SCIENCE CHINA
Information Sciences

• RESEARCH PAPER •

June 2014, Vol. 57 062406:1–062406:12
doi: 10.1007/s11432-013-4969-4

A high speed multi-level-parallel array processor for vision chips

SHI Cong[1,2], YANG Jie[1], WU NanJian[1]* & WANG ZhiHua[2,3]

[1]*State Key Laboratory for Superlattices and Microstructures, Institute of Semiconductors, Chinese Academy of Sciences, Beijing* 100083*, China;*
[2]*Department of Electronic Engineering, Tsinghua University, Beijing* 100084*, China;*
[3]*Institute of Microelectronics, Tsinghua University, Beijing* 100084*, China*

Received December 28, 2013; accepted March 4, 2014; published online April 16, 2014

Abstract This paper proposes a high speed multi-level-parallel array processor for programmable vision chips. This processor includes 2-D pixel-parallel processing element (PE) array and 1-D row-parallel row processor (RP) array. The two arrays both operate in a single-instruction multiple-data (SIMD) fashion and share a common instruction decoder. The sizes of the arrays are scalable according to dedicated applications. In PE array, each PE can communicate not only with its nearest neighbor PEs, but also with the next near neighbor PEs in diagonal directions. This connection can help to speed up local operations in low-level image processing. On the other hand, global operations in mid-level processing are accelerated by the skipping chain and binary boosters in RP array. The array processor was implemented on an FPGA device, and was successfully tested for various algorithms, including real-time face detection based on PPED algorithm. The results show that the image processing speed of proposed processor is much higher than that of the state-of-the-arts digital vision chips.

Keywords vision chip, array processor, multi-level-parallel, high speed image processing, face detection

Citation Shi C, Yang J, Wu N J, et al. A high speed multi-level-parallel array processor for vision chips. Sci China Inf Sci, 2014, 57: 062406(12), doi: 10.1007/s11432-013-4969-4

1 Introduction

Vision chip is playing a more and more important role in high speed real-time vision applications [1–3]. This chip integrates image sensor and parallel processing circuits on a single chip. It eliminates speed bottlenecks in serial image data transmission and processing of the traditional vision systems. It can perform the image processing at a frame rate of over 1000 fps quickly. The vision chip can be widely applied in industry automation, intelligent transportation, robotic vision and so on.

A large number of vision chips have been developed [4–23]. Among these vision chips, the programmable vision chips with digital processors usually show better performances, such as high processing speed, flexibility and robustness [14–19]. Many programmable vision chips use the pixel-parallel architecture with two-dimensional (2D) processing element (PE) array. Each PE consists of a pixel photodiode and corresponding processing circuits. The PE elements in the array operate in single-instruction multiple-data (SIMD) fashion [14,15]. Actually the image resolution and processing flexibility of the vision chips

*Corresponding author (email: nanjian@red.semi.ac.cn) based on the architecture are limited by the finite chip area and the local connection between the PE elements. On the other hand, some other programmable vision chips based on the architecture with one dimensional (1D) parallel processor-array were reported [16,17]. Those vision chips separated image pixel array and 1D row- (column-) parallel processor (RP) array. This architecture increases the resolution of the image data and enhances the processing flexibility. But, its image processing speed is lower than that of the vision chip with 2D PE array.

Recently, a novel vision chip based on multiple levels of parallel processors was reported [18]. This chip integrates image pixel array, 2D pixel-parallel PE array, 1D row-parallel RP array and non-parallel RISC MPU. The PE array performs pixel-parallel algorithms. The RP array performs row-parallel algorithms. The embedded MPU can perform the non-parallel processing algorithm and manage the chip operation. They can complete image sensing, low-, mid-, and high-level image processing, respectively. This chip overcomes drawbacks of previous vision chip architectures. However, each PE can only perform the operation of the data variables from the nearest-neighbor PEs and itself directly. It cannot directly visit the next near neighbor PEs in the diagonal directions. Moreover, the two distant RP processors in RP array cannot visit each other quickly. Therefore it is difficult to speed up the pixel-level filtering operations and some global operations.

This paper proposes a novel multi-level-parallel array processor for programmable vision chips. This array processor consists of PE array, RP array and SIMD instruction decoder. The PE element can visit the nearest and next near neighbor PE elements and perform the corresponding operations. The RP array includes novel skipping chain and binary boosters. The PE and RP arrays significantly speed up low-level local operations and mid-level global operations. The vision chip based on the array processor would be more suitable for high speed vision applications with complicated low- and mid- level processing. The rest of this paper is organized as follows. Section 2 describes the architecture of proposed array processor. The detailed circuits of PE cell and RP cell are presented in Section 3. Section 4 shows physical implementation and evaluation results of this processor. Finally, we draw our conclusions in Section 5.

2 Vision chip architecture and array processor

A typical architecture of programmable vision chip with low- to high-level image processing ability is depicted in Figure 1. This chip integrates image pixel array, 2D pixel-parallel PE array, 1D row-parallel RP array and non-parallel RISC MPU. The image data is captured on the image pixel array and quickly converted to digital image signals by analog-to-digital (ADC) array. Then the PE array and RP array perform pixel-parallel low-level processing and row-parallel mid-level processing, respectively. The RISC MPU performs non-parallel high-level processing and manages the chip operation. The PE array is different from the pixel array in size for chip area reduction [18]. And flexible mapping relationship between pixels and PEs can be established. This paper will mainly focus on the PE array and RP array with their SIMD instruction decoder and treat the arrays as an independent kernel. It is called multi-level-parallel array processor in Figure 1. We design a novel multi-level-parallel array processor and improve the performance of the vision chip significantly.

