Design of Concurrent Error-Detectable VLSI-Based Array Dividers

Thou-Ho Chen , Liang-Gee Chen and Yi-Shing Chang

Department of Electrical Engineering, National Taiwan University
Taipei, Taiwan 10764, R.O.C.

Abstract

A building-block design of VLSI-based array dividers with concurrent error detection by REcomputing using Partitioned Architecture (REPA) is proposed in this paper. The basic concept is that the divide array can be divided into two identical parts and its operation can be completed by using one part through two iterative calculations. With two such parts, we can design a CED (concurrent error detection) scheme of space redundancy approach and the detecting action is achieved at each iteration. The design is better than other previous designs such as RESO and AL, in terms of area requirement, time penalty, error detection capability and detecting period. Advanced analysis in m partitions is also included. The experiment results are attractive especially for various designs with application-specified tradeoffs between speed performance and area cost.

1 Introduction

Conventionally, system reliability has been achieved by two basic strategies: fault avoidance and fault tolerance. In the case of fault avoidance, the reliability can be improved by the use of high-reliability components which are provided through the sophisticated testing schemes. Those schemes can identify permanent faults only, but not transient faults. This inadequacy has motivated the development of fault-tolerance techniques, which can tolerate both permanent and transient faults during the system operation. Concurrent error detection (CED) technique is a better choice to do fault tolerance due to its lower cost. Many CED techniques [1]-[4] by using either space redundancy, time redundancy or hybrid redundancy have been presented. A process type of designing CED-capable arithmetic units, the standard building-block design which is originated from the concept of the standard cell approach to circuit design, is popularly utilized [2][5]. It facilitates the automatic design process and is easily applied to design fault-tolerant system. In this paper, an alternative building-block design of array divider with concurrent error detection by recomputing using the partitioned architecture (REPA) is proposed. Based on the swapping method [4], the basic idea is that the regular arithmetic array can be divided into two identical half-parts. The original function is then completed through 2-step computations and error detection can therefore be achieved during each computation step by organizing the space redundancy scheme. Comparison with the RESO (recomputing with shifted operands) [1] and the AL (alternating logic) [3] has revealed the superiority of the proposed method. In addition, generic design of m-partition, REPA-m, is deduced and analyzed. Extensions to error correction scheme and other array structures are also discussed. The area and time delay overheads comparison provided in this paper is based on the evaluation of gate-count and gate-delay requirements, respectively. For lack of considering the interconnection problem, only a rough estimate is derived. However, the means of comparison is available since the interconnection attributes is too various to be tractable in general and its efficiency depends on strategies of routing and placement.

2 Methodology

The structure of cellular array divider is very regular such that we can directly divide it into two identical n-by-(n/2) sub-dividers. Each sub-divider can complete the full division by executing twice. Conventionally, the space-redundant scheme uses two sets of hardware with comparison to perform the detective action. Using two n-by-(n/2) sub-dividers to organize a space-redundant scheme, results of division will be obtained through two half-computations but error detection can be achieved during each half-computation. Fig. 1 shows the CED scheme of n-by-n nonrestoring array division (also denoted as NRD here) [6] using REPA. At first, divisor bits d_0~d_{n-1} are input to all cells of the divider for both half-computations. In the first half-computation, the 2-1

1063-6404/92 $3.00 © 1992 IEEE

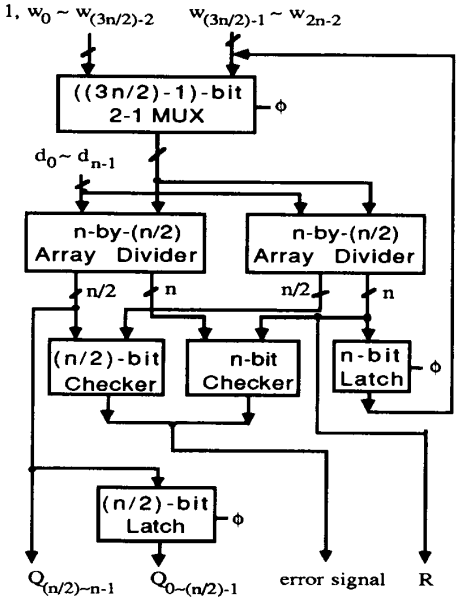

Fig. 1 The CED scheme of n-by-n NRD using REPA.

