AN AREA-EFFICIENT MEDIAN FILTERING IC FOR IMAGE/VIDEO APPLICATIONS*

Po-Wen Hsieh, Jer-Min Tsai, and Chen-Yi Lee
Dept. of Electronics Eng. and Institute of Electronics,
National Chiao Tung University,
1001, University Road, Hsinchu 300, Taiwan, ROC

ABSTRACT

An area-efficient IC for high-throughput median filtering applications is presented in this paper. This IC implements a modified delete-and-insert sorting algorithm which is very efficient for running order statistics applications. In hardware design, we first map the algorithm onto a regular PE structure, where each PE consistis of shift register, comparator, and some control gates. Then we conduct full-custom circuit/layout design of the PE to meet performance requirement. A proto-type chip for 64 input samples is implemented and tested. Results show that clock rate up to 50 MHz can be achieved using a 1.2 μm CMOS double metal technology. Two outstanding features of this IC are: *(1) any specified order of input patterns can be produced within one clock cycle; (2) each chip can handle at most 64 data and can be cascaded as the number of sorted data is over 64.* Thus this IC releases the bottle-neck of median search in hardware realization for many system designs, making real-time performace achievable.

1. INTRODUCTION

Image median filters are well known for being able to remove impluse noise [1], to preserve image edges, and recently to improve coding efficiency [2] as well as to enhance color signal processing [3]. Many algorithmic approaches with hardware implementation can be found in the literature [1, 4,5,6] and they show that the applicability of this median filtering technique is very application-dependent. Moreover different windows and pixel locations may be required to achieve specified behavior for different target applications.

In principle, these algorithms and methods can be classified into two categories: word-level and bit-level as discussed in [6]. In this paper, only the word-level median filters are concerned since they offer high throughput capability as required in many real-time image/video systems. However a very cost-effective hardware solution to meet this goal is often difficult to achieve and hence system performance becomes degraded to allow trade-off between hardware cost and achievable performance. For example, a fast median filter based on bubble sorting algorithm can be found in [5]. By means of a set of processing elements or PEs, the required values can be obtained with a latency of N cycles, where N is the number of input samples. However the hardware overheads of a parallel bubble sorter increase rapidly with the number of input samples. In addition to this sorting kernel, it is necessary to provide extra hardware in the form of a data buffer to rearrange input samples for the parallel processing and hence increase the memory bandwidth. Another solution is a message passing method [8] realized on systolic array architecture [9]. Both deletion and insertion messages pass through the systolic arrays until certain conditions are encountered. Although the hardware complexity depends on the number of input samples (N), the latency remains the same as that needed in the parallel bubble sorter. This latency of N cycles may not be allowed when real-time performance is concerned.

In this paper, we present a more cost-effective hardware solution which can be integrated with other hardwares without degrading overall system performance. This is achieved by reducing the latency of median search on a set of input samples so that the median can be obtained immediately without causing a stall on the data flow. For example, the median can be obtained right after the last sample is presented and then immediately passed to next stage. In section 2, an algorithm suitable for such an implementation is first introduced. The main idea is to partition the sorted items into two groups and then perform shfit operations on one of the groups according to the operation mode. Section 3 presents a shiftable content address memory (SCAM) VLSI architecture which is a processor element (PE) based structure. Each PE contains two different cells– one (sort-cell) stores sorted items and the other (compare-cell) compares current input sample with the sorted items so that it can be placed appropriately to reach high speed sorting. A proto-type chip based on this design approach is given in section 4 to evaluate design efficiency. Also some comparisons with available approaches are included to highlight the performance of the SCAM VLSI architecture for sorter-based applications.

*WORK SUPPORTED BY THE NATIONAL SCIENCE COUNCIL OF TAIWAN, ROC UNDER GRANT NSC-82-0404-E009-184.

Manuscript received March 8, 1993

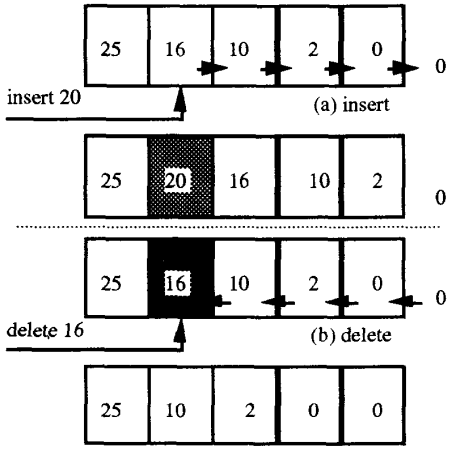

Fig.1. Illustration of the Delete and Insert sorting operations.

