Asynchronous Approximation of a Center of Gravity for Pixel Detectors' Readout Circuits

Piotr Otfinowski, *Member, IEEE,* Grzegorz W. Deptuch, *Senior Member, IEEE*, and Piotr Maj, *Member, IEEE*

***Abstract*— This paper presents an implementation of asynchronous approximation of a center of gravity of a binary object on a focal plane of a pixel detector. The direct field of its application is dealing with charge sharing in processing of signals from semiconductor X-ray hybrid pixel detectors. The developed algorithm is called the center of gravity in a temporal object (COGITO), standing for approximation of a geometrical COGITO that is a charge cloud sampled by a pixel detector. Its operation resembles image processing for finding a center of gravity of a binary object in an image. The presented circuitry operates entirely in the digital domain—the analog pixel front end is not included. Thus, this paper does not show a complete, self-consistent solution to X-ray or particle detectors. The key details of the concept and its implementation, followed by measurements obtained with a test chip designed in a 55-nm CMOS process, are presented. The algorithm can deal with arbitrarily large objects and allocation of a hit to a single pixel can be achieved in a few tens of nanoseconds.**

***Index Terms*— Asynchronous circuits, charge sharing, dynamic vision sensors, focal-plane tracking, pixel detectors, single photon counting.**

I. Introduction

DIVISION of charge clouds generated in the detector active volume between two or more readout pixels is called charge sharing [1]–[3]. It may lead to duplicated or missed registrations of incident X-ray photons. These problems become significant especially in small pixel pitch or thick detectors. Therefore, the existence of charge sharing is one of the limiting factors in decreasing the pixel's size in radiation imaging systems [4], [5].

Several attempts to deal with charge sharing have been undertaken by a few design groups. The solutions known from [6]–[12] differ in some details; however, their common underlying principle is to compare analog properties of the signals of pixels involved in a charge shared event, find a pixel with the largest amount of charge, and assign an event to it.

Two approaches are known that differ by the time required getting the comparisons resolved and the allocation of the registration accomplished. The first method [6]–[8] compares times-over-threshold features of the processed signal and requires about 2.02 μs for registering a single radiation event [8], while the second method [10]–[12] compares amplitudes of signal pulse and allows registration of an event typically in about 1.01 μs [11].

It is worth noticing that both prior methods target effectively estimation of a center of gravity of a group of pixels that received and processed fractional charge signals resulting from a charge shared event. Unquestionably, a direct comparison of analog signals seems to be an intuitive way for a precise estimation of a pixel where collection of charge has peaked. On the other hand, such an approach leads to an increased circuit complexity and requests a significant amount of the analog functionality, which usually needs calibration and compensation of circuit imperfections resulting from process mismatches.

Leaving direct comparisons of individual signals from neighboring pixels behind, as indeed an intuitively unbiased way of finding a pixel with a maximum charge collected, it is possible to notice that such a pixel typically lies in the center of a group of pixels that collect fractional charges from a charge shared event. Thus, no comparisons are needed if only a delineated group of pixels that received fractional signal can be identified, because a procedure of finding a pixel that should register an event can be nothing else but a version of the iterative minimization procedure known from image processing. This fundamental statement leads to a conclusion that the postulated functionality can entirely be implemented in the digital domain.

A first realization of the proposed functionality is disserted in this paper. The design of an integrated circuit implementing the algorithm for an asynchronous approximation of a center of gravity in a temporal object, named COGITO, is presented. The designed circuit first builds a bordered group of pixels that have participated in a charge shared event. It keeps such a group growing while more pixels have their signal above threshold. Then, it processes such a group when all participating pixels have their individual signals below a threshold again. Finally, it starts the minimization procedure that ends with one pixel in the center of the group, where an event gets eventually allocated. In order to achieve low processing times, high-energy efficiency, and avoid building clock distribution

Manuscript received June 5, 2017; revised August 22, 2017 and December 18, 2017; accepted December 31, 2017. Date of publication February 1, 2018; date of current version April 23, 2018. This paper was approved by Associate Editor David Stoppa. This work was supported in part by the National Science Center under Contract DEC-2014/13/B/ST7/01168 and in part by the Fermilab (operated by Fermi Research Alliance, LLC) under Contract DE-AC02-07CH11359 with the U.S. Department of Energy. *(Corresponding author: Piotr Otfinowski.)*

P. Otfinowski and P. Maj are with the Department of Measurement and Electronics, AGH University of Science and Technology, 30-059 Krakow, Poland (e-mail: potfin@agh.edu.pl; piotr.maj@agh.edu.pl).

G. W. Deptuch is with the Department of Measurement and Electronics, AGH University of Science and Technology, 30-059 Krakow, Poland, and also with the Fermi National Accelerator Laboratory, Particle Physics Division, ASIC Development Group, Electrical Engineering Department, Batavia, IL 60510 USA (e-mail: deptuch@agh.edu.pl).

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/JSSC.2018.2793530

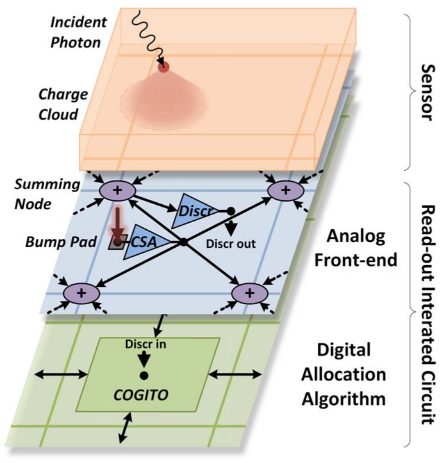

Fig. 1. Schematic of a hybrid pixel detector with the COGITO algorithm.

trees, the logic responsible for the minimization procedure is fully asynchronous.

