Trigger-Wave Asynchronous Cellular Logic Array for Fast Binary Image Processing

Przemysław Mroszczyk , *Student Member, IEEE*, and Piotr Dudek , *Senior Member, IEEE*

***Abstract*—This paper presents the design and the VLSI implementation of an asynchronous cellular logic array for fast binary image processing. The proposed processor array employs trigger-wave propagation and collision detection mechanisms for binary image skeletonization, and Voronoi tessellation. Low power, low area, and high processing speed are achieved using full custom dynamic logic design. The prototype array consisting of 64×96 cells is fabricated in a standard 90 nm CMOS technology. The experimental results confirm the fast operation of the array, capable of extracting up to 2.78×10^6 skeletons per second, consuming less than 1 nJ/skeleton. The asynchronous operation enables circular wave contours, which improves the quality of the extracted skeletons. The proposed asynchronous processing module consists of 24 MOS transistors and occupies $5.5\ \mu\text{m} \times 7.4\ \mu\text{m}$ area. Such array can be used as a co-processing unit aiding global binary image processing in standard pixel-parallel SIMD architectures in vision chips.**

***Index Terms*—Cellular processor array, CMOS, CNN, image processing, skeletonization, trigger-wave propagation, vision chips.**

I. Introduction

MASSIVELY parallel processor arrays based on the single instruction multiple data (SIMD) principle are used for efficient execution of low-level image processing algorithms. Such tasks exhibit a high degree of parallelism with only nearest neighborhood communication [1]–[7]. In particular, "processor-per-pixel" architectures enable design of compact, high-performance, and low-power integrated vision systems realized using digital [3]–[6] and mixed processing elements [7], [8]. Furthermore, various realizations employing cellular neural networks (CNN) [9]–[12] and analogue logic automata [13] have been proposed. The potential for very high processing speed in such systems is often limited by off-chip data transfer, when the amount of input and output data (i.e., captured and processed images) remains practically the same. In many applications, however, (e.g., robotics, surveillance etc.) only a set of abstract features of objects describing their structures, locations, and interactions is essential for the next processing step. Object feature extraction can reduce the off-chip data transfer but often employs medium-level algorithms requiring global image processing tasks such as geodesic reconstruction, distance transformation, skeletonization, etc. The execution of such algorithms in SIMD processor arrays is generally an iterative process, resembling a wave propagating across the image, where only the pixels located on the wave-fronts perform computation. In principle, the propagation can be seen as a generic computational engine applicable to a variety of image processing tasks. Many morphological operations (e.g., hole filling, closed curve detection, etc.) have already been implemented using asynchronous logic arrays [14]–[17] or CNN-based machines [19], [20], [28]. However, efficient realizations of complex tasks, such as skeletonization and tessellation, still remain a challenge.

This paper extends our simulation-based work from [18], discussing the idea of hardware implementation of the trigger wave propagation and wave front collision detection in binary image processing. In this paper, we provide experimental results obtained from the measurements of the prototype test array fabricated in a standard 90 nm CMOS technology and consisting of 64×96 pixel-processors, operating according to a non-synchronous, data-driven scheme with no global clock signal involved. The measurements are supplemented with an extensive discussion on design issues, testing methodology, and benchmarking with comparison to other works.

Section II of this paper discusses different approaches to binary image skeletonization, Section III shows the system design and implementation, Section IV presents the results and design issues, and Section V concludes the paper.

II. Skeletonization

In the following section, we will consider binary images, where objects in the foreground are already segmented and clearly distinct from the background [Fig. 1(a)]. In general, a skeleton consists of lines and curves creating continuous structure describing the shape and size of an object. Skeleton is usually defined as a set of points equally distant from the edges and associated with the "ridges" on the distance transformation map [Fig. 1(b)]. These points can also be interpreted as the centres of bitangent circles drawn into the object [Fig. 1(c)], or as locations of collisions between isotropic waves triggered from the boundary and propagating to the inside of the object [Fig. 1(d)] [21], [22]. There are several methods of binary image skeletonization, exhibiting different levels of complexity and returning slightly different results. The most common ones, usually considered in the VLSI realizations, are based on iterative thinning or distance transformation [23], [24].

When mapped upon regular 2D cellular processor arrays, thinning algorithms require only logical operations and memory of one bit per pixel. However, a number of templates, sometimes of two pixels radius may be required, increasing the complexity of the corresponding hardware realization. In the case of the distance transformation, every processing

Manuscript received June 25, 2014; revised September 18, 2014 and October 06, 2014; accepted October 08, 2014. Date of publication November 06, 2014; date of current version January 26, 2015. This work was supported by the Engineering and Physical Sciences Research Council (EPSRC) of the U.K. This paper was recommended by Associate Editor B.-D. (Brian) Liu.

The authors are with the School of Electrical and Electronic Engineering, University of Manchester, Manchester, M13 9PL, U.K. (e-mail: przemyslaw.mroszczyk@postgrad.manchester.ac.uk; pdudek@manchester.ac.uk).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCSI.2014.2363511

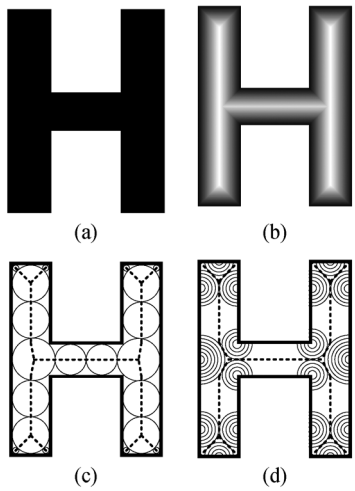

Fig. 1. (a) Binary object, (b) skeleton denoted by the "ridges" on the distance transformation map, (c) method of bitangent circles, (d) method of trigger-wave propagation.

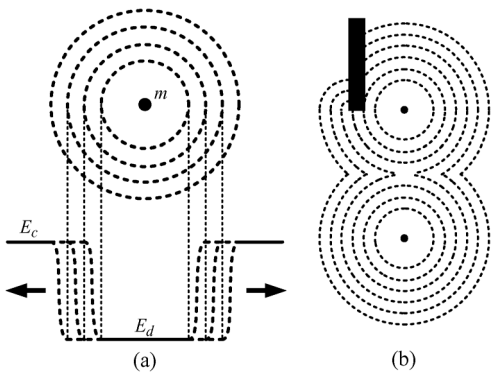

Fig. 2. Autowave properties (dotted contours illustrate wave-fronts at successive time intervals): (a) the expansion of the wave-front in an active medium starting from trigger point marker m (the wave-front separates charged and discharged areas of energies E_c and E_d respectively), (b) annihilation of two colliding waves and diffraction on the object.

element has to store information about its distance relative to the object's edge, which involves numerical computation and makes the implementation image-size variant. Skeletonization based on trigger-wave propagation and wave-front collision detection was initially proposed in [22]. In that approach, each boundary point triggers a wave that propagates to the inside of the object. It can be assumed that the points where the waves collide form a skeleton. The majority of works considering CNN based solutions are mainly inspired by the properties of a light-sensitive chemical nonlinear system, a variant of the Belousov-Zhabotinski medium, and try to build its electronic hardware replicas [19]. The nonlinear autonomous waves (autowaves) observed in such systems exhibit a series of interesting properties. They propagate with a constant speed, utilizing the locally stored energy of the active medium, thus they can expand infinitely, preserving their initial amplitudes and contours. The wave-front leaves the medium behind the front "discharged," inhibiting any other propagation [Fig. 2(a)]. When two wave-fronts meet, the propagation cannot proceed and the waves annihilate. The basic properties of autowaves are graphically illustrated in Fig. 2(b). Autowave propagation is a natural phenomenon and can also be observed as combustion waves, nerve impulses, and epidemic spreads [22].

