A GENERAL PURPOSE VLSI MEDIAN FILTER AND ITS APPLICATIONS FOR IMAGE PROCESSING

Mustafa Karaman, Levent Onural, and Abdullah Atalar

Electrical and Electronics Eng. Dept., Bilkent University
POB. 8, 06572 Maltepe, Ankara, Turkey

ABSTRACT

A general purpose median filter configuration consisting of two single-chip median filters is proposed. One of the chips is designed for the applications requiring variable word-length and variable window size whereas the other one is for real-time applications. The architectures of the chips are based on the odd/even transposition sorting. The chips are implemented in 3-μm M^{2}CMOS by using full-custom VLSI design techniques. The chips together with a reasonable external hardware can be used for the realizations of many median filtering techniques. In this paper, the VLSI design procedure of the chips and their applications to different median filtering techniques for image processing are presented.

1. INTRODUCTION

The median of an odd number of elements is defined as the middle element when the elements are sorted. Output of a median filter is the median of its input data, and the resulting nonlinear smoothing filter can filter out the impulsive noises from signals and images while preserving the edge-information [1]. Such filters are frequently used in many signal and image processing applications. In terms of impulsive noise suppression, edge preservation, and ease of design, the performance of median filters are better than the other smoothing filters such as linear filters [2] and generalized mean filters [3].

In 1-D and 2-D standard median filtering applications, a window of size w, w is odd, moves on the sampled values of the signal or image, and then the median of the samples within the window is computed and written as the output element at the location of the center of the window. Theoretical analysis and applications of the median filters can be found in the literature [4,5]. Mostly, median filters are implemented in general purpose computers [6,7]. However, there are also hardware implementations for faster filtering purposes [8]. Because of the low VLSI cost of sorting structures, most of the hardware median filtering algorithms are based on sorting [9].

In order to increase the performance of the median filters for particular applications, various techniques such as weighted [10], separable [11], recursive [12], adaptive-length [13], generalized [14], selective [15], hybrid [16] median filtering techniques have been developed. The computation of the median of a group of elements is the fundamental operation in all those techniques. Thus, the standard median filters are used as the basic components for the realizations of other techniques.

The window size of the median filter and the word-length of the elements are not the same in different applications. Also, the required speed of the filtering operation varies depending on the application. In order to meet these changing demands, a general purpose VLSI median filter unit which consists of two single-chip median filters, one extensible and one real-time, is designed. The extensible median filter chip is designed for the applications requiring variable word-lengths and variable window sizes whereas the real-time median filter chip is for the real-time median filtering applications. The architectures of the chips are bit-level pipelined systolic structures based on the odd/even transposition sorting. The chips are implemented in 3-μm M^{2}CMOS by using full-custom VLSI design techniques. In the following sections, the architectures, VLSI implementations, and some possible applications of the chips are presented.

