MIT Open Access Articles

A CMOS Imager With a Programmable Bit-Serial Column-Parallel SIMD/MIMD Processor

The MIT Faculty has made this article openly available. **Please share** how this access benefits you. Your story matters.

Citation: Yamashita, Hirofumi, and Charles G. Sodini. "A CMOS Imager With a Programmable Bit-Serial Column-Parallel SIMD/MIMD Processor." IEEE Transactions on Electron Devices 56.11 (2009): 2534–2545. © Copyright 2009 IEEE

As Published: <http://dx.doi.org/10.1109/ted.2009.2030718>

Publisher: Institute of Electrical and Electronics Engineers (IEEE)

Persistent URL: <http://hdl.handle.net/1721.1/71887>

Version: Final published version: final published article, as it appeared in a journal, conference proceedings, or other formally published context

Terms of Use: Article is made available in accordance with the publisher's policy and may be subject to US copyright law. Please refer to the publisher's site for terms of use.

A CMOS Imager With a Programmable Bit-Serial Column-Parallel SIMD/MIMD Processor

Hirofumi Yamashita, *Member, IEEE*, and Charles G. Sodini, *Fellow, IEEE*

Abstract—An imager with an integrated fully programmable bit-serial column-parallel processor is proposed to meet the demand for a compact and versatile system-on-imager chip for consumer applications. The on-imager processor is targeting a computationally intensive low-level image processing task. The processor is physically arranged as a densely packed 2-D processing element (PE) array at an imager column level. The digital processor has a multiple-instruction-multiple-data (MIMD) architecture configuring multiple column-parallel single-instruction-multiple-data (SIMD) processors. The prototype imager chip with 128×128 pixels and 4×128 PE array designed with $0.6\text{-}\mu\text{m}$ technology was fabricated, and its functionality was tested. The estimation of performance level of the proposed processor architecture with an advanced technology such as the $0.09\text{-}\mu\text{m}$ process technology shows that the proposed imager chip architecture has a potential of giga sum operations per second per square millimeter class processing performance.

Index Terms—Array signal processing, CMOS image sensors, CMOS integrated circuits, image processing, intelligent sensors, parallel processing, programmable circuits.

I. INTRODUCTION

A CMOS imager with an integrated digital image processing is a promising imaging system for consumer applications such as mobile imaging, video conferences, and surveillances, where low cost and small system size are required [1]–[3]. One solution is a full implementation of camera functionality on an imager chip. However, the digital blocks for fully implemented camera function consume considerable amount of silicon area. Since low cost and small size are a crucial demand for a consumer imaging system, a fully implemented imager chip solution is not the optimum solution for consumer applications.

In a typical image processing task, more than 50% of the total computing power is consumed by a low-level image processing task. Typical low-level image processing tasks require hundreds of operations per pixel for a captured image with a small scale of memory. General-purpose computers such as microprocessors do not match these low-level image processing tasks since large hierarchical memory and flexibility for sequential processing capability in a general-purpose computer

are not valuable. Yet, general-purpose computers can afford image processing requiring sophisticated algorithms with frame memories. Since most low-level processing tasks are repetitive and are performed by using neighboring pixel data, an image processor architecture that utilizes parallelism should be valuable [4], [5].

In this paper, an imager with an integrated fully programmable bit-serial column-parallel processor is proposed to meet the demand for a compact versatile system-on-imager chip for consumer applications [6]. The proposed image processing architecture uses the system partitioning approach, where computationally intensive pixel-rate processing alone is implemented on an imager chip to target low-level image processing tasks. The processor is physically arranged as a densely packed 2-D processing element (PE) array at an imager column level. The digital processor has a multiple-instruction-multiple-data (MIMD) architecture configuring multiple column-parallel single-instruction-multiple-data (SIMD) processors. The fully programmable image processor incorporated in an imager chip is advantageous over the alternative special-purpose on-imager digital signal processor (DSP) solutions because of its flexibility for an image processing algorithms. There is no need to redesign hard-wired logic whenever an image processing algorithm has to be changed because altering an image processing algorithm can be accomplished by software if a processor is programmable. There have been several approaches for on-imager processors to meet the demand for compact and versatile consumer imaging applications. However, the on-imager processing architecture that is most appropriate for consumer imaging applications still remains to be seen. The purpose of this paper is to provide a candidate on-imager processing architecture that would be one of the solutions for the challenge.

The proposed image sensor architecture is described in Section II. The processor architecture is presented in Section III. The task sharing in the MIMD/SIMD processor is discussed in Section IV. Experimental results for a prototype imager chip with 128×128 pixels and 4×128 PE array designed with $0.6\text{-}\mu\text{m}$ technology are described in Section V. In Section VI, performance of the proposed processing architecture is compared with that of alternative on-imager processing architectures and is also compared with that of a current DSP chips. Conclusions are provided in Section VII.

II. IMAGE SENSOR ARCHITECTURE

Fig. 1 shows the proposed image sensor architecture. The digital processor on the image sensor chip has a column-parallel architecture [7]–[10]. All the data paths in the imager chip are

Manuscript received January 6, 2009; revised June 10, 2009. Current version published October 21, 2009. The review of this paper was arranged by Editor J. Nakamura.

H. Yamashita is with the System LSI Division, Semiconductor Company, Toshiba Corporation, Yokohama 247-8585, Japan (e-mail: hirofumi.yamashita@toshiba.co.jp).

C. G. Sodini is with the Massachusetts Institute of Technology (MIT), Cambridge, MA 02139-4307 USA (e-mail: Sodini@mit.edu).

