1982

Design and implementation of a single-chip 1-d median filter

Kemal Oflazer
Carnegie Mellon University

CMU-CS-82-115

DESIGN AND IMPLEMENTATION OF A SINGLE-CHIP 1-D MEDIAN FILTER

Kemal Oflazer

Department of Computer Science
Carnegie-Mellon University
Pittsburgh, PA, 15213

April 1982

ABSTRACT

The design and implementation of a 1-Dimensional median filter in VLSI is presented. The device is designed to operate on 8-bit sample sequences with a window size of 5 samples. Extensive pipelining and employment of systolic concepts at the bit level enable the chip to filter at rates up to 10 Mega-samples per second. The chip is designed to be implemented with a $\lambda = 2.5\mu$ NMOS technology and is 6.2 mm by 5.0 mm in size. A circuit configuration for using the chip in approximate 2-D median filtering is also presented.

This research is supported in part by the Office of Naval Research under Contracts N00014-76-C-0370,NR 044-0422 and N00014-80-C-0236,NR 048-659, in part by the National Science Foundation under Grant MCS 78-236-76 , and in part by the Defense Advanced Research Projects Agency (DOD), ARPA Order No. 3597, monitored by the Air Force Avionics Laboratory under Contract F33615-81-K-1539.

Table of Contents

1. Introduction
2. Systolic Algorithms and Structures
3. The 1-Dimensional Median-Filtering Algorithm
    3.1. High-Level Structure of the Algorithm
    3.2. Hardware Implementation of the Algorithm
4. The Chip
5. Application to 2-Dimensional Image Processing
6. Evaluation and Conclusions
7. Acknowledgements

List of Figures

Figure 3-1: High level structure of the algorithm 3
Figure 3-2: The basic compare-and-swap unit 4
Figure 3-3: Internal structure of the odd/even-transposition sort network 6
Figure 4-1: The floor plan of the chip 7
Figure 5-1: Hardware structure to implement the approximate 2-D median filtering 7

1. Introduction

Median filtering is a nonlinear signal smoothing operation in which the median of a window of size $w = 2n + 1$ replaces the sample at the middle of the window. Medians computed in this way tend to follow the polynomial trends in the original sequence while sharp discontinuities of short duration are filtered out. Further properties of median filtering have been described in [1] while [2] describes its application to speech processing. Recently, an algorithm for real-time median filtering has been presented in [3]. Systolic algorithms for one- and multi-dimensional median-filtering operations and the more general case of computing running-order statistics have been recently proposed by Fisher [4].

This work presents the design and implementation of a VLSI chip for the 1-dimensional median-filtering operation. The device is designed to operate on 8-bit sample sequences with a window size of 5 samples. Extensive pipelining and employment of systolic concepts at the bit level enable the chip to have a very high throughput, i.e. the chip can be clocked at rates up to 10 Mhz and produce one median every clock cycle after an initial delay to fill the pipeline. The chip is designed to operate as a shift register in a system environment, filtering data coming from the source before going into the actual computing system.

2. Systolic Algorithms and Structures

Rapidly advancing VLSI technology offers system designers a very high potential for parallel operations. However, in order to exploit this potential, algorithms to be implemented with VLSI computing structures should have regular and simple communication schemes. This is mainly due to the fact that communication, especially irregular communication, is costly in VLSI in terms of the chip area that communication channels (i.e. wires) occupy. Furthermore, to reduce the design time, these algorithms should employ a rather small number of basic building blocks (or cells) from which larger systems can be built.

A class of parallel algorithms that exhibit such regular structures are *systolic algorithms*. Systolic algorithms for various computational problems have been described in [4, 5, 6, 7, 8]. Systolic data structures for priority queue operations and connectivity problems have been proposed in [9] and in [10] respectively. The general architectural principles of systolic computation systems have been discussed by Kung in [11]. In general, systolic algorithms and the underlying hardware structures implementing them have very regular neighbor-to-neighbor communication schemes. They utilize their inputs many times through pipelining and multi-directional data flow and hence do not make heavy bandwidth demands on system memories.

