

Monolithic Mixed-Mode Implementation of Sum-of-Product Arrays for Performing Binary Morphological Image Processing

Ronald G. Spencer and Edgar Sanchez-Sinencio

Electrical Engineering Department, Texas A&M University, College Station, TX. 77843-3128
rspencer@tamu.edu

Abstract- This paper presents a monolithic mixed-mode implementation of sum-of-product arrays that are capable of performing morphological image processing in silicon. The digital equations of the two basic morphological operations, erosion and dilation, are rewritten in a local, mixed-mode form to facilitate hardware implementation and results are shown for a 3x3 selective element array fabricated in silicon.

1. INTRODUCTION

Morphology is the study of structure in images [1] and [2]. It is based on the concept of transforming images by probing them with smaller images known as selective elements to determine intersections and unions. Two very significant advantages of morphological processing are the ability to discover fundamental structures in images and the localized simplicity of the operations. As a result, morphological processing can be easily implemented in hardware to assist with such tasks as image analysis and video coding [3].

The two fundamental morphological operations are erosion and dilation. In the case of erosion, the objective is to see how the smaller image fits within the larger image. In the case of dilation, the objective is to find the union of the smaller images with local regions of the larger image. Using different combinations and permutations of erosion and dilation, aggregate operations can be performed, such as opening and closing which can be used to remove noise and locate objects. [1].

Although morphological image processing can be implemented in a variety of ways, such as using cellular neural networks, the architecture presented here is dedicated to this task; therefore, it is simpler and more compact. It is also immune to non-idealities such as process variation and noise.

2. METHODS

2.1 Digital Formulation

The two fundamental morphological operations, dilation and erosion can be expressed in terms of Minkowski addition and subtraction which are given by equations 1 and 2 [1].

$$S + E \equiv \bigcup_{i,j \in D(S)} \text{TRAN}(E;i,j) \quad (1)$$

$$S - E \equiv \bigcap_{i,j \in D(E)} \text{TRAN}(S;i,j) \quad (2)$$

where S is the original binary image, E is the binary selective element image, $\text{TRAN}(\bullet;i,j)$ is the translation of image \bullet by i,j , and $D(\bullet)$ is the domain of image \bullet which is the set of coordinates of all TRUE pixels in image \bullet . Dilation is defined to be identical to Minkowski addition as shown by equation 3 and erosion is defined to be Minkowski subtraction of $-E$ from S as shown by equation 4 where $-E$ is equal to E rotated by 180 degrees.

$$\text{Dilate}(S, E) \equiv S + E \quad (3)$$

$$\text{Erode}(S, E) \equiv S - (-E) \quad (4)$$

An alternate form of erosion which is more easily visualized shows that a given pixel in the eroded image is TRUE if and only if the translated version of the selective element image, E , is fully contained within the original image, S :

$$[\text{Erode}(S, E)](i, j) = \begin{cases} 1 & \text{if } \text{Tran}(E;i,j) \ll S \\ 0 & \text{otherwise} \end{cases} \quad (5)$$

As an example, let S and E be defined by equation 6 where the dots denote the origin.

$$S = \begin{pmatrix} 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 0 \bullet & 1 & 1 & 1 & 0 \end{pmatrix} \quad E = \begin{pmatrix} 1 & 1 \bullet \end{pmatrix} \quad (6)$$

To dilate S with respect to E , E must be translated by the domain of S which is:

$$D(S) = \left\{ \begin{pmatrix} 1,1 \end{pmatrix}, \begin{pmatrix} 2,1 \end{pmatrix}, \begin{pmatrix} 1,0 \end{pmatrix}, \begin{pmatrix} 2,0 \end{pmatrix}, \begin{pmatrix} 3,2 \end{pmatrix}, \begin{pmatrix} 3,0 \end{pmatrix} \right\} \quad (7)$$

which calls for six translations of E :

$$E_{1,1} = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 0 \bullet & 0 & 0 & 0 \end{pmatrix} E_{2,1} = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 \bullet & 0 & 0 & 0 \end{pmatrix} \quad (8a)$$

$$E_{1,0} = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 \bullet & 1 & 0 & 0 \end{pmatrix} E_{2,0} = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 \bullet & 1 & 1 & 0 \end{pmatrix} \quad (8b)$$

$$E_{3,2} = \begin{pmatrix} 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 0 \bullet & 0 & 0 & 0 \end{pmatrix} E_{3,0} = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 \bullet & 0 & 1 & 1 \end{pmatrix} \quad (8c)$$

The final result is the *union* of the translated images:

$$\text{Dilate}(S, E) = \begin{pmatrix} 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 0 \\ 1 \bullet & 1 & 1 & 1 \end{pmatrix} \quad (9)$$

On the other hand, to *erode* S by E , S must be translated by the domain of the rotated version of E ,

$$D(-E) = \{(0,0), (1,0)\} \quad (10)$$

which calls for two translations of S :

$$T_1 = \text{TRAN}(S, 0, 0) = \begin{pmatrix} 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 0 \bullet & 1 & 1 & 1 & 0 \end{pmatrix} \quad (11a)$$

$$T_2 = \text{TRAN}(S, 1, 0) = \begin{pmatrix} 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 \\ 0 \bullet & 0 & 1 & 1 & 1 \end{pmatrix} \quad (11b)$$