Versatility of the presented solution lies in the fact that it accepts binary information about simultaneous occurrence of an event, occupying certain area of an array and compacts this information to a single pixel located as close as possible to its center of gravity. It does not require any additional clock or strobe signals. The analog front end and following digital back-end blocks can be application-specific, what is particularly attractive considering the latest advances in 3-D integration technologies [13]. Moreover, applications of the presented solution are not limited to X-ray imaging; it can also be adopted in particle detection systems. A charged particle, e.g., traversing electron or alpha particle, can be detected by a cluster composing of 4×4 readout channels or more. Also, when charge collection is achieved through thermal diffusion, not through drift, e.g., in MAPS, cluster sizes reach 5×5 pixels. The algorithm can be implemented directly in the sensor readout circuit, allowing for very fast image processing in various other applications, e.g., focal-plane processing [14], [15] or dynamic vision sensors [16], [17].

This paper is organized in five sections. Section II introduces the principles of the COGITO algorithm operation and its functional blocks, while Section III contains its implementation details. Section IV and V present the prototype integrated circuit architecture, the measurement methodology, and the measurement results. Finally, Section VI summarizes the presented method and the obtained results.

II. COGITO Algorithm Features

A. Analog Front End

The COGITO algorithm operates mainly in the digital domain. The analog signal processing is limited to a front-end charge sensitive amplifier (CSA), a summing circuit, and a discriminator, which output is the only signal connecting the digital and analog domains, as presented in Fig. 1.

The presented method relies on analog signal summing. Each pixel sums CSA output signals coming from four pixels: from itself and its three neighbors, located at its upper side (N), its upper-left corner (NW) and its left-hand side (W). This approach effectively quadruples the area of a single pixel. It allows building a signal as if the total amount of charge was collected by such a virtual larger pixel, when the charge is divided among 2, 3, or 4 adjacent pixels, at the cost of the doubled noise level seen at the discriminator's input, but fully preserving the signal magnitude, whereas the latter may be decreased up to four times in the typical approach, without charge summing.

In addition, looking at Fig. 1, it can be noticed that summing virtually shifts the signal in the lower right (SE) direction, from the center of a single pixel to the corner between the four pixels from which the charge is summed.

B. COGITO Algorithm Overview

The operation of the COGITO algorithm is fully asynchronous and is based only on the signals coming from the discriminators, i.e., no external clock or strobe signals are required. The operation comprises three phases.

1) *Maximization Phase:* Pixels collecting a charge cloud are identified and logically organized into one bordered group, hereinafter called an "object."
2) *Minimization Phase:* The object formed in the preceding phase undergoes iterative minimization, i.e., shrinks toward its center.
3) *Resolution Phase:* A single pixel estimating the center of gravity of the object is selected.

The maximization phase starts when a discriminator in any pixel crosses a threshold. If at that time another discriminator in one of eight neighboring pixels crosses a threshold too, this neighboring pixel also joins the newly created object. The object keeps growing and expanding over adjacent pixels, whose discriminators cross a threshold. Eventually, when there are no new pixels and discriminators of all pixels already being a part of the object return below the threshold, the maximization phase is finished.

The maximization phase realizes an equivalent function of peak detector for the object in terms of its size. Pixels are added to an object as their discriminators go above a threshold, and the object's state is frozen at its largest size, before advancing to the next phase.

The algorithm begins the minimization phase if the object's size is larger than 2×2 pixels. During this phase pixels are forced to leave an object in a specific order, beginning with pixels at its boundary and moving toward the center of an object. Each pixel asynchronously checks its position within an object and—if it meets certain conditions (explained below)—it generates a request signal to leave an object. After all its neighboring pixels respond with the acknowledge signals, the pixel is removed from an object. The remaining pixels reevaluate their new positions and the process continues.

The algorithm enters the resolution phase when the object's size is equal to or smaller than 2×2 pixels. Such size is chosen, because it correlates with the area from where the signal is summed. During this phase, a pixel approximating the center of gravity of the evaluated object is found and afterward all pixels forming the object return to the idle state.

The presented algorithm can handle objects of any size and shape. Pixels forming a single object may be connected by

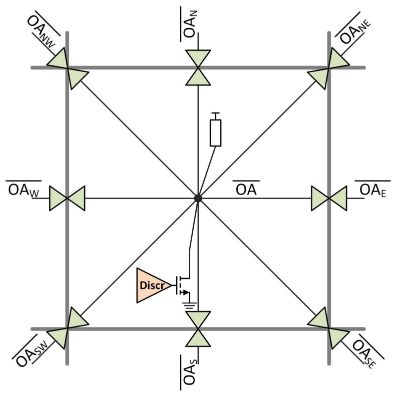

Fig. 2. Implementation principle of the OA signal.

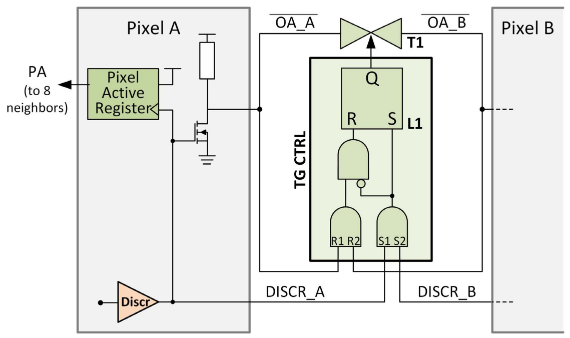

Fig. 3. Partial OA signal logic between two neighboring pixels.

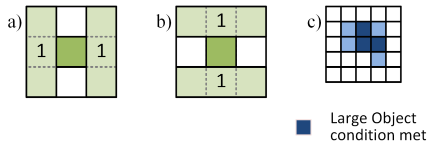

Fig. 4. LO condition. (a) and (b) Test patterns. (c) Example of an object.

their edges or corners. Many different objects may exist at the same time in a detector, but they must be separated by at least one pixel not being a part of any object; otherwise, they are merged into one. This also implies that a pixel may be a part of only one object at a time.

III. COGITO—Detailed Algorithm Implementation

A. *Maximization Phase*

The purpose of the maximization phase is building of an object. This phase lasts as long as at least one pixel in the object has its discriminator still over a threshold. The last discriminator in an object returning below the threshold ends the maximization phase and all pixels in this object should have information about this fact available. The signal carrying this information, hereinafter called Object Active (OA), should be compliant with a dynamic character of forming of objects of any size and shape and should have the minimal propagation delay. Also, the OA signal cannot be implemented straightforward in a CMOS logic as each pixel should be able to both receive and send the OA signal from and to any of its neighbors.