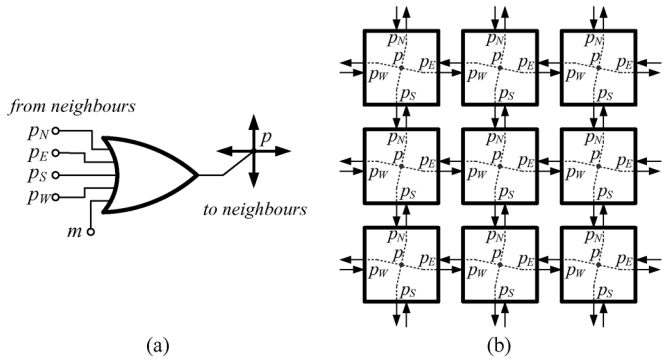

Fig. 3. (a) Schematic diagram of the propagation OR gate and (b) the network connectivity of the pixel-parallel logic array realising the wave propagation concept.

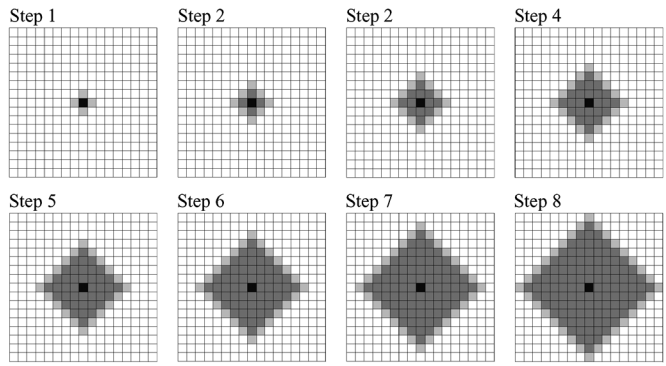

Fig. 4. The expansion of the propagation wave triggered from the centre of the array (color map used in the figure: white □—charged (not yet activated) cell, light gray ■—cell that receives the propagation signal, medium grey ■—dischrged (activated) cell, black ■—cell triggered from the marker).

III. Hardware Implementation

A. Propagation Gate

In VLSI systems, the idea of trigger-wave propagation is typically implemented using a pixel-parallel array of OR gates. Each cell connects to only four orthogonal neighbors denoted as p_N, p_E, p_S and p_W, which provides a sufficient amount of information for the algorithm while keeping the hardware realization simple (Fig. 3).

Such a circuit remains stable when the inputs and outputs of all the gates are in low logic state. The output p of a particular OR gate can be set up to a high logic state (activated) using the additional input m, individually triggering a selected gate according to the formula:

$$p = p_N \vee p_E \vee p_S \vee p_W \vee m \tag{1}$$

In such a case, the neighbors of the activated gate will detect the high logic states on their inputs and turn on, triggering their respective neighbors. The expansion of the activated region around the triggered cell resembles a propagation mechanism which continues until all the gates in the array are in the high logic state (Fig. 4). After that, the array remains stable and a new propagation is not possible until the gates are again set to the low logic state (i.e., the array requires initialization).

Usually, the arrays of OR gates used in image processing require some additional features which, for example, enable the definition of a propagation space or set a particular direction of the propagation [14]–[18]. In this paper, we consider a design of a test array dedicated for image skeletonization, therefore, we

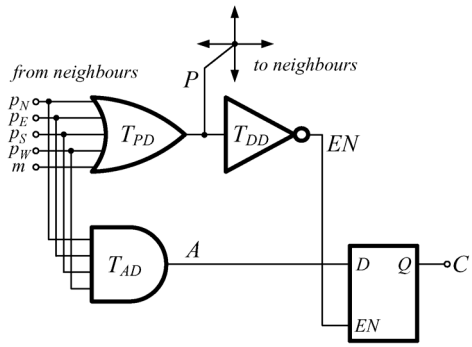

Fig. 5. The logic diagram of the proposed cell with the propagation and the collision-detecting circuit.

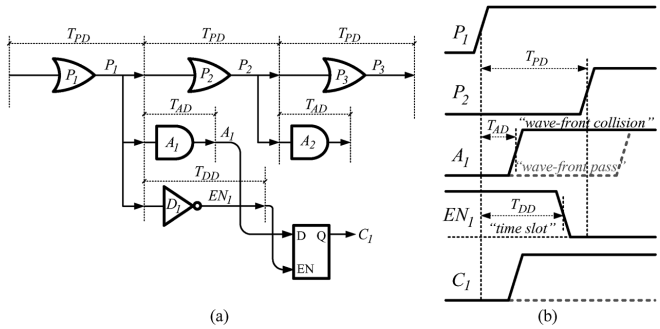

Fig. 6. The timing analysis of the propagation chain: (a) schematic of a 1D array illustrating timing relations, (b) timing diagram (dashed traces of A_1 and C_1 show the rejection of the "wave-front pass" case).

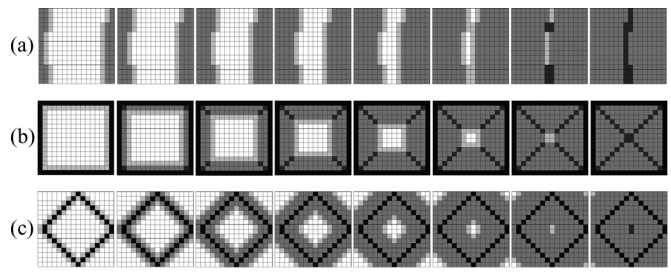

Fig. 7. The operation of the propagation and collision detection mechanisms in different cases: (a) the collision of two irregular parallel wave-fronts (the resulting collision line is of one or two pixels width depending on whether there was odd or even number of cells between colliding wave-fronts), (b) square (the proposed mechanism correctly recognises the collision points), (c) 45° rotated square (the proposed collision detecting mechanism does not recognize collisions because all the cells in the wave-front have at least one inactive (white) neighbor). The gray scale color map used in the figure: white □—inactive (charged) cell, light gray ▫—cell that receives the propagation signal, medium grey ■—activated (discharged) cell, dark grey ■—cell that detects collision, black ■—cell triggered from the marker.

assume that the propagation space is set to the full array, and the binary input image will correspond to the logic states of markers m used to trigger propagation.

B. Collision-Detecting Gate

In the CNN-based methods [21], [28], or in the iterative thinning approach [22], the execution of the skeletonization algorithm resembles the mechanism of propagation, which terminates shortly before the wave-fronts meet. In order to inhibit the propagation before the collision, each cell needs to monitor the state of its neighborhood within at least two pixels radius. In the solution proposed in this paper, however, we allow the wave-fronts to meet and annihilate, and employ a separate logic circuit detecting collisions, based on the states of the cell and its four nearest neighbors. If the logic state of all four neighbors of a particular cell turns to "1," it may mean that two independent wave-fronts met. However, it could also mean that a wave has simply passed by this cell. Therefore, a logical AND operation on the state of the neighboring cells, determines only the necessary but not sufficient condition for the collision occurrence, given by the equation:

$$A = p_N \wedge p_E \wedge p_S \wedge p_W \tag{2}$$

Signal A in (2) indicates collision before or quickly after the cell becomes activated. On the other hand, when a single wave is passing through the cell, the signal A in (2) will indicate only the "wave-front pass" condition. Therefore, it is critical to determine the specific time when the collision condition is valid and should be saved. The logic diagram of the proposed cell with the propagation and collision-detecting mechanisms is presented in Fig. 5. The corresponding diagram illustrating the timing relations between signals is presented in Fig. 6. For simplicity, a 1D array (i.e., a chain of the propagation gates) is analyzed.