MUX transfers one and dividend bits $w_0 \sim w_{(3n/2)-2}$ as input operands to the first row of cells and then two sets of lower-half quotient bits are derived. Further, both lower-half quotients are compared each other to do error detection and one of the both must be latched for the final quotient. Also the intermediate result generated in the last row should be checked the same as the lower-half quotient but be further latched for input to the first row in the subsequent half-computation step. During the second half-computation, dividend bits $w_{(3n/2)-1} \sim w_{2n-2}$ and the intermediate data stored in the latch are transferred to the diagonal row and first row, respectively, by the multiplexer. Thus, the upper-half quotient and the final remainder are generated and both also need to be checked.

3 Comparison with Others

Comparison made in this section is addressed at the building-block level. Two effective CED-capable designs of array divider by time redundancy have been presented previously: RESO [1] and AL [3]. RESO-(l, m) uses the recomputing scheme of shifting the dividend by l bits and the divisor by m bits for achieving error detection. The capability of error detection increases with the values of l and m, but large l and m will result in more extra cells required. In AL approach, the original circuit must be modified as possessing the property of self-duality and then uses the complemented input for the recomputation. Though it has only little area overhead, the fault assumption is confined to single cell fault. Another single stuck-at-fault-based CED scheme of array dividers applies the data complementation strategy [7] and it utilizes the modified cells which result in requiring larger chip area. The proposed CED design requires no cell modification, and can detect errors resulting from single-cell and multiple-cell fault models confined to one of the both sub-dividers. This assumption makes our architecture more general than the traditional stuck-at fault model and powerful than only single-cell fault model. Furthermore, it needs negligible time overhead including multiplexing and latching time delays. Other features include: it only needs half division cycle to do error detection, which is of importance for high-speed real-time systems; and it leads to small error latency.

In order to do area evaluation, in this study, we utilize the standard cell library of CMOS technology in the Lcells generator with the GDT tools [8]. For moderate evaluation, we define α as the unit area taken in the P-N-transistor pair followed by τ as the unit pair delay corresponding to one level of the logic circuit. The P-N-transistor pair counts and gate delays of those typical cells and modules used to realize those CED designs are listed in Table 1. Table 2 shows the comparisons in terms of area overhead, time overhead, fault model and detecting period for RESO-(2, 3), AL and REPA implementations. The area overhead is defined as the percentage increase over that of the original structure without CED capability, and time overhead too. Area overheads can be formulated as $1/n - 2/(7n^2)$ for REPA, $6/n + 142/(21n^2)$ for RESO and $5/(3n) - 10/(21n^2)$ for AL, respectively.

4 Design Analysis and Extensions

Based on the principle of partitioning approaches, the array divider can be divided into many identical sub-dividers. This section analyzes the REPA-m design of partitioning approach and describes the extension to design of error correction.

4.1 Design Analysis

As mentioned in section 2, an n-by-n NRD can be directly divided into m identical n-by-(n/m) sub-dividers. Since each n-by-(n/m) sub-divider can calculate m times to complete the full n-by-n division, this partitioned design is called REPA-m. It requires m iterative sub-divsions and can be checked during each iteration. The schematic diagram of n-by-n NRD using REPA-m is shown in Fig. 2. The 2-to-1 multiplexer (MUX) is used to transfer dividend bits $w_0 \sim w_{n-2}$ to the first row. After

Table 1. Cells used for evaluation of area and time.

Cell or Module	NOT	2-input NAND	2-input NOR	2-input AND	2-input OR	2-input XOR	2-input XNOR	1-bit Latch	2-1 MUX	1-bit CAS *	1-bit D-FF *	4-1 MUX *	8-1 MUX *	16-1 MUX *	32-1 MUX *
P-N count (α)	1	2	2	3	3	5	5	3	6	21	8	91	15	36	75
Delay (τ)	1	1	1	2	2	3	3	2	2	4	4	8	2	2	4

* Modules constructed with the Lcells Standard Cell Library, Silicon Compiler Systems Corporation.

