INTERNATIONAL JOURNAL OF CIRCUIT THEORY AND APPLICATIONS
Int. J. Circ. Theor. Appl. (2010)
Published online in Wiley Interscience (www.interscience.wiley.com). DOI: 10.1002/cta.679

Asynchronous cellular logic network as a co-processor for a general-purpose massively parallel array

Alexey Lopich*,† and Piotr Dudek

School of Electrical and Electronic Engineering, The University of Manchester, P.O. Box 88, Manchester M60 1QD, U.K.

ABSTRACT

This paper demonstrates an implementation of an asynchronous cellular processor array that facilitates binary trigger-wave propagations, extensively used in various image-processing algorithms. The circuit operates in a continuous-time mode, achieving high operational performance and low-power consumption. An integrated circuit with proof-of-concept array of 24×60 cells has been fabricated in a $0.35\,\mu m$ three-metal CMOS process and tested. Occupying only $16 \times 8\,\mu m^2$ the binary wave-propagation cell is designed to be used as a co-processor in general-purpose processor-per-pixel arrays intended for focal-plane image processing. The results of global operations such as object reconstruction and hole filling are presented. Copyright © 2010 John Wiley & Sons, Ltd.

KEYWORDS

asynchronous logic; cellular processor; wave computations; vision chip; asynchronous image processing

Correspondence

*Alexey Lopich, School of Electrical and Electronic Engineering, The University of Manchester, P.O. Box 88, Manchester M60 1QD, U.K.

†E-mail: a.lopich@manchester.ac.uk

Contract/grant sponsor: EPSRC; contract/grant number: EP/D029759/1

Received 11 April 2008; Revised 8 December 2009; Accepted 8 December 2009

1. INTRODUCTION

The ongoing research on massively parallel processor arrays [1–5] proves that they can be efficiently utilized for low-level image processing, due to a high degree of data parallelism and the locality of the required computations. The fine-grain parallelism of low-level image-processing algorithms is particularly suited to processor-per-pixel architectures [6–9]. These devices often integrate photo-sensors and pixel-processors on a single 'vision chip', providing image acquisition and pre-processing front-end and reducing the computational power requirements on the rest of the computer vision system.

The majority of pixel-parallel processor arrays operate according to a single-instruction multiple data (SIMD) paradigm, which has been demonstrated to be effective for operations where the resulting pixel value is explicitly expressed as a function of its bounded neighbourhood [10].

However, restricting the application range of such devices only to low-level image-processing algorithms results in a bottleneck during an off-chip data transmission, as the output result of low-level pre-processing algorithms is represented by the image of the same size as the input image. In contrast, medium-level processing algorithms reduce the initial image information to a set of more abstract descriptors (e.g. area, perimeter, coordinates, etc.). Thus, execution of such algorithms on a chip would lead to a significant reduction of the output data. As opposed to low-level processing, a number of medium-level image-processing operations (e.g. object reconstruction, distance transform, hole filling and other morphological and segmentation routines) involve global data flow across pixel network. These operations can be represented as an iterative process, where a pixel value, while determined as a function of local neighbourhood on each iteration, implicitly depends on the entire array of pixels. The meaningful processing during each iteration is typically carried out only in a limited number of 'active' pixels, at a front of a propagating wave of activity (a flood-filling or a grass-fire algorithm is the classical examples of this process). Geodesic reconstruction, illustrated in Figure 1, is one of the examples of employing such wave propagations in image processing. When implemented on a fully parallel, synchronous processor-per-pixel SIMD array, such algorithm requires many iterative steps ($\mathbf{O}(N^2)$ for $N \times N$

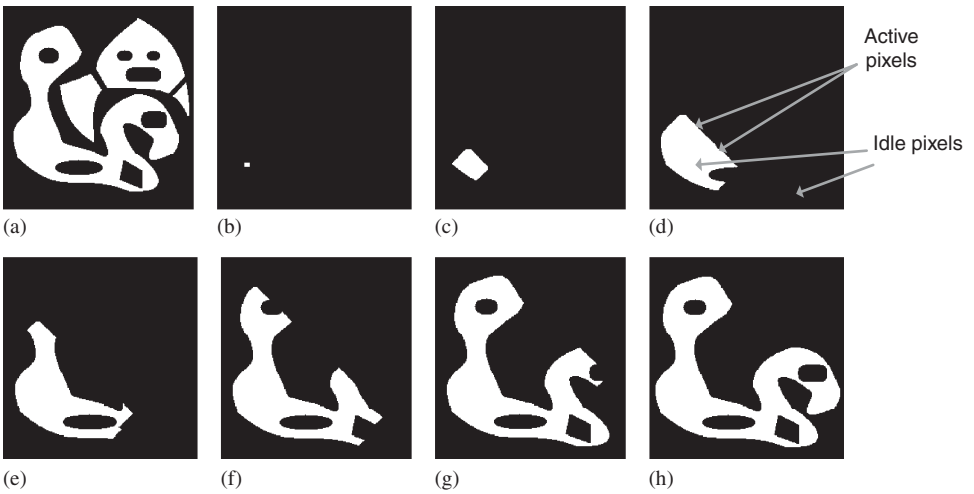

Figure 1. Geodesic reconstruction of the selected binary object: (a) input image (mask); (b) initial state; (c)–(g) the progress of wave-propagating activity; and (h) reconstructed object. The processing is required only in the wave-front pixels at the boundary of expanding marked region.

image) executed concurrently on all $N \times N$ processors, and therefore it has a high computational cost (and correspondingly inefficient use of hardware and high-power consumption). In order to optimize global operations on cellular processor arrays, it is necessary to look for approaches alternative to SIMD [11–14].