The architecture of the proposed multi-level-parallel array processor is given in Figure 2. It consists of a 1-bit-grained $M1 \times M2$ pixel-parallel PE array, an 8-bit-grained $M1 \times 1$ row-parallel RP array, and an SIMD instruction decoder. The decoder receives SIMD instructions from the RISC MPU, and generates hardware control signals for PE array or RP array which are active exclusively. The PE array operates in SIMD manner. That is, all PE elements run with same instruction and different data in their own local memories. Each PE element can visit its four nearest–neighbor PEs on east (E), south (S), west (W), north (N), as well as the four next near neighbor PEs on northeast (NE), southeast (SE), northwest (NW) and southwest (SW). Compared with previous vision chips, our PE array reduces local operation time, because data variables in diagonal neighbor PEs can be operated directly. Although the PE array can finish most of the low-level processing in pixel-parallel, the row-parallel RP array is still needed. The PE array can only perform linear operations such as addtion, subtraction and logical

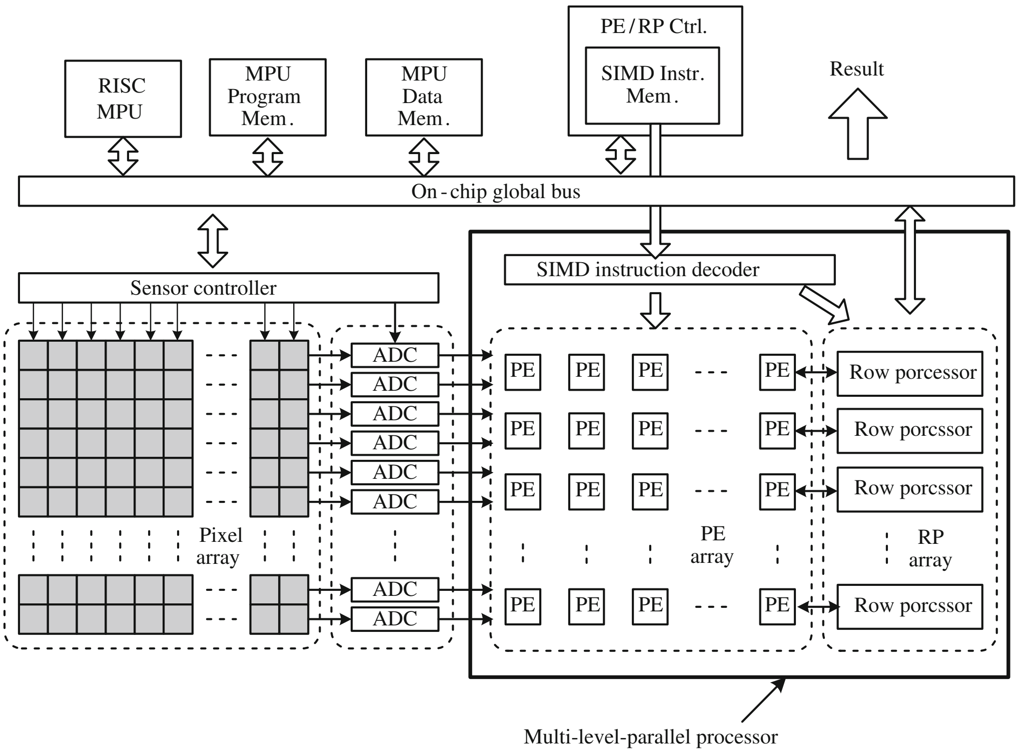

Figure 1 Programmable vision chip architecture.

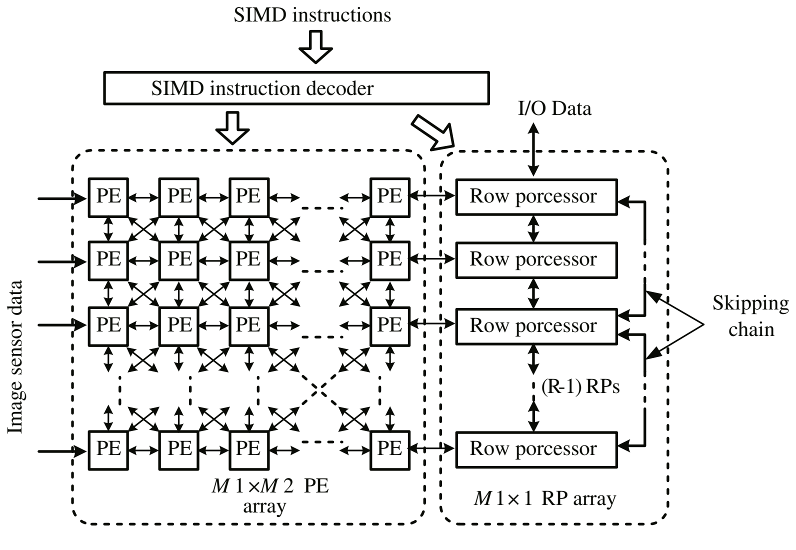

Figure 2 Multi-level-parallel processor.