Employment of systolic concepts at the low-level implementation of logic circuits for various simple functions (like addition and comparison) also leads to regular structures that have small propagation delays (independent of the size of the circuit) and require no broadcasting. Such circuits are suitable as building

blocks in higher-level pipelined structures. A previous chip employing such concepts is the pattern matching chip described in [12].

In the last few years various special purpose chips employing systolic algorithms have been designed at Carnegie-Mellon University. These include a pattern matching chip [12], an image processing chip [13], and a tree processor for database applications [14].

3. The 1-Dimensional Median-Filtering Algorithm

The 1-D median-filtering algorithm implemented is differs from the one described in [4] in the sense that it uses the odd/even-transposition sort [15, 16, 6] as the high-level algorithm and exploits systolic data flow concepts at the bit level to achieve a very high throughput. After an initial delay to fill the pipeline, the chip can produce one median over a sliding 5-wide window at every clock period. The logic design enables the use of a clock period that is long enough to cover the propagation delays of five NMOS gates. However, due to technological limitations, the method employed is suitable only for small window sizes (3 to 7) because the network implementing the pipelined odd/even-transposition sort requires area proportional to the square of the window size. The systolic algorithms presented in [4] require area linear in window size but they need more complex circuits.

3.1. High-Level Structure of the Algorithm

At the high level, the algorithm, and hence the underlying hardware that implements it, consists of an input stage which generates the successive window elements from the incoming sample stream, and a pipelined sort stage which performs the odd/even-transposition sort on the elements of successive windows (see Fig. 3-1). Shamos, in [17], has proposed similar circuits for median finding; in fact, a circuit proposed there for a window of size 5 uses fewer of comparators than the circuit presented here, but Shamos' circuit structure is not regular.

The input stage is basically a shift register. At every clock, it reads one sample value from the input and discards the sample value read five clocks earlier. This effectively slides the window of the filter over the incoming sample stream. Hence, a new window is presented to the odd/even-transposition sort network at every clock.

The odd/even-transposition sort network is a pipelined structure consisting of *compare-and-swap* stages that operate on even and odd pairs of window elements[1] alternately. Five such alternating stages implement

[1] Even pairs have indices of the form (n , n+1) where n is even, while for odd pairs n is odd.