In order to increase the speed and efficiency of global operations, one should either introduce a very complex network of global interconnections or optimize the flow through the network of locally interconnected cells. The first approach becomes very difficult for a physical realization, since the sizes of processed images (and correspondingly the complexity of interconnections in processor arrays) are continuously increasing, whereas the number of routing layers, available for interconnections, is limited. The second approach, based on local communication only, yields itself better to a VLSI implementation. In case of iterative wave-propagating algorithms a promising approach involves local triggering of processor cells by their neighbours, so that they only perform the required computations when the results of computations in their neighbouring cells become available. As the timing of the entire operation is data driven, asynchronous methodologies can be employed. Apart from speeding up the propagations, asynchronous execution provides savings in power consumption as only active 'wave-front' pixel cells perform the actual processing, whereas others remain in the idle mode.

Global operations and cellular wave computing have been considered in the context of cellular nonlinear networks (CNNs) [15]. Application examples of grey-scale and binary trigger-wave propagations implemented using the CNN paradigm have been proposed [16, 17]. However, the classical CNNs have proven to be impractical for hardware implementation. A number of processor-per-pixel array architectures, designed specifically for asynchronous data-flow processing, have been presented in the literature, including the work on grey-scale morphology [13], segmentation [14], skeletonization [18] and rank-order filtering [19]. However, the algorithm-specific designs not only have a limited functionality, but also occupy relatively large silicon area, which is the major drawback when used as a co-processor for a general-purpose cellular processor array. Therefore, it is important to identify the global operation that is extensively used in the computer vision and has a simple hardware realization. Binary propagations (Figure 1) can be efficiently employed for basic object reconstruction and hole filling [20] as well as a component of more complex algorithms such as watershed segmentation [21] and convexity analysis. For example, in pixel-parallel implementation of retinal vessel tree extraction [22], discrete binary hole filling operation contributes 30% to the overall processing time. In order to explore the benefits of using an asynchronous sub-network within a general-purpose processor-per-pixel cellular array, we have designed a general-purpose asynchronous/synchronous processor array [23]. We have also implemented an asynchronous cellular logic array (ACLA) [20] chip as a proof-of-concept design for the evaluation of the actual performance of asynchronous processing. This paper is an extended version of the work presented in [24] and reports the experimental results obtained with this implementation.

2. PROCESSOR ARRAY ARCHITECTURE

Limited functionality and restriction on programmability make asynchronous cellular logic networks not very suitable for general-purpose image processing. Therefore, we propose a vision chip architecture that adopts omni-synchronous approach, thus benefiting in terms of programmability and versatility while in the SIMD mode (discrete-time, synchronous operation) and achieving the

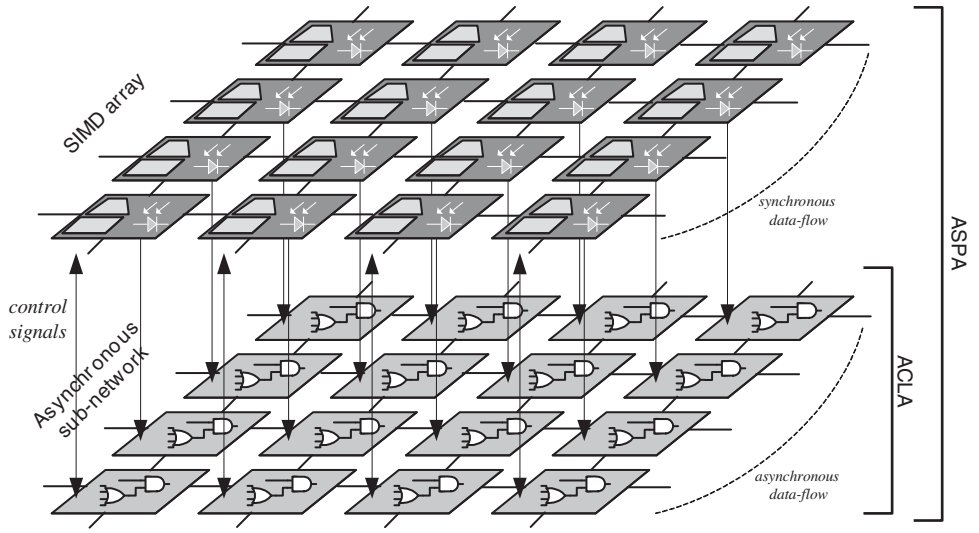

Figure 2. Asynchronous/synchronous processor array architecture—sub-networks with separate data flows.

maximum performance in global operations when configured as a single combinatorial circuit (continuous-time, asynchronous operation).

