Realization of a Programmable Rank-Order Filter Architecture Using Capacitive Threshold Logic Gates

İ. Hatırnaz, F. K. Gürkaynak and Y. Leblebici

Worcester Polytechnic Institute
Department of Electrical and Computer Engineering
Worcester, MA 01609-2280

Abstract

We present a new architecture to realize a fully programmable rank order filter (ROF), based on Capacitive Threshold Logic (CTL) gates. Variants of ROFs, especially median filters, are widely used in digital signal and image/video processing and image enhancement. The CTL realization of the majority gates used in the ROF architecture allows the filter rank and the window size to be user-programmable, using a much smaller silicon area. The overall filter architecture is also simplified significantly, compared to conventional realizations of digital median filters.

1 Introduction

The rank order filter (ROF) is a non-linear digital filter which determines the r-th ranking element in a given window consisting of binary encoded input words. Variants of ROFs are widely used in digital signal and image/video processing. Especially, median filters have found many applications in digital image enhancement, such as eliminating blur in the edges and suppressing noise in an image [1][2].

In recent years, some innovative structures for rank-order-filters have been presented , which are mostly based on majority-decision algorithms [3][4]. Yet, the majority function is typically hard to realize using conventional Boolean building blocks, since it requires a large number of gates and a large logic depth. Consequently, such structures suffer from speed and area limitations, especially if the window size becomes larger than 10 words.

In this paper, we present a new architecture to realize a fully programmable ROF, based on Capacitive Threshold Logic (CTL) gates. The CTL realization of the majority gates [5] used in the ROF architecture allows the filter rank and the window size to be user-programmable, using a much smaller silicon area. The overall filter architecture is also simplified significantly, compared to conventional realizations of digital median filters. The outline of a simple bit-serial algorithm for rank ordering is presented in Section 2. In Section 3, the implementation of a programmable ROF architecture is discussed. The conclusions are summarized in Section 4.

2 The Rank Ordering Algorithm

2.1 Algorithm Description

A bit-serial algorithm first proposed by Kar and Pradhan [6] was chosen to implement the programmable rank-order filter using capacitive threshold logic gates. In this algorithm, the problem of finding a rank-order-selection for n-bit long words is reduced to finding out "n" rank-order-selections for 1-bit numbers.

The algorithm starts by processing the most significant bits (MSB) of the m=(2N + 1) words in the current window to yield the MSB of the desired filter output. This output is then compared with the other MSBs of the window elements. The vectors whose MSB is not equal to the filter output have their MSB propagated down by one position, replacing the less significant bits of the corresponding words. This process is continued for the following bits. Thus, any bit that is not equal to the corresponding stage output is propagated down to the lesser significant positions, until the least significant bit is processed.

0-7803-5471-0/99/$10.00©1999 IEEE

The algorithm can be summarized as follows :

```
begin
  for l := 1 to n do
    begin
      -- sum[l] is the sum of the
      -- l-th bit of all the numbers
      sum[l] := 0;
      for j := 1 to s do
        sum[l] := sum[l] + ajl;
      if sum[l] <= s - i then
        -- selecting the i-th ranked bit
        selector[l] := 0;
      else
        selector[l] := 1;
        -- where the selector[l] is the l-th
        -- bit of the i-th ranked number
      for j := 1 to s do
        begin
        if ajl != selector[l] then
          for q := (l + 1) to n do
            ajq := ajl;
        end
    end
    i-th-ranked-number := selector;
     -- selector is the concatenation of
     -- selector[1],...,selector[n]
end
```

Table 1: An example of the rank-ordering algorithm. Values in parantheses denote values being propagated down to lesser significant positions.

Words :	#1	#2	#3	#4	#5	**Result**
B_7(MSB)	1	0	1	0	0	0
B_6	0(1)	1	1(1)	1	1	1
B_5	1(1)	1	1(1)	1	0	1
B_4	1(1)	0	0(1)	1	0(0)	1
B_3	1(1)	1(0)	0(1)	0	1(0)	0
B_2	0(1)	0(0)	0(1)	1	0(0)	1
B_1	1(1)	0(0)	1(1)	0	1(0)	0
B_0(LSB)	0(1)	1(0)	0(1)	1	1(0)	1

Table 1 shows an example where, five 8-bit words (denoted #1 through #5 with decimal values of 184, 105, 194, 117 and 75 respectively) are being rank-ordered using the algorithm given above. The window size is m=5 and the rank is r=3, indicating that the third smallest among these five numbers is being found in 8 steps. Note that the main bit-level operation at each step amounts to a majority (rank) decision among n bits of the same bit-plane. In the example, the result corresponds to word #4 which has the decimal value of 117.