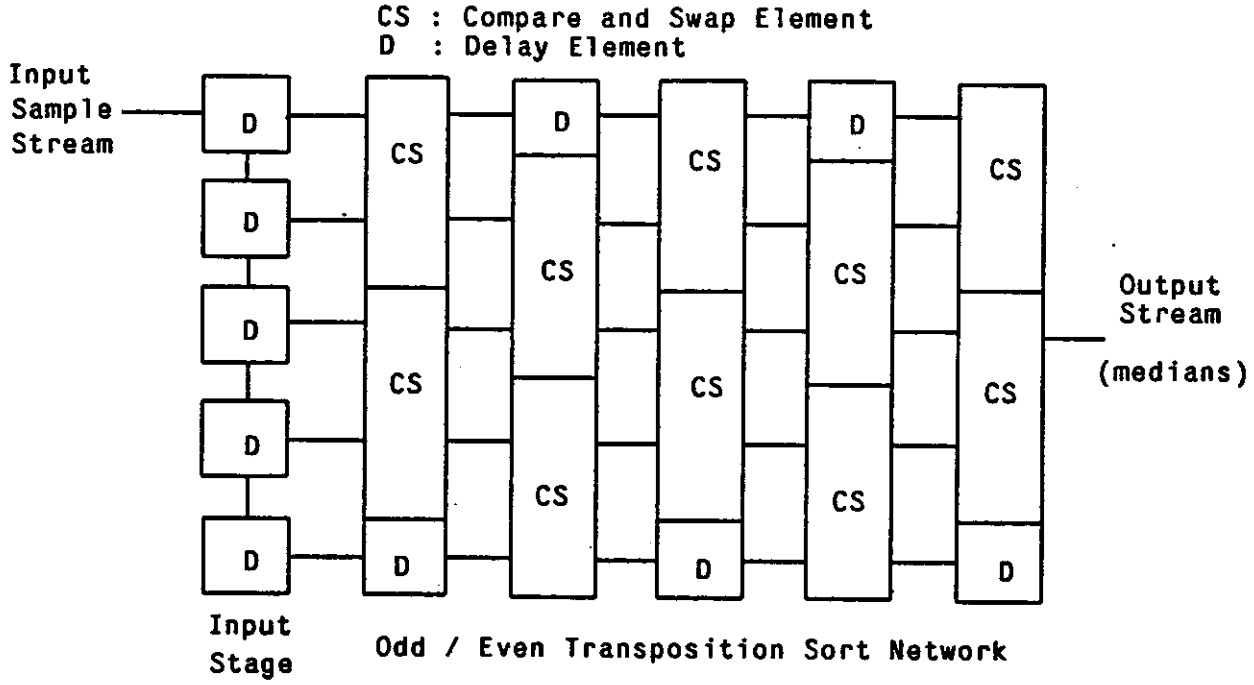

Figure 3-1: High level structure of the algorithm

the odd/even-transposition sort for five sample values. Since the sample values in a window pass through one stage of the odd/even-transposition sort network in one clock, it is possible to pipeline the sorting of successive windows through the network.

Each stage of the odd/even-transposition sort network consists of 2 8-bit compare-and-swap units ($\lfloor$ *window size* / 2 $\rfloor$ units in general) and one delay element to store the window sample value that does not get compared at that stage, due to the fact that the window size is odd.

Each 8-bit compare-and-swap unit compares the pair of 8-bit numbers at its input and interchanges them if necessary so that the larger of the numbers is at the "top". At the output of the last stage, the window elements will be sorted such that the largest will be at the "top" .

3.2. Hardware Implementation of the Algorithm

The structure of the odd/even-transposition sort network described above has certain undesirable characteristics if directly mapped into hardware. In the compare-and-swap units, the swapping of the inputs can only be done after the result of the entire 8-bit comparison has been computed. However, this requires waiting for a long propagation delay through 8 stages of bitwise comparators.

It is possible to get rid of this propagation delay by employing systolic concepts at the bit level. This involves breaking up the compare-and-swap operation into steps and then distributing them over time by skewing the bits of the numbers being compared with delay elements so that each pair of bits arrives at their bitwise comparator at the same time the subresult of the comparison of their more significant counterparts arrives.

The basic element to implement the compare-and-swap operation is the bitwise *compare-and-swap* unit. The functional description of this unit is given in Fig. 3-2. It is a bit comparator followed by two multiplexers which pass the larger of the inputs to the A output and the smaller to the B output, if E_{in} is asserted. Otherwise it unconditionally swaps or passes the inputs depending on whether L_{in} is asserted or not. It also passes "downward" the cumulative subresult of the comparison to the less significant stages.