operations. However, in some algorithms such as the grayscale morphology, the nonlinear operations of max/min extracting are necessary, and additional dedicated hardwares are required. If these nonlinear functions are also integrated into the PE array, a large amount of area would be consumed. Therefore, it is suitable to be performed by the RP array in a row-parallel fashion. Furthermore, some global operations in mid-level processing only have row-parallelism and require fast processing on multi-bit data format. If these operations are performed by the PE array in a bit-serial manner, the performance would decrease drastically. The RP array also operates in SIMD manner. Each RP processor can exchange data with its nearest upper and lower neighbor RPs, as well as with PEs in the same row. Moreover, some RP

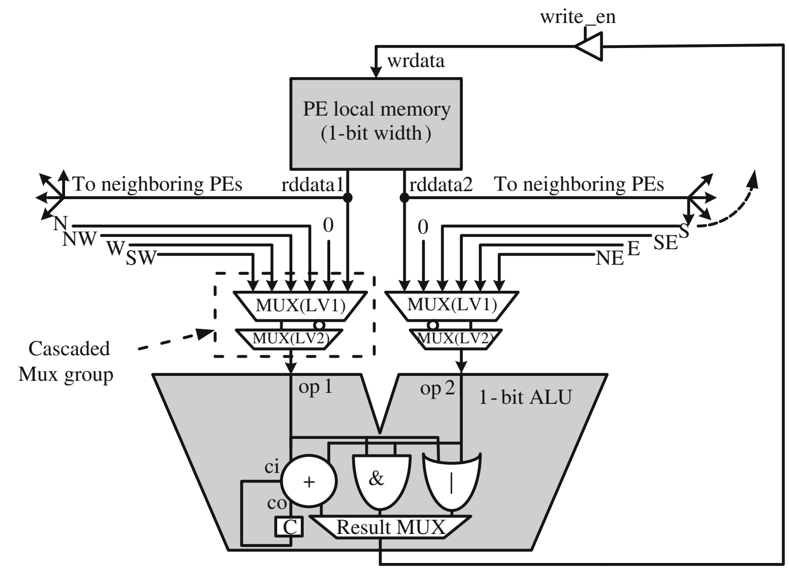

Figure 3 PE element circuit.

processors are also located on a skipping chain. These RPs can directly visit distant RPs on the same chain and operate their data variables to accelerate some global operations, as is shown in Figure 2. The optimal value of skipping interval R is $\sqrt{M1}$. This will be further explained with an example in Subsection 3.2. The RP array is also used as I/O interface when external data access is required. Image data can be either directly fed into PE array in word-parallel, or through RP array in word-serial. Because the general-purpose multi-core CPUs only exhibit some thread-parallelism and are not suitable to utilize the inherent massive data parallelism in low- and mid-level processing algorithms, the multi-level-parallel PE array and RP array are essential to high speed image processing and cannot be replaced by the technique of multi-core CPUs.

3 Design of circuits

3.1 Processing element

As is shown in Figure 3, the PE element consists of a 1-bit ALU, a 1-bit-width triple-port local memory and two cascaded multiplexer groups. The ALU includes a full adder, a carry register, an AND gate, an OR gate and a multiplexer for final result selection. It performs arithmetic addition, AND and OR operations between two input operands and sends the results into the local memory. The carry register is used to store the adder carry for multi-bit addition/subtraction operation in a bit-serial manner by the 1-bit ALU. The operands of op1 and op2 are taken from the left and right cascaded multiplexer groups, respectively. The first level multiplexer (LV1) in the left (right) multiplexer group selects one from the local memory data, the north (south) PE data, the northwest (southeast) PE data, the west (east) data, the southwest (northeast) PE data and 1-bit constant of "0", and outputs it into the second level multiplexer (LV2). Then the LV2 multiplexer in the left (right) multiplexer group decides whether to pass the candidate itself or its inverted value as the first (second) ALU operand op1 (op2) directly. Actually the multiplexer has a pre-inverting operation function and speeds up some algorithms such as XOR efficiently. There are one writing port and two reading ports in the local memory. The outputted rddata1 and rddata2 from the local memory are fed into the left and right cascaded multiplexer groups, the nearest-neighbor and next near neighbor PEs, respectively.

This simple PE element circuit can perform powerful and fast operations required for most pixel-parallel image processing algorithms. We take a typical Edge Detection algorithm for example [23]. This algorithm first filters the image with a Laplacian template as shown in Figure 4(a), and then performs

