Increasing Implementability of β-driven Threshold Checkers

V. Varshavsky
NNT Israel
vviktor@zahav.net.il

I. Levin
Tel Aviv University
Israel
i.levin@ieee.ogr

V. Ostrovsky
Tel Aviv University
Israel
vios@post.tau.ac.il

Implementation of checkers on the basis of threshold functions is problematic due to low stability of component threshold schemes. For increasing the implementability, the paper investigates two methods of improving stability of threshold based checkers. One of the methods is majorization of the threshold schemes. The second method is so-called calibration of the mentioned schemes. Both of the methods are described and evaluated from the point of expected implementability.

The basic difficulty of the checker's synthesis is the fact that minterms corresponding to the checker's codewords are unordered. Universal checkers do not check whether a word belongs to the complete set of codewords, but may check whether a word has some specific property, which distinguishes a codeword from a non-codeword. Usually, a number of "1" (or "0") positions in the word is checked to this end. In the Berger code, this number is set in control positions of the code; in the *m-out-of-n* code, this number is defined by the code itself. A considerable portion of a universal checker is occupied by a device computing the number of "1" positions in a binary word. It has been shown by Lala [1] that digital adders, if used for the above purpose, result in a considerable overhead. Threshold elements using analog circuits for computing the "1" positions may effectively reduce the overhead [2]. However, there are some considerable difficulties on the way of using such implementations. As long as β-driven threshold element [3] is an analog system in its main part, correctness of its functioning is apparently associated with the precision of manufacturing and stability of the parameters' maintenance during the operation process. This assertion is purified by the definition of implementability, that determines the limitations on the class of threshold functions; therefore correct (faultless) work can be guaranteed with the given technological limitations and operating conditions. It was shown in paper [3], that the implementability can be estimated as: $\min(\eta, \sum_{j=0}^{m-1} \omega_j - \eta)$, where: ω_j - the weight of the j–th input; η - threshold.

The present paper investigates ways of increasing the implementability of the specific threshold based checkers on an example of implementing of the following threshold function:

$$B(X,Y) = Sign(\sum_{j=0}^{n-1} x_j + \sum_{i=0}^{k-1} 2^i \cdot y_i - n)\text{, where:}$$

x_j - information bits, y_i - control bits and n – a number of the information bits.

The circuit implementing this function may be a portion of a checker of the Berger code, or of a checker of the *m-out-of-n* code, or even of a checker of the context-oriented code.

The paper proposes two methods allowing increasing the implementability: a) by introducing a relatively small overhead for preliminarily exclusion of defective circuits; b) by using a specific procedure of calibration.

The first method is proposed upon considering a method of majority reservation. The PSPICE MC Simulation results appear to show that the majority reservation cannot be used as universal method since it works with quite a high percentage of defective circuits. It therefore looks reasonable to use full exclusion of defective circuits instead of majority signal recovery.

The second of the proposed methods for increasing implementability is based on a specific learning procedure that is known from studying β-driven threshold elements as a basis for CMOS artificial neurons. It is known from those studies that the learning procedure allows implementing the threshold elements with the threshold value more then 200. The procedure of increasing a threshold element's implementability by compensation of deviation of a technological parameter we call as calibration. The calibration method seems much more promising in comparison with the method of majority reservation. It guarantees significant growth of the threshold checkers implementability. The procedure of calibration is presented and corresponding circuits are proposed.

The simplest way of recalibration without loss of operation productivity is using a couple of threshold elements: one of them is operating and the second one is calibrating. Notice, that the checker is, actually, tested during the calibration that can be considered an additional advantage of the proposed approach.

References

[1] P. Lala. *Self-Checking and Fault Tolerant Digital Design*, Morgan Kaufman Publishers, 2001.

[2] S. J. Piestrak, "Design of Encoders and Self-Testing Checkers for Some Systematic Unidirectional Error Detecting Codes", *Proceedings of the 1997 Workshop on Defect and Fault-Tolerance in VLSI Systems (DFT '97).*

[3] V. Varshavsky, "β̃-Driven Threshold Elements", *Proceedings 8th Great Lakes Symposium on VLSI*, Lafayette, Lo, USA, Feb. 19-21 1998, IEEE Computer Society, pp 52-58.

Proceedings of the 9th IEEE International On-Line Testing Symposium (IOLTS'03)
0-7695-1968-7/03 $17.00 © 2003 IEEE