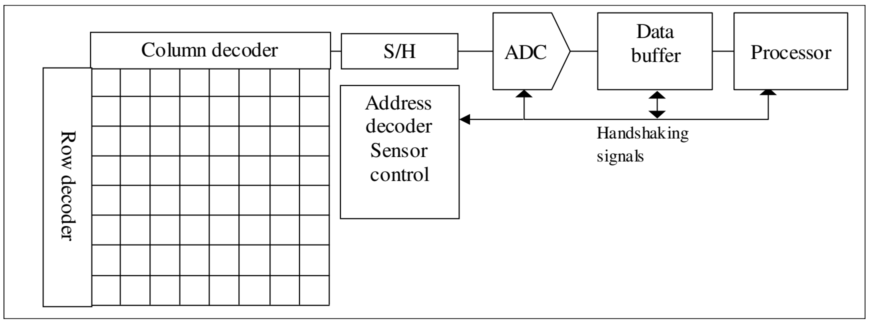


Figure 7. The figure shows the proposed sensor interface containing the analog sensor pad array, analog to digital converter, buffering, and necessary handshaking signals.

A. *Electrical sensor interface*

As to the electrical interfacing, there are uncountable options [16]-[20]. In our work, we have considered so far three basic integration type circuit configurations (Figure 8). These configurations are the active pixel sensor (APS), separated integrating capacitor, and capacitive transconductance amplifier (CTIA) cases. Each type has their strengths and weaknesses in terms of area, sensitivity, noise, and linearity [16]-[20].

In our former work [4], we have employed the separated integrating capacitor architecture in order to achieve higher speed. This interface is now attached to high sensitivity III-V compound semiconductor diode array. In the Xenon V3, that is to be described, the APS architecture is selected. This structure gives the most compact design, and the smallest sensor pitch. We intend to use this interface type with classic silicon and InP diode arrays.

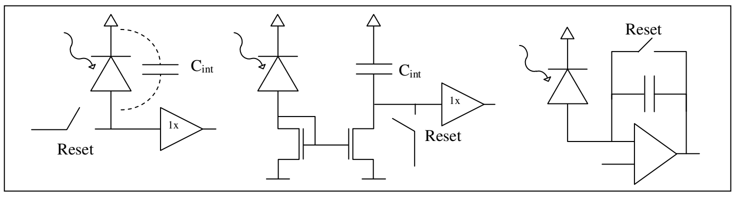


Figure 8. Three considered integrated photocurrent sensor architecture for different photodiode and application types (from left to right: active pixel sensor, separated integrating capacitor, and capacitive transconductance amplifier)

B. Mechanical sensor interface

The great advantage of the 3D bonding technology (Figure 9) is that it enables close to 100% fill factor without using up the valuable silicon space from the processors. The other advantage comes from the freedom of the choice of using different material for sensor and processor layer. This is very important, because the dedicated sensor silicon technologies are typically not the best for building high density processor arrays, and those technologies, which are excellent for building processors are not light sensitive enough. Moreover, besides silicon sensor, the utilized bump bonding technology enables the usage of exotic sensor materials sensitive in different wavelengths [26]. As it can be seen in the figure, this is a face-to-face bond type.

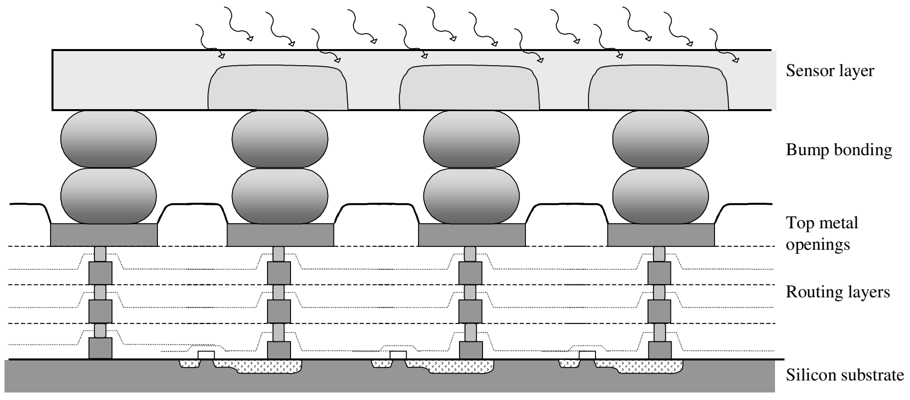


Figure 9. Illustration of the Indium Bump Bonding method.