The chosen way to implement the OA signal is a wired-or-logic, active low. Each pixel has its separate OA signal, with a pull-up resistor and a pull-down driver. The latter activates when a discriminator is over a threshold and the OA signal is being locally pulled down. Furthermore, OA signals between adjacent pixels are connected by means of transmission gates (T-gates). The idea is depicted in Fig. 2.

When two adjacent pixels belong to an object, the T-gate between them is enabled and their OA signals become connected together. Then, it is enough that only one pixel has its discriminator over a threshold to pull the OA signal down for all pixels in an object. When discriminators of all pixels in an object return below a threshold, the OA signal gets pulled-up by the resistors in the entire object at the same time, as it is kept connected by T-gates. Therefore, a rising OA signal edge marks the end of the maximization phase.

A simplified logic circuit, controlling T-gates at each pixel border, is presented in Fig. 3. The T-gate T1 is enabled when both discriminators are over a threshold at the same time as the RS latch L1 is set by a logical conjunction of two discriminator signals. The T-gate T1 is kept enabled until the OA signals of both pixels become inactive (high level) as a logical conjunction of two OA signals resets the RS latch L1.

In addition to the OA signal, each pixel has an internal pixel active register (PA). Its purpose is to maintain the pixel state through the following phases of the algorithm operation, after the OA signal is deactivated. If a pixel has its PA register set, it means that this pixel belongs to an object. The PA register is set by the rising edge of the discriminator signal, during the maximization phase, and it is cleared through the algorithm operation in one of the following phases. The state of the PA register is propagated to all eight neighboring pixels via a CMOS logic, as it is shown in Fig. 3.

B. *Minimization Phase*

During the minimization phase, an object shrinks. Shrinking means that pixels leave an object in a specified order, by resetting their PA registers. This phase continues until an object fits into the size of 2 × 2 pixels.

In order to measure an object's size, each pixel forming an object monitors PA registers' states of its eight immediate neighbors and matches them with a specific pattern, in a way that is presented in Fig. 4. This pattern is given by the condition, hereinafter called the large object (LO) condition that requires the following.

1) At least one of the three neighbors on the left-hand side and at least one of the three neighbors on the right-hand side have their PA register set [Fig. 4(a)].
2) At least one of the three neighbors on the upper side and at least one of the three neighbors on the lower side have their PA register set [Fig. 4(b)].

Meeting of the LO condition by a pixel can be interpreted that it is adjacent to at least two pixels that are not immediate neighbors, translating to the conclusion that the size of such object is larger than 2×2 pixels. In Fig. 4(c), an example of an object is presented, where the pixels meeting the LO condition are marked in dark blue.

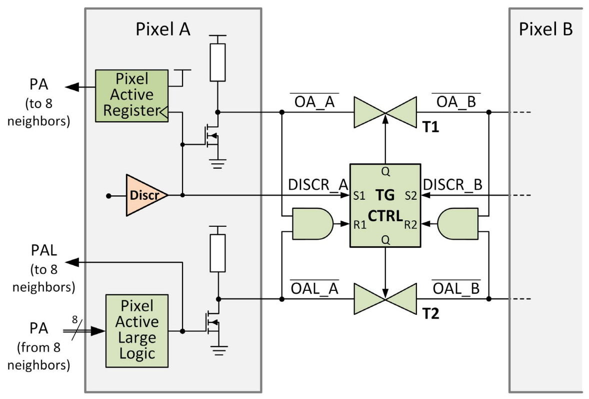

Fig. 5. Complete OA and OAL signals logic at the boundary between two pixels.

To indicate that a pixel meets the LO condition, the PA large (PAL) signal is introduced. This signal is set by each pixel asynchronously, by a combinational logic.

If any pixel in an object has the PAL signal active, the minimization phase shall continue. This information has to be available to all pixels of an object, as in the case of the OA signal during the maximization phase. In order to realize this functionality, the OA Large (OAL) signal is introduced. It is implemented in the same way as the OA signal, i.e., in the active low wired-or-logic, with signals of adjacent pixels being connected by T-gates. Each pixel which has its PAL signal set pulls the OAL signal down.

The complete logic managing an object's state, present at each pixel border, is depicted in Fig. 5. Both T-gates T1 and T2, for the OA and OAL signals, are enabled at the same time, when the discriminators of two pixels are over a threshold simultaneously. The T-gates remain enabled as long as either the OA or OAL signal is active (low level). The PAL signal is passed to all eight neighboring pixels in the CMOS logic.

As long as the OAL signal is active, pixels located on the outside boundary of an object are to progressively leave it. Each pixel asynchronously checks its location, basing on the PA and PAL signals from its eight neighbors. A pixel chosen to leave an object must meet the three conditions, hereinafter called the leave conditions. The first leave condition requests that as follows.

1) A pixel's three neighbors in the row above it and the neighbor on the right-hand side must have their PA register cleared.
2) A pixel's three neighbors in the row below it and the neighbor on the left-hand side must have their PA register cleared.

This situation is depicted in Fig. 6(a) and (b), respectively, where the neighbors involved in the condition are indicated as "0." Fig. 6(c) presents an example of an object, in which pixels meeting the first leave condition are marked in dark blue.

The first leave condition results in directional thinning of an object. The thinning direction is not chosen arbitrarily, but it correlates with the virtual signal shift due to the summing, shown in Fig. 1, and it is transversal to it.

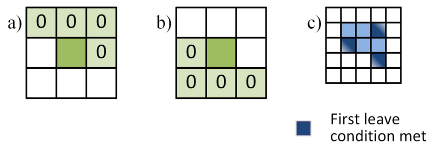

Fig. 6. First leave condition. (a) and (b) Test patterns. (c) Example of an object.

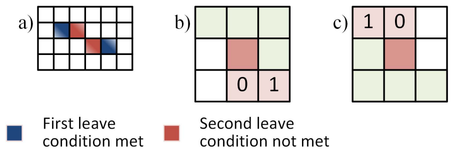

Fig. 7. Second leave condition. (a) Example of an object. (b) and (c) Test patterns.