When the propagation gate P_1 of the first cell in the chain receives a signal from its preceding neighbor (or from the marker m), it sets the propagation bit P_1 to the high state after a certain delay time T_{PD}. This indicates the activation state of this cell denoting the beginning of the "time slot" when the collision condition in (2) is valid. If the collision condition is met (i.e., all the neighbors of A_1 are in the high state), A_1 turns to the high state after a certain propagation delay T_{AD}. The state of the signal A_1 determines the state for the collision bit C_1. The corresponding D-latch remains "transparent" when the signal EN_1 is in the high state. This signal is generated by the inverting delay gate D_1 and turns to the low state after the delay time T_{DD}. This terminates the "time slot" and latches the value of A_1. As the propagation progresses, all the inputs of the AND gate receive signals from their neighbors, however, the value of the respective collision bit C_1 can only be modified during the "time slot" T_{DD} [Fig. 6(b)]. For correct operation, it must be ensured that: $T_{AD} < T_{DD} < T_{PD}$. It is important to note that, in the dedicated VLSI hardware implementation, the operation of such structure will be sensitive to process parameter variation (mismatch) and noise. As a result, the neighboring cells may not produce correct logic states right at the beginning of the time slot, but slightly later with some random time offset. For that reason, the delay time T_{DD}, controlled by the bias voltage V_{DELAY}, should be adjusted experimentally to assure correct extraction of the unbroken collision lines of at most two pixels width, depending on whether there is an even or odd number of pixels in between parallel wave-fronts (see Fig. 17). The exemplar case showing the collision of two such waves propagating from the opposite directions is presented in Fig. 7(a).

The proposed solution is similar to iterative thinning, but far more simplified. Instead of a set of matching templates, we define only one—logical AND of the nearest neighborhood state. However, the use of only one template may occasionally lead to confusions, especially for non-frontal collisions, where at the meeting point of two wave-fronts, not all the neighboring cells are activated. Two cases of the propagation with non-frontal collisions triggered from edges of a square and a 45° degrees rotated square, are shown in Fig. 7(b) and 7(c) respectively. In the first case, collisions are detected correctly because all the neighboring cells of the points where two perpendicular wave-fronts

meet are activated at the same time. In the second case, however, the perpendicular waves collide, but the wave-fronts are 45° angled to the cell lattice, therefore each cell has at least one inactive neighbor, where the collision should normally be detected. Nevertheless, despite the simplicity of the proposed method, it produces satisfactory results, especially for natural objects of irregular shapes, as shown in Section IV. Most importantly, it can be carried out asynchronously, further improving the result quality and array performance. In a synchronous implementation, each of the (1) and (2) is executed repeatedly for each pixel in the image assuming one iteration per clock cycle.

C. Circuits

The realization of the proposed asynchronous processing module (APM), based on the schematic diagram from Fig. 5 and using standard CMOS logic gates, is possible but rather inefficient due to the large area occupied. Therefore, similarly to our previous works in [14], [17], [18], dynamic logic design has been chosen for compact circuit realization. The schematic diagram of the proposed propagation and collision-detecting cell is shown in Fig. 8. It is functionally equivalent to the structure from Fig. 5, assuming that the transitions of signals p_N, p_E. p_S and p_W are always from "0" to "1" (which is the case here). The size of each transistor is given as the ratio of the channel width to the channel length in micrometers. The sizes of the transistors were chosen to address the leakage and parameter mismatch issues discussed further in this section.

The circuit is realized using a two-phase dynamic logic approach, therefore, each cell has to be initialized before the array can perform any operation. During the initialization phase, signal V_P (precharge) is set to V_{DD} discharging the parasitic capacitances of nodes P and C through M_6 and M_{24}, respectively, and the capacitances of NOR and $NAND$ nodes are precharged to V_{DD} through M_8 and M_{22}. After that, signal V_P is set to "0" which terminates the precharge phase and the array is ready to process the input image (i.e., the array is able to carry out one propagation cycle). The sizes of the keepers M_8 and M_{22} were optimized in simulations to minimize the short-circuit currents during switching and assure that the leakage currents of the corresponding pull-down networks will not affect the state of the circuit after initialization.

If any of the input signals turns to a high state (the cell receives the propagation signal), the node NOR discharges turning on M_7 and setting the propagation bit P to V_{DD}. If all the inputs turn to the high state, the node $NAND$ can be discharged depending on the state of the signal EN (*enable*). This signal is generated by the inverting delay gate with delay time controlled by voltage V_{DELAY}. As long as P remains in a low state, transistor M_{14} pulls up the signal EN to V_{DD} "enabling" the AND-Latch gate. This enables the latch to "follow" the output A of the AND gate. The high state of this signal (when all the inputs p_N, p_E, p_S, and p_W are in a high state) discharges the node $NAND$ setting the collision bit C to a high state. Once this node is discharged, the output state cannot be changed until the next initialization cycle. Such a limitation does not inhibit the cell from proper operation because all the signals received from the neighbors can change only once (from the low to the high logic state).

A critical parameter of the circuit is the duration of the time slot defining the time the AND-Latch gate remains transparent

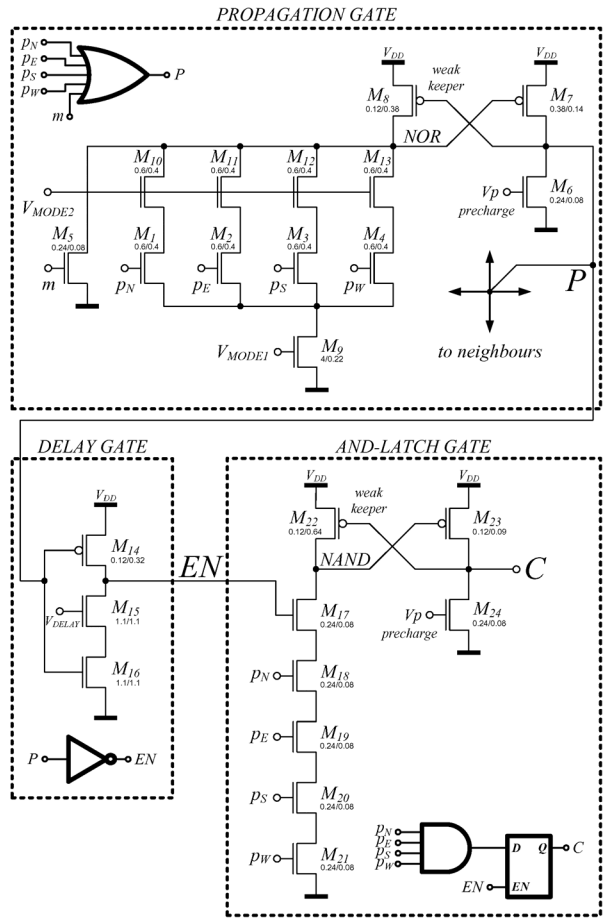

Fig. 8. Schematic diagram of the APM cell consisting of propagation gate, delay gate, and AND-Latch gate (transistor dimensions W/L are given in micrometers).

for the signal A [Fig. 6(b)]. When the signal P turns to a high state (denoting the beginning of the time slot) the output capacitance of the delay gate (node EN) starts discharging. While the discharge time depends on the current controlled by the transistor M_{15}, the time slot length can be tuned using voltage V_{DELAY} biasing the gate of this transistor.