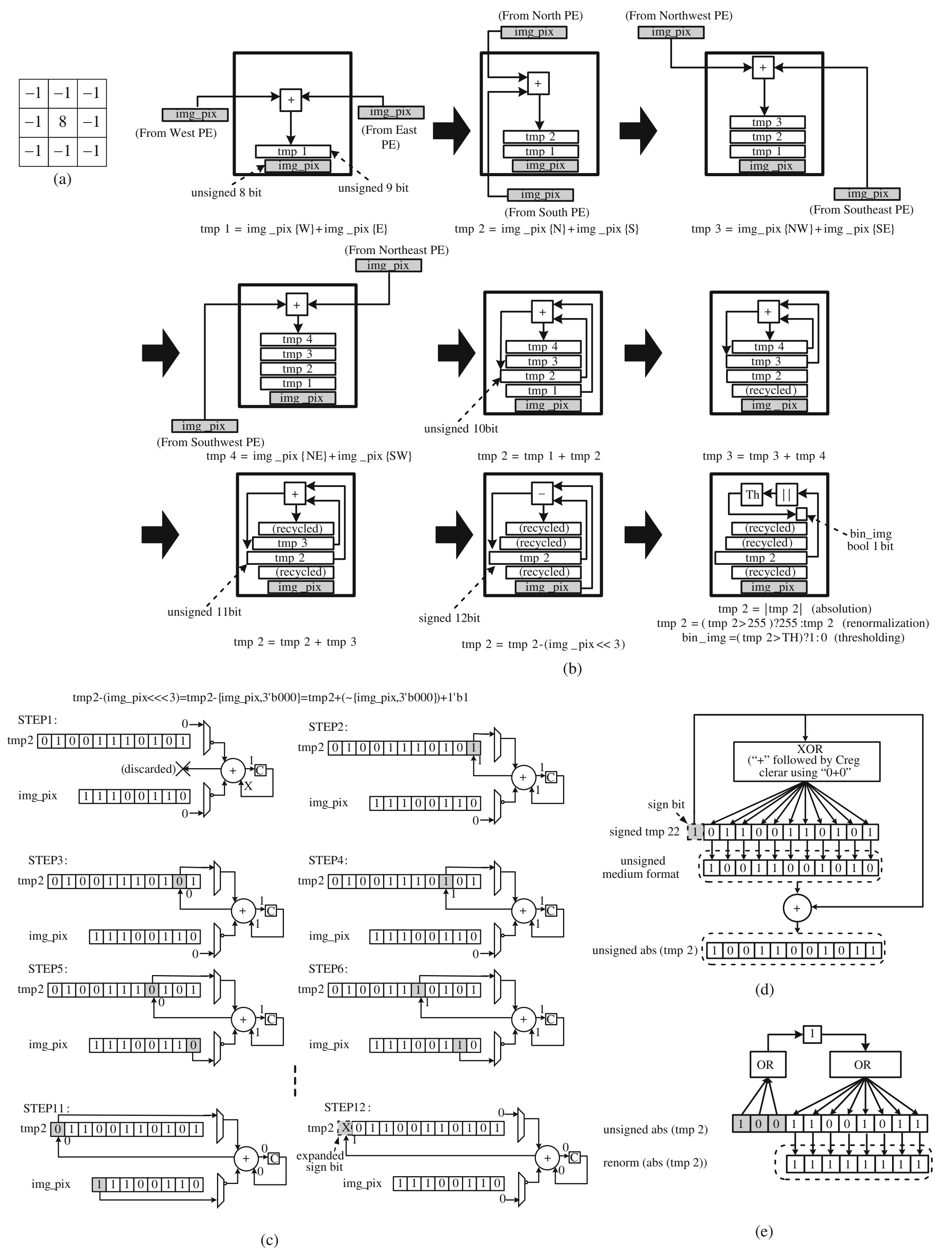

Figure 4 PE operations. (a) Laplacian template; (b) edge detection algorithm flow; (c) shift-subtraction on PE, one step per cycle; (d) absolution operation; (e) renormalization operation.

the absolution operation of the result within each pixel. The ×8 operation can be realized by left-shifting 3 bits. Next, a renormalization operation truncates these absolute values to range [0,255] for possible

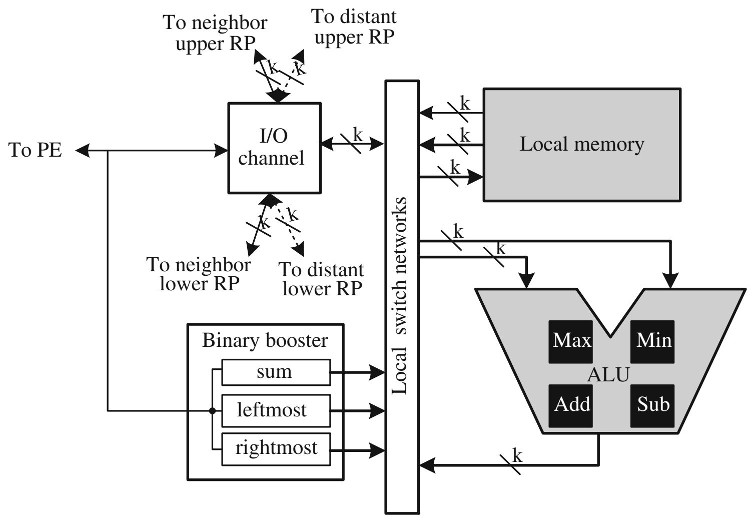

Figure 5 RP processor circuit.