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/TED.2009.2030718

Fig. 1. Proposed imager architecture.

implemented at the imager column level. The pixel in the pixel array can be either with a four-transistor architecture or with a three-transistor architecture. Image data in a pixel array are read out row by row. Processing is performed in the way that the imager row data are processed simultaneously at each data path. Correlated double sampling (CDS) circuitry is arranged per column to reduce pixel-to-pixel fixed-pattern noise. Sampled signals are digitized with a column-parallel analog-to-digital converter (ADC). Digitized images are transferred to the next-stage digital processors. The digital processor has an MIMD architecture configuring multiple column-parallel SIMD processors. Each SIMD processor is comprised of a 1-D array of PEs. Physically, PEs are arranged as a fine-grained densely packed 2-D PE array. The layout pitch of all the data paths is matched to the pixel column pitch in order to minimize silicon area. Image data flow from the pixel array to the output register along the column direction; thus, no extra wiring for long-distance data transfer is required. The proposed dense array of column-parallel data path architecture realizes high computational power with a minimal silicon area. Each SIMD processor works with a single stream of instructions, and each PE in the same SIMD processor share the same instruction stream. Processing tasks are shared with SIMD processors with an MIMD control from a controller. The host processor delivering instructions to processors can be an external intelligent host, such as a general-purpose processor, or it can be a command controller on an image sensor chip.

Imager pixel data are transferred to the processor at every imager horizontal period. For example, one horizontal period is $67 \mu\text{s}$ for 30 frames per second (fps) video graphics array (VGA) imaging. When the embedded processor clock cycle is 20 MHz, 1330 instruction cycles are allowed for one horizontal period. Data are transferred from the imager to a PE bit by bit, and a data transfer is completed by $b + 1$ instruction cycles, where b is the bit depth of the image data. Processed data are transferred from the PE to the output multiplexer (MUX) bit by bit, and a data transfer is also completed by $b + 1$ cycle. The ratio of the instruction cycle used for the data transfers

Fig. 2. On-imager processing architecture comparison.

to the total instruction cycle for one horizontal period is 1.4% when the 30-fps VGA image, 20-MHz clock, and 8-bit data are assumed. The processing power can be increased by concatenating SIMD processors along the column direction. The balance of the inter-PE communication and processing will be discussed in Section III. The way of task sharing among SIMD processors to maximize array utility is discussed in Section IV.

The advantage of column-parallel processing architecture is shown in Fig. 2. Fig. 2 compares three types of on-imager processor architecture. In a pixel-level architecture, a processor is implemented per pixel or block of pixels [11]–[13]. The advantage of the pixel parallel architecture is that the bandwidth between pixel and processor is maximum and high-speed nearest neighbor processing can be expected. The drawback of the pixel parallel architecture is that the pixel size tends to be larger than in the other architectures since the pixel processor consumes considerable amount of silicon area. Thus, high image resolution with a large pixel array with a moderate chip size is not available. In a chip-level architecture, since there is no processor in a pixel array, image resolution can be maximum, but consequently, the bandwidth between pixel and processor is minimum. There is less systematic advantage in

Fig. 3. Functional representation of PE.

this architecture over a two-chip solution with serial processing DSPs on an extra chip. A column-parallel architecture implements processors at a pixel array column level, where both medium class of parallelism and high resolution image can be obtained. For imaging systems requiring both image resolution and processing speed, a column-parallel architecture is the best solution to meet those demands.

III. PROCESSOR ARCHITECTURE

One of the important features for the on-imager processor used for the imaging system shown in Fig. 1 is full programmability to afford a variety of processing tasks. No specific-purpose data path architecture is used in the PE. Full programmability in the PE is provided with a general-purpose arithmetic logic unit (ALU). Fig. 3 shows the functional representation of the PE [14]. The PE comprises a logic unit and a local-memory column. The memory column is pitch matched to the logic unit. A bit-serial function generator is implemented in the logic as a general-purpose ALU. Three latches (A, B, and C) provide inputs to the function generator. The function generator can produce all the 256 three-input Boolean functions, which are selected with eight control signals (f7–0). Dynamic logic is used in the function generator to reduce the logic silicon area. Latch D provides input to the memory cell. Latch E enables write buffer. This write-enable structure for the memory column allows data-dependent processing in an image processing algorithm. Two kinds of inter-PE communication paths are connected to the PE. One is the path connecting two neighboring PEs in a row direction (right, left). The other is a 1-bit bus connecting PEs on the same column of the PE array for a communication among SIMD processors. The timing diagram of the PE is shown in Fig. 4.

In order to implement a per-column PE array structure, the processor employs a 1-bit-wide data communication path. Fig. 5 shows the PE array configuration in the processors. Each PE has a bit-serial structure, and all the communication paths in the PE array are 1 bit wide. Data in the PE array are transferred bit by bit among PEs during processing. The 1-bit-wide communication path structure reduces the processor array

Fig. 4. Timing diagram for the PE.

Fig. 5. Data communication path in the PE array.

silicon area used for data communication. There are two kinds of communication paths in the PE array as described before. One is for the right and left communication paths, connecting neighboring PEs in the same SIMD processor. The other is an inter-SIMD communication path that is arranged at every column. The inter-SIMD communication path has a 1-bit-wide bus structure. All the PEs on the same column are connected with the column bus. The column bus is used for data transfer for task sharing among SIMD processors. The column bus is also used for data transfer from ADC to SIMD processors. One of the PEs on the same column and ADC can drive the column bus, and any of the PEs and output MUX can receive the bus data at a time. Compared to the previously reported column-parallel processors [7], [9], [10], where one row of SIMD processors with a bit-parallel ALU is implemented, the proposed processor has a simpler PE array architecture. A fine-grained 2-D PE array connected with the 1-bit-wide communication path allows both compact processor structure and high computational power.

Fig. 6(a) summarizes the instruction cycles for a unit operation in the PE. Fig. 6(b) shows the example for an intra-SIMD sum operation procedure, which is one of the typical primitive operations in the PE. The procedure example shown in Fig. 6(b) computes the sum between a 4-bit value in a memory column in a PE and a 4-bit value in the memory column in the left neighbor PE. Instruction 1 loads 1 into latch E to enable the write buffer and loads the least significant bit of the operand, i.e., M0, from a memory into latch A. Instruction 2 transfers the left neighbor PE's M0 value to latch B and loads the least significant bit of the operand, i.e., M4, from a memory into latch A.