It was observed that the fabrication mismatch is one of the major contributors affecting the operation of the circuit and degrading the quality of the extracted images. It introduces the variability of the timing parameters of the array, which are critical in terms of the correct extraction of the collision lines. Matching between devices can be improved by transistor scaling (enlarging). The obtained improvement, however, comes at the price of increased silicon area and consumed power caused by leakage and increased junction, gate and node capacitances. Therefore, only the critical transistors in the APM cell, which have the greatest effect on the timing parameter variability, were scaled in order to keep the design area small. It was observed that the propagation speed uniformity across the array depends mostly on matching between transistors in the propagation gate whereas the precision of the generated time slot can be improved by proper scaling of the transistors in the delay gate (Fig. 8). In particular, the evaluation transistors M_{1-4}, M_7, and M_{16}, and the current limiting transistors (biased from the analogue voltage sources) M_{9-13} and M_{15}, are dominant contributors to the performance degradation. The mismatch optimization of the APM was performed based on

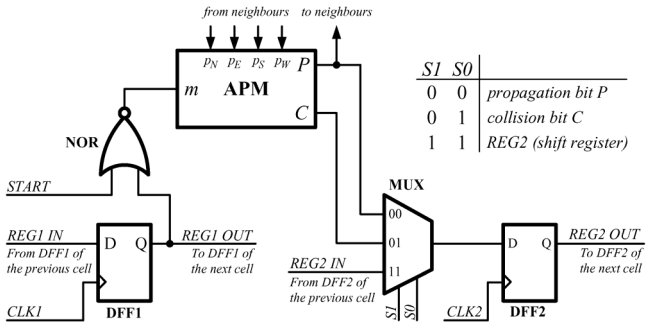

Fig. 9. Schematic diagram of the processing element with APM and I/O logic.

the simulation results of a reduced size array using statistical Monte Carlo MOS transistor models provided by the foundry. The goal was to eliminate the artefacts in the resulting images caused by the random variability of the timing parameters of the cells in the array. Transistors M_{1-5}, M_{10-13}, and $M_{15,16}$ in the actual circuit were oversized due to the available space in the APM layout. Furthermore, the employment of the output split inverter (OSI) in the design of a delay gate, proposed in our previous works [18], [26], rather than the current starved inverter, significantly reduced the variability of the critical timing parameters of the array (time slot) without additional area increase. The variability of the timing parameters of the AND-Latch gate does not affect the operation of the circuit, therefore the weak keeping transistor and the output stage were sized to assure that the gate operates correctly whereas transistors M_{17-21} in the NAND pull-down network were slightly widened only to speed up the gate switching process when the collision condition is met.

D. Implementation

A test array consisting of 64 × 96 APM cells was designed and fabricated in a standard 90 nm CMOS technology. In order to test the array, each cell accommodates also a synchronous I/O circuit consisting of two D-type flip-flops, a multiplexer and two logic gates. The schematic diagram of the processor used in the implementation of the test array is shown in Fig. 9. The input image bit is stored in DFF1, and if it is set to "0" (image background), it triggers the APM on the falling edge of the global *START* signal. The APM generates two signals P and C corresponding to the propagation and collision bits respectively. Depending on the state of lines $S1$ and $S0$ (see table in Fig. 9), either bit P or C can be saved in DFF2. In the array, both D flip flops are serially connected with their respective neighbors and form two separate shift registers REG1 and REG2. The register REG1 is used to shift the input image into the array (one bit per processor). The second register (REG2) can be used to send the captured result off the chip when both signals $S1$ and $S0$ are in the high state. The rising edge of the clock signal $CLK2$ is used to capture the value of bit P or C in DFF2 after or during the processing, which enables the observation of intermediate results of propagation. The design of the processing cell (including the APM, I/O logic, and signal routing) in the array occupies 12.5 μm × 12.5 μm; the layout of the cell is shown in Fig. 10. The proposed APM cell consists of 24 MOS transistors and occupies 5.5 μm × 7.4 μm which is less than the area of three D-type flip-flops designed in this technology.

The additional inverter in the bottom right corner is, by default, inactive (the output is floating and the input is tied to "0").

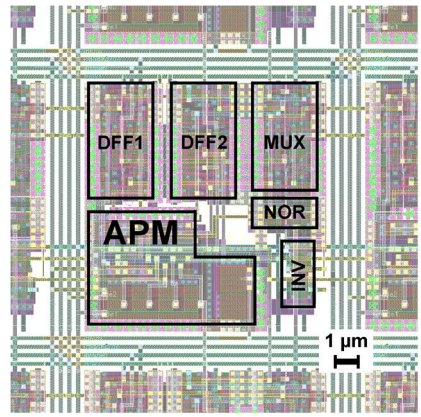

Fig. 10. The layout of the processing cell including the APM and I/O logic (three last top metal layers not shown, core area: 12.5 μm × 12.5 μm).

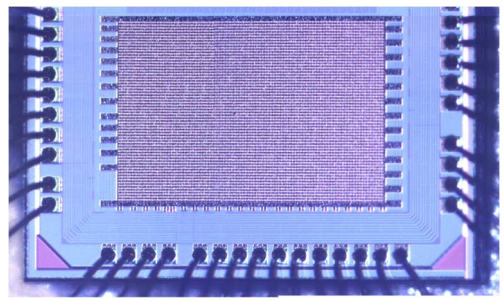

Fig. 11. Micrograph showing the prototype array of 64 × 96 processing cells (size: 840 μm × 1200 μm).

In some cells, these inverters were used as buffers in the build of the signal distribution networks assuring the uniform propagation times of *START* and *CLK2* signals. These networks are based on the "H-pattern" routing topology, assuring equal distances between the input pads of the chip and all clock inputs of DFF1 and DFF2 in the array. The uniformity of the falling edge of the *START* signal is critical for correct operation of the array and affects the quality of the obtained skeletons. Proper distribution of the CLK2 signal is also important when images of the intermediate states of the array are required. The chip micrograph, showing the test array consisting of 64 × 96 processing cells occupying total area of 840 μm × 1200 μm, is presented in Fig. 11.

IV. Results

A. Test System

A test system was designed to generate the programming sequences and control signals for the fabricated chip and to provide communication with a PC (Fig. 12). The design is based on the KCPSM3 (Xilinx PicoBlaze) controller with I/O interfaces and RAM memories implemented on a Spartan 3 XC3S200 FPGA chip. For communication, the RS-232 serial interface was chosen due to its simplicity and sufficient speed for this particular test application. The program stored in the ROM of KCPSM3 is a command interpreter working in a text mode, executing commands received from the host, and sending back the results. In particular, the program memory manager module (PMM) can be accessed and any user's program can be uploaded to the separate 18 × 1 k PRAM memory (program RAM), and executed by switching between PRAM and ROM. This allows one to develop and debug the software for KCPSM3 controller "online" without repeating the synthesis and implementation

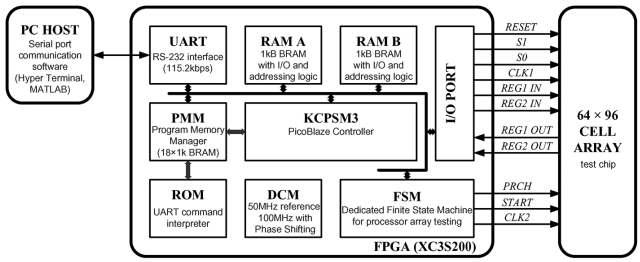

Fig. 12. Block diagram of the test system.

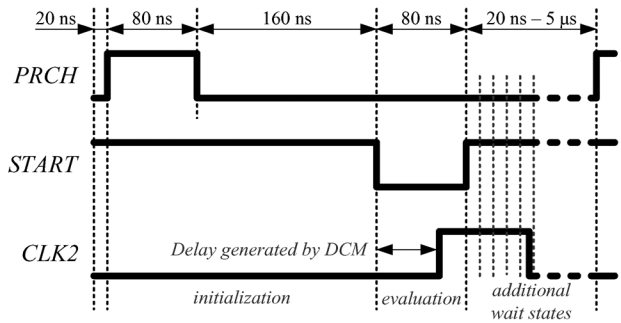

Fig. 13. Time diagram showing the initialization and evaluation sequences generated by the dedicated FSM.

steps. Also, a dedicated MATLAB application was built to simplify the image data exchange and visualization. The two 1kB memory blocks, RAM A and B, are used to store the binary input and output (result) images respectively.