The system consists of an array of processing elements (PE), placed in a four-connected rectangular grid. Every PE is a simple digital processor with arithmetic and logic unit (ALU) and general-purpose registers. Eight 8-bit registers and an accumulator in the ALU enable every PE to store 64 bits of data. The PE operates as a datapath with register-transfer operations controlled by an instruction word. Operating in the SIMD mode, all PEs execute the same instruction broadcast by a central controller.

Simple synchronous digital architecture of the cell facilitates all arithmetic and logic operations, thus enabling efficient execution of most of the low-level image-processing algorithms. In order to extend functional capabilities of the array from local to global operations, every cell is endowed with a simple functional unit, called *propagation chain*, which provides a tool for binary trigger-wave propagations across the entire array, controlled by 'flag' and 'mask' indicators. Structurally, these asynchronous units can be considered as a separate sub-network (Figure 2) and this asynchronous network has been implemented as a separate chip for detailed evaluation of the performance and power characteristics.

It should be noted that the proposed architecture contains further mechanisms for asynchronous data transfers and operations as outlined in [23]; however, the discussion of these is outside the scope of this paper. In the following sections we focus on the ACLA part of the proposed architecture and its prototype silicon implementation.

3. ACLA—OPERATION PRINCIPLES

The operation of every cell in the asynchronous sub-network is based on a Boolean function that is fundamental to geodesic reconstruction. Let us consider a four-connected rectangular grid of cells and let us assign two Boolean values y and m to each cell. The value y corresponds to the initial pixel value (Figure 1(a)) or *mask* ('1'—foreground, '0'—background). The value m corresponds to the current *state* of the pixel ('1'—pixel is marked, '0'—unmarked), which describes the dynamic behaviour of the array. The process of reconstruction is carried out by iterative calculation in each cell of the following Boolean function:

$$m=(m_N \vee m_S \vee m_W \vee m_E \vee m_I)y \tag{1}$$

where m_I is the initial state (Figure 1(b)) and m_N, m_S, m_W and m_E correspond to current states of four nearest neighbours. At the end of this global operation all pixels of initially marked objects are marked. This operation has been illustrated in Figure 1.

In order to provide further control over the propagation, it is possible to constrain interconnection locally in every cell, so that the final function is as follows:

$$m=(c_N m_N \vee c_S m_S \vee c_W m_W \vee c_E m_E \vee m_I)y \tag{2}$$

where (c_N, c_S, c_W, c_E) is a locally stored interconnection mask vector. In this way we achieve reconfigurable topology, i.e. the direction of the propagation is constrained by local data in every pixel. Such a modification enables the usage of binary trigger-wave propagations in the algorithms that process 8-bit grey-scale image data. The example of employing this operation in watershed transformation can be found in [21]. Further applications of controlled propagations in image processing have been demonstrated in [7, 14, 20].

4. IMPLEMENTATION

We have fabricated a proof-of-concept chip that incorporates an array of cells, which execute operations described

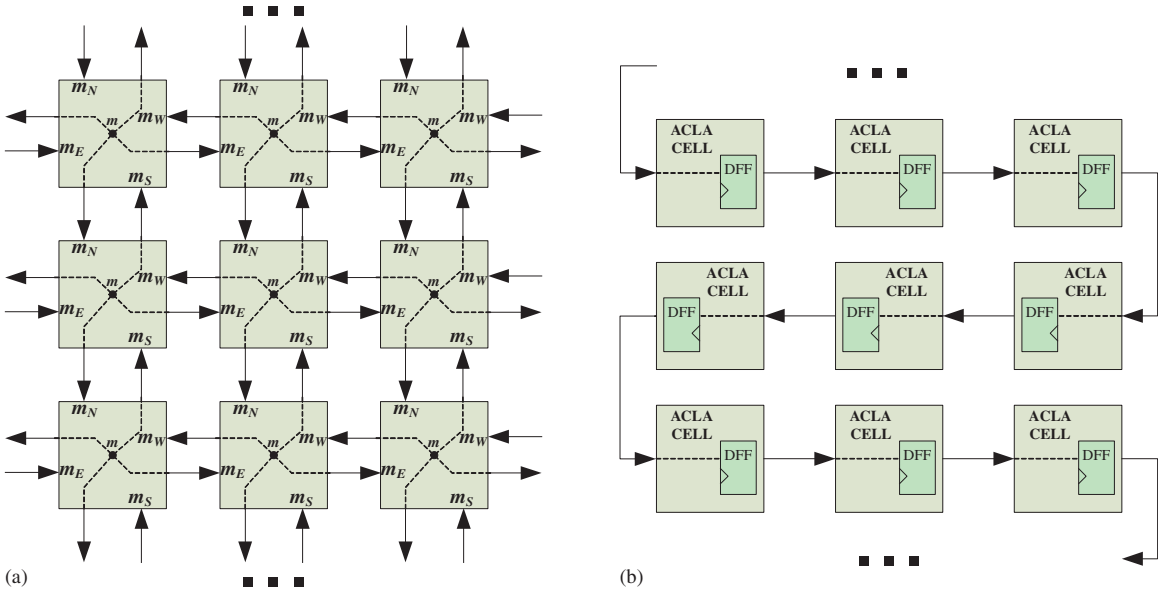

Figure 3. (a) Network connectivity of the ACLA chip and (b) shift-register data I/O organization in the ACLA chip.