From design point of view, the requirements for enabling 3D bump bonding technology are quite straightforward. It usually requires an array of standard openings on the passivation and sensor substrate connection ring around the array. The topological rule set, such as the opening size, minimal sensor pitch, and distance from unrelated bonding are easy to fulfill as they are far within the nowadays technology resolution (typically 5-10 um range).

V. ASIC IMPLEMENTATION

The Xenon V3* is an ASIC implementation of the proposed architecture. In this chapter, we describe this implementation details.

A. Overview

The Xenon V3 comprises a single array of 8 by 8 full-featured processors of this generalized architecture, set for sensing and processing 64 by 64 pixel sized images. With other words, the ratio between the processor and the associated sensor array is 1:64. Each of the processors' tiny sensor array has a resolution of 8 by 8 pixels. In order to evaluate the cluster operation of the architecture, we have integrated it into a smart camera, that contains among other components a high performance DSP and an FPGA as well [10], [25].

The used technology is 0.18 um drawn feature sized, contains one poly, and six metal layers offered

* The implemented IP is owned by Eutecus, Inc.

by UMC. The design contains more than half million transistors in total. Figure 10 shows the floorplan of the Xenon V3 overlaid on a die microphoto. Table I. concludes the general features of the design.

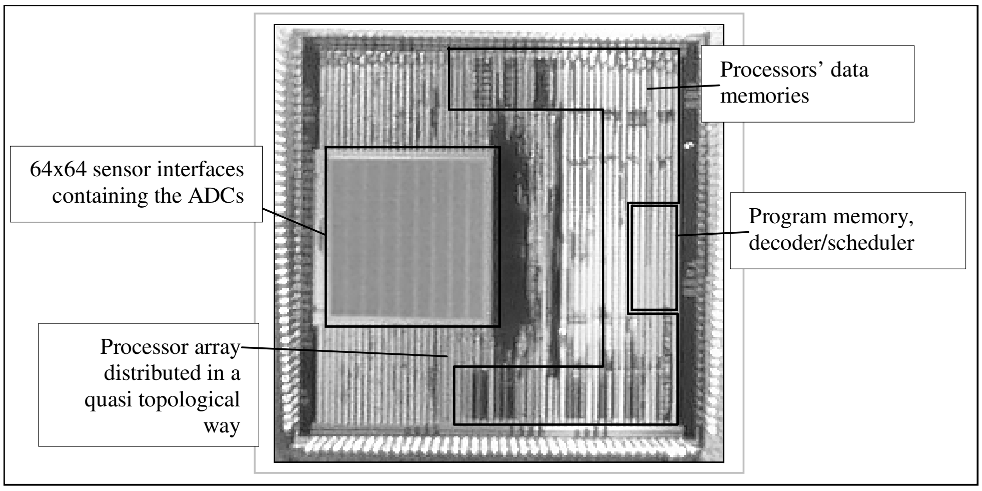

Processors' data memories
64x64 sensor interfaces containing the ADCs
Program memory, decoder/scheduler
Processor array distributed in a quasi topological way

Figure 10. The floorplan of the Xenon V3. (Source: Eutecus Inc.)

TABLE I. GENERAL FEATURES

Technology	UMC 0.18 um 1P6M generic process
Die size	5x5 mm^2
Equivalent gates	596 Kgates
Pixel array size	64x64
Processor array size	8x8
Area per sensor interface and processor	0.23 mm^2
Peak power consumption (digital array)	35 mW
I/O bus width	32-bit
I/O bus bandwidth	320 Mbyte/sec
Continuous conversion rate	100 Kframes per second

B. *Processor array*

The processor array and the chip level integration have been implemented by following the classic standard cell based digital design flow (i.e. synthesis, place, routing, post-layout verification).