The reset and I/O sequences controlling *REG1* and *REG2* registers are generated entirely by the KCPSM3 by reading and writing to particular pins in the generic I/O PORT module. The initialization and evaluation sequence for the array is generated by a dedicated finite state machine (FSM) unit and shown in Fig. 13. For the system clock frequency 50 MHz, the minimum time resolution of the FSM is 20 ns. It starts the cycle by bringing all the lines to their initial state (first 20 ns), precharges the array (80 ns) and generates *START* signal to begin propagation. The inserted delay of 160 ns after the *PRCH* falling edge before the *START* falling edge is necessary due to ringing observed on the power rails affecting the extracted skeletons. The duration of the evaluation phase (when the *START* signal is in low state) is fixed to 80 ns, which is sufficient for the array to finish the propagation. The rising edge of the *CLK2* signal is used to capture the processed image to the register *REG2*. In order to observe intermediate results of the propagation, an additional circuit generating the rising edge of *CLK2* signal shifted in phase with reference to the system clock was designed and implemented using the phase shifter from the dedicated DCM module on the FPGA. It enables the shifting of the rising edge of *CLK2* within the entire evaluation phase with about $+/-75$ ps time resolution. After the evaluation phase, the FSM can generate from 1 to 256 wait states of 20 ns each before it returns to the initialization state. This enables the control of the length of the generated cycle from 360 ns to 5.46 μs.

B. Experimental Results

The operation of the fabricated chip was verified in a laboratory environment using the test system described above and a set of voltage regulators providing required biases and power supplies (see Table I). The bias voltages V_{DELAY}, V_{MODE1} and V_{MODE2} were adjusted to assure circular contours of the

TABLE I
THE SUPPLY AND BIAS VOLTAGES USED FOR TESTING THE FABRICATED CHIP

Parameter	Value	Remarks
V_{DD}	930 mV	Power supply voltage (APMs)
V_{CC}	930 mV	Power supply voltage (control and I/O logic)
V_{DELAY}	400 mV	Delay gate bias voltage
V_{MODE2}	510 mV	Propagation gate bias voltage
V_{MODE1}	320 mV	Propagation gate bias voltage
V_{IO}	2.5 V	I/O supply voltage of the test chip

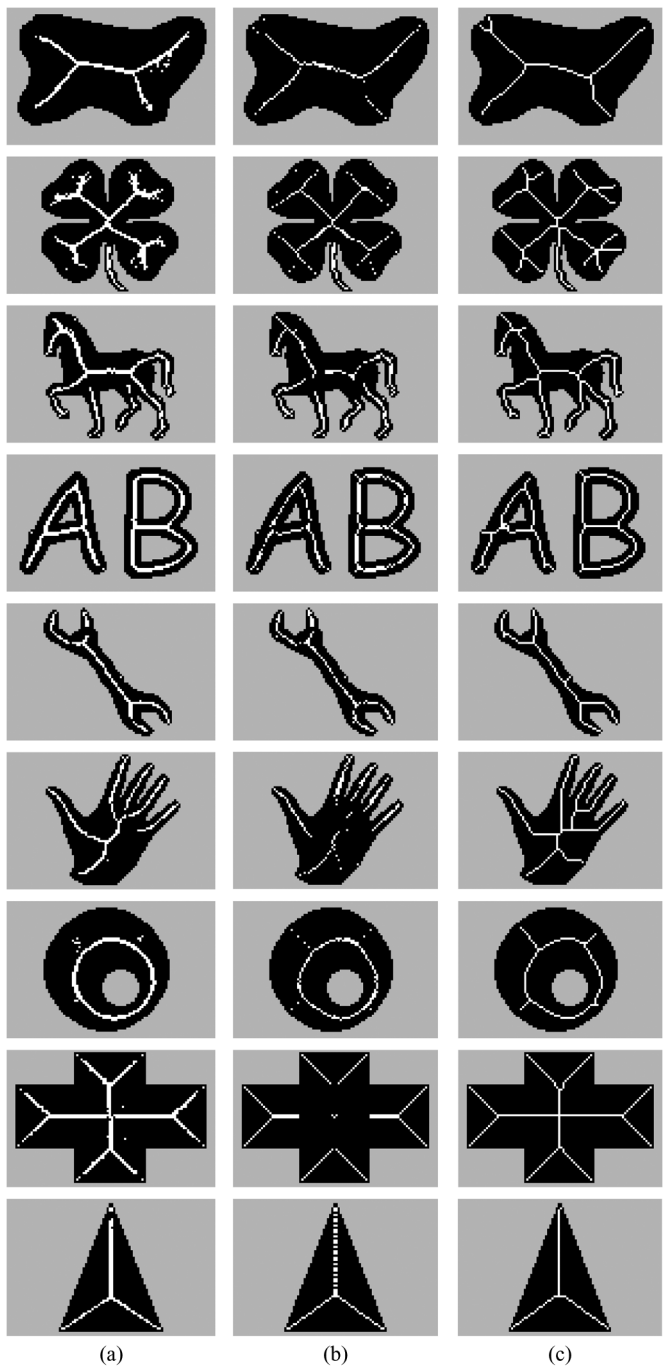

Fig. 14. The skeletons of different objects extracted by: (a) the asynchronous array on the test chip, (b) the synchronous implementation of the propagation and collision detection algorithm, (c) the iterative thinning method.

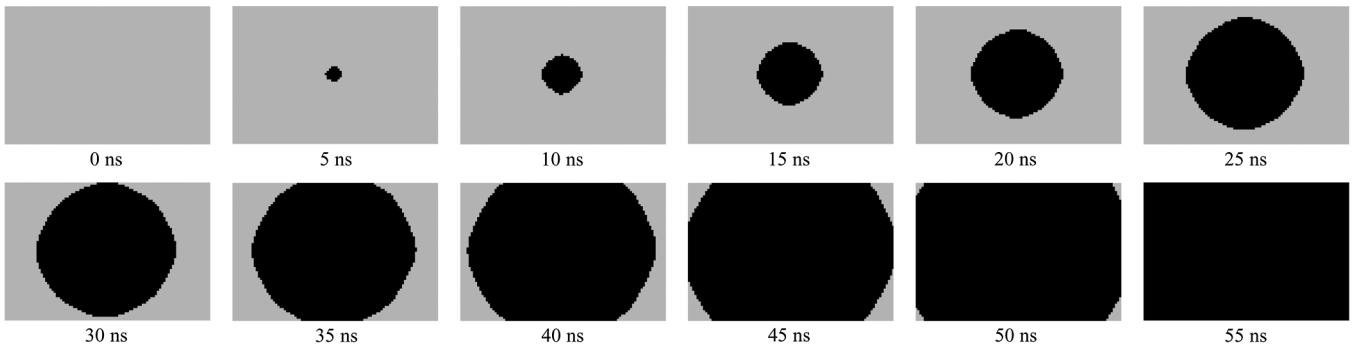

Fig. 15. The intermediate states of the propagation wave triggered from a single pixel in the middle of the array and captured after each 5 ns.

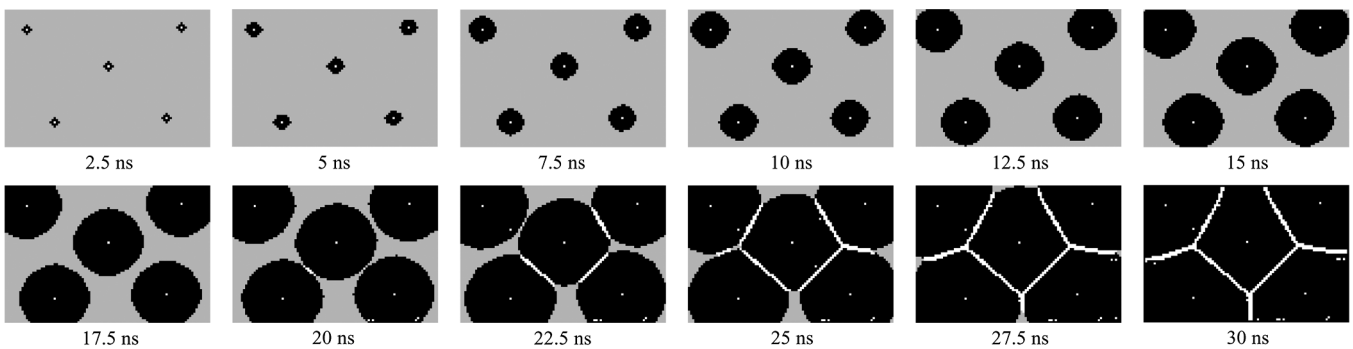

Fig. 16. The intermediates steps of the propagation and collision detection captured after each 2.5 ns showing the generation of the Voronoi diagram.

propagation waves [25] and the best quality of the extracted collision lines. In practice, only V_{DELAY} and V_{MODE2} voltages will require individual tuning, whereas for the remaining ones, standard core supply voltages can be used.