Operation	Notation	Instructions per b bit Operation
sum	$A \leftarrow A + B$	3b
copy	$A \leftarrow B$	2b
write	$A \leftarrow x$	b + 1
signed product	$P \leftarrow D \times r$	$2b^2$
move	$A \leftarrow B$ (Inter-PE, Intra-PE)	2b

Upper case : parallel data, Lower case : scalar

(a)

	Operation	Logic Activity	Memory Activity	Boolean Function	Interconnect Function	Latch
1	Read	$E \leftarrow 1$	$A \leftarrow [M0]$	1	---	A, E
2	Interconnect	$B \leftarrow \text{Left}$	$A \leftarrow [M4]$	---	Left	A, B
3	Sum	$D \leftarrow A + B$	$[S0] \leftarrow D$	$A + B$	---	D
4	Carry	$C \leftarrow A \wedge B$	$A \leftarrow [M1]$	$A \wedge B$	---	A, C
5	Interconnect	$B \leftarrow \text{Left}$	$A \leftarrow [M5]$	---	Left	A, B
6	Sum	$D \leftarrow A + B + C$	$[S1] \leftarrow D$	$A + B + C$	---	D
7	Carry	$C \leftarrow (A \wedge B)U(A \wedge C)U(B \wedge C)$	$A \leftarrow [M2]$	$(A \wedge B)U(A \wedge C)U(B \wedge C)$	---	A, C
8	Interconnect	$B \leftarrow \text{Left}$	$A \leftarrow [M6]$	---	Left	A, B
9	Sum	$D \leftarrow A + B + C$	$[S2] \leftarrow D$	$A + B + C$	---	D
10	Carry	$C \leftarrow (A \wedge B)U(A \wedge C)U(B \wedge C)$	$A \leftarrow [M3]$	$(A \wedge B)U(A \wedge C)U(B \wedge C)$	---	A, C
11	Interconnect	$B \leftarrow \text{Left}$	$A \leftarrow [M7]$	---	Left	A, B
12	Sum	$D \leftarrow A + B + C$	$[S4] \leftarrow D$	$A + B + C$	---	D

(b)

SIMD1 Instruction	SIMD1 Operation	Logic Activity	Memory Activity	Boolean Function	Interconnect Function	Latch
1	Read	---	$A \leftarrow [M0]$	---	---	A
2	Interconnect	$\text{Bus} \leftarrow A$	---	---	Tri-state "1"	---
3	---	---	---	---	---	---
4	Read	---	$A \leftarrow [M1]$	---	---	A
5	Interconnect	$\text{Bus} \leftarrow A$	---	---	Tri-state "1"	---
6	---	---	---	---	---	---
7	Read	---	$A \leftarrow [M2]$	---	---	A
8	Interconnect	$\text{Bus} \leftarrow A$	---	---	Tri-state "1"	---
9	---	---	---	---	---	---
10	Read	---	$A \leftarrow [M3]$	---	---	A
11	Interconnect	$\text{Bus} \leftarrow A$	---	---	Tri-state "1"	---
12	---	---	---	---	---	---

SIMD2 Instruction	SIMD2 Operation	Logic Activity	Memory Activity	Boolean Function	Interconnect Function	Latch
1	Read	$E \leftarrow 1$	$A \leftarrow [M4]$	1	---	A, E
2	Interconnect	$B \leftarrow \text{Bus}$	---	---	Bus	B
3	Sum	$D \leftarrow A + B$	$[S0] \leftarrow D$	$A + B$	---	D
4	Carry	$C \leftarrow A \wedge B$	$A \leftarrow [M5]$	$A \wedge B$	---	A, C
5	Interconnect	$B \leftarrow \text{Bus}$	---	---	Bus	B
6	Sum	$D \leftarrow A + B + C$	$[S1] \leftarrow D$	$A + B + C$	---	D
7	Carry	$C \leftarrow (A \wedge B)U(A \wedge C)U(B \wedge C)$	$A \leftarrow [M6]$	$(A \wedge B)U(A \wedge C)U(B \wedge C)$	---	A, C
8	Interconnect	$B \leftarrow \text{Bus}$	---	---	Bus	B
9	Sum	$D \leftarrow A + B + C$	$[S2] \leftarrow D$	$A + B + C$	---	D
10	Carry	$C \leftarrow (A \wedge B)U(A \wedge C)U(B \wedge C)$	$A \leftarrow [M7]$	$(A \wedge B)U(A \wedge C)U(B \wedge C)$	---	A, C
11	Interconnect	$B \leftarrow \text{Bus}$	---	---	Bus	B
12	Sum	$D \leftarrow A + B + C$	$[S3] \leftarrow D$	$A + B + C$	---	D

(c)

Fig. 6. (a) Instruction cycles for a unit operation. (b) Intra-SIMD sum operation. (c) Inter-SIMD sum operation.

The left neighbor value is transferred via an intra-SIMD transfer path. Instruction 3 computes the least significant bit of the sum, i.e., S_0 , and stores it in a memory through latch D. Instruction 4 computes the carry value, loads the result in latch C, and reads the second bit of the operand, i.e., M_1 , from a memory into latch A. Instructions 5 and 6 complete the processing for the second bit position. Instructions 7–9 complete the processing for the third bit position, and instructions 10–12 complete the processing for the fourth bit position. The intra-SIMD sum procedure consumes $3b$ operation cycles, where b is the number of bits used. In the proposed bit-serial ALU design, the sum procedure completed in a single PE (no data transfer among PE is involved) requires three memory activities—two read operations and one write operation per bit. The sum procedure exemplified in Fig. 6(b) shows that there is no additional instruction cycle needed for a data transfer in an intra-SIMD computation among neighboring PEs.

Fig. 6(c) shows an example of an inter-SIMD sum procedure. Two SIMD processors are involved in this inter-SIMD sum procedure. In the inter-SIMD sum procedure, SIMD1 delivers an operand bit into the column bus via a tri-state buffer. The delivered bus data are received by SIMD2, and SIMD2 computes the sum between the operand in SIMD1 and the operand in SIMD2. The inter-SIMD sum procedure also consumes $3b$ operation cycles. The sum procedure exemplified in Fig. 6(c) shows that there is no additional instruction cycle needed for a data transfer in an inter-SIMD computation among PEs on the same column bus.