Table 2. Comparison of three CED designs.

term \ way / bits		RESO-(2,3)	AL	REPA
(area, time) overhead (%)	16	(40.0, 154.5)	(10.2, 110.6)	(6.1, 1.0)
	32	(19.4, 126.1)	(5.2, 105.0)	(3.1, 0.2)
	64	(9.5, 112.8)	(2.7, 102.4)	(1.6, 0.1)
	128	(4.7, 106.3)	(1.3, 101.2)	(0.8, ~0)
fault model		single-cell	single-cell	multiple-cell
detecting period		2 division cycles	2 division cycles	0.5 division cycle

first sub-division, it is used to transfer the intermediate data from the latch in the remaining iterations except the last one. Other dividend bits w_{n-1}~w_{2n-2} is transferred by the m-1 MUX with n/m bits at each iteration from the lower significant bits to the more significant bits. The quotient with n/m bits is derived and checked at each sub-division. The remainder is generated by the last row in the last iteration.

In the Fig. 2, area A and time T requirements of n-by-n NRD with REPA-m can be evaluated by the following formulae:

$$A = (2n^2/m)*a_{\text{CAS}} + n*a_{\text{2-1 MUX}} + (n/m)*a_{\text{m-1 MUX}} + ((mn+n)/m)*a_{\text{checker}} + n*a_{\text{latch}} + n*a_{\text{D-FF}}$$

$$T = n^2*t_{\text{CAS}} + m*t_{\text{m-1 MUX}} + m*t_{\text{checker}}$$

where a_{unit} and t_{unit} denote the area required and time delay per one bit unit, respectively. Fig. 3(a) and (b) show the area overhead-ratio and time overhead-ratio percentage, defined as the modified with CED-capability to the original structure, for 16-bit, 32-bit, 64-bit and 128-bit wordlength systems. It is clearly that only negligible time overhead, about 4%, is required and furthermore it can result in large area decrease. The larger m is, the more REPA-m benefits. Fig. 3(c) shows AT^2 overhead-ratio distributions of four wordlength systems and all decreases with increasing m. This is attributed to the good regularity of the array, simple intermediate data dependency to be divided, and large area and time requirements in itself. Though the area and time delay of interconnections are not considered, the experiment results of Fig. 3 demonstrate that REPA-m should be a

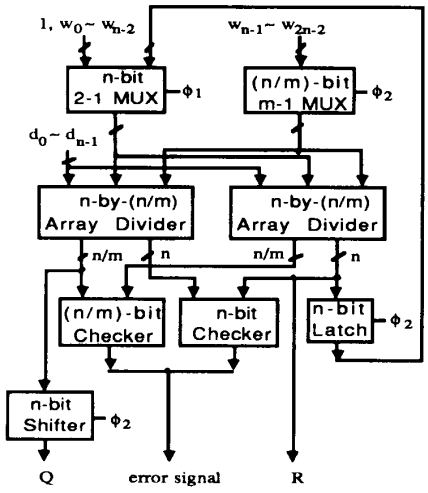

Fig. 2 The CED scheme of n-by-n NRD using REPA-m.

cost-effective CED design. In addition, various designs of specific m values depended on area-time tradeoff may be attractive for some special-purpose applications.

4.2 Extensions

The REPA method not only can be used for designing with CED-capability but also can be extended to design with error correction. Since the sub-dividers are identical, taking three such sub-dividers with triple-modular-redundancy (TMR) organization is available for achieving fault tolerance. Still n/m quotient bits is obtained at each iteration and all must be voted for the final quotient. For the same evaluation as mentioned above, it can incur a lower area cost without significantly sacrificing speed [9]. Furthermore, REPA can be applied to the restoring array divider and other regular array structures such as array multipliers [6]. Implementing a concurrent error-detectable array multiplier, the original structure must be modified before utilizing REPA method, for reducing extra computing time during iterative operations. The efficiency is not good as that of divider because the cellular regularity and intermediate data dependency of

processing flow in multiply array are inferior compared to divide array.

5 Conclusions

In this paper we have presented a CED design in array dividers using REPA. The results of performance analysis and evaluation have shown that the proposed design is better than previous designs in terms of detecting period, area and time overhead required. REPA-m can incur a lower area cost with increasing m, without resulting in large speed penalty. Owing to these features, the proposed approach should be very attractive for designing CED-capable and/or fault-tolerant VLSI-based array dividers. Future work will lay an emphasis on study of area and time overhead problems in the cases of large m by implementing a REPA-m in the form of solid state LSI's.