The skeletons obtained from the designed asynchronous array were compared with the results from two skeletonization algorithms. The first algorithm is the implementation of the propagation and collision detection mechanisms, but executed synchronously. The second algorithm is more complex and extracts skeletons based on the iterative thinning using 8 structuring-element pairs [27]. Its software realization form Image Processing Toolbox for Matlab (*bwmorph* function called with a parameter *skel*) is used here as reference. The skeletons of several "natural" objects and geometric shapes, extracted by the fabricated circuit and computed using the aforementioned algorithms, are presented in Fig. 14.

The differences in skeletons extracted by iterative thinning [Fig. 14(c)] and the software realization of the propagation-based method [Fig. 14(b)] result mainly from the simplicity of the proposed mechanism. The differences between the synchronous (software) and asynchronous implementations of the propagation-based method result from more circular contours of the trigger-waves generated in hardware. In a synchronous implementation, a wave propagates with a constant cell-to-cell speed and it takes it twice as much time to reach the nearest cell on the diagonal direction than the nearest cell in the cardinal direction. Therefore, the contour of the wave-front triggered from a single cell resembles a 45° rotated square (diamond, see Fig. 4) indicating anisotropic wave propagation In the asynchronous realization, the diagonal cell receives signals from two nearest neighbors simultaneously, which shortens the propagation time through this cell. As a result, the contour of the trigger-wave is closer to circular as shown in Fig. 15 indicating isotropic propagation. The bias voltages V_{MODE1} and V_{MODE2} can be used to tune the timing parameters of the propagation gate, and hence, the shape of the generated wave-fronts [25]. The effects of the circular propagation in the extraction of a Voronoi diagram are presented in Fig. 16.

The quality of the obtained skeletons and tessellation diagrams depends on the V_{DELAY} voltage, regulating the delay time slot for each cell in the array (see Fig. 6). The experimental results showing skeletons of the same objects extracted for different V_{DELAY} bias voltages are presented in Fig. 17. For lower values of V_{DELAY} the generated time slot becomes longer and cells may start to misclassify the "wavefront pass" condition as a collision. As a result, the detected collision lines become wider and, due to the fabrication mismatch, can also be surrounded by some "loose" pixels resembling the presence of noise. For higher values of V_{DELAY}, the extracted collision lines become fractured and eventually disappear. Such artefacts, contrary to noise, remain stationary. The presence of random noise has been observed in the circuit but its level is very low and practically does not affect the obtained results. The experiments with the chip used in the measurements showed that the best skeletons can be extracted for $V_{DELAY} \approx 400$ mV. It can be observed in Fig. 17 that the variability of the bias voltages below $+/-10$ mV has practically negligible impact on the quality of the obtained results.

The operation of the prototype array was also verified for different supply voltages V_{DD} and different temperatures. In the experiments, bias voltages V_{DELAY}, V_{MODE1}, and V_{MODE2}, and the supply V_{CC} were constant (see Table I). The results of the V_{DD} variability within range $+/-100$ mV, corresponding to the relative variability of $+/-10\%$, are presented in Fig. 18. The supply voltage affects the propagation speed which decreases for smaller and increases for higher values of V_{DD}. As a result, the time slot of the collision detecting mechanism, will either be too short or too long, resulting in incomplete skeletons

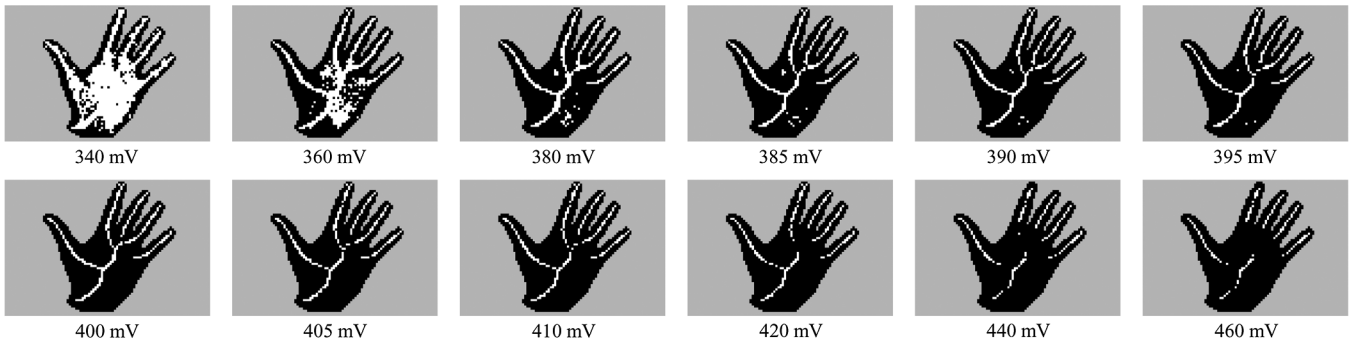

Fig. 17. The skeletons of four images extracted for different V_{DELAY} bias voltages from 340 mV to 460 mV.

Fig. 18. Skeleton extracted for supply voltages V_{DD} equal: (a) 830 mV, (b) 930 mV (nominal value), (c) 1030 mV.

Fig. 19. Skeleton extracted assuming constant bias and supply voltages for temperatures: (a) $T = 25\ ^{\circ}\mathrm{C}$ (room temperature), (b) $T = 40\ ^{\circ}\mathrm{C}$, (c) $T = 60\ ^{\circ}\mathrm{C}$.

[Fig. 18(a)] or excessive number of artefacts [Fig. 18(c)] in the extracted images. This can be fixed by proper adjustment of the V_{DELAY} voltage. In the experiments, when the array worked at $V_{DD} = 830$ mV and 1030 mV, the correct skeleton results (the same as for the nominal bias and supply conditions from Table I) could be obtained for V_{DELAY} equal 370 mV and 430 mV respectively. The effects of the temperature variation on the quality of the obtained results are presented in Fig. 19. The operation of the chip was tested in three temperatures: 25 °C (room temperature), 40 °C, and 60 °C assuming constant bias and supply voltages from Table I. In the experiments it was observed that the effects of temperature variability are less severe and can easily be compensated by the V_{DELAY} bias voltage adjustment. In particular, correct skeletons were obtained for $V_{DELAY} = 410$ mV and 420 mV for the temperatures 40 °C and 60 °C respectively.

C. Design Issues

The quality of the obtained results is mainly affected by the voltage oscillation (ringing) on the power supply rail. This occurs after a larger group of cells switches at the same time, i.e., during the precharge phase, and right after the falling edge of the *START* signal triggers the propagation from the markers. In order to reduce the impact of these oscillations, additional banks of capacitors were used on the test PCB, and also the additional delay of 160 ns was inserted to assure that the power rail settles before the evaluation phase. Ringing caused by the falling edge of the *START* signal cannot be easily reduced. It modulates the propagation speed of the array, affecting the operation of the collision detecting circuit. The effects of the propagation speed variability are shown in Fig. 20. It can be observed that triggering from all the background pixels increases the oscillation of the power rails, which impedes the correct recognition of the collisions. One way of reducing this undesirable effect is to reduce the number of marker pixels and trigger the propagation only from the object's boundary. This requires additional logical operations on the input and output images, such as binary edge detection and masking, but these can easily be performed by a standard SIMD processor. Also, proper adjustment of the voltages V_{MODE1} and V_{MODE2} helped to reduce the propagation speed and further decrease the ringing effects on the supply rails (see Table I).