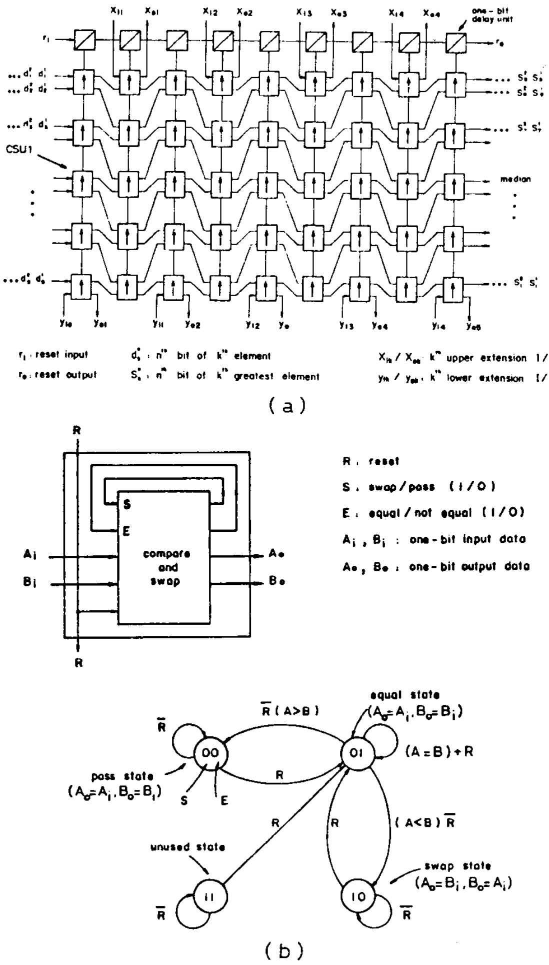
2. ARCHITECTURES

2.1 Extensible Median Filter Architecture

The extensible median filter is an odd/even transposition sorting network which is a pipelined regular structure consisting of 9 *compare-and-swap* stages (Fig.1.a). Each stage consists of 5 bitwise *compare-and-swap* units. Each of these units compares two one-bit numbers at its inputs and interchanges them if necessary so that the larger one is at the "top". At the output of the last stage, the data will be sorted such that the largest will be at the top, and the median will be in the middle. At each clock, one bit from each word (total of 9 bits) enter the network and one bit of the median is obtained at the output. The flow is from the most significant bits toward the least significant bits both at the input and at the output. Because of the bitwise serial data flow, this structure allows arbitrary word-length, L.

The bitwise compare-and-swap unit (CSU1) is a finite state machine which has three legal operation states: *equal*, *pass*, and *swap*. CSU1 is set to the equal state at the end of each data word by a reset signal. Thus the reset signal flows through the stages of the network at a rate of one stage per clock cycle by means of the pipelined delay units. CSU1 stays in equal state as long as its inputs are equal. However, it locks itself into one of the pass or swap states depending on its inputs and stays in that state until it is reset. The state diagram and the operations at different states are given in Fig.1.b.

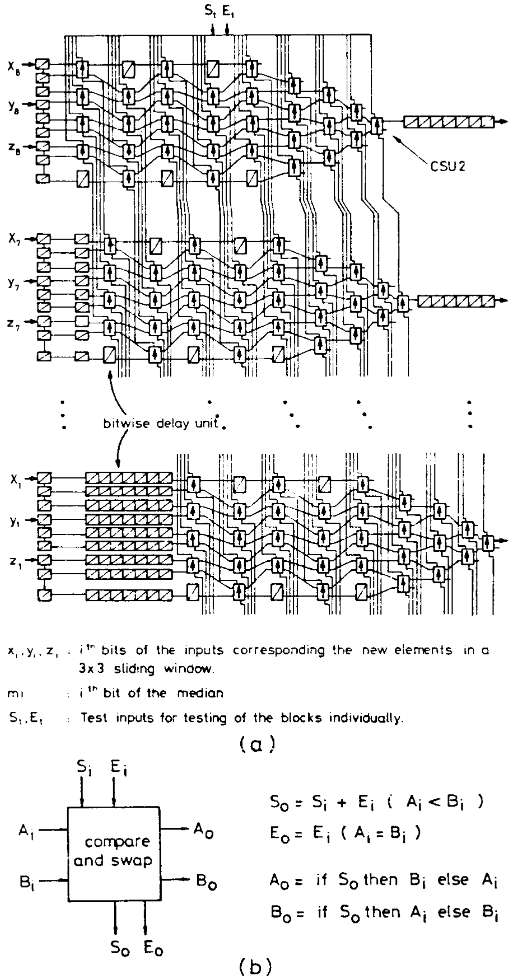
In the extensible median filter structure given in Fig.1.a, the *upper* and *lower extension* I/O's ($x_{i/o}$'s and $y_{i/o}$'s) are used to extend the filter to larger window sizes. For $w = 9$, the upper and lower extension inputs are connected to logic 1's and logic 0's so that the corresponding compare-and-swap units act as delay units. On the other hand, the design allows the interconnections of many of these chips to form median filters for $w > 9$.