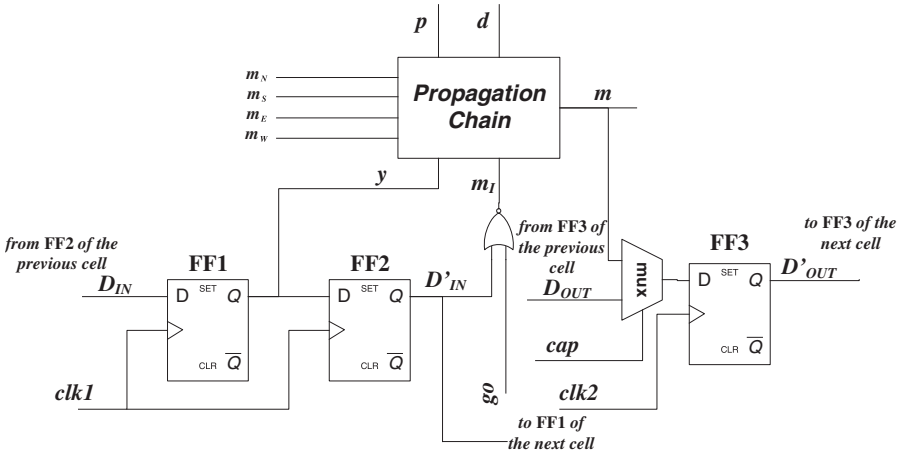

Figure 4. ACLA cell.

in the previous section in an asynchronous manner. The array is organized according to a rectangular grid arrangement of 24×60 cells with each cell interconnected to its four neighbours (Figure 3(a)). The architecture of each cell is presented in Figure 4.

The idea of the ACLA cellular processor array is that each cell would be combined with a sensor and a general-purpose processor; therefore, it is assumed that the input to the ACLA array would be pixel-parallel (Figure 2). However, since the test chip has been designed without a sensor part, a serial mechanism for data input/output was introduced. Each cell contains memory elements for input and output data, which are connected in a serial manner (Figure 3(b)). From this perspective the array acts as two 24×60-bit shift-registers (for input and output). The input register is constructed by chaining flip-flops FF1 and FF2 from each cell (Figure 4) in a snake-like pattern so that the output of the FF2 is connected to the input of the FF1 in the next cell (Figure 3(b)). The flip-flop FF1 is used for storing the input mask and the flip-flop FF2 for the initial state. The output mechanism is built in a similar way. The input and output memory is controlled by separate clocks ***clk1*** and ***clk2,*** correspondingly.

The propagation chain cell, which implements Equation 1, is shown in Figure 5. Its circuitry is based on domino logic. Both input mask and initial state are stored locally and applied to the inputs $\boldsymbol{y}$ and $\boldsymbol{m_1}$, correspondingly. Initially, outputs of all cells in the asynchronous network (node $\boldsymbol{m}$) are discharged by applying logic '1' to the input $\boldsymbol{d}$, and the storage node **X** is precharged in every cell by

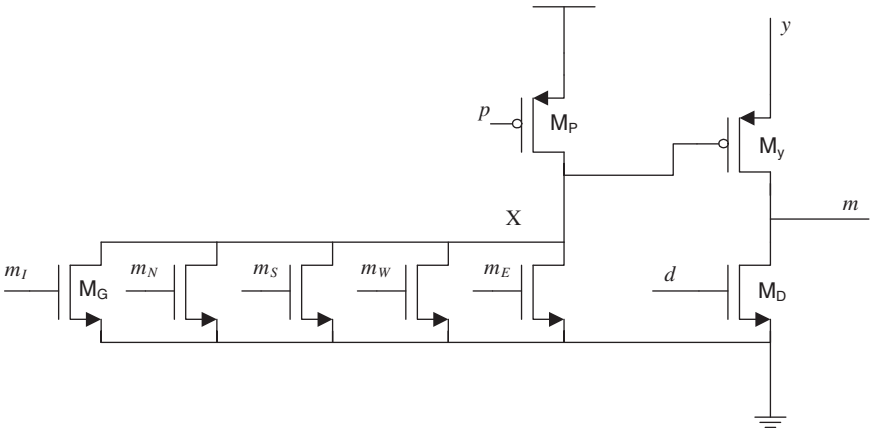

Figure 5. Schematic of the propagation chain.