However, a simple data move operation among PEs requires some amount of additional instruction cycles in a processing task. It consumes $2b$ clock cycles, where two memory activities are involved—one read and one write. The balance between data transfer cycles and processing cycles in a typical color processing task is estimated in order to see the burden of a data transfer. In a color processing algorithm assumed by the authors, which includes interpolation, color matrix, detail signal, and color coding, the ratio of the inter-PE data transfer cycles during processing to the total processing cycles is 7.6% for an 8-bit operation. The impact of the bit-by-bit data transfer on the total processing time is relatively small.

The advantage of a bit-serial architecture is its reduced hardware complexity. A simple analysis shows that a processing speed per silicon area of a bit-serial architecture and that of a bit-parallel architecture might be similar. A bit-parallel architecture consumes eight times larger silicon area than a bit-serial architecture does, but a bit-parallel architecture affords eight-pixel processing in the processing time that a bit-serial architecture completes a single pixel processing task. In a column-parallel architecture, the bandwidth between a row of pixel arrays and a row of PE arrays is maximum when a PE is arranged per imager column. When a bit-parallel architecture is used, an appropriate design is that a bit-parallel ALU is arranged per several columns since arranging a bit-parallel PE into a narrow imager column is not a practical solution. In this case, additional hardware such as a MUX is needed in the data path. A simple bit-serial architecture in which a PE is arranged per imager column requires no additional hardware, and thus, reduced hardware complexity is achieved.

The proposed processor architecture allows full flexibility in the image processor. The PE with a general-purpose ALU produces all the Boolean functions. The bit-serial data path architecture allows flexible signal word length because there is no specific bit-wide data path in this architecture. A variety of pixel rate image-processing algorithms can be implemented in this processing architecture.

IV. TASK SHARING IN MIMD/SIMD PROCESSOR

One of the important points in the SIMD/MIMD processing architecture is the way of maximizing the PE array utility. When the total processing tasks are exactly shared with the four SIMD processors all the time, maximum processing performance is attained. If the processing sequences are invariant during processing, the MIMD-controlled simple pipeline processing scheme affords to allocate the processing tasks to the four SIMD processors. However, sequences of processing could be changed during processing. For example, processing parameters such as parallel data used in an adaptive processing could vary while images are being captured. It is a difficult task for a host processor to allocate processing tasks on-the-fly to SIMD processors so that maximum performance can be attained.

The per-column bus architecture for inter-SIMD communication offers a simple solution to this problem by utilizing a data transfer among SIMD processors. An example of a processing scheme to maximize the PE array utility with this processing architecture for typical color processing is shown in Fig. 7(a). In Fig. 7(a), a processing scheme that has a column with four SIMD processors performing two 3×3 -kernel filtering (i.e., interpolation and Laplacian filtering for the detail signal) is shown. Each SIMD processor receives three rows of imager data row by row. The first 3×3 -kernel filtering (interpolation) in SIMD1 is performed with pixel-array row 2–row 4 data. The processing time for 8-bit data interpolation with a 20-MHz clock is $36 \mu\text{s}$. After the color interpolation is completed, the first point calculation (color matrixing) is performed. The processing time for 8-bit data color matrixing with a 20-MHz clock is $67 \mu\text{s}$. The result of the color matrixing in SIMD1 is broadcasted on a column bus once the result is obtained. Two SIMD processors (i.e., SIMD2 and SIMD3) working for consecutive imager row data, which require the color-matrixed data for row 3 in SIMD1, receive the broadcasted bus data at one time. The second 3×3 -kernel filtering (Laplacian) in SIMD2 starts right after the color matrix result in SIMD2 is produced. Since the color-matrixed data from the SIMD1 arrives before one imager horizontal period, there is no wait time in processing in SIMD2. After the result of the second 3×3 -kernel filtering (Laplacian) is obtained, the second point calculation (color difference coding) in SIMD2 is performed. After the color difference coding, the color signals are transferred from SIMD2 to the output MUX through the column bus. The processed image data for each imager row are read out row by row from the PE array. One row period for the 30-fps VGA imager is about $67 \mu\text{s}$. As shown in Fig. 7(a), one SIMD consumes more than three imager horizontal periods to complete per-pixel color processing task. SIMD2 waits for the

Fig. 7. (a) Per-pixel processing scheme maximizing PE array utility. (b) Memory activity for color processing procedure example.

next imager data (row 9) and starts processing right after the row 9 imager data arrive. As shown in Fig. 7(a), four SIMD processors exactly and evenly share the total processing task for one frame image since a processing task for one pixel is performed with one SIMD processor.

By employing this processing scheme, the total processor silicon area becomes scalable with the increase in the number of pixel. The ratio of the processor silicon area to pixel-array silicon area remains as the number of pixel increases. The design

approach for this SIMD/MIMD processor architecture becomes just as simple as concatenating a row of SIMD processors to meet the required processing speed for the image processing tasks. The number of SIMD processors in this SIMD/MIMD architecture can be determined to meet the projected maximum amount of processing tasks. The burden of the processor hardware design can be greatly reduced in this approach.