2.2 Structure and Realization of the Algorithm

The bit-serial operation flow of the algorithm described above suggests a very simple bit-level pipelined data path architecture.

The hardware implementation of the ROF algorithm consists of two main blocks:

1. The Modifier/Selector(propagator) block whose function is to store and to shift the actual data and to calculate the selector signal for the next processing block.
2. The Majority or Rank Decision block which determines the output bit as a function of n bits.

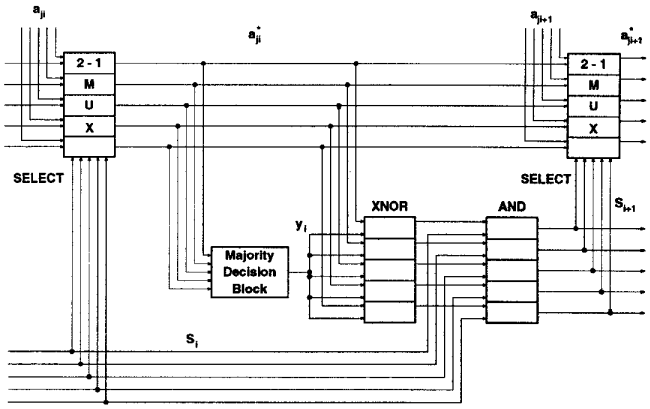

Figure 1: A 1-bit slice implementation of the Kar-Pradhan rank ordering algorithm (for m=5).

A 1-bit slice of the algorithm's proposed implementation is shown in Figure 1, as a pipeline block for filtering 5 words of arbitrary bit-length. First, the Majority Decision block finds the filter-slice output "y_i" from a_{ji}. In the

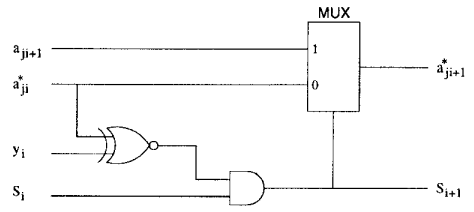

Figure 2: Gate-level schematic of the modifier/selector block

Modifier/Selector block (Fig.2) an XNOR operation is performed on the output bit "y_i" and the corresponding bit "a^*_{ji}", in order to determine whether or not the two bits are identical. The result of this XNOR operation is then combined (AND operation) with the select signal originating from the previous block. This provides the information if the data bit taken from the previous block is a propagating one or not. If the data bit is a propagating one, then the new select signal will be 0, indicating that this data bit will continue propagating unchanged through the following stages. Otherwise, the select signal will only depend on the result of the comparison of the filter-slice output with the current data bit. Identical 1-bit filter slices can be used in sequence (cascade configuration) in order to process input vectors of arbitrary bit-length.

3 Implementation of the Programmable ROF Architecture

3.1 Bit-Level Pipeline Structure

As mentioned earlier, the Kar-Pradhan rank-order algorithm reduces the rank ordering problem to determining "n" rank order selections for 1-bit numbers. This bit-serial approach makes it possible to increase the throughput by using a bit-level pipeline architecture. Thus, for a pipelined filter with 8-bit word length, the throughput would increase by a factor of eight compared to the unpipelined version.

3.2 Realization of the Majority Decision Block

The design of the modifier/selector block shown in Fig.2 is straight-forward using conventional CMOS logic gates, whereas the majority decision block presents a bottleneck with conventional methods. Since the majority decision is in fact a threshold function, it can be most efficiently implemented with threshold logic. Therefore, capacitive threshold logic gates offer a more efficient realization of the majority function compared to conventional logic gates [5]. For the realization of the rank ordering algorithm, the capability of using a programmable majority decision gate is highly desirable. The CTL based realization of the majority function also offers that capability at almost no additional cost. As presented in [5], a 31-input programmable majority gate based on CTL is almost three times faster than a full custom standard CMOS realization [7] and occupies approximately one third of the area which results in a area-delay performance increase of nearly an order of magnitude. In addition, the CTL-based majority gate can be easily integrated with the conventional CMOS gates used in the architecture, since the input and output signals are fully CMOS compatible. The compacted layout of the 31 input programmable majority gate is shown in Figure 3. The dimensions of this majority block are $195\mu m$ x $75\mu m$.