It can be noticed that applying the first leave condition alone can result in splitting of an object into two parts in some cases. An example of such a situation is shown in Fig. 7(a). The pixels marked in red, despite of the fact that they fulfill the first leave condition, should not be allowed leaving an object. This leads to the introduction of the second leave condition, which complements the first one and requires that as follows.

1) A pixel meeting the first leave condition must not have its lower right neighbor PA register set and lower neighbor PA register cleared.
2) A pixel meeting the first leave condition must not have its upper-left neighbor PA register set and upper neighbor PA register cleared.

These complementary conditions are depicted in Fig. 7(b) and (c), respectively.

The third leave condition blocks pixels from leaving an object when its size fits into 2×2 pixels, by requiring that a pixel must have at least one neighbor with PAL signal set. This condition is redundant to the condition for entering the minimization phase, which bases on the OAL signal. However, it is necessary to make the system independent of the delay caused by the deactivation of the OAL signal, which is larger compared to how fast the pixels can leave the object. Without the third condition an object would be completely destroyed before the algorithm would proceed to the last phase, i.e., the resolution phase.

A pixel meeting all the leave conditions generates the leave request (LR) signal, but does not leave an object instantaneously. Before it can do so, it needs to check if any of its neighbors is not selected to leave too. To deal with such a situation, when two adjacent pixels generate the LR signals simultaneously, asynchronous arbiters are used to determine the precedence [18], [19]. An arbiter circuit is located between each pair of neighboring pixels, as it is shown in Fig. 8.

A pixel meeting the leave conditions sends its LR signal to eight arbiter circuits at its boundary ("REQ$_X$ out"). In return, it is allowed leaving an object only when it receives acknowledge signals ("ACK$_X$ in") from all eight arbiters. In addition, when a pixel leaves an object, all adjacent pixels must have their PAL values reevaluated before another pixel can leave too.

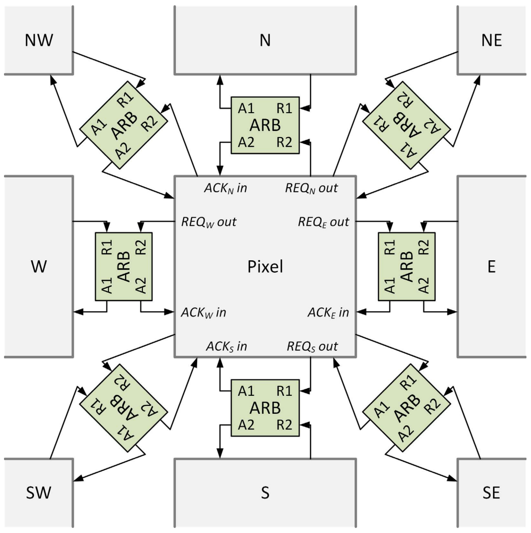

Fig. 8. Simplified connections of asynchronous arbiters between pixels (labels of signals at the diagonals are omitted for clarity).

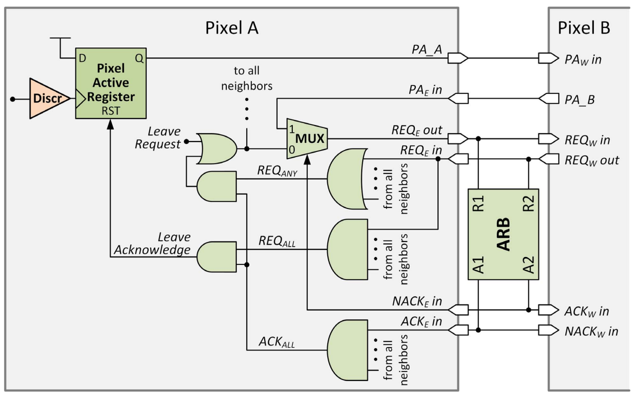

Fig. 9. Schematic of the minimization algorithm asynchronous logic between two neighboring pixels at the E–W border (selected to serve as an example).

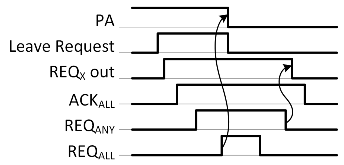

Fig. 10. Leave procedure timing diagram.

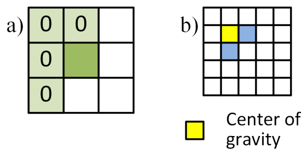

Fig. 11. Condition for pixel approximating the center of gravity. (a) Test pattern. (b) Example of an object.

Otherwise, pixels could generate LR signals too early and leave an object in a wrong order, which would result in splitting an object into two parts or deactivating it completely. Therefore, after a pixel has left an object, it blocks its neighbors from leaving until they recalculate their new locations.

The complete procedure for a pixel to leave an object is based on a four-phase asynchronous handshake protocol. Each pixel uses both arbiter's request and acknowledge signals at all its eight borders. Therefore, such a configuration makes a pixel aware of its neighbors' states. The detailed schematic of the logic responsible for the pixel leaving process at a single border is shown in Fig. 9.

An example of a timing diagram of the leave procedure is depicted in Fig. 10. If a pixel meets all the leave conditions, in the first phase, it generates the LR signal and sends it to all neighbors and to the adjacent arbiters. The LR signal passes through a multiplexer (MUX) in a pixel. In the idle state and if a pixel is pointed by an arbiter to leave, the MUX's output "REQ_X out" carries the LR signal value. Otherwise, when a pixel receives a "$NACK_X$ in" signal from an arbiter, meaning that its corresponding neighbor is leaving an object, a pixel is disallowed to leave and the LR signal is overwritten by its neighbor's PA register state ("PA_X in")—the purpose of such behavior is explained below.

Next, the pixel waits for eight "ACK_X in" signals from arbiters and additionally for eight "REQ_X in" signals, reflecting its own PA register state seen by its eight neighbors. Only after it receives these signals ("ACK_{ALL}" and "REQ_{ALL}") it can leave the object by resetting its PA register ("Leave Acknowledge"). However, to prevent adjacent pixels from subsequently leaving an object, it does not clear its "REQ_X out" signals yet.