Fig. 7(b) exemplifies the memory activity in the PEs for the color processing shown in Fig. 7(a). PE starts one-pixel

Fig. 8. (a) Prototype imager chip die and pitch-matched layouts. (b) Prototype imager chip circuit diagram and 1 H timing.

processing with a column of three rows of image data. The 3×3 -kernel color interpolation is performed, for instance, according to the following algorithm:

$$\begin{aligned}
 R'(n, i) &= R(n, i) + 0.5 \{ R(n-1, i) + R(n+1, i) \\
 &\quad + R(n, i-1) + R(n, i+1) \} \\
 &\quad + 0.25 \{ R(n-1, i-1) + R(n+1, i-1) \\
 &\quad + R(n-1, i+1) + R(n+1, i+1) \} \\
 G'(n, i) &= G(n, i) + 0.25 \{ G(n-1, i) + G(n+1, i) \\
 &\quad + G(n, i-1) + G(n, i+1) \} \\
 B'(n, i) &= B(n, i) + 0.5 \{ B(n-1, i) + B(n+1, i) \\
 &\quad + B(n, i-1) + B(n, i+1) \} \\
 &\quad + 0.25 \{ B(n-1, i-1) + B(n+1, i-1) \\
 &\quad + B(n-1, i+1) + B(n+1, i+1) \}
 \end{aligned}$$

where R' , G' , and B' are interpolated data, R , G , and B are captured image data, n is an imager row number, and i is an imager column number. In the above example, only nearest neighbor pixel data are used for a color interpolation. At every pixel, the same interpolation algorithm shown above is applied to produce each primary color data. Three rows of pixel data are stored in a PE memory, and interpolation is completed with intra-SIMD communication. In the next processing stage, color matrixing is performed. Laplacian filtering is performed after the color matrixing. Three rows of color-matrixed green signal are stored in a PE memory, and Laplacian filtering is completed with the right and left PE green signals by using intra-SIMD communication. Color coding is achieved after the Laplacian filtering. As described in the processing example described in Fig. 7(b), one row of PEs in a SIMD processor can perform a 3×3 processing task. Larger kernel size processing such as 5×5

Fig. 9. Data path architecture in the prototype imager chip from ADC to output MUX.

or 7×7 can be performed in a single SIMD processor when a memory in a PE has enough capacity to hold more rows of data. Thus, the memory size in a PE is dominated by the processing kernel size. The number of PE rows does not limit the size of a processing kernel size but dominates a processing speed in this SIMD/MIMD architecture.

V. PROTOTYPE IMAGER CHIP EXPERIMENTAL RESULTS

The prototype imager chip was designed with the proposed imager architecture described in Sections I–IV. The prototype chip incorporates a 128×128 pixel array, a column-parallel CDS/ADC, and the four column-parallel SIMD processors. In the prototype imager chip, the number of SIMD rows is four, which was determined to meet the processing speed for 30-fps VGA color processing, including color interpolation, color matrixing, aperture correction, and color coding.

The imager chip die is shown in Fig. 8. A three-transistor architecture pixel with a charge spill gate to increase sensitivity is used [15]. CDS circuitry is arranged per column to reduce fixed-pattern noise. Sampled signals are digitized with a column-parallel ADC. One ADC is implemented per four columns to minimize column pitch, and four pixel data in the same imager row are digitized successively in one row period. The ADC architecture is a cyclic ADC [15]. The output bits from ADC are produced bit by bit from the most to the least significant bit. PE is arranged per pixel column. A DRAM is used as a PE memory to minimize column pitch. A twin-cell DRAM with a planar MOS capacitor is used in the prototype imager chip. The DRAM with MOS capacitor structure is used because it can be implemented with a standard logic process. A pitch-matched layout of the prototype imager chip is shown in Fig. 8(a). A circuit diagram from the pixel array to the PE through CDS and ADC for the prototype chip is shown in Fig. 8(b). The operational timing for CDS and ADC is also shown in Fig. 8(b).

A 1-bit-wide shift register is prepared to transfer external parallel data to each column of PE (Fig. 9). A shift register

Fig. 10. Image processing example of the imager chip. (a) Captured raw image. (b) Edge image with Sobel filtering.

TABLE I
PROTOTYPE IMAGER CHIP CHARACTERISTICS

Clock	20 MHz
Power Supply	3.3 V
Technology	0.6 μm 3-Metal CMOS
Chip Size	5.8 x 12.6 mm²
Pixel Array	
Number of Pixel	128 x 128
Pixel Size	18 x 18 μm^2
Processor	
Number of PE	4 x 128
PE Size	18 x 1200 μm

cell is connected to each column bus. External parallel data are used for conditional processing. One of the examples for a conditional processing by using parallel data is a data transfer from a per-four-column ADC to a per-column PE. The outputs from one ADC are delivered to the four columns of the PE array at one time, and one out of four columns of PEs selectively receives the corresponding imager column data according to the flag data delivered to the PE array by the shift register. With this data transfer procedure, no hardware demultiplexer (DMUX) between the ADC and the PE array are required. The combination of a cyclic ADC and a bit-serial PE array architecture produces less redundancy in the data transfer scheme from the ADC to the next-stage PE array. Processed data are transferred out from a 128- to a 16-output MUX. The bit-serial data path architecture with the cyclic ADC and the bit-serial PE architecture allow a flexible signal word length, because there is no specific bit-wide data path in this architecture.

In the previously reported column-parallel processor, PE is provided at every 16 columns [8]. The bit-parallel processor architecture used in the previous work makes it difficult to implement a per-column PE array structure since a multibit bus architecture covers considerable PE column pitch. The bit-parallel processor architecture requires additional components such as a MUX in between the ADC and the PE array in order to sort out image data from the ADC to the multicolumn PE. The proposed bit-serial architecture produces less redundancy in the data transfer schemes.

In an imager chip functionality test, the microcontroller for the processor is prepared on a test board. Bit-by-bit low-level instructions are directly produced on a PC before processing,

TABLE II

(a) PERFORMANCE COMPARISON WITH AN ALTERNATIVE ON-IMAGER PROCESSOR. (b) PERFORMANCE COMPARISON WITH AN ALTERNATIVE DSP

	This work (Prototype chip)	8PE SIMD Bank with 1k byte Memory [8]	Per pixel analog PE [13]	PASIC [7]
Design Rule	0.6 μm 3-Metal	0.6 μm 3-Metal	0.6 μm 3-Metal	1.6 μm
Clock Frequency	20MHz	25 MHz	2.5 MHz	20 MHz
Pixel pitch	18 μm	---	98.6 μm	40 μm
Bit depth	Flexible	8 bit	---	8 bit
Processor architecture	Column parallel (Bit serial)	Column Parallel (Bit parallel)	Pixel parallel	Column Parallel (Bit serial)
Operations per pixel (640 x 480 30fps)	220 Sum per pixel (8bit)	87 Op. per pixel	---	---
Sobel Filtering	52.6 μsec (8bit, 1PE)	---	11.6 μsec (1PE)	---
# of Sum / sec. per column	3.3 MOPS	---	---	0.77 MOPS (Sum:3b+2, 20MHz)
Power consumption	45 mW (4 x 128 PE)	20 mW (8 SIMD Bank)	85 μW (1 PE)	---