References

[1] J. H. Patel and L. Y. Fung, "Concurrent error detection in multiply and divide arrays," *IEEE Trans. Computers*, Vol. C-32, pp. 417-422, April 1983.

[2] B. W. Johnson, J. H. Aylor and H. H. Hana, "Efficient use of time and hardware redundancy for concurrent error detection in a 32-bit VLSI adder," *IEEE Journal of Solid-State Circuits*, Vol. 23, pp. 208-215, Feb. 1988.

[3] C. L. Wey, "Concurrent error detection in array dividers by alternating input data," in *Proc. IEEE Int. Conf. Computer Design (ICCD)*, pp. 114-117, 1991.

[4] L. G. Chen and T. H. Chen, "Computation with simultaneously concurrent error detection using bi-directional operands," in *Proc. IEEE Int. Conf. Computer Design (ICCD)*, pp. 128-131, 1989.

[5] D. A. Rennels, A. Avizienis and M. Ercegovac, "A study of standard building blocks for the design of fault-tolerant distributed computer systems," in *Proc. 8th IEEE Ann. Conf. Fault-Tolerant Computing (FTCS)*, pp. 144-149, 1978.

[6] K. Hwang, *Computer arithmetic: principles, architecture, and design.* Wiley, New York, 1979.

[7] K. Furuya, Y. Akita and Y. Tohma, "Logic design of fault-tolerant dividers based on data complementation strategy," in *Proc. 13th IEEE Fault Tol. Comp. Symp. (FTCS)*, pp. 306-313, 1983.

[8] *Lcells Generator Library User Guide, Generator Development Tools*, Ver. 3, Silicon Compiler Systems Co., 1987.

[9] T. H. Chen, L. G. Chen and Y. S. Jehng, "A partitioning approach to design fault-tolerant arithmetic arrays," in *Proc. 11th IEEE International Phoenix Conference on Computers and Communications (IPCCC)*, pp. 432-439, 1992.

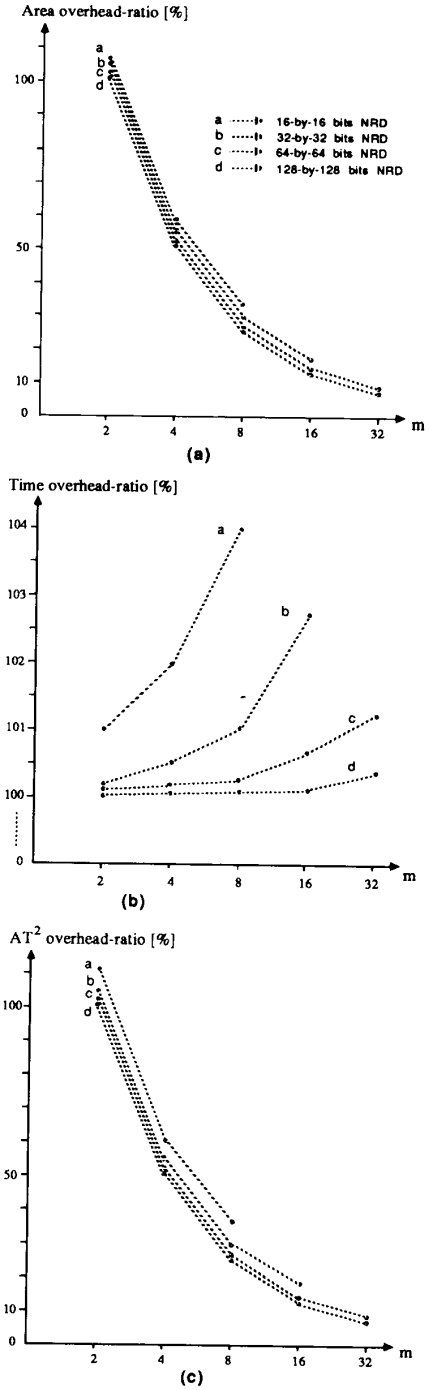

Fig. 3 Evaluation in NRD with error detection using REPA-m: (a) gate-count distribution; (b) time-delay distribution; (c) AT^2 cost distribution.