In the second phase, as long as at least one of the "REQ_X in" signals is set ("REQ_{ANY}"), a pixel keeps the "REQ_X out" signals set and locks the arbiters in the same state, blocking the adjacent pixels from leaving. Finally, when all adjacent pixels clear their "REQ_X out" signals, meaning that they have recalculated their new locations, a pixel which has already left an object also clears its "REQ_X out" signals and the leave procedure for this pixel is finished.

C. Resolution Phase

When the object size is equal to at most 2×2 pixels, the pixel approximating the center of gravity is chosen as the one in the upper-left corner of the remaining object. This condition is chosen in a way to oppose the virtual signal shift, caused by the signal summing, shown in Fig. 1.

The condition whether a pixel is positioned at the upper-left corner is tested by each pixel asynchronously, by checking if PA registers' states of its four neighbors—at its upper, upper-left, left-hand and lower left side—are equal to "0," as it is presented in Fig. 11(a). The pixel decided to approximate the center of gravity of the object example from Fig. 11(b) is marked in yellow.

Regardless of the minimization phase occurred or not, one of the OA or OAL signals is eventually pulled-up. A logical conjunction of these two signals is computed and during its rising edge the pixel meeting the condition is registered as algorithm's result, i.e., being the center of gravity.

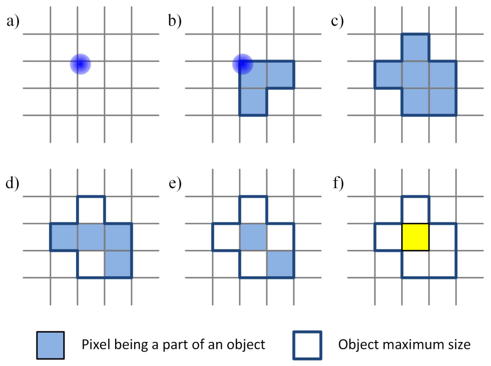

Fig. 12. Example of the algorithm operation. (a) Occurrence of an event, (b) start of the maximization phase, (c) final object, (d) minimization phase, (e) end of the minimization phase, and (f) pixel estimating the center of gravity found.

D. Concise Flow of the COGITO Algorithm Operation

The flow of the COGITO algorithm is illustrated in Fig. 12. A single cycle of the algorithm operation proceeds as follows.

1) An event occurs in a sensor and charge is collected.
2) Discriminators in pixels cross a threshold. The maximization phase starts, an object is created and the OA signal becomes active. Adjacent pixels, whose discriminators have also crossed a threshold, join the object. If the object size exceeds 2×2 pixels, the OAL signal is activated too.
3) When all discriminators in the object return below a threshold, the OA signal becomes inactive. The rising edge of the OA signal marks the end of the maximization phase. If the OAL signal has been activated, the algorithm proceeds to the minimization phase, otherwise it proceeds directly to the resolution phase.
4) The minimization phase starts and pixels begin to leave an object in a specific order.
5) When an object size is equal to or smaller than 2×2 pixels, the OAL signal becomes inactive. Its rising edge marks the end of the minimization phase and the resolution phase commences.
6) A pixel estimating the center of gravity is found among the remaining pixels of an object.

IV. Integrated Circuit Prototype

To verify the COGITO algorithm operation, a prototype integrated circuit has been designed and manufactured in a 55-nm CMOS process. The chip has a size of 1.57×1.57 mm^2 (including bond pads) and it includes an array of 16×16 pixels with 50-μm pixel pitch. Such a size is large enough to cover the realistic sizes of the charge clouds.

The circuit contains only the digital part, responsible for the algorithm operation. The pixel schematic is shown in Fig. 13. A digital MUX is connected to fifteen test signals, imitating discriminators' outputs of the preceding analog front end. One of these signals can be selected as an input in each pixel independently. The test signals are generated off-chip and are distributed to the pixels via a separate bus.

Each pixel has a two-bit counter (CNT), which is incremented each time an event is allocated to the pixel. The pixels

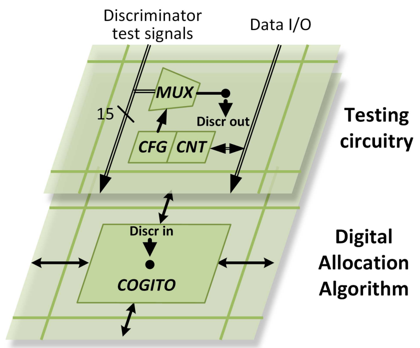

Fig. 13. Schematic of the prototype integrated circuit pixel.

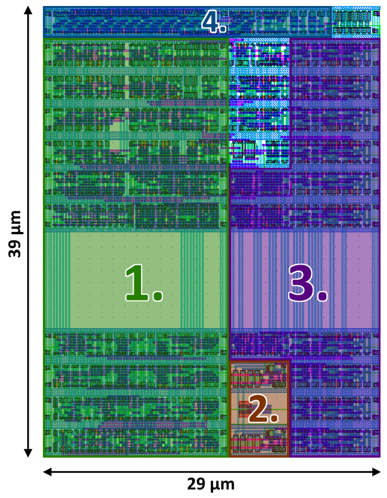

Fig. 14. Layout of a single pixel. 1) COGITO algorithm logic. 2) Pull-up circuits. 3) Configuration registers. 4) Input signal MUX.

also contain a set of R/W registers (CFG), which store the configuration of the MUX's input and temporal state of OA and OAL signals. The data is read from and written to the chip serially, by means of a scan chain.

The pixel logic is supplied from the 1.2-V power supply, while the I/O LVDS drivers use the 1.8-V power rail.

The mask layout of a single pixel, with the specific logic block highlighted, is shown in Fig. 14. Only layers up to the metal three are shown for clarity. The whole pixel logic occupies the area of 29 μm $\times$ 39 μm (including an 8 μm-wide signal routing channel). The logic responsible solely for the COGITO algorithm operation occupy the area of only 16 μm $\times$ 28 μm. The remaining logic is used for the OA signals pull-up circuits, configuration registers, data readout and testing circuitry. The photograph of the chip bonded to the test PCB is shown in Fig. 15.