(a)

	This work (Experimental Chip)	This work (w/ advanced technology)	Programmable 512GOPS Stream Processor [18]	Programmable 512GOPS Image Processor [17]
Processor architecture	Column parallel PE array	Column parallel PE array	16 Lane DPU (DSP chip)	64 PE x 4
Processor Area	0.0216 mm ² per PE (0.6 μm DR)	0.0010 mm ² Per PE (0.13 μm DR)	16 x 3.2 mm ² per 16 Lane DPU (0.13 μm DR)	12.8 mm x 11.94 mm (0.065 μm DR)
Operations per second	0.833 MOPS per PE (20 MHz)	33.3 MOPS per PE (800 MHz)	512 GOPS per 16 Lane DPU (800 MHz)	512 GOPS Per 64 PE x 4 (500 MHz)
3 x 3 Convolution Filter (8 bit)	4218 (cycle/pixel) 0.011 (pixel/cycle/mm ²)	4218 (cycle/pixel) 0.234 (pixel/cycle/mm ²)	0.15 (cycle/pixel) 0.130 (pixel/cycle/mm ²)	-
Power Consumption	87.9 μW per PE (3.3V, 20 MHz, 0.6 μm DR)	-	10.5 W per chip (1.0V(core), 0.13 μm DR)	783 mW per chip (1.0V, 250 MHz, 0.065 μm DR)

(b)

and those were stored on a test board memory. The instruction sets were sent to the imager chip from the test board during processing and were directly used by the on-chip processor. For the purpose of the functional test of the PE array, the external parallel test data were used. The test data were transferred to each column of PEs through the 128-stage shift register. PE processing functionality was checked by testing simple arithmetic operations. The PE on the prototype chip was found to be fully functional. Basic image processing tasks are performed

with the imager chip to demonstrate the processor performance. Fig. 10 shows the captured raw image by the prototype imager chip and the edge extracted image with Sobel filtering achieved by the on-imager processor. The edge extracted image is successfully obtained. Quantitative measurements for image quality testing were not done. The overall image quality was mostly degraded by the camera lens performance, but no obvious digital noise was observed in the reproduced images from this experimental imager chip. The operational clock frequency

is 20 MHz. Since a PE executes one instruction per clock cycle, 0.83 mega sum operations per second (MOPS) by a single PE is achieved with a 50-ns clock cycle. The prototype imager chip potentially affords pixel-rate color processing tasks such as color interpolation, color matrixing, aperture correction, and color coding for an 8-bit VGA image at 30 fps by performing ~ 220 sum operations per pixel with a 20-MHz clock.

The imager chip was fabricated with 0.6 μm double-poly triple-metal CMOS technology. A 3.3-V power supply is used both for digital blocks and analog blocks. The power consumption of the image processor on the prototype chip was not measured but is estimated as 87.9 $\mu\text{W/PE}$. The pixel size is 18 $\mu\text{m} \times 18 \mu\text{m}$. The column size of 18 μm is not dominated by the size of the PE. The expected minimum PE column pitch with 0.6- μm technology is estimated as 7.2 μm . The column size of this experimental chip is dominated by the size of the op-amps used both in CDS and ADC circuitry. The prototype imager chip characteristics are summarized in Table I.

VI. PERFORMANCE ISSUES

The performance of the proposed processing architecture is compared with alternative processing architectures in order to see the performance level of the proposed processing architecture.

The performance comparison between the prototype imager chip and the other on-imager processor is summarized in Table II(a). By assuming 30-fps VGA image processing, the prototype imager chip can afford 220 sum operations per second (OPS)/pixel with a 20-MHz clock, whereas the alternative digital column-parallel processing architecture, i.e., SIMD banks with data streaming buffer per 16 pixel columns, performs 87 OPS/pixel with a 25-MHz clock [8]. The pixel parallel analog PE architecture achieves 11.6 μs for Sobel filtering, which is better than 52.6 μs for Sobel filtering with this work's prototype imager chip [13]. However, the pixel parallel analog PE uses 128 transistors per pixel and consumes a 98.6 $\mu\text{m} \times 98.6 \mu\text{m}$ unit pixel area. The comparably large pixel size suggests that the architecture is suitable for machine vision applications but is not suitable for an ordinary consumer imaging application. The performance of the bit-serial column-parallel processor embedded on a 128 $\times$ 128 pixel array is also compared [7]. The number of sum operations per second for the proposed architecture is 3.3 MOPS, whereas that of the alternative column-parallel architecture is 0.77 MOPS. A pixel parallel digital vision chip for machine vision applications can achieve 18 objects per 1 ms target tracking with a 128 $\times$ 128 pixel array [16]. The pixel parallel digital processor uses 84 transistors per PE, and the pixel pitch is 80 μm . This architecture is also most suitable for machine vision applications.

The performance comparison between the proposed processing architecture and the current DSP chip designed specifically for image processing has been made [17], [18]. The comparison is summarized in Table II(b). The prototype imager chip was fabricated with 0.6- μm technology, whereas the up-to-date DSP chips were designed and fabricated with more advanced technology. Thus, the performance level of the proposed processing architecture with advanced technology is estimated in order to

Fig. 11. Estimation for processing speed per PE Si area.