2. ALGORITHM DESCRIPTION AND TRANSFORMATION

Since median search can be decomposed into two stages– (1) sorting input samples and (2) selecting specified order from sorted sequence, and the former is much more complex than the latter, throughput can only be improved when complexity of the sorting stage is overcome. We select the delete-and-insert algorithm [8] due to its low memory bandwidth requirement and being suitable for raster-scan sequence.

The Delete-and-Isert Sort Algorithm The delete-and-insert sort algorithm is one of the many available sort algorithms. Its operations can be described as follows. For data insertion as shown in Figure 1(a), the current input sample is known to be 20 and should be placed in between 25 and 16 which are already sorted and stored in a memory. Each time when a new sample is given, it can be placed according to this scheme. After all samples are processed, the sorted sequence can be obtained from the memory. For data deletion, it only needs to identify the sorted item whose value is equal to the input sample's value as shown in Figure 1(b). Thus for any running order operations, this delete operation becomes useful since only those samples out of the mask region need to be removed and new input samples need to be inserted. In addition to these features, this algorithm can also provide ascending sequence if smaller items are placed at left side instead of right side as used for descending sequence (as illustrated in Figure 1).

This algorithm description implies that a lot of comparisons are needed in order to allocate a position in which the input sample can be correctly placed or removed. Moreover a lot of move operations are needed once the input sample is inserted or deleted. For real-time image/video processing, either insert or delete operation has to be done within a very stringent timing constraint. Thus if this algorithm is implemented on general-purpose computer, the result is obviously not suitable for real-time applications and hence the algorithm has to be modified. A modified version, called optimized delete-and-insert or ODI sort algorithm is thus developed. It fully explores parallelism so that a very high throughput requirement can be obtained by exploiting VLSI advantages such as speed and density.

The ODI Sort Algorithm As described above, the bottleneck of enhancing the delete-and-insert sorter's throughput lies in two factors– *(1) identification of insert/delete target and (2) data movement among sorted items.* The complexity of these two factors becomes higher as the number of input samples increases. However if we explore fully parallelism inherent in the algorithm, then this bottleneck can be coped with. Here it is found that for each input sample, N comparisons are needed in order to find the position where the input sample should be placed or removed. This parallel-comparison process can be realized on a content addressable memory (CAM) to identify the target to be inserted or deleted. Once the target is identified, the next step is to perform data movement on part or all the sorted items. For example, Figure 1(a) illustrates the insertion of data item whose value is 20. For those sorted items whose value is less than or equal to the input item, they have to move one position right; while in data deletion, those sorted items whose value is less than or equal to the input item have to move one position left as shown in Figure 1(b). In other words, the sorted items are divided into two groups– one is the LE group (i.e. sorted items less than or equal to input sample) and the other is the GT group (i.e. sorted items greater than input sample). Thus the data movement can be replaced by shift operations working on the LE group instead of read/write operations on memory and in particular, these shift operatios can be executed in parallel. Thus this modified ODI algorithm operates on every sorted items simultaneously and generates any order if both concepts of content-addressable memory and shift registers are used.

3. ARCHTIECTURE AND CIRCUIT DESIGN ISSUES

In the previous section, we discussed how the ODI algorithm solved the bottleneck to speed up throughput by means of the support of content addressable memory and shift registers. In this section, we discuss in more detail how the ODI algorithm is mapped onto the shiftable content-address memory (SCAM) architecture which is very suitable for VLSI implementation.

The PE-Based Structure Suppose that there are N samples to be sorted, we can obtain any specified order right after the last sample is given. This is the specification of our high-speed median filter design. We also assume that the sorted sequence is in a descending way. Thus N storage spaces are needed to store the sorted samples. Initially all contents of SCAM are reset to zero. For each input sample,

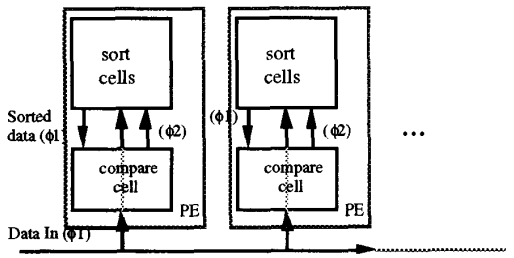

Fig.2. The shiftable content address memory is realized on a PE structure.