The processors contain all of the optional modules, which have been described in the architecture introduction. Each processor contains 512 bytes of data memory. These memories are dual port SRAM IPs. These are not very efficient from silicon area point of view; however, this approach leads to the most robust digital ASIC implementation and helps to retain the portability of the source code.

Table II. shows anticipated execution time of different image processing operators running on the ASIC implementations. The performance is compared to a 1GHz DSP of Texas Instruments [28]. The figures show the per pixel execution times in nanosecond supposing the same image size, properly optimized codes, and excluding the external I/O. As it can be seen, the high-end DSP has a larger computational performance. However, we have to consider the used technology, the silicon area, and the power consumption differences to fairly compare the two designs.

TABLE II. PERFORMANCE COMPARISION. FIGURES SHOWS EXECUTION TIME OF THE LISTED IMAGE PROCESSING OPERATIORS IN NANOSECOND NORMALIZED TO A SINGLE PIXEL.

Operation	Xenon V3	Texas DSP 6415 @ 1GHz
Sobel operator	1.51	1.39
Convolution (3x3 kernel)	2.03	1.54
Convolution (9x9 kernel)	7.81	6.16
Local minima (3x3 kernel)	1.03	0.85
Binary dilation/erosion	0.76	0.84
Skeletonization	6.05	25.82

C. *Sensor interface*

The sensor interface is a combination of both mechanical and electrical constructs. As the mechanical arrangement should follow the sensor array regular topology, the interface is designed to be pitch matched to a rectangular grid. The illustrative floorplan of the interface can be seen in Figure 11.

The complete interface array is packed into a closed layout portion of the chip, near to the analog pad ring section, in order to isolate it from the digital noise, injected by the processor array (Figure 10).

1) Mechanical interface

The pads are 5 micron diameter openings in a 32 micron pitch. These pads are arranged to a 64x64 regular grid. Around the 64x64 grid, there are few more lines of pads, which are used to connect the common cathode voltage of the photo-diodes in the array to a reference point.

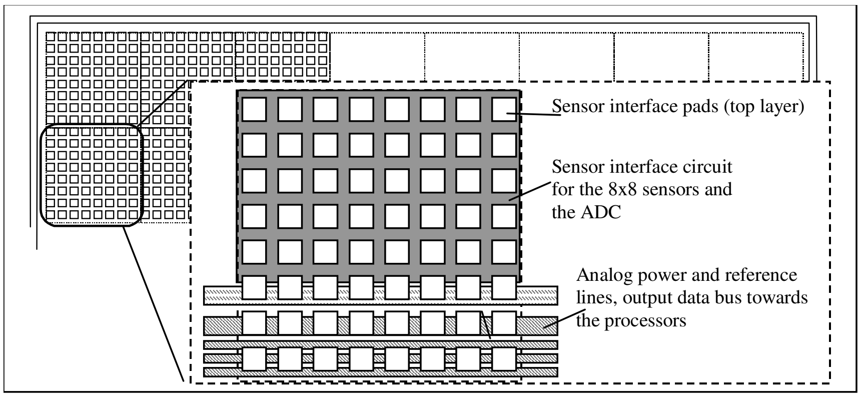

Sensor interface pads (top layer)
Sensor interface circuit for the 8x8 sensors and the ADC
Analog power and reference lines, output data bus towards the processors

Figure 11. The floorplan of the 64 by 64 sensor interface pads are arranged to 8x8 blocks. The drawing also shows the conditioning and conversion circuitry; and the power, ground, and data lines.

2) Electrical interface

The external photosensor read-out interface circuit is designed to handle anode-connected photodiodes in integration mode (Figure 12). As integration capacitor, the photodiodes self-capacitance is used. Its operation starts with a reset phase and than integration phase, in which the diode's junction capacitance is discharged by the photocurrent. As the diodes are to be connected at their anodes to the circuit (while their cathode is shortened together), the voltage appears on the output increases.

