Concurrent Error Detection in Array Dividers by Alternating Input Data

Chin-Long Wey

Department of Electrical Engineering
Michigan State University
East Lansing, MI 48824

ABSTRACT Concurrent error detection (CED) schemes utilizing time redundancy can keep chip area and interconnect to a minimum. An efficient time redundancy scheme, RESO, for array dividers has been reported. Under the same cell fault model, an alternating time redundancy CED scheme using alternating logic (AL) approach is proposed. The key to the detection of faults using AL approach is determining that at least one input combination exists for which the error does not result in alternating outputs. Results of this study show that the proposed design achieves the same CED capability as RESO implementation yet with a lower area overhead. Due to the simplicity and low area overhead the proposed AL approach will be very attractive to the design of fault-tolerant VLSI-based system.

I. Introduction

With the ever-increasing complexity of digital applications, the issue of reliability has become very important in today’s VLSI designs. Reliability can be improved by sophisticated testing schemes to weed out faulty circuits [1]. However, such offline or static tests can identify permanent faults only, but not transient faults. It is obvious that mechanism for concurrent error detection (CED) must be installed to detect such faults before they cause undesirable results.

All CED schemes detect errors through conflicting results generated from operations on the same operands. CED can be achieved through space or time redundancy, or space/time hybrid redundancy [2-8]. Time redundancy employs only one single set of hardware to carry out the repeated operations. Since the same hardware is used, the repeated operation, in the presence of faults, is liable to produce the same erroneous result as that of the first step. To avoid this problem, the operands must be coded in the repeated cycle, and the result thus obtained must be decoded back to the appropriate form for meaningful comparison. Two simple time redundancy techniques have been reported: RESO (Recomputing with Shifted Operands) [2] and AL (Alternating Logic) [9]. While RESO uses arithmetic shift as the encoding function, AL employs the complementation operator as the encoding function.

The alternating logic concept can be applied to any combinational logic circuits that possess the property of self-duality. A combinational circuit is said to be self-dual if and only if \( f'(x) = f(x') \), where \( f \) and \( x \) is the output function and input vector of the circuit. For a self-dual circuit, the application of an input \( x \) followed by the complemented input \( x' \), produces outputs that alternate between 1 and 0. The key to the detection of faults using the AL approach is determining that at least one input combination exists for which the fault does not result in alternating outputs [10].

In this paper, the problem of concurrent error detection in array dividers using AL approach is investigated. Two array dividers are considered: Nonrestoring array divider (NRD) and restoring array divider (RSD) [11]. Conventional NRD and RSD designs are not self-dual [12] and cannot implement AL approach directly. A design of CED-capable array dividers was proposed in [12] that assumed the single stuck-at fault model and required larger chip area due to the cell modification. This paper considers a cell fault model that is more general than the traditional stuck-at fault model. In the next section, a design of concurrent error detectable RSD is presented. Followed by the design of NRD with CED capability. Finally, a conclusion is given in Section IV.

II. Design of Concurrent Error Detectable NRD

Consider a 4-by-4 NRD, as shown in Figure 1(a), that consists of controlled add/subtract (CAS) cells, where \( n_i, d_i, q_i, \) and \( r_i \) are represented as the numerator, denominator, quotient, and remainder bits. The cell functions are represented by the following mapping form [11],

\[
DXYZ \rightarrow PS, \quad S = X@Y(D)@Z \quad (1)
\]

Let \( m \) be the operation mode, \( m=0 \) for the uncomplemented version of input data, and \( m=1 \) for the complemented one. Let \( S_0 \) (\( S_1 \)) and \( P_0 \) (\( P_1 \)) be the logic values of the S-output and P-output for \( m=0 \) (\( m=1 \)), respectively, i.e., \( DXYZ \rightarrow P_0S_0 \) and \( DX'Y'Z' \rightarrow P_1S_1 \). The following truth table for a CAS cell shows that the circuits for both P-output and S-output do not satisfy the self-duality.