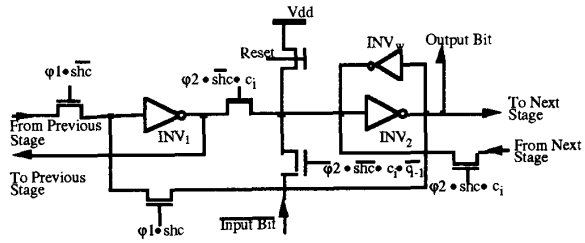

Fig.3. Circuit diagram of the sort-cell. Note that a weak inverter is placed at INV_2 to overcome leakage paths.

the content of each storage space is read out and compared with the input sample in order to identify the position where the input sample should be placed. Once the position is allocated, the content of each storage space in the LE group has to be shifted. This implies that the architecture consists of N processor elements (PEs) and each of which contains two basic cells– (1) sort-cell and (2) compare-cell (as shown in Figure 2). The former is a shift register containing the sorted data and can be shifted right or left while the latter is designed to provide control signals orchestrating how the former should be operated.

Detailed Architecture and Circuit Design for the Sort Cell From the previous discussion, it can be found that the sort-cell should have the following functionalities:

1. shift data right,
2. shift data left,
3. load data,
4. data remains unchanged,
5. reset content.

Here items 1, 3, and 4 are required in insertion operations, while items 2 and 4 are needed in deletion operations. Item 5 is only used when a new input set is to be processed. With these defined functionalities, the corresponding circuit for such a cell can be easily derived as shown in Figure 3. Only 3 inverters and 6 pass transistors are needed for each bit cell. The first inverter acts as an internal buffer for data pre-shifting to enhance clock rate (see the last paragraph of this section for more details). Note that a two-phase non-overlapping clocking strategy is used here and all control gates are functioning at $\phi 2$.

Detailed Architecture and Circuit Design for the Compare Cell This cell is designed to generate all required control signals for those used in the sort cell. Here different control signals are generated respectively when the following conditions are encountered:

- shift data right (*shr*): This occurs when data insertion is under execution. However only those sorted items whose value is less than or equal to (or belong to the LE group as defined above) that of current input sample will be shifted right and hence the carry (C_i in *ith* compare-cell) is demanded. Thus the *shr* is activated whenv *shc* is low (for insertion) and C_i is high.
- shift data left (*shl*): This *shl* is activated when data deletion is performed (i.e. *shc* is high) and C_i is high.
- load data (*load*): Since only one PE requires this signal, i.e. the input sample can only be placed at one correct position, the generation of *load* signal becomes more complex than the previous two control signals. Not only *shc* and C_i are considered, but also the carry from the previous stage (C_{i-1}) should be taken into account. This is obviously since the correct storage space for current input sample is in between those items in the GT group (or greater than the input sample, i.e. C_{i-1}=0) and LE group (or those items less than or equal to the input sample, i.e. C_i=1).

To speed up the carry generation, a carry-look-ahead technique [10] is exploited. The overall circuit diagram for this compare-cell is shown in Figure 4.

Operation and Clocking Stragety To speed up the clock rate, a 2-phase non-overlapping clock is exploited here. All the control signals are generated at $\phi 1$ and data shift operations are performed at $\phi 2$. In addition, a *pre-shift* strategy is also exploited to improve clock speed. This strategy shifts sorted items to the buffer (the first inverter INV_1 of the sort-cell) of next PE during $\phi 1$ and then are conditionally stored at $\phi 2$. The detailed operation for data deletion on the sort-cell is illustrated in Figure 5. Note that the first inverter of each sort cell acts as a buffer during pre-shifting as needed for both shift right and left operations.

4. EVALUATION AND DISCUSSIONS

To evaluate this SCAM architecture, a proto-type chip for maximum of 64 input samples has been fabricated and tested. In this section, we first present some experimental and test results about this ODI chip and then provide some comparisons with the two approaches described in the Introduction.