V. Measurement Results

A. Measurement Methodology

Before each test, first, pixels forming an object of a desired shape were assigned a selected test signal, while the remaining ones had their inputs inactive. Second, a discriminator signal strobe was sent to the selected pixels, initiating the

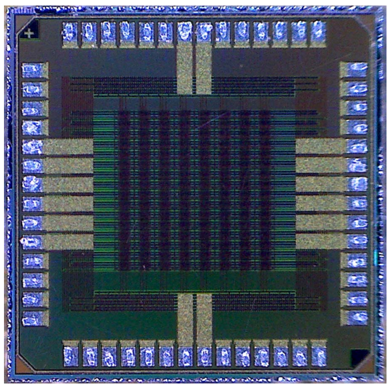

Fig. 15. Photograph of the test chip (1.57 × 1.57 mm^2).

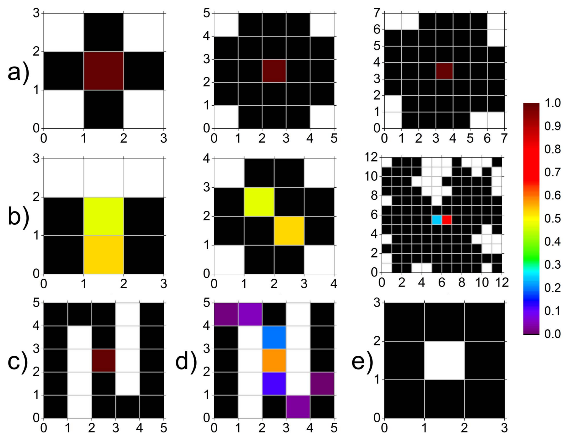

Fig. 16. Examples of tested objects. (a) Objects with odd number of pixels across. (b) Objects with even number of pixels across. (c) Long, narrow object, and identical discriminators' signals timing. (d) Long, narrow object, and randomized discriminators' signals timing. (e) Object with enclosed inactive pixel.

object formation. Finally, the results of the algorithm operation were readout from the chip and the pixels selected as the centers of gravity were identified.

B. Center of Gravity Approximation

To verify the operation of the circuitry fulfilling the COGITO algorithm, resolving of various objects was examined. Pixels to form an object were assigned discriminators' signals with random durations, ranging from 20 to 150 ns. The signals had random start times, assuring, however, that all signals overlap for at least 20 ns in order to be able to build the objects. Objects were also created randomly in various positions across the whole array of pixels. Some examples of the measurement result are presented in Fig. 16. The 2-D graphs show the objects, while colors of pixels reflect probabilities of becoming the center of gravity approximation with black and dark red meaning that a pixel was correspondingly never and always selected as the center of gravity, according to the plotted legend. Each graph shows the averaged result of at least 10 000 measurement iterations.

For symmetrical objects with odd number of pixels across, as presented in Fig. 16(a), the algorithm always gave the same results, regardless of the position of the object in the array, and timing of the discriminator signals. For objects, whose center of gravity lied between two pixels, the result depended

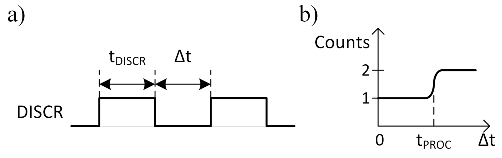

Fig. 17. COGITO algorithm object processing time measurement. (a) Discriminator's signal and (b) obtained S-plot of hit count versus time delay Δt.

on the position of an object in the array, due to different signal propagation times and intrinsic arbiter offsets, caused by mismatch. Such a result was expected. Examples of such objects are presented in Fig. 16(b). Even LOs, constructed on array of 12 × 12 pixels, were processed correctly, with small variation of the center of gravity approximation.

For some fanciful objects in a form of long, narrow lines, the results were less obvious at the first glance. For the sake of some analysis, Fig. 16(c) shows an s-shaped object resolving when identical discriminator signals were applied to all its pixels, while Fig. 16(d) shows the resolving results for the same object when signals were randomized according to the procedure described above. It was noticed that if pixels at one end of an object had their discriminator signals deactivated before other pixels at the opposite end of an object, they were removed from an object earlier, because the OA line could be pulled-up faster in these pixels due to the resistive nature of the transmission gates connecting the OA line between the pixels composing an object. The final effect is visible as occasional shifting of the result toward one of the ends in the s-shape objects instead of yielding exactly its geometrical center of gravity. Such a result is actually beneficial in detector readout circuits, as a shift toward the pixels with longest discriminator signals gets the result closer the center of a radiation event.

With no exception, all probabilities for all objects from Fig. 16(a)–(d) always sum up to 1, which means that always a single pixel was selected as the center of gravity approximation.

Objects similar to the one shown in Fig. 16(e) were found to lead to no results. Such objects are unlikely to occur during the typical operation for the projected applications in single photon-counting pixel detectors; however, they may be caused by non-responsive pixels in the array. One of the possible solutions consists of, first, identifying these dead-pixels and, second, flagging them, by setting an appropriate bit in the configuration register. Flagged pixel would automatically join an object in the maximization phase, by activating its OA signal, if at least its two adjacent neighbors have their OA signal active.

C. Object Processing Times

The benchmark for the algorithm is the time required processing individual objects. In order to measure it, two identical objects were defined, one after another, with a variable delay Δt between their appearances. If the delay was shorter than the processing time, the two objects were merged and a single event was detected, whereas, if the delay between the two objects' creation times was longer than the processing time, two separate events were registered.

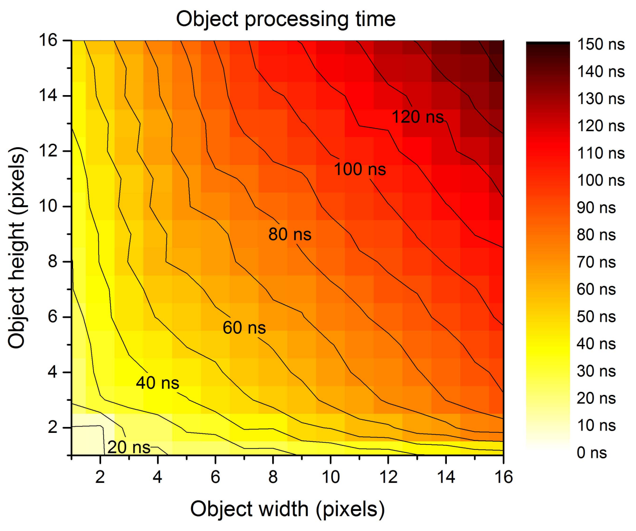

Fig. 18. Measurement results of object processing time versus object size (rectangular shape).