visual debugging; finally, the thresholding operation segments out the pixels on edges. Figure 4(b) focuses on the process of filtering an image with Laplacian template shown in Figure 4(a), and shows how PE array finishes this task. Although only one PE is drawn for clarity, in fact, the same process is simultaneously executed on whole PE array in SIMD manner. In Figure 4(a), the variable named img_pix represents an 8-bit pixel value in the image, and E, W, S, N, etc in following bracket indicate the neighbor relationship among corresponding variables. The variables named tmp1-tmp4 are used to store temporary results during processing. The memory space for variables is dynamically allocated, expanded and recycled throughout the task. The memory allocation can be traced manually or by a dedicated compiler during programming phase. Compared with [18], the clock cycles consumed for Laplacian filtering on our PE array is largely reduced from over 190 down to 80. Figure 4(c) further explains the details of shift-subtraction in the second last stage of Figure 4(b). It is notable that shift-subtraction consumes only one more cycle (for pre-setting carry register to 1 using “1+1” expression) than addition of same bit-precision with the help of: 1) pre-inverting function in multiplexer groups, and 2) the introduction of constant “0” as ALU operands to eliminate real physical shift overhead. In the last stage of Figure 4(b), absolution is realized by XORing (1-bit addition followed by clearing carry register using “0+0” operation) the sign-bit with each of the other bits to build an unsigned format, and then adding the sign-bit to previous unsigned results, as shown in Figure 4(d). The absolute value is renormalized to [0,255] by: 1) first acquiring OR results of all bits located higher than the 8th bit-position to form a signature, and 2) then ORing the signature with each of the lowest 8 bits to produce the renormalized value, as shown in Figure 4(e).

3.2 Row processor

The RP array is mainly used to perform mid-level image processing [18]. The RP processor circuit is shown in Figure 5. It consists of a multi-bit ALU, a multi-port local memory, an on-the-fly binary booster, a local switch network and I/O channel for communication with nearest neighbor RP processors and the PE elements in the same row. These block circuits are connected by the local switch network. If one RP processor belongs to the skipping chain, its I/O channel would also support communication with two distant RP processors on the same chain. The ALU performs nonlinear operations and statistic operations, such as max/min selection for grayscale mathematical morphology and summation calculation for global feature extraction. The serial-to-parallel and parallel-to-serial data conversions between PE and RP are accomplished in the I/O channel.

The RP array can accelerate statistical feature extraction on the binary images. If the binary image is processed by a k-bit ALU, the padding operation from 1-bit binary image data to a k-bit format will result in timing overhead. In this work, we carefully analyze the commom operations of the most

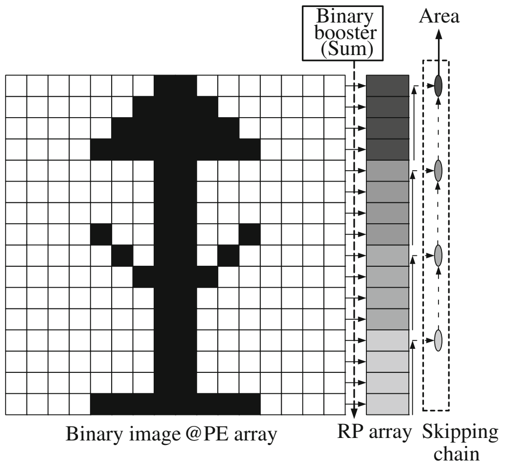

Figure 6 Fast calculation of binary image area on RP array.

frequently used binary image features, such as area, moment, bonding box and so on, and then employ a dedicated binary booster in each RP to directly perform the pixel number summation and the leftmost (rightmost) coordinate extraction operation of the active pixel data of PEs in the same row. An active pixel data is a 1-bit data of logic "1" in a binary image. Then ultimate binary features can be obtained by inter-RP processing operations. The row-parallel binary feature is extracted on-the-fly when a whole binary image on PE array is shifted into the RP array binary boosters column by column, without passing through I/O channel. The whole processing is finished in only $M2$ clock cycles, which is very fast. The bit-width k of RP ALU depends on the maximum gray-scale precision and the size of PE array. We assume the maximum gray-scale precision is K-bit, and the size of PE array is $M1$ rows by $M2$ columns. Because RP ALU will take some nonlinear operations on the image pixel data (e.g. max/min selection), k must be larger than or equal to K. On the other hand, the maximum numeric values collected by binary booster in each RP will not exceed $\log_2(M2)$ bits, so k must also be lager than this for processing convenience. In conclusion, we have

$$k \geqslant \max(K, \log_2 M2). \tag{1}$$

During some linear processing where data bit-width is larger than k, we can use same operations for multiple times to accomplish it, just as the bit-serial operation manner in PE elements. In a typical implementation where $M1$=$M2$=64, and the maximum gray-scale precision is standard 8-bit, k is inferred to be 8 as a result of (1).

Figure 6 shows how RP array speeds up area calculation for a binary image by binary boosters and skipping chain. Compared with previous RP array [18], a 12× acceleration is achieved. The selection of optimal skipping chain interval R can lead to fastest global operations. Here we assume an atom operation that equally takes one "unit" time between two RPs, which is always the case in real applications. We then assume a global operation based on that atom operation, for example, to sum up the feature data in each RP. As can be seen from Figure 6, the time consumed on skipping chain is

$$T = R + \frac{M1}{R}. \tag{2}$$

Then the optimal R_{opt} leading to smallest T is obviously the square root of $M1$, that is

$$R_{\text{opt}} = \sqrt{M1}. \tag{3}$$

However, sometimes (3) would yield non-integer R_{opt}. In such situation, R_{opt} should be rounded to its nearest integer. As in the same typical case described above where $M1$=$M2$=64, the optimal R should be set to 8 according to (3).