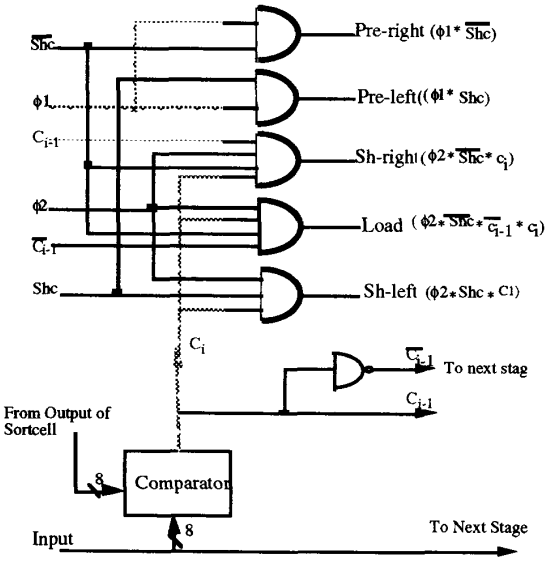

Fig.4. Circuit diagram of the compare-cell. The carry is generated by the carray-look-ahead technique to enhance clock speed.

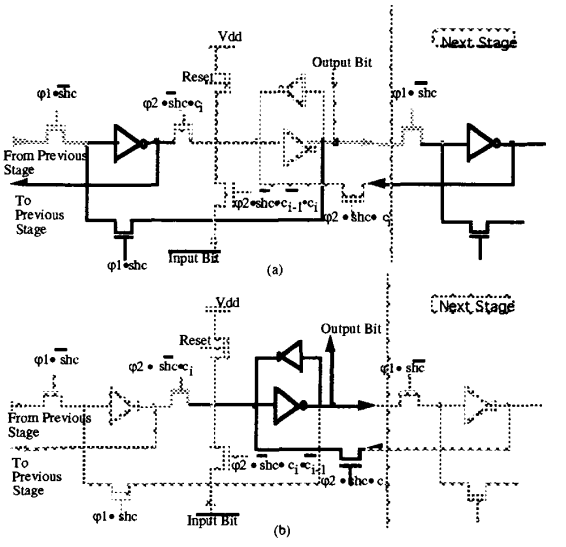

Fig.5. Delete data flow on the sort-cell. (a) pre-shift stage at $\phi 1$ and (b) data shift at $\phi 2$. Note the dark lines indicate how the circuit works at different phase.

Design of the ODI Chip Block diagram of this chip is given in Figure 6. This chip can be used either as a high-speed data sorter or as an image median filter. Input samples are first sorted through the shiftable content address memory. Then the specified order, such as median, can be identified from the sorted items by means of a dynmäic selection circuit. For raster scan images, this chip provides an optimal solution since the required order can be obtained right after the last sample is given. It should be noted that using the pre-shift strategy, both phases ($\phi 1$ and $\phi 2$) are quite balanced. Experimental results show that clock rate up to 50 MHz can be achived. Die photo of this chip is shown in Figure 7. Some key features of this ODI chip are given below:

- Each chip can handle 64 samples and can be cascaded for more samples when data sorting is considered.
- Any specified order can be obtained right after the last sample is given, i.e. without latency. This is very useful for pipelined architecture design.
- Running order can be handled by exploiting both the deletion and insertion operations.
- Design is very regular and hardware complexity is linearly proportional to the number of input samples.

Detail specifications of this chip are given below:

- Die size: 5053 μm × 4774 μm;
- Pin-count: 67;
- Transistor-count: N-type: 13060, P-type: 10141, Totally: 23201;
- Clock rate: 50 MHz;
- Technology: 1.2 μm CMOS double-metal technology.

Comparisons with Other Approaches We only compare the results based on the bit-parallel approaches as given in the Introduction since they are more practical for real-time image/video applications.

The parallel bubble sorter as decribed in [5] requires a lot of hardware as the number of input samples increases. In addition, its input format has to be adjusted so that a set of input samples can be fed simultaneously to the sorter arrays. This implies that large memory bandwidth is needed. In addition, given a set of N input samples, the required order can only be obtained right after N cycles. Obviously, our ODI chip does outperform than this bubble sorter in terms of area and latency.

The ROS sorter from [8] based on systolic architecture offers similar hardware complexity compared to our design. However our design offers higher throughput and expansibility.

In addition to the specific features of less area and no data latency, our design also provides a testable strategy since test patterns can easily be applied to the SCAM and detected from the selection circuit or from both shift output ports.

5. CONCLUSION

In this paper, we have preseneted a high-throughput median filter design based on the ODI algorithm and the SCAM architecture. This chip is obtained by first exploring the

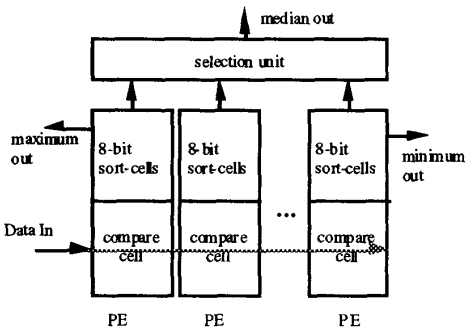

Fig.6. Block diagram of the ODI chip.