make a comparison of the performance level of the proposed image processing architecture with that of the current DSP. It is assumed that the PE silicon area can be scaled with a process design rule. The 512 giga sum operations per second (GOPS) stream processor chip is comprised of 16 data parallel unit (DPU) with five ALUs per lane [18]. The DSP chip can complete a 3 $\times$ 3 2-D convolution filter (8-bit pixels) per 0.15 cycle/pixel. The DSP chip is fabricated with 0.13- μm technology, and the unit DPU silicon area is 3.2 mm^2 . The performance metric used for the comparison is the number of pixels per clock cycle per unit silicon area for a 3 $\times$ 3 convolution filter with 8-bit data. The proposed processing architecture is capable of a 3 $\times$ 3 convolution filter (8-bit data) per 4218 cycles/pixel by a single PE. The prototype chip is fabricated with 0.6- μm technology, and the unit PE silicon area is 21 600 μm^2 . By assuming that the PE silicon area is scaled with a design rule, the PE area would be 1014 μm^2 when 0.13- μm technology is used. The performance level of the proposed architecture with 0.13 μm design rule can be estimated as 0.23 pixel/cycle/ mm^2 (4218 cycles/pixel for 8-bit data and 1014 $\mu\text{m}^2/\text{PE}$ with 0.13- μm technology). The alternative DSP solution provides 0.13 pixel/cycle/ mm^2 (0.15 cycle/pixel for 8-bit data and 51.2 mm^2 per 16-lane DPU). The estimation proves that the simple fine-grained array processor designed for imager column level processing produces comparable performance with the alternative programmable DSP processor when low-level image processing is a target processing task.

Fig. 11 shows the dependence of MOPS per silicon area occupied by PE on the process design rule. The estimation was based on the prototype imager chip performance, 0.83 MOPS per PE for 8-bit data with a 20-MHz clock frequency. It is also assumed that the Si PE area can be scaled with a process design rule. As shown in Fig. 11, 714 MOPS/ mm^2 for a 12-bit operation can be achieved with 0.18- μm technology and with a 50-MHz clock frequency. On the other hand, 5.72 GOPS/ mm^2 for a 12-bit operation can be achieved with 0.09- μm technology and with a 100-MHz clock frequency. These performance levels are roughly comparable with the performance level of up-to-date programmable digital image processor of 512 GOPS with a 12.8 $\text{mm} \times 11.94 \text{ mm}$ die size, or other examples [17], [18].

Fig. 12. Imager chip aspect. (a) Prototype imager chip $18\ \mu\text{m} \times 18\ \mu\text{m}$ pixel 128×128 pixel-array $0.6\text{-}\mu\text{m}$ DR 20-MHz 8-bit operation. (b) VGA imager chip $18\ \mu\text{m} \times 18\ \mu\text{m}$ pixel 640×480 pixel-array $0.6\text{-}\mu\text{m}$ DR 20-MHz 8-bit operation. (c) Five-megapixel imager chip $2.7\text{-}\mu\text{m}$ pixel pitch 2560×2068 pixel-array $0.09\text{-}\mu\text{m}$ DR 50-MHz 12-bit operation.

The aspect of the imager chip implementing the proposed SIMD/MIMD processor is shown in Fig. 12. As described in Section IV, by employing the proposed processing scheme, the total processor silicon area becomes scalable with an increasing number of pixels. The ratio of the processor silicon area to the pixel-array silicon area remains as the number of pixels increases, if the processing task is the same. The aspect of the prototype imager chip with a 128×128 pixel array is shown in Fig. 12(a). Since the prototype chip was targeted to meet the 30-fps VGA color processing tasks, the ratio of the processor area to the pixel-array area is large. The full VGA chip aspect for the same assumption with the prototype chip is shown in Fig. 12(b). The 5-Mpixel imager chip with the proposed processor is shown in Fig. 12(c). Finally, $0.09\text{-}\mu\text{m}$ process technology, 12-bit word length, and $2.7\text{-}\mu\text{m}$ pixel pitch are assumed. With these assumptions, the ratio of the processor area to the pixel-array area is sufficiently small. This estimation proves that if advanced CMOS technology is used, the proposed imager architecture achieves both high processing speed and high image resolution with a small pixel pitch.

VII. SUMMARY AND CONCLUSION

An imager with a column-parallel MIMD/SIMD processor configured with a general-purpose data path has been proposed. The imager architecture provides full programmability for a variety of image processing applications. The pitch-matched fine-grained PE array configuration offers a compact system-on-imager solution. A 128×128 pixel prototype imager chip with the four column-parallel SIMD processors was fabricated. Image acquisition and processor functionality was tested, and full functionality was demonstrated. The performance level of the proposed processing architecture with advanced process technology shows that the proposed imager chip architecture is one of the candidates for an imager with a fully programmable high-speed processing performance for a variety of consumer imaging applications.

ACKNOWLEDGMENT

The authors would like to thank J. C. Gealow for his contribution to the PE core design, S. J. Decker for providing his imager technology, M. Ishikawa for his test board design and for helping with imager testing, and Dr. I. Masaki and Dr. D. McGrath for the fruitful discussions. The authors also appreciate the encouragements from their colleagues at Microsystems Technology Laboratories, MIT, particularly I. Fujimori, C.-C. Wang, P. Acosta-Serafini, K. Fife, Z. Talib, and S. Dacy.