Table 1 FPGA resource ultilization

FPGA device	Cyclone-III EP3C120F484C8
Total logic elements	76905/119088 (65%)
Total registers	28089
Total pins	40/284 (14%)
Total memory bits	1034246/3981312 (26%)
Embedded multiplier 9-bit elements	0/576 (0%)
Total PLLs	1 / 4 (25%)

Table 2 Parameters in FPGA implementation

Parameter	Value	Description
$M1$	64	Number of rows in PE and RP array
$M2$	64	Number of columns in PE array
R	8	The interval of RP array skipping -chain
k	8	The datapath bit-width of RP ALU
P	64	The depth of PE local memory
Q	32	The depth of RP local memory

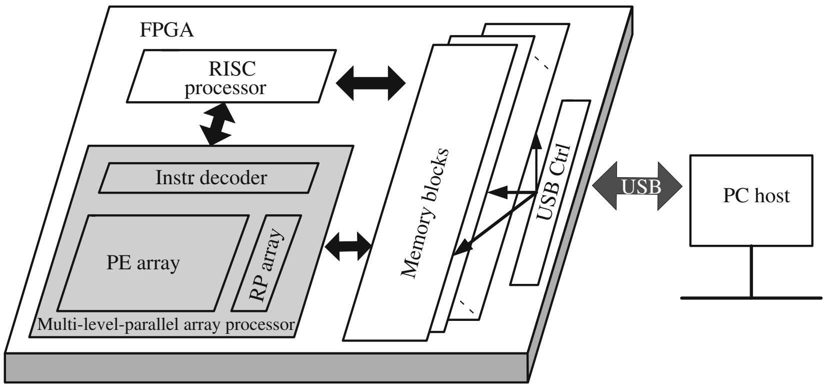

Figure 7 FPGA implementation.

4 Implementation and performance

4.1 FPGA Implementation

In order to evaluate proposed array processor, the proposed multi-level-parallel array processor and other blocks (RISC processor, several program and data memories) were implemented on an FPGA device. The information about consumed FPGA hardware resource is listed in Table 1. And the important parameters of the processor implementation are listed in Table 2. These parameters are easily scalable for different applications. Static timing analyzer (STA) showed that the implemented processor can operate at a maximum clock frequency of 50 MHz. The FPGA device is linked to a personal computer (PC) through USB channel to build an evaluation system, as shown in Figure 7. Instructions for multi-level-parallel array processor and RISC processor were first developed and compiled on PC, and then sent to memories on FPGA device. The PC was also responsible for sending testing images and video frame sequences to FPGA memories. Then the testing image (frame) data was immediately fed into the array processor through a DMA-like path to avoid system performance degradation. The RISC processor performs three kinds of operations: 1) issuing instructions to multi-level-parallel array processor; 2) sending processed images and extracted features to PC for monitor and debug; 3) managing chip operation.

Table 3 Performance comparison

	Ref. [18] 100 MHz ASIC, PE array + RP array		Ref. [19] 75 MHz ASIC, PE array		This work 50 MHz FPGA, PE array + RP array		Expected future 100 MHz ASIC, PE array + RP array	
	Clock cycles	Micro seconds	Clock cycles	Micro seconds	Clock cycles	Micro seconds	Clock cycles	Micro seconds
3×3 weighted avg.	66	0.660	N/A	N/A	30	0.600	30	0.300
3×3 nonweighted avg.	83	0.830	N/A	N/A	39	0.780	39	0.390
3×3 Sobel	62	0.620	420	5.590	31	0.620	31	0.310
3×3 Laplacian	194	1.940	N/A	N/A	80	1.600	80	0.800
3×3 Binary erosion	6	0.006	10	0.130	4	0.008	4	0.004
3×3 Binary dilation	6	0.006	10	0.130	4	0.008	4	0.004
Global thresholding	19	0.190	35	0.468	9	0.180	9	0.090

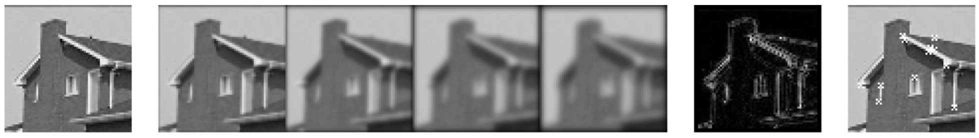
(a) (b) (c) (d)

Figure 8 Processing result of pixel-parallel algorithms. (a) Original image; (b) heat diffusion results at discrete iteration times of n= 1, 5, 10, 16, respectively; (c) DoG result; (d) SUSAN corner detection result, the detected corners are labeled with "×".

4.2 Pixel-parallel algorithms

We evaluated the array processor performance for some basic parallel image processing algorithms [24]. The size of testing image is 64 × 64, which is the same as the PE array size in the array processor. Table 3 shows its performance and comparison results with other vision chips [18–19]. We did not take into account the impact of different array sizes on performance, because these array processor architectures can all be easily scaled to any size without performance loss on these pixel-parallel algorithms. So the only fair metric is the processing time on each array for a frame whose size is the same as the array. For example, in proposed array processor, 64×64 images are used for testing, while in [18] and [19], 32×128 and 19×22 images are used, respectively. We showed consumed processing time in terms of both clock cycles and micro seconds. The proposed array processor takes much fewer cycles to finish these algorithms, though their absolute consumed time is about the same (see Table 3). However, this is due to the clock frequency limitation on FPGA implementation. If in future we transplant proposed array processor onto an ASIC implementation using, for example, 0.18 μm CMOS technology as the author did in [18], a higher clock frequency of 100 MHz would be achieved. And the absolute processing speed can be much improved, just as listed in the last column of Table 3.