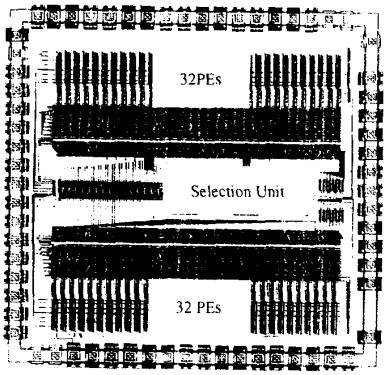

Fig.7. Plot of the ODI chip design.

inherent parallelism and then exploiting shift operations to achieve high speed sorting so that any order of input samples can be obtained. Using the shiftable content address memory architecture, we have developed a very cost-effective hardware solution for median search as well as for data sorting. Final test results show that the ODI chip does outperform than available approaches for median search in both area and throughput. Also this area-efficient solution can be integrated with other hardware when median serarch is demanded in system development. We are currently looking into the applicability of exploiting this high-speed sorting architecture for adaptive coding which is often used in entropy coding to enhance compression ratio.

Acknowledgement: The authors would like to thank their colleagues within the VLSI/CAD group of NCTU for many fruitful discussions. Also the MPC support from Chip Implementation Center (CIC) of NSC is acknowledged.

REFERENCES

[1] A.K. Jain, *"Fundamentals of Digital Image Processing"*, Prentice-Hall, 1989.

[2] D.H. Kang, J.H. Choi, Y.H. Lee, and C. Lee, *"Applications of a DPCM System with Median Predictors for Image Coding"*, IEEE Trans. on Consumer Electronics, Vol. 38, No. 3, pp. 429-435, Aug. 1992.

[3] H. Rantanen, M. Karlsson, P. Pohjala, and S. Kalli, *"Color Video Signal Processing with Median Filters"*, IEEE Trans. on Consumer Electronics, Vol. 38, No. 3, pp. 157-161, Aug. 1992.

[4] H.M. Lin and A.N. Jr. Willson, *"Median Filters with Adaptive Length"*, IEEE Trans. on CAS, Vol. 35, No. 6, pp. 675-690, June 1988.

[5] J. Offen and R. Raymond, *"VLSI Image Processing"*, McGraw-Hill, 1985.

[6] C.L. Lee and C.W. Jen, *"Bit-sliced Median Filter Design Based on a Majority Gate"*, IEE Proc-G V139, No 1, pp. 63-71, Feb. 1992.

[7] V.V. Bapeswara Rao and K. Sankara Rao, *"A New Algorithm for Real-Time Median Filtering"*, IEEE Trans. on ASSP, Vol. ASSP-34, pp. 1674-1675, No. 6, Dec. 1986.

[8] A.L. Fisher, *"Systolic Algorithms for Running Order Statistics"*, in Signal and Image Processing, Dept. of Computer Science, Carnegie Mellon University, Pittsburgh, July 1981.

[9] H. T. Kung, *"Why Systolic architectures"*, IEEE Computer, Vol. 15, no 1, Jan., 1982.

[10] N. Weste and K. Eshraghian, *"Principles of CMOS VLSI Design- A Systems Perspective"*, Addison-Wesley, 1985, pp. 320-335.

Po-Wen Hsieh was born in Kaoshiung City, Taiwan, on November 10, 1967. He received the B.S. degree in Electronics Engineering from National Chiao Tung University in Jan. 1991. He is now a graduate student in the Institute of Electronics, National Chiao Tung University. His research interests include VLSI circuit design and image processing.

Jer-Min Tsai received the B.S. degree in Electrical Engineering from National Cheng Kung University, Tainan, Taiwan in 1989. He is now a graduate student in the Institute of Electronics, National Chiao Tung University. His research interests include high-speed circuit, VLSI architecture, and data compression.

Chen-Yi Lee received the PhD in Electrical Engineering from Katholieke University Leuven (KUL), Belgium. From 1986 to 1990, he was with IMEC/VSDM, working in the area of architecture synthesis. Since Feb. 1991, he has been an associate professor in the Dept. of Electronics Engineering at the National Chiao Tung University, Hsinchu Taiwan. His research interests mainly include video/image coding, digital communications, VLSI architectures, and ASIC design methodology.