The final result is the *intersection* of the translated images:

$$\text{Erode}(S, E) = \wedge(T_1, T_2) = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 \bullet & 0 & 1 & 1 \end{pmatrix} \quad (12)$$

Although global translation, union, and intersection demonstrate dilation and erosion very well, they are not well suited for hardware implementation because each operation must be computed for the whole image and stored for subsequent operations which requires much memory. The alternative is local processing which facilitates the computation of a single output pixel without needing to carry over information to the next set of operations. To this end, if the assumption is made that

the pixel at $(0,0)$ in selective element E is always TRUE (which is almost never a restriction) then dilation and erosion can be expressed as the union and intersection of ANDs, respectively as shown in equations 13 and 14 with E given by equation 15.

$$[\text{Dilate}(S, E)](i, j) = \bigcup_{j'=-J}^J \bigcup_{i'=-I}^I [S_{j+j', i+i'} \wedge E_{-j', -i'}] \quad (13)$$

$$[\text{Erode}(S, E)](i, j) = \bigcap_{j'=-J}^J \bigcap_{i'=-I}^I [\overline{S_{j+j', i+i'} \wedge E_{j', i'}}] \quad (14)$$

$$E = \begin{pmatrix} E_{-J, -I} & \cdot & E_{-J, I} \\ \cdot & 1 & \cdot \\ E_{J, -I} & \cdot & E_{J, I} \end{pmatrix} \quad (15)$$

For reasons that will become clear later, erosion can also be written as a sum-of-products (SOP):

$$[\text{Erode}(S, E)](i, j) = \overline{\bigcup_{j'=-J}^J \bigcup_{i'=-I}^I [S_{j+j', i+i'} \wedge E_{j', i'}]} \quad (16)$$

Now that these digital operations have been explicitly localized they can be more easily rewritten to so as to lend themselves to hardware implementation.

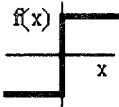
2.2 Mixed-Mode Formulation

Equations 13 and 16 are typical Boolean SOP expressions. To evaluate these expressions exclusively in the digital domain would require $(2I+1)(2J+1)$ ANDs and $(2I+1)(2J+1)-1$ ORs for each one. However, if a value of 1 is assigned to logic TRUE and 0 to logic FALSE, the OR operation is the same as thresholding an analog summation. Since analog summers can be implemented as current inputs feeding into a single resistance, a substantial savings in hardware can be gained by implementing SOP expressions in mixed-mode. By performing the summation in the analog domain, no digital OR-gates are needed. Instead, single-transistor transconductors can be used which feed current into a single resistance. This is the motivation for recasting the fully digital expressions into mixed-mode expressions.

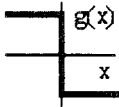
Dilation can be reformulated as a mixed-mode operation by replacing the union operators in

equation 13 with summers and thresholding the result with a non-inverting step function as shown in equation 17 and figure 1a:

$$[\text{Dilate}(S, E)](i, j) = f \left[\sum_{j'=-J}^J \sum_{i'=-I}^I [S_{j+j', i+i'} \wedge E_{-j', -i'}] \right] \quad (17)$$



(a)



(b)

Fig. 1. Non-inverting (a) and inverting (b) step functions.

The mixed-mode equivalent of erosion can be obtained in a like manner with an inverting step function replacing the inversion of equation 16,

$$[\text{Erode}(S, E)](i, j) = g \left[\sum_{j'=-J}^J \sum_{i'=-I}^I [\overline{S_{j+j', i+i'} \wedge E_{j', i'}}] \right] \quad (18)$$

where $g(x)$ is an inverting hard-limiting step function as shown in figure 1b.

2.3 Hardware Implementation

Since dilation and erosion can each be expressed as a sum of products, they can be expressed by the same circuit topology. As shown in figure 2, a NAND gate and an p-type transistor can be used to implement a single product in an SOP expression. Out of the four possible combinations of inputs, only one of them produces a FALSE NAND-gate output and, as a result, turns on the p-type "tattletale" transistor, feeding current into the gate-drain (diode) connected n-type transistor. The gate-drain connected transistor serves as a common transresistor for all product cells in the SOP array. If the tattletale transistor in any product cell turns on, then a voltage appears at the common node which is then thresholded and inverted by a push-pull inverter. So if any part of the selective element intersects with the local region of the image being processed, the tattletale transistors turn on and the output of the whole expression is FALSE. Depending on the operation being performed, the output of the push-pull inverter will either be inverted (dilation) or not (erosion).