We have also performed several more complex algorithms, such as heat diffusion equation. Difference of Gaussian (DoG) and SUSAN corner detector, on the proposed array processor to evaluate its flexibility at high speed. Figure 8(a) is the 64 × 64 original image. And Figure 8(b) shows the image results processed by heat diffusion equation algorithm at several discrete iteration times n= 1, 5, 10, 16, respectively. This algorithm simulates the heat diffusion phenomenon in the real world, and each image pixel value represents the temperature on that location. As a result, each pixel value would diffuse to its surroundings in this algorithm, and the image would be more and more blurred along time (discrete iterations). This algorithm can be applied as a noise suppression kernel in the Partial Differential Equation (PDE) image processing methodology [25]. Figure 8(c) shows the image result processed by difference of Gaussian operator. This operator first uses two Gaussian filters with different derivations to smooth the

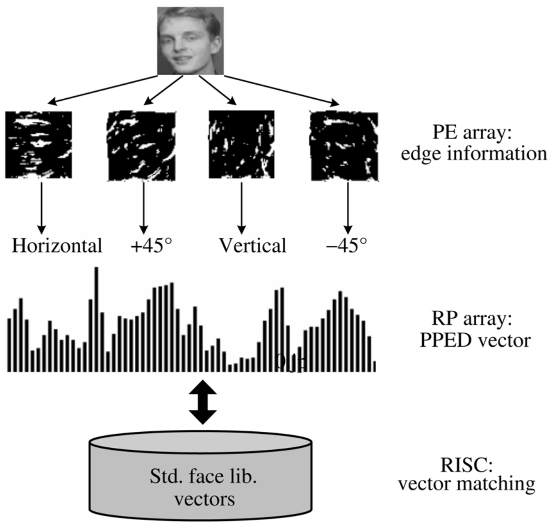

Figure 9 PPED algorithm for face detection.

image respectively, and then subtracts one of the smoothed images from the other. The DoG operator can be used to detect edges or keypoints, and is the foundation of currently popular scale-invariant algorithms [26]. Figure 8(d) shows the result of SUSAN corner detector [27]. This detector holds a circular template at a particular location in the image, and counts the number of template-covered pixels that have inconspicuous gray difference from the pixel at template center. Then the center pixel is labeled as a corner if counted number is smaller than a pre-defined threshold. When the circular template scans all pixels as the center, all corners are detected. This detector can be carried out at a high speed on the proposed array processor in pixel-parallel. The detected corners are labeled with symbol "×" in Figure 8(d). The processing time for heat diffusion equation (per iteration), DoG operator and SUSAN detector (radius = 3 measured in Manhattan metric) on the proposed array processor is 1.220 μs, 3.056 μs and 11.760 μs, respectively. The processing speed is very fast.

4.3 Face detection

We used the proposed array processor to perform a real-time face detection operation. The face detection is based on PPED feature extraction algorithm [28]. The PPED algorithm first produces directional edge information from the original image, and then compresses the edge information into a 64-dimension feature vector. Next, a vector match is carried out between this PPED vector and human face library vectors. Finally, the match result decides whether a human face exists in the original image. The processing flow is roughly described in Figure 9. The edge information detection and feature vector generation are performed by PE array and RP array of the array processor. The matching operation is carried out by the RISC processor implemented in the evaluation FPGA device. The proposed array processor only consumes about 1770 clock cycles to generate PPED feature vector for a 64 x 64 image. While the previous work [18] needs 12000 cycles for a 128×128 image, that equals to about 3000 cycles for each 64×64 sub-image. Therefore, the proposed array processor is about 1.7 times faster than the previous digital vision chip. Since PPED feature extraction takes most of processing cycles in face detection, the proposed array processor largely speeds up the application. We have already successfully implemented the PPED-based face detection algorithm into the metro pedestrian statistics system [29].

5 Conclusion

This paper proposed a high speed multi-level-parallel array processor for programmable vision chips. The array processor contains 2D PE array for pixel-parallel image processing and 1D RP array for row-parallel image processing. The two arrays both operate in an SIMD fashion. Each PE element can visit its nearest and next near neighbor PEs and speeds up local operations in low-level image processing. Meanwhile, the novel skipping chain and binary boosters in RP array accelerate global operations in mid-level image processing. The PE array size and RP array size are scalable for different vision application requirements. We found the optimal bit-width $k= 8$ for RP processor and optimal interval $R= 8$ for RP array skipping chain.This array processor was implemented on an FPGA device. And various parallel image algorithms including real-time face detection were used to evaluate the processor performance. The results show the proposed array processor outperforms the state-of-the-arts digital programmable vision chips in terms of low- and mid- level image processing speed.

Acknowledgements

This work was supported by National Natural Science Foundation of China (Grant Nos. 60976023, 61234003), and Special Funds for Major State Basic Research Project of China (Grant No. 2011CB932902).