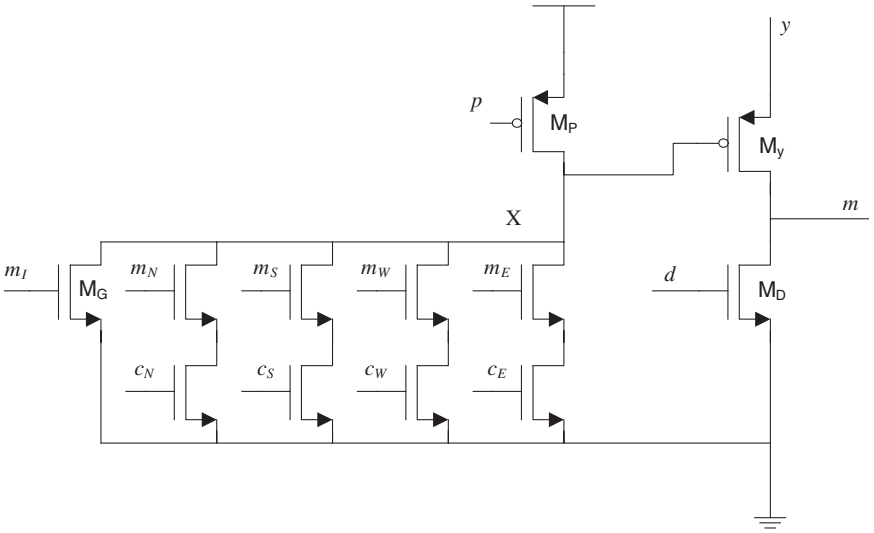

Figure 6. Propagation chain with local direction control.

setting the input ***p*** into logic '0'. After the initial steps are taken, signals ***p*** and ***d*** are withdrawn and the propagation is initiated by setting the ***go*** signal (Figure 4) to logic '0' so that the inverted value of the initial state is applied to the input $\boldsymbol{m_I}$. In order to eliminate static power dissipation, it is important to make sure that the mask signal (in the current setup ***go*** signal) is not applied at the same time as ***p*** and ***d*** signals. By discharging the node **X** the value ***y*** is applied to the output ***m***, thus triggering four neighbouring cells (if $\boldsymbol{y}$ = '1'), and so on. This operation finishes when all pixels of the object defined by ***y*** and marked by $\boldsymbol{m_I}$ are selected ($\boldsymbol{m}$ = '1'). An example of propagation chain with local control over neighbourhood connectivity, described by Equation 2, is presented in Figure 6. After the propagation is complete (or during the propagation to capture the intermediate data for experimental purposes), the result is latched into the output flip-flop FF3. The input ***cap*** (Figure 4) is used as a select signal for a multiplexer, so that after the result is stored, the input of the flip-flop

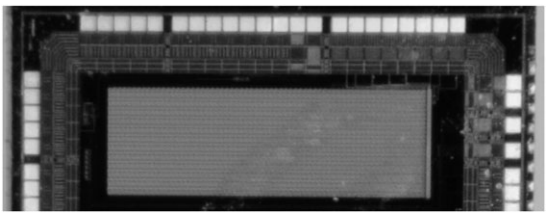

Figure 7. Microphotograph of the ACLA chip.

FF3 is connected to the output of the same flip-flop in the neighbouring cell to enable shifting out the result.

The chip, shown in Figure 7 is fabricated in a 0.35 μm three-metal layer CMOS process by austriamicrosystems. The size of the cell is $25 \times 25\,\mu m^2$, of which the propagation chain cell comprises only $16 \times 8\,\mu m^2$. The layout of the cell is illustrated in Figure 8. Such a small size makes it attractive to incorporate this circuit as a co-processor to a general-purpose processing cell. In the processor array reported in

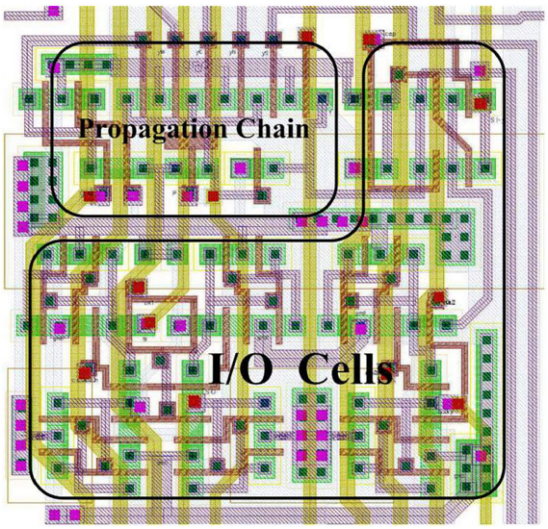

Figure 8. Cell layout.

[23] such asynchronous propagation unit occupies only 1% of the $100 \times 117\,\mu m^2$ cell area. It should be emphasized that in the vision chip it is essential to ensure that the effect of photo-induced leakage currents does not lead to false triggering. Therefore, additional shielding has been applied to all nodes that discharge node ***m*** and precharge/discharge node **X**. The latter also improves the protections against signal coupling.