<table>
<thead>
<tr>
<th>DXYZ</th>
<th>P S</th>
<th>DXYZ</th>
<th>P S</th>
</tr>
</thead>
<tbody>
<tr>
<td>0000 00</td>
<td>1000 01</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0001 01</td>
<td>1001 10</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0010 01</td>
<td>1010 00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0011 10</td>
<td>1011 01</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0100 01</td>
<td>1100 10</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0101 10</td>
<td>1101 11</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0110 10</td>
<td>1110 01</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0111 11</td>
<td>1111 10</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

CH3040-3/91/0000/0114$01.00 © 1991 IEEE
Theorem 2: detectable.

2.2. Cell Fault Model

A cell fault model is considered. The fault assumptions are: (1) in an array, at most one basic cell (CAS cell, or XOR gate) is faulty at a time; (2) the fault in a basic cell can be either permanent or transient; and (3) the fault may alter the cell’s output functions in any arbitrary way, as long as the faulty cell remains a combinational circuit.

Based on these fault assumptions, two cases can be identified at the outputs of a CAS cell in both operation modes: (a) both P-outputs and/or S-outputs are alternated; and (b) P-outputs and/or S-outputs are not alternated. Since the CAS is a self-dual, the former case identifies a fault-free cell, and the latter one detects a faulty cell. The latter case is equivalent to that the P-output and/or S-output data are temporarily stuck-at 1 or 0 in both operation modes. (Such a error is referred to as stuck-at (s-a) error.) Therefore, any error caused by single/multiple permanent (transient) fault(s) in one cell can be identified as the following error types, as shown in Figure 2:

Type 1 error: only S-output has a s-a error.
Type 2 error: only P-output has a s-a error.
Type 3 error: both S- and P- outputs have s-a faults.

Theorem 2: The modified NRD is concurrent error detectable.

Lemma 1: If DXYZ --> P0S0 and DX'Y'Z' --> P1S1 (3)

by (3), S0=(S0)' and P1=(P0)' [13], both S-output and P-output satisfy the self-duality. Thus, the CAS cell is self-dual.

2.1. Self-dual NRD Design

Figure 1(b) shows a self-dual NRD design that consists of CAS cell array, XOR gates, and equality checkers. The quotient bit equality checker compares the quotient bit q0, obtained in the mode m=0, with q1, obtained in m=1. Similarly, the remainder bit checker compares r0 with r1. The quotient bit output q0 is used as the D-input for both operation modes. Finally, the XOR gates are added to the rightmost cells in order to allow the Z-inputs to be alternated.

Theorem 1: The modified NRD is self-dual.

Due to the limitation of space, the proofs of all theorems and lemmas are omitted, but it is available in [13]. Figure 1(b) also illustrates the application of alternating inputs to the modified NRD. The alternated quotient and remainder outputs show that the modified NRD satisfies the self-duality. It should be mentioned that the CAS cells in the modified NRD are exactly the same as that in the original NRD. Thus, this implementation requires no cell modification.

2.2. Cell Fault Model

The truth table also gives a very important message that the circuits for both P-output and S-output can be self-dual if the same D-input is applied to both operation modes, i.e., the mappings for both operation modes are

DXYZ --> P0S0 and DX'Y'Z' --> P1S1 (3)

by (3), S0=(S0)' and P1=(P0)' [13], both S-output and P-output satisfy the self-duality. Thus, the CAS cell is self-dual.

2.3. Design Evaluation

Figure 1(c) illustrates the application of alternating inputs to the modified NRD, where the nonalternated outputs at q2, r2, and r3 identify the existence of a faulty cell in that array.

The proposed n-bit NRD design, as shown in Figure 1(b), requires (4n-1)'s 2-input XOR gates for q1's, d1's and the rightmost cells, and two equality checkers for q1's and r1's. On the other hand, RESO implementation requires one additional row and three additional cells per row in the original array, where RESO-(2,3) is assumed. The extra hardware includes (n+3)(n+1)-n2 (=4n+3) additional CAS cells, four shifters of length (n+3)-bits, and two equality checkers of (n+3)-bits. Assuming the complexity of each bit of the shifter and the checker to be approximately half that of a CAS cell [3], the relative increase in the original divider circuit is (7n+12)/n2. Assuming also the complexity of a XOR gate to be nearly quarter that of a CAS cell, thus, the area overhead in AL implementation is (2.5n-0.75)/n2. Table 1 compares the area overhead for both AL and RESO implementations. Results show that the area overhead in the AL implementation is about 1/3 that of RESO implementation. Also, with extra 8% hardware overhead, the 32-bit modified NRD can achieve error detection in real-time applications. The proposed design has the following salient features: (1) no cell modification; (2) low overhead; and (3) technology independent. Thus, the design concept is readily implemented to existing NRD designs for achieving the concurrent error detection.