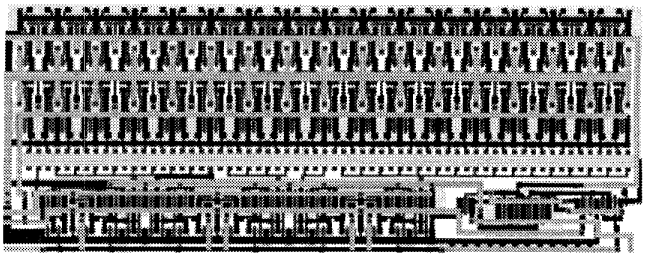

Figure 3: Layout of the majority gate based on CTL.

3.3 Overall System Architecture

The programmable Rank-Order-Filter algorithm described above can be implemented using modifier/selector blocks based on standard CMOS, and majority function blocks based on threshold logic.

The proposed architecture has the following specifications:

- 8-bit word length, 31 word maximum window size.
- Programmable rank and window size.
- 8-stage pipelined architecture to increase throughput.

The specifications essentially describe a high-speed, multi-purpose rank-order filter. The most important characteristic of the filter is programmability of both rank and window size which are not easily realizable with conventional approaches.

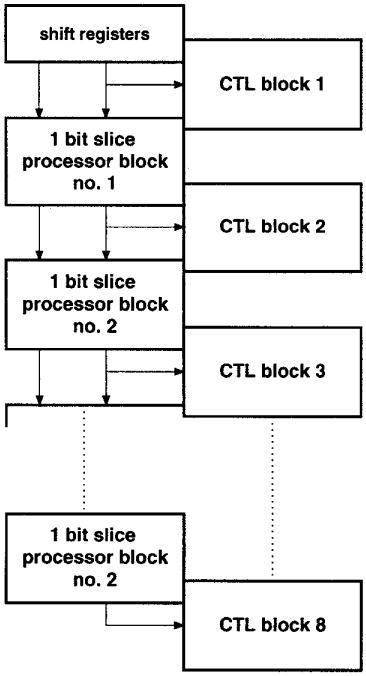

Figure 4: Datapath floorplan of the 8-bit programmable ROF.

A simple program control block is responsible for evaluating the input control words for window size and rank programming. Depending on the result of the evaluation phase, this block either programs the CTL gates or in case of an error, sets the error flag retaining the current settings of the CTL gates. The window size control logic consists only of a 5 input NOR gate because the only invalid word is 00000, a null window. A single word window is also considered a valid input which makes the ROF equivalent to a digital all-pass filter. The threshold control logic will look for invalid control words like a rank of 0 and ranks that are less than the current window size.

The floorplan of the 8-bit programmable ROF which actually reflects the dataflow can be seen in Figure 4.

4 Conclusion

In this paper, we have presented a new architecture for realizing a fully programmable ROF, based on the Kar-Pradhan rank ordering algorithm and Capacitive Threshold Logic (CTL) majority gates. The bit-serial realization of the rank ordering algorithm offers a simple pipelined filter architecture which is highly modular and easily expandable. The CTL realization of the majority gates used in the ROF architecture allows the filter rank and the window size to be user-programmable, resulting in a much smaller silicon area. In addition, the CTL based majority gates enable a much simpler overall filter architecture compared to conventional digital median filter realizations.

References

[1] D.S. Richards, "VLSI median filters", *IEEE Trans. Acoust., Speech, Signal Processing*, vol. 38, pp.145-152, January, 1990.

[2] J.P. Fitch, E.L. Coyle and N.C. Callagher, "Median filtering by threshold decomposition", IEEE Trans. Acous. Speech, Signal Proc., vol 32, pp.1183-1188, 1984.

[3] A. Gasteratos, I. Andreadis, Ph. Tsalides, "Realization of rank order filters based on majority gate", *Pattern Recognition*, vol.30, no. 9, pp 1571-1576, 1997.

[4] C.L. Lee and C.W. Jen, "Bit-sliced median filter design based on majority gate", *in Proc. Ins. Elec. Eng.-G*, vol 139, pp.63-71, 1992.

[5] Y. Leblebici, F.K. Gurkaynak, D. Mlynek, "A compact 31-input programmable majority gate based on capacitive threshold logic", *in Proc. IEEE Int. ASIC Conference 1998*, pp. 281-285.

[6] B.K. Kar, D.K. Pradhan, "A new algorithm for order statistic and sorting", *IEEE Trans. on Signal Processing*, vol. 41, pp.2688-2694, August 1993.

[7] E.E. Swartzlander, Jr., "Parallel counters", *IEEE Trans. Comput.*, vol.34, pp. 1021-1024, Nov. 1973.