The circuit works on 3.3V, and has been built with thick gate oxide transistors only. This choice enables higher signal swing and operation that is more robust. In order to increase the reliability, the circuit encompasses basic ESD (electrostatic discharge) protection as well.

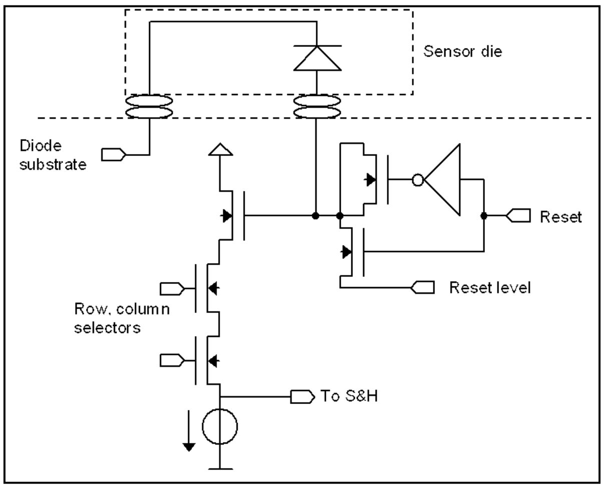


Figure 12. The implemented sensor interface.

3) AD conversion

The employed ADC is an 8-bit successive approximation type with current steering DAC. It has a 1V useful single-ended input range at 1.8V power supply and its effective number of bits is 7.2. The nominal operation speed is 8 Msamples per second. This conversion rate enables to reach less than 10μs conversion time for the connected array. As the interfaces work in parallel, this time is the conversion rate of the whole 64 by 64 image.

For test purposes, the control and the output of the AD converters can be accessed through an external bus also resulting in nothing but a 64 by 64 high speed camera chip.

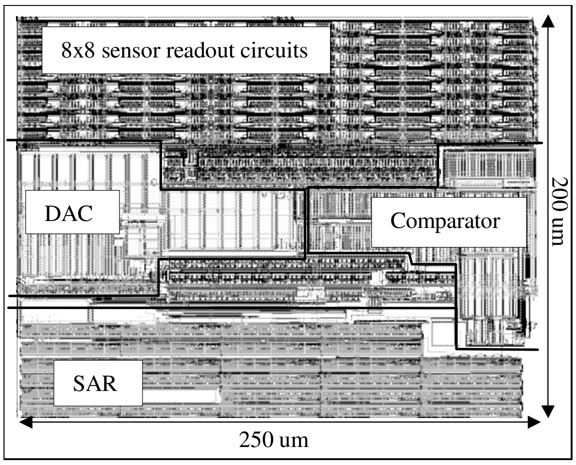


Figure 13. The layout and the size of the signal conditioning and conversion circuitry of the sensor interface. (Source: Eutecus Inc.)

VI. CONCLUSIONS

In this paper, we have given a deeper insight to our continuous efforts to integrate the focal plane sensing and tightly coupled processing. As we have shown, a general architecture has been developed to be flexible and customizable in low level and system level, and to be a versatile host for various exotic sensory integration.

We have demonstrated the proposed architecture by a functional ASIC implementation. This system is capable to continuously acquire up to 100 thousand images per seconds. Meanwhile, a large variety of image processing primitives can be efficiently executed on its processors, like convolution, look-up table, diffusion, rank order filters, contour detection.

ACKNOWLEDGMENT

This research was founded by US DoD small business R&D and transition program funding for Eutecus, Inc; Grant of the National Science Fund of Hungary (OTKA); the multidisciplinary doctoral school at the Faculty of Information Technology of the Pázmány Peter Catholic University.