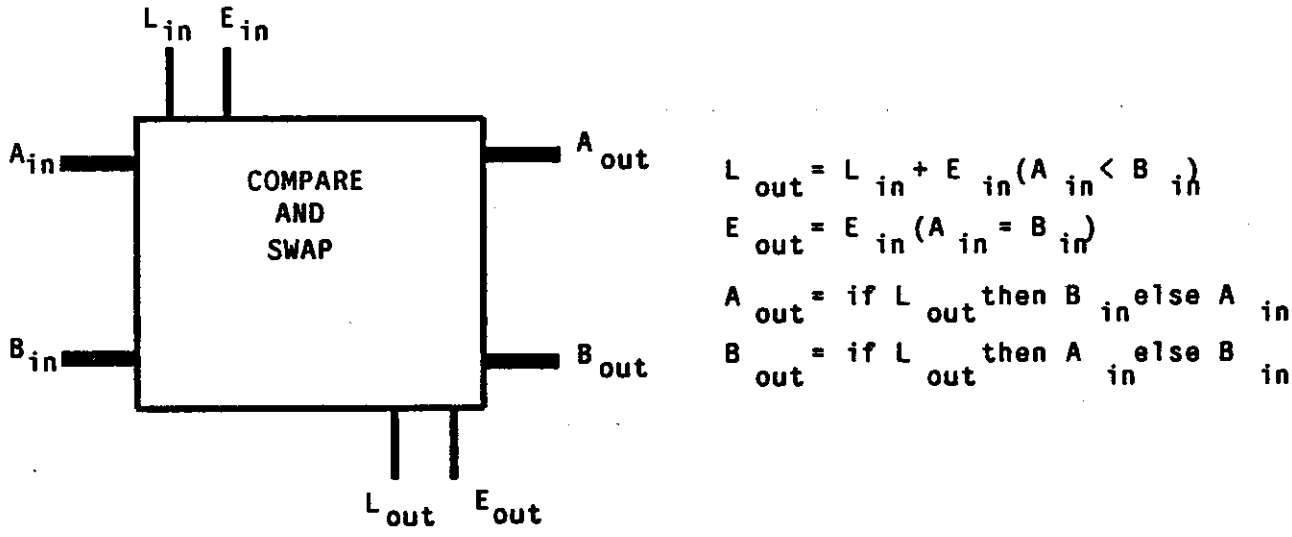

Figure 3-2: The basic compare-and-swap unit

The 8-bit wide compare-and-swap units implemented with the units described above also distribute the swap operation over time along with the comparisons. So at the end each bitwise compare-and-swap operation, the outputs of the bit compare-and-swap units will be same as they would be if all the bitwise swaps were done simultaneously after waiting for the final comparison result. This is easy to see if we note that if a less significant compare-and-swap unit decides that all the input bits should be swapped, then the inputs to more significant stages should have been equal hence passing them without swapping would not matter.

The implementation of the odd/even-transposition sort network exploits the observations presented above. Furthermore, the comparisons of the next stage of the odd/even-transposition sort network can be started

immediately once the comparisons and swaps of the first bits of the preceding stage are done. This observation leads to an internal block structure of the odd/even-transposition sort network given in Fig 3.3. It should also be noted that in the resulting structure, the first bits of the sorted outputs are available even before the comparisons of the first stage of the odd/even-transposition sort network are completed.

4. The Chip

The chip employing the method described in the preceding section has been designed to be implemented with an NMOS process with λ of 2.5 microns. The basic methodology and the design rules presented in [18] have been used throughout the design and layout process. The outline of the floor plan of the resulting chip is given in Fig. 4-1.The dimensions of the chip are approximately 6.2 mm by 5.0 mm .

It uses 21 pins : 8 for input, 8 for output, 2 for the two phases of the clock, 1 for V_{dd} , 1 for Ground and 1 for the substrate bias; hence it can be packaged in a 24 pin package.

As of this writing, the chip has been laid out completely and design-rule checks have been made. Circuit level simulations of the circuits making up the sort network have been done. Currently the chip is being fabricated by the ARPA facility coordinated by USC-ISI .

5. Application to 2-Dimensional Image Processing

Although the design is not directly applicable to 2-Dimensional median filtering operation, a cascade configuration using these chips can be used for approximate median filtering of 2-D images as suggested by Shamos [17]. The basic idea is to find the medians of the rows of an $n \times n$ window and then compute the median of the medians. It is shown in [17] that A_m, the median of the medians of such a window, has the property that

$$rank(A_m) \geq (n^2 + 2n + 1) / 4$$

and

$$rank(A_m) \leq (3n^2 - 2n + 3) / 4$$

This result indicates that such a configuration is guaranteed to filter out the upper and lower quartiles of the samples in the window. Simulation results obtained by Shamos also indicate that for $n = 5$ $Prob\,(rank(A_5) = 13) \approx 0.2900$ and $Prob\,(12 \leq rank(A_5) \leq 14) \geq 0.7200$.