(a)

(b)

Figure 1: The extensible median filter: a) architecture, b) compare-and-swap unit (CSU1).


St Et
X8
y8
z8
m8
CSU2
X7
y7
z7
m7
bitwise delay unit
X1
y1
z1
m1
xi, yi, zi : i^th bits of the inputs corresponding the new elements in a 3x3 sliding window.
mi : i^th bit of the median
St, Et : Test inputs for testing of the blocks individually.
Si
Ei
Ai
Bi
compare and swap
Ao
Bo
So
Eo
So = Si + Ei (Ai < Bi)
Eo = Ei (Ai = Bi)
Ao = if So then Bi else Ai
Bo = if So then Ai else Bi

(a)

(b)

Figure 2: The real-time median filter: a) architecture, b) compare-and-swap unit (CSU2).

The extensible median filter generates its outputs with a delay of $w + L$ clocks; and after the network is full, it finds one L-bit median per L clocks. Although, the resulting speed may be sufficient for the real-time median filtering of 512×512 frames with $L < 5$, it is not enough for the real-time filtering of 1024×1024 frames with $L > 1$.

2.2 Real-Time Median Filter Architecture

The real-time median filter is designed by interconnecting 8 odd/even transposition sorter blocks in parallel [9] (Fig.2.a). In this network, the data enter in such a way that the most significant bits go to the first block, the second most significant bits to the second block, and so on. The bitwise compare-and-swap unit used in this network is slightly different than that of the extensible one, because the"swap" or "pass" information flows from upper to lower block so that the compare-and-swap unit takes this information, uses, updates and sends it out (Fig.2.b). For proper timing, the delay units are included at the input and output of the network.

The real-time median filter has nine 8-bit data inputs and it generates one 8-bit median per clock. At every clock, three new elements enter the chip, corresponding to the new elements of a sliding 3×3 window. Since the clock period is determined by the delay of one compare-and swap unit (CSU2), recent VLSI technology allows the implementation of CSU2 at a speed larger than the real-time operation rate for the 1024×1024 frames with $L = 8$.

3. CHIPS

Both of the extensible and real-time median filter architectures are regular arrays of the bitwise compare-and-swap units. Also, their internal communication schemes are simple and regular. This makes the VLSI implementations easy and straightforward [17,18]. The architectures are mapped to hardware by using standard CMOS logic style [19] in 3-μ double metal n-well process. For generation of the chip layouts, and their simulations, full-custom VLSI CAD tools [20,21] are used: *magic* for layout editing, *Spice*, *Rnl*, and *Esim* for simulations. The overall layouts of the chips are shown in Fig.3.

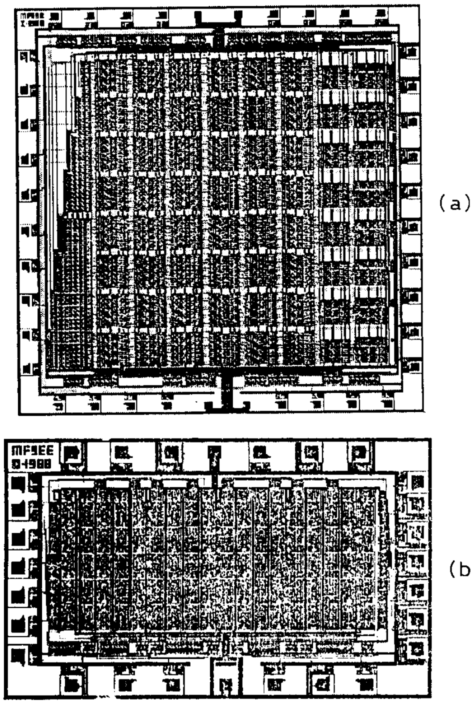


Figure 3: The layouts of the median filter chips: a) real-time, b) extensible.