III. Design of Concurrent Error Detectable RSD

Consider a 4-by-4 RSD, as shown in Figure 3(a), that consists of controlled subtractor (CS) cells. The cell functions are represented by the following mapping form,

DXYZ --> PS, where S = D'X@Y@Z+DX (4)
and P = X'Y+X'Z+YZ

Lemma 2: The circuit for P-output is self-dual, but for S-output is not self-dual.

As given in [11], Figure 3(b) shows a modified RSD design that consists of modified CS (MCS) cells. The cell functions are represented by the following mapping,

DXYZA --> PSB, where S = D'X@Y@Z+DX (5)
and P = X'Y+X'Z+YZ
and B = A@Y@Z

The mappings for the both operation modes are

DXYZA --> P0S0B0 and DX'X'Y'Z' --> P1S1B1 (6)

Lemma 3: The MCS cell is self-dual.
In our implementation, the extra pin overhead for \( a_i \)'s and \( b_i \)'s in Figure 3(b) are reduced by connecting \( a_i \)'s to the inputs \( n_i \)'s and connecting \( b_i \)'s to the equality checkers, as shown in Figure 3(c). So, no extra external pins are needed in this implementation.

**Theorem 3:** The modified RSD is self-dual and concurrent error detectable.

The proposed \( n \)-bit RSD design requires \( (3n-1)'s \) 2-input XOR gates, three equality checkers, and \( n^2 \) MCS cells. According to the cell layouts shown in Figure 4, a MCS cell is nearly 5% more than the original CS cell. Thus, the area overhead is \( 0.05+\frac{2.25n-0.75}{n^2} \). On the other hand, the area overhead for RESO implementation, with RESO-(2,3), is \( (7n+12)/n^2 \). Table 1 compares the area overhead for both AL and RESO implementations. Results show that, with extra 12% hardware overhead, the 32-bit modified RSD is capable of detecting errors in real-time applications. Results also show that AL implementation requires less chip area than RESO implementation.

### IV. Conclusion

Design of self-dual array dividers with concurrent error detection (CED) capability using alternating input data approach is presented. Errors are detected by the nonalternated remainder or quotient bit outputs with the application of alternating input data. Results show that, with less 8% area overhead, the proposed 32-bit nonrestoring array divider (NRD) achieves the concurrent error detection where the cell fault model is assumed. In addition, since the proposed NRD design requires no cell modification, it is very easy for designers to convert existing NRD designs with various cell design alternatives to achieve the CED capability. Results also show that the proposed 32-bit restoring array divider (RSD) design requires 12% for achieving the concurrent error detection. The overhead required in the proposed design is much lower than both RESO implementation and the AL implementation proposed in [12].

### REFERENCES


**Table 1. Comparison of Area Overhead -- NRD**

<table>
<thead>
<tr>
<th>Type</th>
<th>16-bit</th>
<th>32-bit</th>
<th>64-bit</th>
</tr>
</thead>
<tbody>
<tr>
<td>RESO</td>
<td>48.4%</td>
<td>23.1%</td>
<td>11.2%</td>
</tr>
<tr>
<td>AL (NRD)</td>
<td>15.4%</td>
<td>7.7%</td>
<td>3.9%</td>
</tr>
<tr>
<td>AL (RSD)</td>
<td>18.8%</td>
<td>12.0%</td>
<td>8.5%</td>
</tr>
</tbody>
</table>

**Figure 2: Error Models and Propagation Paths.**
Figure 1: Non-Restoring Array Divider (NRD): (a) Schematic Diagram; (b) Application of Alternating Input Data; and (c) Faulty Circuit.

Figure 2: Non-Restoring Array Divider (NRD): (a) Schematic Diagram; (b) Application of Alternating Input Data; and (c) Faulty Circuit.

Figure 3: Restoring Array Divider (RSD): (a) Schematic Diagram; (b) & (c) Modified RSD.

Figure 4: Physical Layout for Divider Cells: (a) CS Cell; and (b) MCS Cell.