Fig. 20. The skeletons of the same object extracted for: (a) wave triggered from all the background pixels, and (b) wave triggered only from the pixels on the border of the image.

In the designed test array the border line cells, unlike the rest, trigger only two or three nearest neighbors. Due to the lower output load of these cells the propagation speed increases along the borders. The intermediate steps of the propagation wave triggered from a column of pixels on the left hand side of the array are presented in Fig. 21. It can be observed that the cells located along the borders propagate faster triggering other cells inside the array. As a result, the wave front bends towards the borders. This could easily be fixed by inserting an additional set of dummy gates around the array balancing the load, or by adding a mechanism to control the propagation space and switching off the border line pixels. It should also be noted that transistors M_{17-21} in the pull-down network of the AND-LATCH gate (Fig. 8) have slightly different gate capacitances due to the body effect resulting from the serial connection and a common substrate potential. Such systematic asymmetry can affect the propagation speed and make it direction dependent. For example, the wave contour in Fig. 21 bends faster at the bottom side than at the top side of the array (i.e., the wave propagates faster to the northeast than to southeast direction). This, however, has a minor effect on the quality of the obtained results as long as transistors M_{17-21} are much smaller than M_{1-4}.

D. Performance and Power

The performance of the prototype array was verified using the dedicated FSM module generating repetitive sequences for the

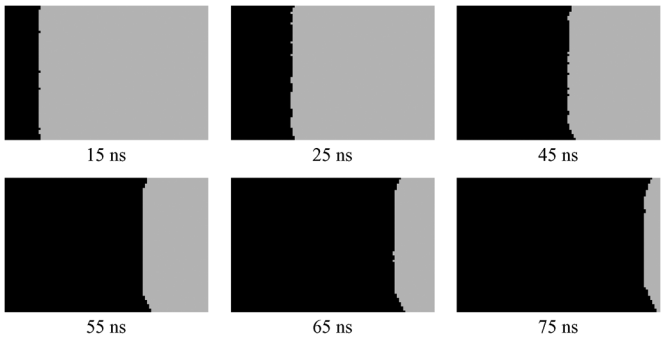

Fig. 21. Bending of the propagation wave contour caused by the unbalanced load of the border cells.

TABLE II
PERFORMANCE AND POWER RESULTS OF THE DESIGNED ARRAY

Input image	I_{APM} @ 2.78 MHz	I_{APM} @ idle
white	0.20 mA	0.31 mA
single dot	1.75 mA	0.31 mA
full size rectangle	1.78 mA	0.32 mA
black	1.95 mA	0.43 mA

initialization and evaluation cycles. The shortest sequence (with no additional wait states), takes 360 ns and the longest one (with 256 wait states) 5.46 μs. In the power estimation, only the current I_{APM}, supplying the APMs, was considered. The correct operation of the prototype chip was verified for constant supply and bias voltages at room temperature (see Table I). During the performance tests, the obtained results were saved to the register REG2 (on the rising edge of *CLK2*) without shifting it out. Only the result captured during the last cycle, was transmitted off chip for verification. For power measurements, four different input images were used: *white* (no markers), *single dot* in the middle (only one marker), *full size rectangle* (markers on the boundary of the array), and *black* (all pixels are markers). The operation of the array was tested at the maximum processing speed (2.72 MHz) and in the idle state. The obtained results are presented in Table II.

In the case of the white input image, there is no propagation triggered in the array. It can be observed, however, that the corresponding current consumption is lower during the operation of the array (0.20 mA) than in the idle state (0.31 mA). This is caused by leakage currents affecting the initial state after the initialization cycle. In particular, nodes P are slowly pulled up through transistors M_7, which eventually triggers a "spontaneous" propagation discharging the array. After such discharge, the state of the array settles depending on the balance between the leakage currents of the transistors. Therefore, to assure low power operation in the idle state, signal *DSCH* should be kept at high logic state. This limits the total leakage current of the APM modules to about 0.20 mA ($\sim$33 nA/pixel). The differences of the current consumption observed during the maximum processing speed, for the last three images in Table II, result mainly from a different number of collision pixels detected in each of the images. Each time a collision is detected, the respective AND-Latch gate is discharged, which requires an additional amount of energy to overcome the weak keeping transistor M_{22} (Fig. 8). Therefore, when the processed image generates the collision pixels, the corresponding supply current will be higher. The black input image was used to measure the worst case of the power consumption when all the pixels are markers, and hence

TABLE III
POWER, PERFORMANCE AND DESIGN PARAMETERS OF THE TEST ARRAY

Parameter	Value	Remarks
Technology	CMOS 90 nm	Dynamic logic full custom
No. transistors APM	24	Sized accordingly to reduce effects of mismatch
APM size	5.5 µm × 7.4 µm	Less than three D flip flops
Processor size	12.5 µm × 12.4 µm	Including APM and I/O
Array size	840 µm × 1200 µm	64 × 96 processor array
Propagation speed	1.1 pixel / ns	1.4 pixel/ns for V_{MODE1} and V_{MODE2} set to V_{DD}
Current consumption (idle state)	33 nA / pixel 50 nA / pixel 70 nA / pixel	*DSCH* in high state propagation propagation and collision
Current consumption (max. speed)	285 nA / pixel 317 nA / pixel	propagation propagation and collision
Processing speed	2.78 Mfps	With no quality degradation
Power consumption	< 1mW / 1 Mfps	> 200 kfps
Min. power consumption	0.2 - 0.4 mW	< 200 kfps, depending on no. of markers in the image

TABLE IV
PERFORMANCE OF THE APM ARRAY AND RELATED HARDWARE REALIZATIONS

Reference	ANN [29]	ACLA [17]	FPGA [16]	CNN [28]	APM
Implementation	0.7 µm	0.35 µm	xc3s1500	180 nm	90 nm
Array size	22 × 26	24 × 60	16 × 16	4 × 4	64 × 96
PE size [µm^2]	140 × 120	16 × 8	48 slices	12.5 ×12.5	5.5 × 7.4
Speed [ns/pixel]	300	0.48	20.6	4	0.91
Energy [pJ/pixel]	35	0.4	---	0.190	0.16

collisions are detect by all the cells in the array. The respective supply current during the operation is 1.95 mA. Also the supply current in the idle state increases due to the increased leakage of the AND-Latch gate when the collision bit C is set to "1" ($\sim$70 nA/pixel). In terms of power, the APM array consumes 1.81 mW at the maximum speed and 0.19 mW in the idle state. The I/O circuit consumes 0.47 mW of constant power and is dominated by leakage currents. The main design parameters and the power performance of the prototype chip measured at 25 °C are presented in Table III. The maximum processing speed of the designed array is mainly limited by ringing on the power rails, requiring an additional 160 ns delay between the slopes of the *PRCH* and *START* signals. Also, the propagation speed of about 1.1 pixels/ns typically requires less than 60 ns time to complete processing of an image. In the experiments, a delay of maximum 80 ns was assumed for testing. At such speed, the array performs 2.78 million initialization and evaluation cycles per second. In practice, the performance will be limited by other factors, such as off chip data transfer. In many typical applications, the design can benefit from a low processing power $<$1 nW/fps. A comparison with related designs is presented in Table IV. In particular, the realizations from [16] and [29] implement a complete skeletonization algorithm whereas [17] and [28] can perform binary wave propagation only.

V. CONCLUSIONS

In this paper, the idea and design of an asynchronous processor array for binary image processing using wave propagation concept have been presented. The experimental results confirmed the correct operation of the proposed circuit capable of processing up to 2.78×10^6 images per second consuming less than 1 nJ/image. The proposed design could be of use as a power-efficient co-processing unit in SIMD architectures and vision chips.