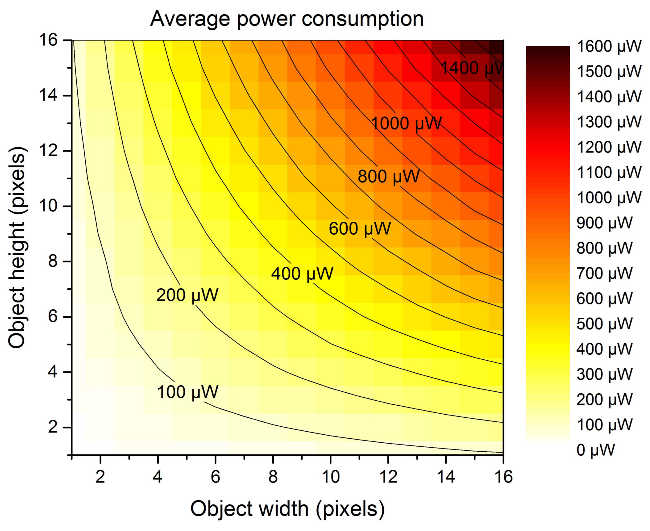

Fig. 19. Measurement results of an average power consumption (entire array—256 pixels) versus object size (rectangular shape) at object occurrence repetition rate of 1 MHz.

In order to measure the minimum Δt resulting in finding two events, two consecutive discriminator's signal strobes were sent to an object. All pixels forming an object were connected to the same test signal. The appropriate discriminator signal is presented in Fig. 17(a). An S-curve was obtained by varying the delay between the strobes. The time at which the S-curve crosses its half-amplitude corresponds to the average processing time t_{PROC} of the given object's shape, as it is explained in Fig. 17(b). The discriminator's strobe signal length was constant, equal to $t_{\text{DISCR}} = 100$ ns and it was not included in the measured processing time.

The processing times for objects of rectangular shapes, with widths and heights ranging from 1 to 16 pixels, were measured. The results are presented in Fig. 18.

The measured processing time varies from 8 ns, for the object size of 2×2 pixels, up to 145 ns, for the size of 16×16 pixels (entire array). Objects with the size of 2×2 pixels or smaller do not require the minimization phase; therefore, their processing time is significantly smaller. Still, the time required processing objects with the sizes of up to 5×5 pixels does not exceed 50 ns. The measurement results allow noticing that the OA and OAL signal pull-up times are on the order of 8 to 9 ns, while a single pixel requires from 3 to 5 ns to leave an object and multiple pixels can be leaving simultaneously. The variation of the given times results from the actual shape of an object and a position of a leaving pixel in an object with respect to its neighbors.

The obtained processing times are significantly lower that the dead times of analog front end. Despite of the fact, which algorithm's operation starts with the falling edge of the discriminator, the processing finishes before the signal reaches its baseline. Thus, the analog front end is the dominant limiting factor in terms of speed.

D. Power Consumption

The power consumption of the circuit implementing the COGITO algorithm was measured under the conditions, where an object of the desired shape was defined with the rate of 1 MHz. These parameters refer to the typical conditions for a single photon counting X-ray detector. All pixels of an object received identical discriminator signals with duration of 100 ns. The data readout was disabled during the measurement.

The power consumption of an entire array, for objects of rectangular shapes with widths and heights ranging from 1 to 16 pixels is presented in Fig. 19. The measured average power consumption ranges from 13.5 μW, for an object consisting of a single pixel, through 20 and 149 μW for objects of 2×2 and 5×5 pixels, respectively, up to 1590 μW for an object of 16×16 pixels. The power consumption per active pixel is approximately equal to 6 μW. As the COGITO algorithm is based entirely on digital signal processing, it has negligible static power consumption.

The reported total static power consumption of the existing circuits ranges from 9.3 μW [5], [6] to 35 μW [11] and 45 μW [10] per pixel. These solutions require—apart from signal summing circuit—additional shaping amplifier and analog comparators [10]–[12] or a low-offset discriminator [6]. These blocks consume about 7 μW or 15% of total power [10], while the energy consumption of the digital part is not explicitly specified.

VI. Summary

This paper presented the implementation details and the measurement results of the method for asynchronous approximation of a center of gravity of 2-D binary objects, COGITO. The algorithm extensively relies on inter-pixel communication and it was developed for the purpose of compensation of charge sharing in single photon counting X-ray hybrid pixel detectors. Nevertheless, the implementation makes it a very universal solution, applicable in any binary image processing system.

The obtained measurement results indicate objects processing times on the order of tens of nanoseconds, while the power consumption is equal to approximately 6 μW per pixel per MHz. The area occupation of the logic implementing the algorithm is equal to 16 μm×28 μm in 55-nm CMOS process, which together with limited required analog circuitry makes the presented method applicable in readout circuits with fast response and fine pixel pitch.

Acknowledgment

The authors would like to thank F. Fahim and Dr. J. R. Hoff from Fermilab, DuPage County, IL, USA, for stimulating discussions and reviewing fundamentals for the presented concept.

References

[1] E. Gatti, A. Longoni, P. Rehak, and M. Sampietro, "Dynamics of electrons in drift detectors," *Nucl. Instrum. Methods Phys. Res. A, Accel. Spectrom. Detect. Assoc. Equip.*, vol. 253, no. 3, pp. 393–399, Jan. 1987.

[2] K. Iniewski, H. Chen, G. Bindley, I. Kuvvetli, C. Budtz-Jorgensen, "Modeling charge-sharing effects in pixellated CZT detectors," in *Proc. Conf. Rec. IEEE Nucl. Sci. Symp.*, Oct./Nov. 2007, pp. 4608–4611.