5. RESULTS AND DISCUSSION

The fabricated circuit has been evaluated in a test system consisting of an FPGA (generation of control signals and managing support devices) and standard test equipment (logic analyzer, pulse generator, oscilloscope, etc.). Correct operation and performance of the ACLA chip have been verified by varying the delay (with resolution step of 5 ps) between the falling edge of the ***go*** signal (initiates the propagation) and the rising edge of the clock ***clk2*** (latches the state of the array). In this way the state of the array can be captured during the propagation. The output data showed that the average processing time per cell is approximately 0.48 ns, which is consistent with post-layout simulation results (Figure 9).

The results of processing are presented in Figure 10. To illustrate the operation of the circuit, the results were captured at various stages during the propagation. Sweeping the time between the ***go*** signal and the ***clk2*** signal, multiple captures were performed, and the grey-level of a pixel in the sampled images plotted in Figure 10 corresponds to the number of 1's and 0's captured in that pixel, across all capture runs at the corresponding sample time. It is interesting to notice that the shape of the propagation front has a spherical rather than a diamond shape. The reason for this phenomenon is the increased speed of switching of the OR gate in the propagation chain circuit (Figure 5) when triggered by several neighbours simultaneously. Owing to some clock distribution issues not all the pixels in the array are accessible and therefore test images are smaller than the original size of the fabricated array. While transistor mismatch and parasitic capacitance may lead to slight differences in propagation speed across the array, the observed effect is very small and in either case, the result of the basic reconstruction/hole filling operation is not dependant on the propagation speed uniformity.

On an equivalent SIMD array (with one clock cycle per iteration) the cell's processing time of 0.48 ns corresponds to a 2.08 GHz clock. Interpolating this result to a 256×256 image, the frame processing time becomes 31.5 µs (the worst case, $N \times N$ iterations, although typically a smaller number could be used), which enables processing binary images at 31×10^3 images/sec and faster. Considering real-time computer vision applications, this allows performing more than 1000 geodesic reconstructions on a 256×256 frame at a frame rate of 30 frames/second. Many propagations can be therefore performed within a single video frame, for example to implement multiple object reconstructions, or use propagations extensively as a component of more complex image-processing operations [21]. Generally, the propagation time depends on the strength of switching transistors M_G and M_P, operating voltage and parasitic capacitance of interconnection wires. Considering the latter two characteristics to be fixed, one should consider the trade-off between cell size and the speed of the asynchronous operation. The total energy consumed by triggered pixel during propagation is 0.4 pJ. The power consumption is primarily contributed by discharging/precharging of gate and drain parasitic capacitances and coupling wire capacitance in the node *m*. While processing images at 31×10^3 frames per second, the power consumption contribution of each cell during asynchronous processing is 12.7 nW, which would result in 0.83 mW for a 256×256 array. The main contribution to power consumption, however, is made by driving global signals ***p*** and ***d***. Based on measurement results during the propagation when all the cells are triggered, the extracted total energy consumed by a cell including the initial setup is 11.39 pJ. For 256×256 array performing 31×10^3 propagations per second this would result in total power consumption of 21.1 mW. The latter figure does not include the contribution of I/O pads.

The performance of the ACLA chip could be favourably compared with arrays that provide similar facilities [7, 25]. The maximum throughput during asynchronous operation for the NSIP [7] chip is reported to be 4×10^6 frames per second, which corresponds to approximately 2 ns delay per cell. Taking into account that the cell size is $118 \times 118\,\mu m^2$ and the design was fabricated in 0.8 µm process, the delay can be scaled down to approximately 0.87 ns. In the case of the chip reported in [25], the propagation delay per cell is 4 ns (however, this chip benefits from threshold capabilities and applies a 3×3 mask to the eight-connected neighbourhood in every pixel).