REFERENCES

[1] A. Zarandy, *Focal-Plane Sensor-Processor Chips*. New York: Springer, 2011.

[2] A. N. Belbachir, *Smart Cameras*. New York: Springer, 2010.

[3] A. Lopich and P. Dudek, "A SIMD cellular processor array vision chip with asynchronous processing capabilities," *IEEE Trans. Circuits. Syst. I, Reg. Papers*, vol. 58, no. 10, pp. 2420–2431, Oct. 2011.

[4] M. Ishikawa, A. Morita, and N. Takayanagi, "High speed vision system using massively parallel processing," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Jul. 1992.

[5] J. E. Eklund, C. Svensson, and A. Astrom, "VLSI implementation of a focal plane image processor—A realization of the near-sensor image processing concept," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 4, no. 3, pp. 322–335, Sep. 1996.

[6] W. C. Zhang, Q. Y. Fu, and N. J. Wu, "A programmable vision chip based on multiple levels of parallel processors," *IEEE J. Solid-State Circuits*, vol. 46, no. 9, pp. 1–16, Sep. 2011.

[7] P. Dudek and P. J. Hicks, "A CMOS general-purpose sampled-data analog processing element," *IEEE Trans. Circuits Syst. II, Analog Digit. Signal Process.*, vol. 47, no. 5, pp. 467–472, May 2000.

[8] J. Poikonen, M. Laiho, and A. Paasio, "MIPA4k: A 64 × 64 cell mixed-mode image processor array," in *Proc. ISCAS*, May 2009.

[9] A. Rodriguez-Vazquez *et al.*, "ACE16k: The third generation of mixed-signal SIMD-CNN ACE chips towards VSoCs," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 51, no. 5, pp. 851–863 , May 2004.

[10] R. Carmona-Galan *et al.*, "A bio-inspired two-layer mixed-signal flexible programmable chip for early vision," *IEEE Trans. Neural Netw.*, vol. 14, no. 5, pp. 1313–1336, Sep. 2003.

[11] A. Rodriguez-Vazquez, S. Espejo, R. Dominguez-Castro, J. L. Huertas, and E. Sanchez-Sinencio, "Current-mode techniques for the implementation of continuous- and discrete-time cellular neural networks," *IEEE Trans. Circuits Syst. II, Analog Digit. Signal Process.*, vol. 40, no. 3, pp. 132–146, Mar. 1993.

[12] A. Paasio, M. Laiho, A. Kananen, and K. Halonen, "An analog array processor hardware realization with multiple new features," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, May 2002, vol. 2, pp. 1952–1955.

[13] K. Chen, J. Leu, and N. Gershenfeld, "Analog logic automata," in *Proc. BioCAS*, 2008, pp. 189–192.

[14] P. Dudek, "An asynchronous cellular logic network for trigger-wave image processing on fine-grain massively parallel arrays," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 53, no. 5, pp. 354–358, May 2006.

[15] A. Astrom, R. Forchheimer, and J. E. Eklund, "Global feature extraction operations for near-sensor image processing," *IEEE Trans. Image Process.*, vol. 5, no. 1, pp. 102–110, Jan. 1996.

[16] A. Lopich and P. Dudek, "Hardware implementation of skeletonization algorithm for parallel asynchronous image processing," *J. Signal Process. Syst.*, vol. 56, no. 1, pp. 91–103, Jul. 2009.

[17] A. Lopich and P. Dudek, "Asynchronous cellular logic network as a co-processor for a general-purpose massively parallel array," *Int. J. Circuit Theory Appl.*, Apr. 2010.

[18] P. Mroszczyk and P. Dudek, "Trigger-wave collision detecting asynchronous cellular logic array for fast image skeletonization," in *Proc. ISCAS*, May 2012, pp. 2653–2656.

[19] C. Rekeczky and L. O. Chua, "Computing with front propagation: Active contour and skeleton models in continuous time CNN," *J. VLSI Signal Process. Syst. Signal, Image, Video Technol.*, vol. 23, no. 2–3, pp. 373–402, 1999.

[20] V. Perez-Munuzuri, V. Perez-Villar, and L. O. Chua, "Autowaves for image processing on two-dimensional CNN array of excitable nonlinear circuits: Flat and wrinkled labyrinths," *IEEE Trans. Circuits Syst. I, Fundam. Theory Appl.*, vol. 40, no. 3, pp. 174–181, Mar. 1993.

[21] H. Blum, "A transformation for extracting new descriptors of shape," in *Proc. Symp. Models Perception Speech Vis. Form*, Cambridge, MA, USA, 1967, pp. 362–380, MIT Press.

[22] V. I. Krinsky, V. N. Biktashev, and I. R. Efimov, "Autowaves principles for parallel image processing," *Physica D*, vol. 49, pp. 247–253, 1991.

[23] E. R. Davies, *Machine Vision: Theory, Algorithms, Practicalities*. Cambridge, U.K.: Cambridge Univ. Press, 1990.

[24] L. Lam, S. W. Lee, and C. Y. Suen, "Thinning methodologies—A comprehensive survey," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 14, no. 9, pp. 869–885, Sep. 1992.

[25] P. Mroszczyk and P. Dudek, "Trigger-wave propagation in arbitrary metrics in asynchronous cellular logic arrays," in *Proc. ECCTD*, Sep. 2013, pp. 1–4.

[26] P. Mroszczyk and P. Dudek, "Tunable CMOS delay gate with improved matching properties," *IEEE Trans. Circuits Syst. I., Reg. Papers*, vol. 61, no. 9, pp. 2586–2595, 2014.

[27] R. M. Haralick and L. G. Shapiro, *Computer and Robot Vision*. Reading, MA, USA: Addison-Wesley, 1992, vol. 1.

[28] J. Flak, M. Laiho, A. Paasio, and K. Halonen, "Dense CMOS implementation of a binary-programmable cellular neural network," *Int. J. Circuit Theory Appl.*, vol. 34, pp. 429–443, 2006.

[29] M. Olah, P. Masa, and A. Lorincz, "A mixed-signal VLSI circuit for skeletonization by grassfire transformation," in *Proc. ICANN '97*, pp. 1205–1210.

Przemysław Mroszczyk received his five-year Masters (mgr inż) degree in electronic engineering from the University of Science and Technology (former AGH), Kraków, Poland, in 2010, and the Ph.D. degree from the University of Manchester, U.K., in 2014. He is currently working as a Research Associate at the University of Manchester (School of Electrical and Electronic Engineering, Microelectronics Design Lab). His research work is in the field of massively-parallel, fine-grain mixed-mode, and asynchronous CMOS VLSI processor arrays for fast and power efficient computation with applications in computer vision and machine learning.

Piotr Dudek (S'98–M'01–SM'06) received his Masters (mgr inż) degree in electronic engineering from the Technical University of Gdańsk, Poland, in 1997, and the M.Sc. and Ph.D. degrees from the University of Manchester Institute of Science and Technology (UMIST), U.K., in 1996 and 2000, respectively. He worked as a Research Associate, and since 2002 as a Lecturer at UMIST/The University of Manchester. In 2009 he was a Visiting Associate Professor in the Department of Electronic and Computer Engineering at the Hong Kong University of Science and Technology. Since 2011 he has been a Reader in the School of Electrical and Electronic Engineering, University of Manchester, leading the Microelectronics Design Lab. His research interests are in the area of integrated circuit design, cellular processor arrays, vision sensors, novel computer architectures, and brain-inspired systems.

Dr. Dudek is a member of IEEE Circuits and Systems Society Technical Committees on Cellular Neural Networks and Array Computing, Neural Systems and Applications, and a Chair-Elect of the CAS Sensory Systems Technical Committee. He has been a member of scientific committees of several international conferences.