[3] J. C. Kim *et al.*, "Charge sharing in common-grid pixelated CdZnTe detectors," *Nucl. Instrum. Methods Phys. Res. A, Accel. Spectrom. Detect. Assoc. Equip.*, vol. 654, no. 1, pp. 233–243, Oct. 2011.

[4] L. Rossi, P. Fischer, T. Rohe, and N. Wermes, *Pixel Detectors—From Fundamentals to Applications*. Berlin, Germany: Springer-Verlag, 2006.

[5] R. Ballabriga *et al.*, "Review of hybrid pixel detector readout ASICs for spectroscopic X-ray imaging," *J. Instrum.*, vol. 11, p. P01007, Jan. 2016.

[6] R. Ballabriga *et al.*, "The Medipix3RX: A high resolution, zero dead-time pixel detector readout chip allowing spectroscopic imaging," *J. Instrum.*, vol. 8, p. C02016, Feb. 2013.

[7] T. Koenig *et al.*, "Charge summing in spectroscopic X-ray detectors with high-Z sensors," *IEEE Trans. Nucl. Sci.*, vol. 60, no. 6, pp. 4713–4718, Dec. 2013.

[8] E. Frojdh *et al.*, "Count rate linearity and spectral response of the Medipix3RX chip coupled to a 300 μm silicon sensor under high flux conditions," *J. Instrum.*, vol. 9, p. C04028, Apr. 2014.

[9] R. Bellazzini *et al.*, "PIXIE III: A very large area photon-counting CMOS pixel ASIC for sharp X-ray spectral imaging," *J. Instrum.*, vol. 10, p. C01032, Jan. 2015.

[10] G. W. Deptuch *et al.*, "An algorithm of an X-ray hit allocation to a single pixel in a cluster and its test-circuit implementation," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 65, no. 1, pp. 185–197, Jan. 2018, doi: 10.1109/TCSI.2017.2713767.

[11] A. Krzyżanowska, G. W. Deptuch, P. Maj, P. Gryboś, and R. Szczygieł, "Characterization of the photon counting CHASE Jr., chip built in a 40-nm CMOS process with a charge sharing correction algorithm using a collimated X-ray beam," *IEEE Trans. Nucl. Sci.*, vol. 64, no. 9, pp. 2561–2568, Sep. 2017.

[12] P. Maj *et al.*, "Measurements of matching and noise performance of a prototype readout chip in 40 nm CMOS process for hybrid pixel detectors," *IEEE Trans. Nucl. Sci.*, vol. 62, no. 1, pp. 359–367, Feb. 2015.

[13] G. W. Deptuch *et al.*, "Fully 3-D integrated pixel detectors for X-rays," *IEEE Trans. Electron Devices*, vol. 63, no. 1, pp. 205–214, Jan. 2016.

[14] W. Miao, Q. Lin, W. Zhang, and N. J. Wu, "A programmable SIMD vision chip for real-time vision applications," *IEEE J. Solid-State Circuits*, vol. 43, no. 6, pp. 1470–1479, Jun. 2008.

[15] J. A. Schmitz, M. K. Gharzai, S. Balkir, M. W. Hoffman, D. J. White, and N. Schemm, "A 1000 frames/s vision chip using scalable pixel-neighborhood-level parallel processing," *IEEE J. Solid-State Circuits*, vol. 52, no. 2, pp. 556–568, Feb. 2017.

[16] T. Serrano-Gotarredona and B. Linares-Barranco, "A 128 × 128 1.5% contrast sensitivity 0.9% FPN 3 μs latency 4 mW asynchronous frame-free dynamic vision sensor using transimpedance preamplifiers," *IEEE J. Solid-State Circuits*, vol. 48, no. 3, pp. 827–838, Mar. 2013.

[17] P. Lichtsteiner, C. Posch, and T. Delbruck, "A 128 × 128 120 dB 15 μs latency asynchronous temporal contrast vision sensor," *IEEE J. Solid-State Circuits*, vol. 43, no. 2, pp. 566–576, Jan. 2008.

[18] C. L. Seitz, "Ideas about arbiters," *Lambda*, vol. 1, no. 1, pp. 10–14, 1980.

[19] C. H. V. Berkel and C. E. Molnar, "Beware the three-way arbiter," *IEEE J. Solid-State Circuits*, vol. 34, no. 6, pp. 840–848, Jun. 1999.

Piotr Otfinowski (M'12) received the M.S. degree in electronics and telecommunications and the Ph.D. degree in electronics from the AGH University of Science and Technology (AGH-UST), Krakow, Poland, in 2009 and 2015, respectively.

From 2011 to 2016, he was a Professor's Assistant with the Department of Measurement and Electronics, AGH-UST. His current research interests include low-power analog-to-digital converters for multichannel measurement systems and hybrid pixel detectors.

Grzegorz W. Deptuch (M'98–SM'07) received the M.Sc. degree in electronics from the AGH University of Science and Technology (AGH-UST), Cracow, Poland, in 1996, the Ph.D. degrees in physics-electronics from Université Louis Pasteur, Strasbourg, France, and AGH-UST, in 2002, and the D.Sc. degree in electronics from AGH-UST in 2014.

He has authored or co-authored about 110 publications and one monograph. He holds four U.S. patents. His current research interests include the design of readout integrated circuits for radiation detectors and radiation detectors instrumentation.

Piotr Maj (M'08) has received the master's degree in electrical engineering and the Ph.D. degree in electronics from the AGH University of Science and Technology (AGH-UST), Krakow, Poland, in 2005 and 2008, respectively.

He focused on the readout electronics for silicon strip detectors. He joined the Department of Measurement and Electronics, AGH-UST, in 2007. He was a member of the Microelectronics Group, AGH-UST, where he was involved in the field of hybrid pixel X-ray detectors. Since 2017, he has been an Associate Professor and continues his research in the field of hybrid pixel detectors trying to utilize the 3-D CMOS and deep sub-micrometer CMOS technologies in the goal of developing single photon-counting X-rays detectors.