The 1-D median-filter chip can be used in the configuration given in Fig. 5-1 to implement the approximate 2-D median filtering. This configuration operates in the following way: the 1-D median-filters

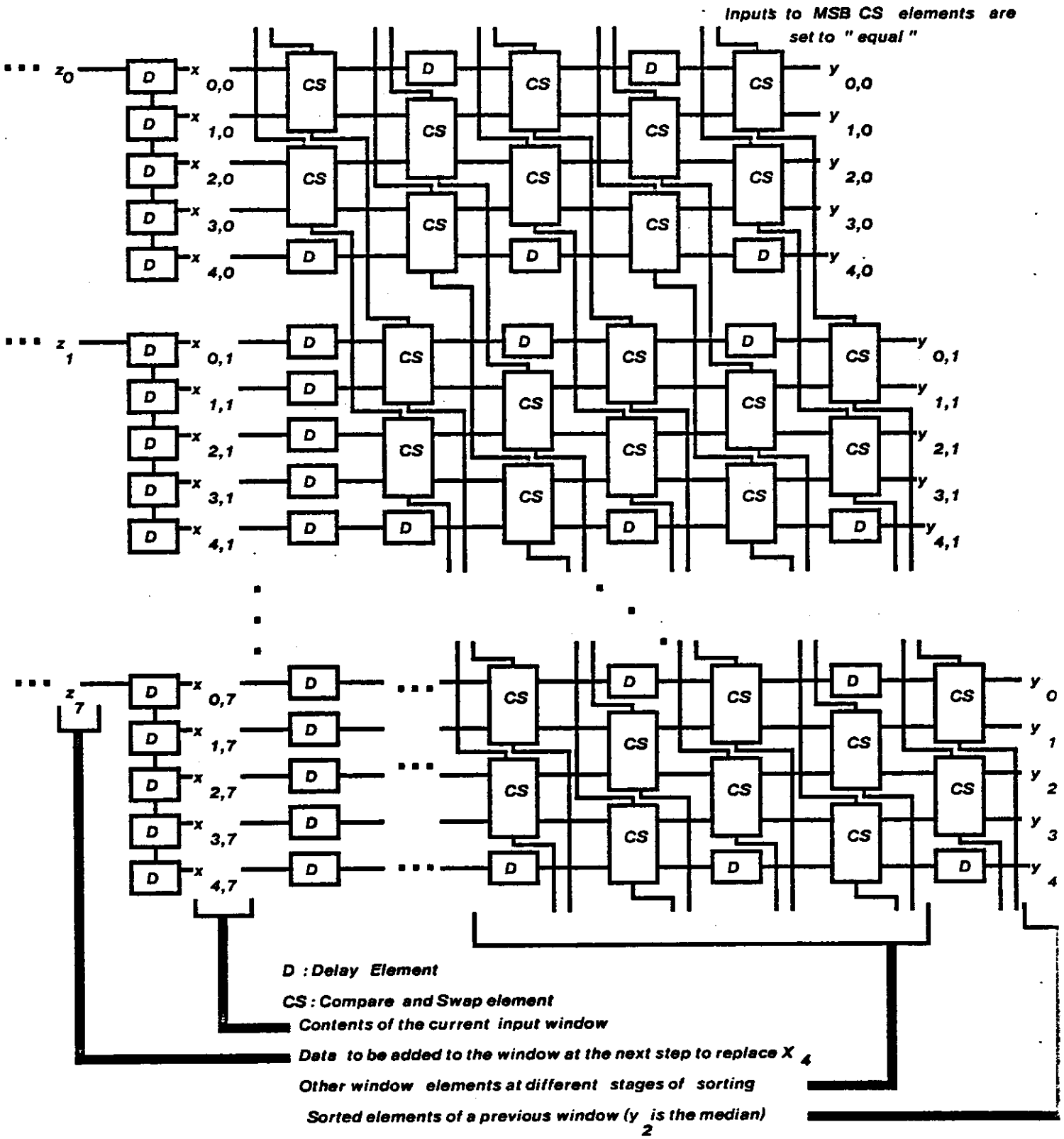

Figure 3-3: Internal structure of the odd/even-transposition sort network

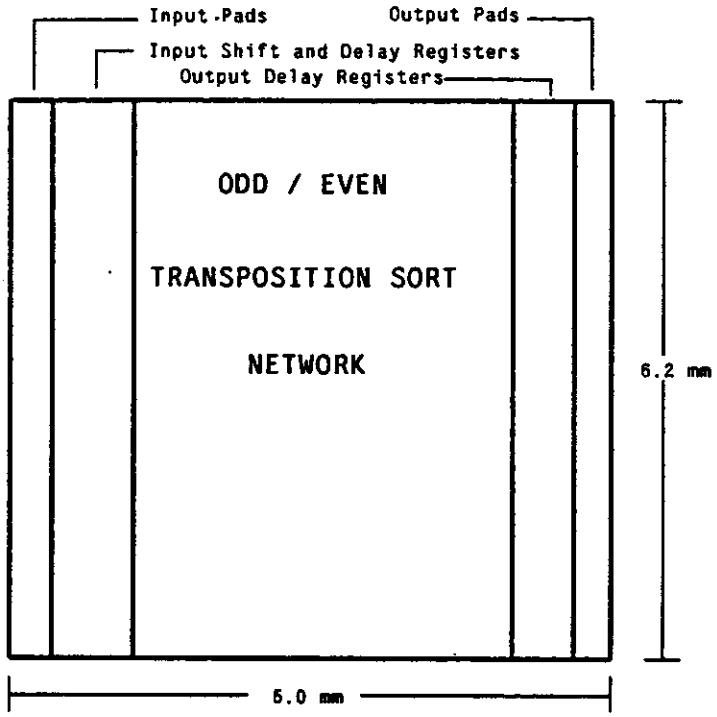

Figure 4-1: The floor plan of the chip

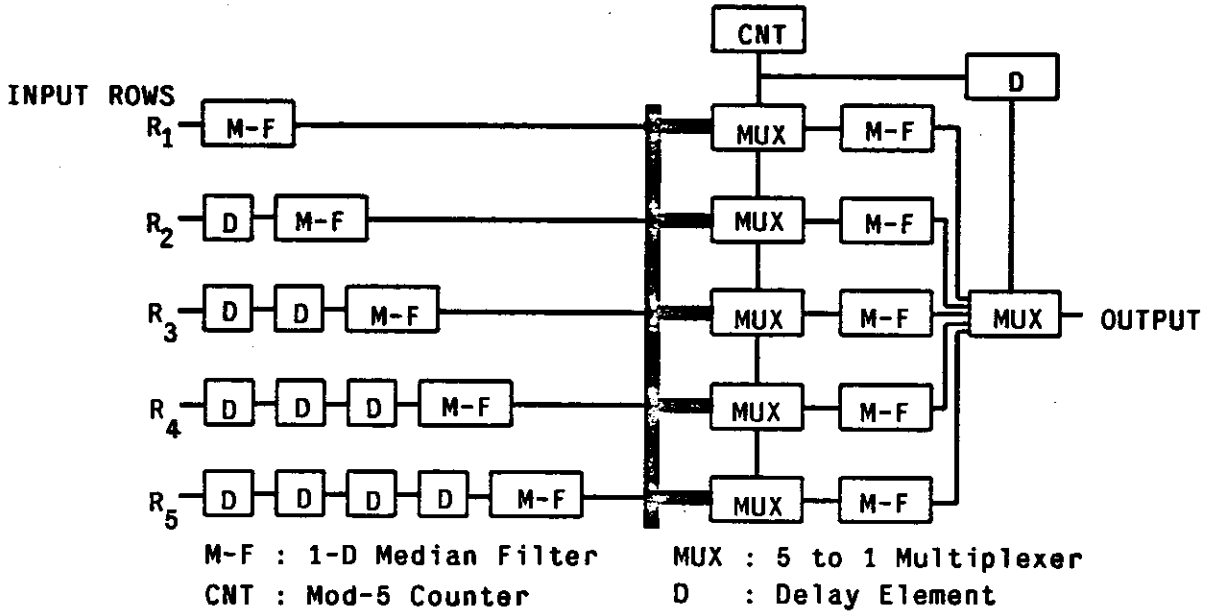

Figure 5-1: Hardware structure to implement the approximate 2-D median filtering

on the left filter the rows of the 5×5 window sliding over the rows and output the medians of the rows skewed in time. The multiplexers serialize the parallel incoming medians into the 1-D median filters on the

right and also pass the select code coming from the upper multiplexers or the counter to the next multiplexer below. The median filters on the right operate on skewed window outputs in parallel, computing the medians of the medians of the rows. However they generate one result every 5 time steps. Finally, the last multiplexer selects and outputs the approximate medians (A_5) coming out of the median-filters. It can be noted that this configuration can filter at at rate of 50 Mega-samples per second.

6. Evaluation and Conclusions