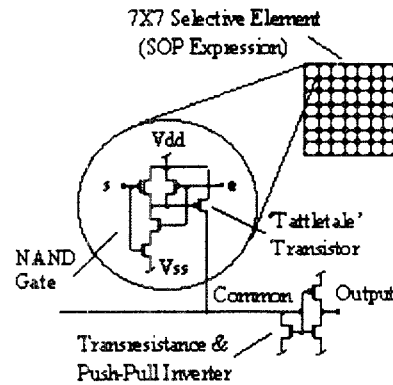


Fig. 2. Transistor-level view of a product cell, common transresistance, and push-pull inverter

Using this architecture, two SOP arrays of different sizes were fabricated in a standard 2.0 μ double-metal, double-poly ORBIT CMOS process with a 40-pin tiny chip package. A layout of the product cell is shown in figure 3 and microphotographs of the actual chip and two arrays of 3x3 and 7x7 are shown in figures 4 and 5.

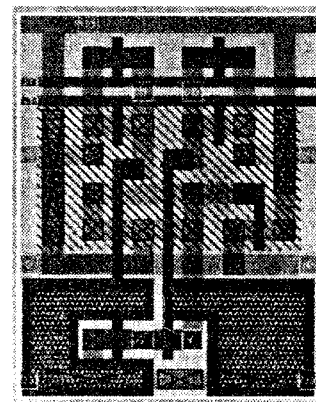


Fig. 3. Layout of one product cell

Due to the limit on the number of pins on a tiny chip, not all inputs can be presented simultaneously to the array; therefore, a multiplexing scheme was developed such that only one row of inputs is presented to the array at any given time. As part of this scheme, poly-poly capacitors were added to each input line on-chip. Due to the fact that each capacitor is finite in area, all rows must be loaded before the voltages in the first row decay. The frequency used for testing was 1 MHz (1 microsecond per row).

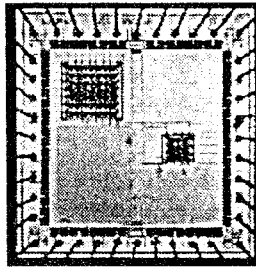


Fig. 4. Microphotograph of the monolithic implementation.

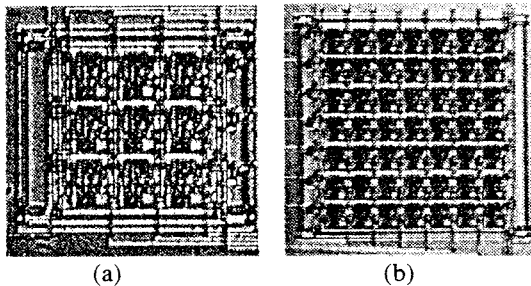


Fig. 5. Microphotograph of the (a) 3x3 and (b) 7x7 selective element arrays.

3. RESULTS

Results are shown for several combinations of 3x3 sub-images and selective elements in table 1. These images include several combinations of images with like columns. Subscripts of e and s stand for the columns of the selective element and sub-image, respectively. As expected, the output of the array is FALSE (low voltage) when the selective element intersects with the sub-image in at least one position. All measurements taken to be FALSE were less than or equal to .2 volts and all measurements taken to be TRUE were greater than or equal to 4.7 volts. The same results were obtained for the 7x7 array.

4. DISCUSSION

A mixed-mode implementation of a sum-of-product array that is capable of performing binary morphological image processing in silicon was presented. This architecture has several advantages. First of all, it is very compact, with each product cell being 78x60 square microns. Secondly, it is very much immune to process variation and noise due to much of the hardware being digital. Thirdly, this architecture can operate at low voltages, thus reducing the power consumption. (A supply voltage of five volts was used during testing only because the

TTL chips used to feed it needed five volts.) The chip can easily operate at two volts.

One disadvantage of this architecture is the fact that overlapping image data must be reloaded each time the array is moved to process another set of pixels in the original image. The throughput could be improved by pipelining the data from column to column thus allowing the circuit to produce an output

Table 1.
Test Results of the 3x3 SOP Array
(All measurements are in volts)

e_x	e_y	e_z	s_x	s_y	s_z	Out
0	0	5	0	0	0	5
0	0	5	0	0	5	0
0	0	5	0	5	0	5
0	0	5	0	5	5	0
0	5	0	0	0	0	5
0	5	0	0	0	5	5
0	5	0	0	5	0	0
0	5	0	0	5	5	0
0	5	0	5	0	0	5
0	5	0	5	0	5	5
0	5	0	5	5	0	0
0	5	0	5	5	5	0
0	5	5	0	0	0	5
0	5	5	0	0	5	0
0	5	5	0	5	5	0
5	5	5	0	0	0	5
5	5	5	0	0	5	0
5	5	5	0	5	0	0
5	5	5	0	5	5	0

at each clock cycle rather than waiting for all clock phases per cycle. Although the complexity of the interface circuitry would be somewhat increased, the throughput rate could be improved.

REFERENCES

- [1] C.R. Giardina and E.R. Dougherty, *Morphological Methods in Image Processing and Signal Processing*, Prentice Hall, Englewood Cliffs (NJ), 1988.
- [2] J. Serra, *Image Analysis and Mathematical Morphology*, Academic Press, 1982.
- [3] P. Salemier, L. Torres, F. Meyer, and C. Gu, *Region-Based Video Coding Using Mathematical Morphology*, (CNNA-94), Rome, pp. 843-857, December 1994.