REFERENCES

- [1] E. R. Fossum, "CMOS image sensors: Electronic camera-on-a-chip," *IEEE Trans. Electron Devices*, vol. 44, no. 10, pp. 1689–1698, Oct. 1997.
- [2] S. G. Smith, J. E. D. Hurwitz, M. J. Torrie, D. J. Baxtr, A. A. Murray, P. Likoudis, A. J. Holmes, M. J. Panaghiston, R. K. Henderson, S. Anderson, P. B. Denyer, and D. Renshaw, "A single-chip CMOS 306×244 -pixel NTSC video camera and a descendant coprocessor device," *IEEE J. Solid-State Circuits*, vol. 33, no. 12, pp. 2104–2111, Dec. 1998.
- [3] M. J. Loinaz, K. J. Singh, A. J. Blanksby, D. A. Inglis, K. Azadet, and B. D. Ackland, "A 200-mW, 3.3-V, CMOS color camera IC producing 352×288 24-b video at 30 frames/s," *IEEE J. Solid-State Circuits*, vol. 33, no. 12, pp. 2092–2103, Dec. 1998.
- [4] P. M. Flanders, D. J. Hunt, S. F. Reddaway, and D. Parkinson, "Efficient high speed computing with the distributed array processor," in *Massively Parallel Computing With the DAP*. Cambridge, MA: MIT Press, 1990.
- [5] D. Parkinson, "The distributed array processor (DAP)," in *Massively Parallel Computing With the DAP*. Cambridge, MA: MIT Press, 1990.
- [6] H. Yamashita and C. G. Sodini, "A 128×128 CMOS imager with 4×128 bit-serial column-parallel PE array," in *Proc. IEEE ISSCC Dig. Tech. Papers*, San Francisco, CA, Feb. 2001, pp. 96–97.
- [7] K. Chen, A. Astrom, and P.-E. Danielsson, "PASIC: A smart sensor for computer vision," in *Proc. 10th Int. Conf. Pattern Recog.*, 1990, vol. 2, pp. 286–291.
- [8] S. H. Hong and W. Yang, "An embeddable low power SIMD processor bank," in *Proc. IEEE ISSCC Dig. Tech. Papers*, San Francisco, CA, Feb. 2000, pp. 192–193.
- [9] R. P. Kleinhorst, A. A. Abbo, A. van der Avoird, M. J. R. Op de Beeck, L. Sevat, P. Wielage, R. van Veen, and H. van Herten, "Xetal: A low-power high-performance smart camera processor," in *Proc. ISCAS*, 2001, vol. 5, pp. 215–218.
- [10] L. Lindgren, J. Melander, R. Johansson, and B. Moller, "A multiresolution 100-GOPS 4-Gpixels/s programmable smart vision sensor for multisense imaging," *IEEE J. Solid-State Circuits*, vol. 40, no. 6, pp. 1350–1359, Jun. 2005.
- [11] D. X. Yang, A. E. Gamal, B. Fowler, and H. Tian, "A 640×512 CMOS image sensor with ultrawide dynamic range floating-point pixel-level ADC," *IEEE J. Solid-State Circuits*, vol. 34, no. 12, pp. 1821–1834, Dec. 1999.
- [12] S. Kleinfelder, S.-I. Lim, X. Liu, and A. El Gamal, "A 10 kframes/0.18 μm CMOS digital pixel sensor with pixel-level memory," in *Proc. IEEE ISSCC Dig. Tech. Papers*, San Francisco, CA, Feb. 2001, pp. 88–89.
- [13] P. Dudek and P. J. Hicks, "A general-purpose processor-per-pixel analog SIMD vision chip," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 52, no. 1, pp. 13–20, Jan. 2005.
- [14] J. C. Gealow and C. G. Sodini, "A pixel parallel image processor using logic pitch-matched to dynamic memory," *IEEE J. Solid-State Circuits*, vol. 34, no. 6, pp. 831–839, Jun. 1999.
- [15] S. Decker, R. McGrath, K. Brehmer, and C. G. Sodini, "A 256×256 CMOS imaging array with wide dynamic range pixels and column-parallel digital output," *IEEE J. Solid-State Circuits*, vol. 33, no. 12, pp. 2081–2091, Dec. 1999.
- [16] T. Komuro, I. Ishii, M. Ishikawa, and A. Yoshida, "A digital vision chip specialized for high-speed target tracking," *IEEE Trans. Electron Devices*, vol. 50, no. 1, pp. 191–199, Jan. 2003.
- [17] S. Arakawa, Y. Yamaguchi, S. Akui, Y. Fukuda, H. Sumi, H. Hayashi, M. Igarashi, K. Ito, H. Nagano, M. Imai, and N. Asari, "A 512GOPS fully-programmable digital image processor with full HD 1080p processing capabilities," in *Proc. IEEE ISSCC Dig. Tech. Papers*, San Francisco, CA, Feb. 2008, p. 312.
- [18] B. K. Khailany, T. Williams, J. Lin, E. P. Long, M. Rygh, D. W. Tovey, and W. J. Dally, "A programmable 512 GOPS stream processor for signal, image, and video processing," *IEEE J. Solid-State Circuits*, vol. 43, no. 1, pp. 202–213, Jan. 2008.

Hirofumi Yamashita (M'01) received the B.S. and M.S. degrees in physics from the Tokyo Institute of Technology, Tokyo, Japan, in 1985 and 1987, respectively.

In 1987, he joined the Research and Development Center, Toshiba Corporation, Kawasaki, Japan, where he studied solid-state imagers. From 1998 to 1999, he was a Visiting Scientist at Microsystems Technology Laboratories, Massachusetts Institute of Technology, Cambridge, where he studied on-chip image processors. In 2000, he joined the System LSI

Division, Semiconductor Company, Toshiba Corporation, Yokohama, Japan, where he has been engaged in the development of CMOS imagers.

Charles G. Sodini (M'82–SM'90–F'95) received the B.S. degree in electrical engineering from Purdue University, West Lafayette, IN, in 1974, and the M.S. degree in electrical engineering and the Ph.D. degree from the University of California, Berkeley, in 1981 and 1982, respectively.

He was a Member of Technical Staff at Hewlett-Packard Laboratories from 1974 to 1982, where he worked on the design of MOS memory. He joined the faculty of the Massachusetts Institute of Technology, Cambridge, in 1983, where he is currently the LeBel

Professor of electrical engineering. He also studied the Hong Kong/South China electronics industry in 1996–1997 and has continued to study the globalization of the electronics industry. He was a cofounder of SMaL Camera Technologies, a leader in imaging technology for consumer digital still cameras and machine vision cameras for automotive applications. Along with Prof. R. T. Howe, he is a coauthor of an undergraduate text on integrated circuits and devices, entitled *Microelectronics: An Integrated Approach* (Prentice–Hall, 1997). His research interests are focused on mixed-signal integrated circuit and systems with emphasis on analog, RF, and millimeter-wave circuit design.

Dr. Sodini has served on several IEEE conference committees, including the International Electron Device Meeting, where he was the 1989 General Chairman. He has served on the IEEE Electron Device Society Administrative Committee and was the President of the IEEE Solid-State Circuits Society from 2002 to 2004. He is currently the Chair of the Executive Committee for the VLSI Symposia.