The design and implementation of a VLSI chip for performing the 1-D median-filtering operation has been presented. The major motivation for this work has been to apply systolic concepts at the bit level in the implementation of logic circuits to construct a digital system with a very high throughput. Also, application of the developed chip to 2-D image processing has been investigated and a configuration for employing it in approximate 2-D median filtering has been proposed.

Although the design developed in this work has a very high throughput, the response time is $k + w$ where k is the number of bits in each sample and w is the window size (so the response time for this specific implementation is 13 clock periods). Furthermore, the design is not practical for larger window sizes because the silicon area for implementing the odd/even-transposition sort network grows as the square of the window size.

7. Acknowledgements

I would like to thank Prof. H. T. Kung for suggesting the chip design and for providing valuable comments and to Allan Fisher, Richard Korf, and Michael Horowitz for providing constructive comments.

REFERENCES

1. Gallagher, N. C., Jr. , Wise , G. L., "A Theoretical Analysis of the Properties of Median Filters," *IEEE Transactions on Acoustics, Speech and Signal Processing*, Vol. ASSP-29, No. 6, Dec. 1981, pp. 1136-1141.

2. Rabiner, L. R. , Sambur, M. R. , Schmidt, C. E. , "Applications of a Nonlinear Smoothing Algorithm to Speech Processing," *IEEE Transactions on Acoustics, Speech and Signal Processing*, Vol. ASSP-23, No. 6, Dec. 1975, pp. 552-557.

3. Ataman, E. , Aatre, V. K. , Wong, K. M., "A Fast Method for Real-Time Median Filtering," *IEEE Transactions on Acoustics, Speech and Signal Processing*, Vol. ASSP-28, No. 4, Aug. 1980, pp. 415-421.

4. Fisher, A. L., "Systolic Algorithms for Running Order Statistics in Signal and Image Processing," *Proceedings of CMU Conference on VLSI Systems and Computations*, Kung, H. T. ,Sproull, R. F. ,Steele, G. L. Jr., eds., Computer Science Press, Carnegie-Mellon University, October 1981, pp. 265-272, To appear also in Journal Of Digital Systems

5. Kung, H.T., "The Structure of Parallel Algorithms," *Advances in Computers, Volume 19*, Yovits, M.C., ed., Academic Press, New York, 1980, pp. 65-112, Also available as CMU Computer Science Department technical report, August 1979

6. Kung H. T., "Let's Design Algorithms For VLSI Systems," *Proceedings of the Conference on Very Large Scale Integration: Architecture, Design, Fabrication*, Seitz, C. L., ed., California Institute of Technology, January 1979, pp. 65-90.