REFERENCES

[1] T.Roska and L.O.Chua, "The CNN Universal Machine: An Analogic Array Computer", *IEEE Trans. Circuits and Systems*, Ser.II., vol. 40, pp. 163-173, 1993
[2] T. Roska, "Computer-Sensors: Spatial-Temporal Computers for Analog Array Signals, Dynamically Integrated with Sensors", *Journal of VLSI Signal Processing,* Vol.23, pp.221-237, 1999
[3] T. Roska and Á. Zarándy: "Proactive Adaptive Cellular Sensory-Computer Architecture via extending the CNN Universal Machine", to be published on the *ECCTD '03 European Conference on Circuit Theory and Design*, 1 - 4 September 2003, Kraków, Poland
[4] P. Földesy, Á. Zarándy, Cs. Rekeczky, and T. Roska „Digital implementation of cellular sensor-computers", *International Journal of Circuit Theory and Applications*, Vol. 34, No. 4, pp. 409-428., July 2006.
[5] L. Li, D. Cochran, and R. Martin, "Target Tracking with an Attentive Foveal Sensor", *Conference Record of the Thirty-Fourth Asilomar Conference on Signals, Systems and Computers*, pp. 182-185, 2000.
[6] Rekeczky, C.; Szatmari, I.; Balya, D.; Timar, G.; Zarandy, A., "Cellular multiadaptive analogic architecture: a computational framework for UAV applications", *IEEE Transactions on Circuits and Systems I*, Volume 51, Issue 5, pp. 864 – 884., May 2004.
[7] http://www.analogic-computers.com/ProdServ/Bi-i/index.html
[8] G. Linan: "ACE16K: an Advanced Focal-Plane Analog Programmable Array Processor", *ESSCIRC 2001 Presentations 27th European Solid-State Circuits Conference*, Villach, Austria, 18-20 September 2001
[9] P.Dudek, "Accuracy and Efficiency of Grey-level Image Filtering on VLSI Cellular Processor Arrays", CNNA 2004, Budapest, pp.123-128, Budapest, July 2004.
[10] "Bi-i Real-time Intelligent Cameras", Analogic-computers Ltd. whitepaper, available at http://www.analogic-computers.com/ProdServ/Bi-i
[11] T. Hamamoto and K. Aizawa: "A Computational Image Sensor with Adaptive Pixel-Based Integration Time", *IEEE Journal of Solid State Circuits*, Vol. 36. No. 4., pp. 580 – 585., April. 2001
[12] R. Wagner, Á. Zarándy and T. Roska "Adaptive Perception with Locally-Adaptable Sensor Array", *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, Vol. 51., No. 5., pp: 1014 - 1023, May 2004.
[13] M. D. Grossberg and S. K. Nayar: "High Dynamic Range from Multiple Images: Which Exposures to Combine?", *Int. Proc. ICCV Workshop on Color and Photometric Methods in Computer Vision (CPMCV)*, Nice, France, October 2003.
[14] S. K. Nayar and T. Mitsunaga: "High Dynamic Range Imaging: Spatially Varying Pixel Exposures", *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, Hilton Head Island, South Carolina, June 2000.
[15] Bolotski, M. Abacus: A Reconfigurable Bit-Parallel Architecture for Early Vision. PhD thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, 1996.