References

1 Aizawa K. Computational sensors–vision VLSI. IEICE Trans Inf Syst, 1999, 82: 580–588

2 Ishikawa M, Ogawa K, Komoro T, et al. A CMOS vision chip with SIMD processing element array for 1 ms image processing. In: Proceedings of IEEE International Solid State Circuits Conference(ISSCC), San Francisco, 1999. 206–207

3 Ohta J. Smart CMOS Image Sensors and Applications. New York: CRC Press, 2007. 1–7

4 Komoro T, Ishii I, Ishikawa M, et al. A digital vision chip specialized for high-speed target tracking. IEEE Trans Electron Dev, 2003, 50: 191–199

5 Oike Y, Ikeda M, Asada K. A 375 × 365 high-speed 3-D range-finding image sensor using row-parallel search architecture and multisampling technique. IEEE J Solid-State Circ, 2005, 40: 444–453

6 Ishii I, Yamamoto K, Kubozono M. Higher order autocorrelation vision chip. IEEE Trans Electron Dev, 2006, 53: 1797–1804

7 Stoppa D, Pancheri L, Scandiuzzo M, et al. A CMOS 3-D imager based on single photon avalanche diode. IEEE Trans Circ Syst-I, 2007, 54: 4–12

8 Leon-Salas W D, Balkir S, Sayood K, et al. A CMOS imager with focal plane compression using predictive coding. IEEE J Solid-State Circ, 2007, 42: 2555–2572

9 Choi J, Han S W, Kim S J, et al. A spatial-temporal multiresolution CMOS image sensor with adaptive frame rates for tracking the moving objects in region-of-interest and suppressing motion blur. IEEE J Solid-State Circ, 2007, 42: 2978–2989

10 Chi Y M, Mallik U, Clapp M A, et al. CMOS camera with in-pixel temporal change detection and ADC. IEEE J Solid-State Circ, 2007, 42: 2187–2196

11 Kim D, Han G. A 200 µs processing time smart image sensor for an eye tracker using pixel-level analog image processing. IEEE J Solid-State Circ, 2009, 44: 2581–2590

12 Horiuchi T K. A low-power visual-horizon estimation chip. IEEE Trans Circ Syst-I, 2009, 56: 1566–1575

13 Zhao B, Zhang X, Chen S. A 64 × 64 CMOS image sensor with on-chip moving object detection and localization. IEEE Trans Circ Syst Vid, 2012, 22: 581–588

14 Komuro T, Kagami S, Ishikawa M. A dynamically reconfigurable SIMD processor for a vision chip. IEEE J Solid-State Circ, 2004, 39: 265–268

15 Miao W, Lin Q Y, Zhang W C, et al. A programmable SIMD vision chip for real-time vision applications. IEEE J Solid-State Circ, 2008, 43: 1470–1479

16 Yamashita H, Sodini C. A CMOS imager with a programmable bit-serial column-parallel SIMD/MIMD processor. IEEE Trans Electron Dev, 2009, 56: 2534–2545

17 Cheng C C, Lin C H, Li C T, et al. iVisual: an intelligent visual sensor SoC with 2790 fps image sensor and 205 GOPS/W vision processor. IEEE J Solid-State Circ, 2009, 44: 127–135

18 Zhang W C, Fu Q Y, Wu N J. A programmable vision chip based on multiple levels of parallel processors. IEEE J Solid-State Circ, 2011, 46: 2132–2147

19 Lopich A, Dudek P. A SIMD cellular processor array vision chip with asynchronous processing capabilities. IEEE Trans Circuits Syst-I, 2011, 58: 2420–2431

20 Rodriguez-Vazquez A, Linan-Cembrano G, Carranza L, et al. ACE16k: the 3rd generation of mixed-signal SIMD-CNN ACE chips towards VSoCs. IEEE Trans Circ Syst-I, 2004, 51: 851–863

21 Dudek P, Hicks P J. A general-purpose processor-per-pixel analog SIMD vision chip. IEEE Trans Circuits Syst-I, 2005, 52: 13–20

22 Dubois J, Ginhac D, Paindavoine M, et al. A 10000 fps CMOS sensor with massively parallel image processing. IEEE J Solid-State Circ, 2008, 43: 706–717

23 Lin Q Y, Miao W, Zhang W C. A 1000 frame/s programmable vision chip with variable resolution and row-pixel-mixed parallel image processors. Sensors, 2009, 9: 5933–5951

24 Gonzalez R C, Woods R E. Digital Image Processing. 3rd ed. NJ: Pearson Education, 2009. 182–184

25 Perona P, Malik J. Scale-space and edge detection using anisotropic diffusion. IEEE Trans Pattern Anal, 1990, 12: 629–639

26 Lowe D G. Distinctive image features from scale-invariant keypoints. Int J Computer Vision, 2004, 60: 91–110

27 Smith S M, Brady J M. SUSAN–a new approach to low level image processing. Int J Computer Vision, 1997, 23: 45–78

28 Yaga M, Shibata T. An image representation algorithm compatible with neural-associative-processor-based hardware recognition system. IEEE Trans Neural Network, 2003, 14: 1144–1161

29 Shi C, Wu N J, Wang Z H. A high-speed vision processor based on pixel-parallel PE array and its applications. In: Proceedings of IEEE Youth Conference on Information Computing and Telecommunications, Beijing, 2010. 57–60