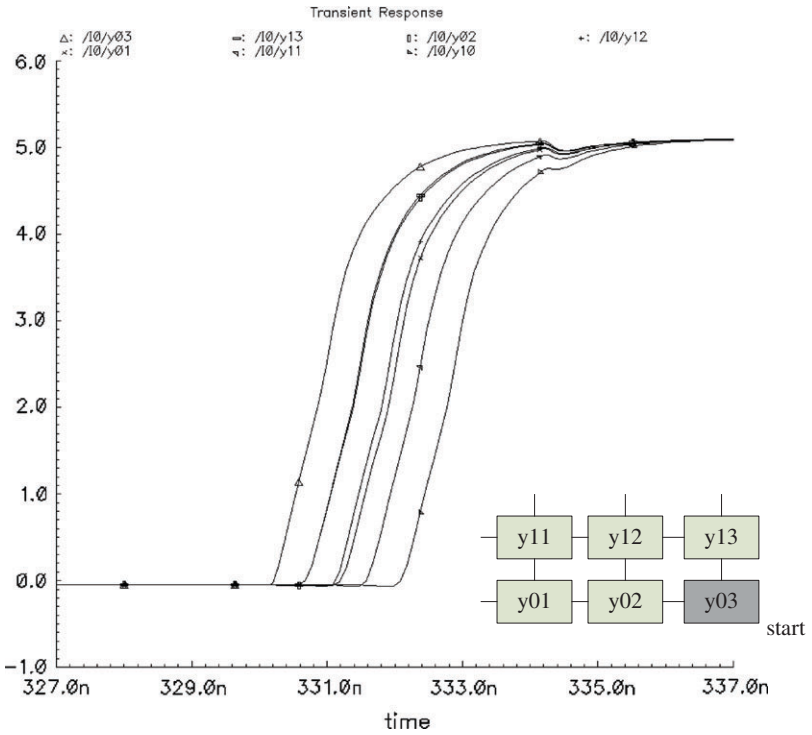

Figure 9. Simulation of the propagating activity—the voltages at nodes *m* of several cells (y01-y03:y11-y13) during propagation initiated from the cell y03.

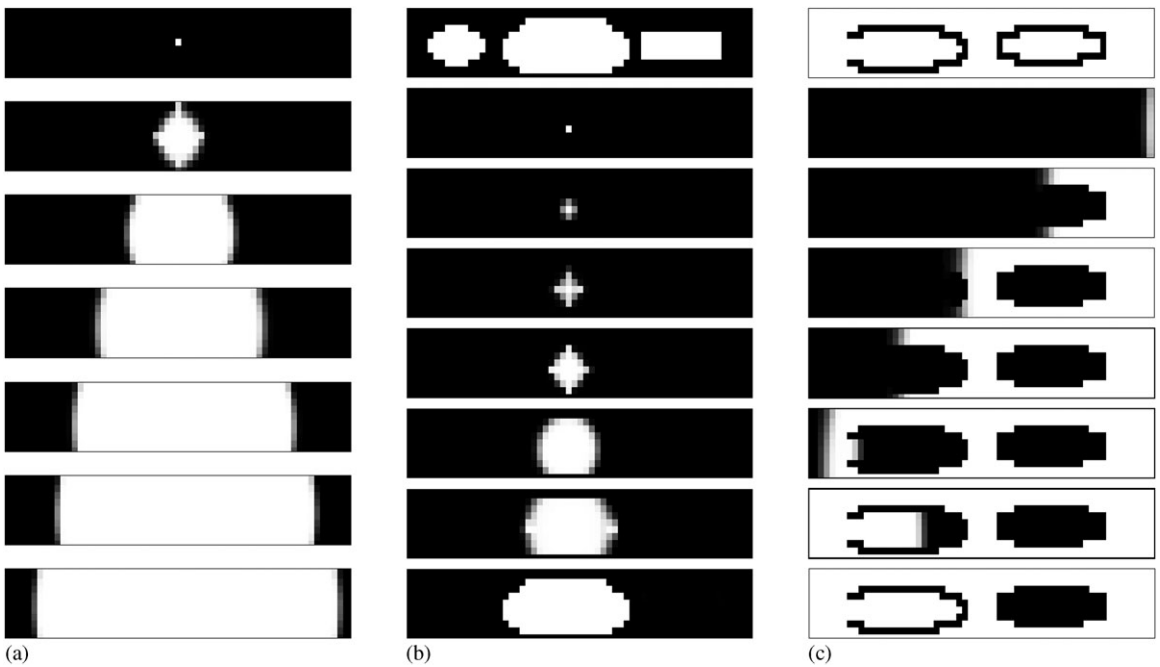

Figure 10. Global operations implemented on the ACLA chip: (a) wave propagation (sample rate 2 ns); (b) object reconstruction (sample rate 1.8 ns); and (c) hole filling (sample rate 5 ns).

6. CONCLUSIONS

An implementation of an asynchronous cellular array has been presented. The designed array combines small cell area, high processing speed and low-power consumption. The performance and power characteristics of the circuit have been evaluated and indicate that it can be efficiently used as a co-processor in a processor-per-pixel array in order to enable global operations at minimum hardware cost. The size of the asynchronous part of the circuit is $16 \times 8\,\mu m^2$ in 0.35 μm technology, which is a small fraction of a typical cellular processor array cell area. the current development of a vision chip in 0.18 μm process indicates that the area of the ACLA cell with additional control over the pixel network discussed in Section III is only 84.8 μm^2. By scaling the cell size down and improving the strength of the driving transistors in the propagation chain, the simulated propagation delay is 0.22 ns. Digital architecture of the asynchronous unit suggests that it is comfortably scalable to even finer technologies, though attention has to be paid to higher leakage off-currents for below 90 nm processes or below. The ongoing research on this topic concentrates on the integration of the asynchronous propagation architecture into general-purpose pixel-parallel processor arrays, and on the development of a new reconfigurable architecture of the general-purpose processing cell that would facilitate asynchronous processing utilizing the same circuitry that is used during the synchronous operation.

ACKNOWLEDGEMENT

This work has been supported by the EPSRC under grant no EP/D029759/1.