[16] A. El Gamal: “High Dynamic Range Image Sensors”, Tutorial at *International Solid-State Circuits Conference*, February 2002, Available: *http://www-isl.stanford.edu/~abbas/group/papers_and_pub/isscc02_tutorial.pdf*
[17] Hui Tian; Fowler, B.; Gamal, A.E.,”Analysis of temporal noise in CMOS photodiode active pixel sensor”, *IEEE Journal of Solid-State Circuits*, Volume 36, Issue 1, pp. 92 – 101, Jan. 2001.
[18] Peter B. Catrysse, Brian A. Wandell, „Optical efficiency of image sensor pixels”, *JOSA*, Volume 19, Issue 8, 1610-1620, August 2002.
[19] El Gamal, A., “Trends in CMOS image sensor technology and design”, *Electron Devices Meeting*, *IEDM '02*, Digest. International, pp. 805- 808, 2002.
[20] B. Schneider et al., “Image sensors in TFA (thin film on ASIC) technology,” in Handbook on Computer Vision and Applications. Boston, MA: Academic, 1999.
[21] Topol, A.W.; Furman, B.K.; Guarini, K.W.; Shi, L.; Cohen, G.M.; Walker, G.F., “Enabling technologies for wafer-level bonding of 3D MEMS and integrated circuit structures”, in *Electronic Components and Technology Conference,* 2004. Proceedings. 54th, Vol. 1, June 2004, pp.: 931 - 938
[22] V. Suntharalingam, R. Berger, J. Burns, C. Chen, C. Keast, J. Knecht, R. Lambert, K. Newcomb, D. O'Mara, C. Stevenson, B. Tyrrell, K.Warner, B. Wheeler, D.Yost, and D.Young, “CMOS image sensor fabricated in three-dimensional integrated circuit technology”, in *IEEE International Solid-State Circuits Conference*, vol. 1, February 2005, pp. 356-357.
[23] M. Koyanagy, Y. Nakagawa, K. W. Lee, T. Nakamura, Y. Yamada, K. Park, and H. Kurino, “Neuromorphic vision chip fabricated using three-dimensional integration technology”, in *IEEE International Solid-State Circuits Conference*, vol. 1, February 2001, pp. 270-271.
[24] Stephan Benthien et al., “Vertically Integrated Sensors for Advanced Imaging Applications”, *IEEE Journal of Solid-State Circuits*, Vol. 35, No. 7, pp. 939-946, July 2000.
[25] Tom R. Halfhill, “Faster Than a Blink”, In Stat Processor Watch, 02/12/2007 http://www.mdronline.com/watch/watch_abstract.asp?Volname=Issue%20%23021207&SID=1826
[26] http://www.solidstatescientific.com/a_home.html
[27] http://www.tezzaron.com
[28] www.ti.com
[29] J. Serra, Image Analysis and Mathematical Morphology. New York: Academic, 1982.
[30] Diamantaras, K.I.; Zimmermann, K.H.; Kung, S.Y., ”Integrated fast implementation of mathematical morphology operations in image processing”, *IEEE International Symposium on Circuits and Systems*, Vol.2, pp. 1442 – 1445, 1-3 May 1990
[31] Edward R. Dougherty and Jaakko Astola, Mathematical Nonlinear Image Processing, Springer, 1992.
[32] M.C. Herbordt, J.B. Cravy, R. Sam, O. Kidwai, C. Lin, “A System for Evaluating Performance and Cost of Massively Parallel Array Designs”, *Journal of Parallel and Distributed Computing*, No. 60 (2), pp. 217-246.
[33] L. Wanhammar, DSP Integrated Circuits, Academic Press, 1998.
[34] Foldesy, P.; Zarandy, A.; Rekeczky, C.; Roska, T., “High performance processor array for image processing”, *IEEE International Symposium on Circuits and Systems, ISCAS 2007,* pp.: 1177 – 1180, 27-30 May 2007
[35] “Pixim Digital Pixel System”, Pixim Inc., 2005.
[36] Robert Johansson, Leif Lindgren, Johan Melander, and Bjrn Mller, “A Multi-Resolution 100 GOPS 4 Gpixels/s Programmable CMOS Image Sensor for Machine Vision”, *IEEE Workshop on Charge Coupled Devices And Advanced Image Sensors*, 2003.
[37] Tonia Morris, Erica Fletcher, Cyrus Afghahi, Sami Issa, Kevin Connolly, Jean-Charles Korta. "A Column-based Processing Array for High-speed Digital Image Processing", IEEE *ARVLSI*, vol. 00, no. , p. 42, 20th 1999.
[38] S. Kleinfelder, SukHwan Lim, X. Liu, and A. E. Gamal,, “A 10 000 Frames/s CMOS Digital Pixel Sensor”, *IEEE Journal of Solid-State Circuits*, Vol. 36, No. 12, pp. 2049- 2059., December 2001.
[39] https://www.opencores.org