7. Kung, H. T. and Leiserson, C. E., "Systolic Arrays (for VLSI)," *Sparse Matrix Proceedings*, Duff, I. S. and Stewart, G. W., eds., Society for Industrial and Applied Mathematics, 1978, pp. 256-282, Also appears as Section 8.3 in *Introduction to VLSI Systems* by Mead and Conway [10]

8. Kung, H. T. and Lehman, P. L , "Systolic (VLSI) Arrays for Relational Database Operations," *Proceedings of ACM-SIGMOD 1980 International Conference on Management of Data*, Santa Monica, California, May 1980, pp. 105-116.

9. Leiserson, C. E., "Systolic Priority Queues," *Proceedings of Conference on Very Large Scale Integration: Architecture, Design, Fabrication*, Seitz, C. L., ed., California Institute of Technology, January 1979, pp. 199-214.

10. Savage, C., "A Systolic Data Structure Chip for Connectivity Problems," *Proceedings of CMU Conference on VLSI Systems and Computations*, Kung, H. T. ,Sproull, R. F. ,Steele, G. L. Jr., eds., Computer Science Press, Carnegie Mellon University, October 1981, pp. 296-300.

11. Kung, H. T., "Why Systolic Architectures?," *IEEE Computer*, January 1982, pp. 37-46.

12. Foster, M. J. and Kung, H. T., "The Design of Special Purpose Chips," *IEEE Computer*, Vol. 13, January 1980, pp. 26-40.

13. Kung, H. T. and Song, S. W., "A Systolic Array Chip for the Convolution Operator in Image Processing," VLSI Memo V046, Carnegie Mellon University, Department of Computer Science, February 1980.

14. Song, S. W., "A Database Machine with Novel Space Allocation Algorithms," VLSI Memo V042, Carnegie Mellon University, Department of Computer Science, February 1980.

15. Habermann, N., "Parallel Neighbor-Sort (Or the Glory of the Induction Principle)," Tech. report, Computer Science Department-Carnegie Mellon University, August 1972.

16. Knuth, D. E., *The Art of Computer Programming-Searching and Sorting*, Addison-Wesley, , Vol. 3, 1973.

17. Shamos, M. I., "Robust picture processing operators and their implementation as circuits," *Proceedings of the ARPA Image Understanding Workshop*, Science Applications, Inc., November 1978, pp. 127-129.

18. Mead, C. and Conway, L., *Introduction to VLSI Systems*, Addison Wesley, 1980.

SECURITY CLASSIFICATION OF (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER: CMU-CS-82-115	2. GOVT ACCESSION NO.: ONR
3. RECIPIENT'S CATALOG NUMBER:	
4. TITLE (and Subtitle): Design and Implementation of a Single-Chip 1-D Median Filter	5. TYPE OF REPORT & PERIOD COVERED: Interim
	6. PERFORMING ORG. REPORT NUMBER:
7. AUTHOR(s): Kemal Oflazer	8. CONTRACT OR GRANT NUMBER(s): N00014-76-C-0370, NR 044-0422 N00014-80-C-0236, NR 048-659
9. PERFORMING ORGANIZATION NAME AND ADDRESS: Carnegie-Mellon University, Computer Science Department, Pittsburgh, PA. 15213	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS:
11. CONTROLLING OFFICE NAME AND ADDRESS:	12. REPORT DATE: April 1982
	13. NUMBER OF PAGES: 13
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office):	15. SECURITY CLASS. (of this report): UNCLASSIFIED
	15a. DECLASSIFICATION/DOWNGRADING SCHEDULE:

16. DISTRIBUTION STATEMENT (of this Report)

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

Approved for public release; distribution unlimited

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE
S/N 0102-014-6601

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)