REFERENCES

1. Lindgren L, Melander J *et al*. A multiresolution 100-GOPS 4-Gpixels/s programmable smart vision sensor for multisense imaging. *IEEE Journal of Solid-State Circuits* 2005; **40**(6):1350–1359.
2. Andrews D, Kancler C, Wealand B. An embedded real-time SIMD processor array for image processing. *Proceedings of the 4th International Workshop on Parallel and Distributed Real-Time Systems*, Los Alamitos, U.S.A., 1996; 131–134.
3. Cucchiara R, Di Stefano L *et al*. The GIOTTO system: a parallel computer for image processing. *Real-Time Imaging* 1997; **3**(5):343–353.
4. Kleihorst RP *et al*. Xetal: a low-power high-performance smart camera processor. *IEEE International Symposium on Circuits and Systems*, NJ, U.S.A., 2001; 215–218.
5. Shin J-K *et al*. A bio-inspired CMOS vision chip for edge detection using an offset-free column readout circuit. *Proceedings of the SPIE—The International Society for Optical Engineering* 2005; **5839**:460–468.
6. Dudek P, Hicks PJ. A general-purpose processor-per-pixel analogue SIMD vision chip. *IEEE Transactions on Circuits and Systems I*: *Fundamental Theory and Applications* 2005; **52**(1):13–20.
7. Eklund J-E, Svensson C, Astrom A. VLSI implementation of a focal plane image processor—a realization of the near-sensor image processing concept. *IEEE Transactions on VLSI Systems* 1996; **4**(3): 322–335.
8. Linan GC *et al*. A 1000 FPS at 128×128 vision processor with 8-bit digitized I/O. *IEEE Journal of Solid-State Circuits* 2004; **39**(7):1044–1055.
9. Komuro T, Kagami S, Ishikawa M. A dynamically reconfigurable SIMD processor for a vision chip. *IEEE Journal of Solid-State Circuits* 2004; **39**(1): 265–268.
10. Dudek P. Accuracy and efficiency of grey-level image filtering on VLSI cellular processor arrays. *IEEE International Workshop on Cellular Nanoscale Networks*, Budapest, 2004; 123–128.
11. Ducourthial B, Merigot A. Parallel asynchronous computations for image analysis. *Proceedings of the IEEE* 2002; **90**(7):1218–1229.
12. Galilee B, Mamalet F *et al*. Parallel asynchronous watershed algorithm-architecture. *IEEE Transactions on Parallel and Distributed Systems* 2007; **18**(1): 44–56.
13. Robinson WH, Wills DS. Efficiency analysis for a mixed-signal focal plane processing architecture. *Journal of VLSI Signal Processing Systems for Signal, Image, and Video Technology* 2005; **41**(1):65–80.
14. Manzanera A. Morphological segmentation on the programmable retina: towards mixed synchronous/asynchronous algorithms. *6th International Symposium on Mathematical Morphology*, Sydney, Australia, 2002; 389–399.
15. Roska T. Circuits, computers, and beyond Boolean logic. *International Journal of Circuit Theory and Applications* 2007; **35**(5–6):485–496.
16. Hillier D, Binzberger V *et al*. Topographic cellular active contour techniques: theory, implementations and comparisons. *International Journal of Circuit Theory and Applications* 2006; **34**(2):183–216.
17. Rekeczky C, Chua LO. Computing with front propagation: active contour and skeleton models in continuous-time CNN. *Journal of VLSI Signal Processing Systems for Signal, Image, and Video Technology* 1999; **23**(2–3):373–402.
18. Lopich A, Dudek P. Architecture of asynchronous cellular processor array for image skeletonization. *Proceedings of the 2005 European Conference on Circuit Theory and Design*, Cork, Ireland, 2005; 81–84.

19. Poikonen J, Paasio A. A ranked order filter implementation for parallel analog processing. *IEEE Transactions on Circuits and Systems I*: *Fundamental Theory and Applications* 2004; **51**(5):974–987.
20. Dudek P. An asynchronous cellular logic network for trigger-wave image processing on fine-grain massively parallel arrays. *IEEE Transactions on Circuits and Systems II*: *Analog and Digital Signal Processing* 2006; **53**(5):354–358.
21. Lopich A, Dudek P. Global operations in SIMD cellular processor arrays employing functional asynchronism. *IEEE International Workshop on Computer Architecture for Machine Perception and Sensing*, Montreal, Canada, 2007; 16–23.
22. Alonso-Montes C, Vilarino DL *et al*. Fast retinal vessel tree extraction: a pixel parallel approach. *International Journal of Circuit Theory and Applications* 2008; **36**(5–6):641–651.
23. Lopich A, Dudek P. Architecture of a VLSI cellular processor array for synchronous/asynchronous image processing. *IEEE International Symposium on Circuits and Systems*, Greece, 2006; 3618–3621.
24. Lopich A, Dudek P. Implementation of an asynchronous cellular logic network as a co-processor for a general-purpose massively parallel array. *European Conference on Circuit Theory and Design*, Seville, Spain, 2008; 84–87.
25. Flak J, Laiho M *et al*. Dense CMOS implementation of a binary-programmable cellular neural network. *International Journal of Circuit Theory and Applications* 2006; **34**(4):429–443.

DOI: 10.1002/cta