According to the simulation results, the extensible median filter chip can run up to a clock frequency of 30 MHz with a power dissipation less than 250 mW at this frequency. The throughput of the chip is about $30/L$ mega medians/s. The chip consists of about 5000 transistors and has an area of 11.7 mm^2 (3 mm×3.9 mm) and 28 pins. On the other hand, the real-time median filter chip can run with a clock frequency up to 40 MHz with a power dissipation less than 800 mW at this frequency. It generates one median per clock so that its throughput is 40 mega medians/s. It consists of about 22000 transistors and has an area of 45 mm^2 (6.8 mm×6.6 mm) and 40 pins.

The testing of the chips are easily accomplished by the *functional* test techniques [22] since the operations of the cells can be selectively probed by using proper test vectors. The test vectors and the expected outputs are generated by using software tools written for these purposes. There are 500 test vectors for the extensible median filter chip, and 12,000 for the other one.

4. APPLICATIONS

In image processing applications, median filters are used mainly for noise suppression and for edge detection. For impulsive noise suppression, standard median filtering technique is a good choice. However, for suppression of nonimpulsive noises other techniques such as adaptive-length, separable, recursive, and weighted median filtering techniques may be more convenient. For edge detection, generalized, hybrid, and selective median filtering techniques are frequently used. In addition, the weighted median filtering can be also used for edge detection by choosing the weight coefficients properly.

The designed median filter chips can be selectively used in a processor environment by means of the chip enable signal that each chip has. Furthermore, one can realize any median filtering technique mentioned above by using the extensible and/or the real-time median filter chips together with or without a reasonable external hardware:

- For the standard median filtering technique, the exact medians of the elements, in a window size $w = 9$ with arbitrary word length L, can be found by using only one extensible median filter chip. For $w > 9$ with arbitrary L, at most $[[w/9]]^2$ ($[[.]]^2$ indicates the smallest greater integer) chips are required to find the exact medians. On the other hand, the real-time median filter chip can find the exact running medians of the elements in a window of a fixed size $w = 9$ with fixed word length $L = 8$ at the real-time rate.

- The extensible median filter is a favorable choice to realize the *adaptive-length* median filters [13], since one can change the window size from 3 to indefinitely large ones by using the extensible median filter chip(s) by applying logic 0's or 1's to unused inputs of the chip(s) appropriately.

- For the realizations of the *weighted* median filters [10], the extensible median filter can be used with a pipelined multiplier to multiply the input data with the weight coefficients. Since all input data of the chip are entered to the chip directly at each move of the window, one can realize an adaptive weighted median filter by changing the weight coefficients at each position of the window on the frame.

- **A pair of the extensible or the real-time median filter chips can be used as a *selective median* filter [15] together with an external control logic consisting of two full-word subtracter and a full-word comparator.**
- **Either the extensible or the real-time median filter chip can be used as a *line-recursive* median filter [13] by loading the window elements from the frame appropriately.**
- **The chips can be used for the realizations of the *separable* median filters [11] without any external hardware.**

5. CONCLUDING REMARKS

A general purpose VLSI median filter unit consisting of two single-chip median filters and its applications are presented. The architectures of the chips are modular and have regular communication schemes which make the VLSI implementations rather easy and straightforward. Both of the architectures are not preferable to be implemented at larger window sizes since the area is proportional to the w^2. We have chosen $w = 9$, because this is the most commonly used window size in two dimensional median filtering applications.

The main contributions of this study are the architecture of the extensible median filter and its VLSI implementation. Another achievement of this study is the implementation of the real-time median filter which can operate at the real-time rate for the 1024×1024 resolution frames.

ACKNOWLEDGMENT

This research was sponsored by NATO's Scientific Affairs Division in the framework of the Science for Stability Programme.

References

[1] J. W. Tukey, "Nonlinear (nonsuperposable) methods for smoothing data," in *Conf. Rec.*, p. 673, *EASCON* 1974.

[2] A. V. Oppenheim and R. W. Schafer, *Digital Signal Processing*, Englewood Cliffs, N.J.: Prentice-Hall, 1975.

[3] A. Kundu, S. K. Mitra, and P. P. Vaidyanathan, "Application of two-dimensional generalized mean filtering for removal of impulse noises from images," *IEEE Trans. Acoustic, Speech, and Signal Processing*, vol. ASSP-32, NO. 3, pp. 600-609, Jun. 1984.

[4] N. C. Gallagher, Jr., and G. L. Wise, "A theoretical analysis of the properties of median filters," *IEEE Trans. Acoustic, Speech, and Signal Processing*, vol. ASSP-29, pp. 1136-1141, Dec. 1981.

[5] E. Ataman and E. Alparslan, "Application of median filtering algorithm to images," Electronics Division, Marmara Research Institute, Gebze, Turkey, Tech. Rep. UI 78/10, Sep. 1978.

[6] E. Ataman, V. K. Aatre, and K. M. Wong, "A fast method for real-time median filtering," *IEEE Trans. Acoustic, Speech, and Signal Processing*, vol. ASSP-28, pp. 415-421, Aug. 1980.

[7] V. V. B. Rao and K. S. Rao, "A new algorithm for real-time median filtering," *IEEE Trans. Acoustic, Speech, and Signal Processing*, vol. ASSP-34, pp. 1674-1675, Dec. 1986

[8] D. L. Knuth, *The Art of Computer Programming-Searching and Sorting*, vol. 3. Reading MA: Addison-Wesley, 1973.

[9] K. Oflazer, "Design and implementation of a single-chip 1-D median filter," *IEEE Trans. Acoustic, Speech and Signal Processing*, vol. ASSP-31, pp. 1164-1168, Oct. 1983.

[10] T. Loupas, W. N. McDicken, and P. L. Allan, " Noise reduction in ultrasonic images by digital filtering, " *The British Journal of Radiology*, vol. 60, pp.389-392, Apr. 1987.

[11] T. A. Nodes and N. C. Gallagher, Jr., "Two-dimensional root structures and convergence properties of the separable median filter," *IEEE Trans. Acoustic, Speech, and Signal Processing*, vol. ASSP-31, pp. 1350-1365, Dec. 1983.

[12] C. G. Boncelet Jr., "Recursive algorithms and VLSI implementations for median filtering," Proc. of IEEE ISCAS'88, pp. 1745-1747.

[13] H. M. Lin and A. N. Willson, Jr., "Adaptive-length median filters for image processing," Proc. of IEEE ISCAS'88, pp. 2557-2560.

[14] Y. H. Lee and S. A. Kassam, "Generalized median filtering and related nonlinear filtering techniques," *IEEE Trans. Acoustic, Speech, and Signal Processing*, vol. ASSP-33, pp. 672-683, Jun. 1985.

[15] S. J. Ko, Y. H. Lee, and A. T. Fam, "Selective median filters," Proc. of IEEE ISCAS'88, pp. 1495-1498.

[16] Y. Neuvo, P. Heinonen, and I. Defee, "Linear-median hybrid edge detectors," *IEEE Trans. Circuits and Systems*, vol. CAS-34, pp. 1337-1343, Nov. 1987.

[17] M. J. Foster and H. T. Kung, "The design of special purpose VLSI chips," *IEEE Computer*, pp. 26-40, Jan. 1980.

[18] H. T. Kung, "Why systolic architectures?," *IEEE Computer*, pp. 37-46, Jan. 1982.

[19] N. Weste and K. Eshraghian,*Principles of CMOS VLSI Design*, Reading MA: Addison-Wesley, 1985.

[20] *Berkeley CAD Tools User's Manual*, EECS Dep., University of California at Berkeley, 1986.

[21] *VLSI Tools Reference Manual*, TR#87-02-01, Release 3.1, NW Lab. Int. Sys., Dep. Computer Sci., University of Washington, Feb.1987.

[22] J. A. Abraham and W. K. Fuchs, "Fault and error models for VLSI," *IEEE Proc.*, vol. 74, pp. 639-654, May 1986.