ELSEVIER

Journal of Systems Architecture 47 (2001) 477–489

JOURNAL OF SYSTEMS ARCHITECTURE

www.elsevier.com/locate/sysarc

Baldwinian learning utilizing genetic and heuristic algorithms for logic synthesis and minimization of incompletely specified data with Generalized Reed–Muller (AND–EXOR) forms

Karen M. Dill *, Marek A. Perkowski

Department of Electrical Engineering, Portland State University, Portland, OR 97207, USA

Accepted 12 November 2000

Abstract

This research applies a new heuristic combined with a genetic algorithm (GA) to the task of logic minimization for incompletely specified data, with both single and multi-outputs, using the Generalized Reed–Muller (GRM) equation form. The GRM equation type is a canonical expression of the Exclusive-Or Sum-of-Products (ESOPs) type, in which for every subset of input variables there exists not more than one term with arbitrary polarities of all variables. This AND–EXOR implementation has been shown to be economical, generally requiring fewer gates and connections than that of AND–OR logic. GRM logic is also highly testable, making it desirable for FPGA designs. The minimization results of this new algorithm tested on a number of binary benchmarks are given. This minimization algorithm utilizes a GA with a two-level fitness calculation, which combines human-designed heuristics with the evolutionary process, employing Baldwinian learning. In this algorithm, first a pure GA creates certain constraints for the selection of chromosomes, creating only genotypes (polarity vectors). The phenotypes (GRMs) are then learned in the environment and contribute to the GA fitness (which is the total number of terms of the best GRM for each output), providing indirect feedback as to the quality of the genotypes (polarity vectors) but the genotype chromosomes (polarity vectors) remain unchanged. In this process, the improvement in genotype chromosomes (polarity vectors) is the product of the evolutionary processes from the GA only. The environmental learning is achieved using a human-designed GRM minimization heuristic. As much previous research has presented the merit of AND–EXOR logic for its high density and testability, this research is the first application of the GRM (a canonical AND–EXOR form) to the minimization of incompletely specified data. © 2001 Published by Elsevier Science B.V.

Keywords: Incompletely specified generalized Reed–Muller forms; AND–EXOR forms; Logic synthesis and minimization; Baldwinian learning; Genetic algorithms

1. Introduction

Natural systems are extremely well adapted to their environments. In nature the structure of all organisms has been designed to provide the capability of solving a multitude of complex problems for both survival and growth, through instinctual, experiential, and intellectual means.

* Corresponding author. Tel.: +503-725-5411; fax: +503-725-4882.

E-mail addresses: dills@worldnet.att.net (K.M. Dill), mperkows@ee.pdx.edu (M.A. Perkowski).

1383-7621/01/$ - see front matter © 2001 Published by Elsevier Science B.V.
PII: S1383-7621(00)00060-6

Over the millennia, it is these capabilities that have proven an effective method of sustaining the existence and propagation of natural organisms. Even as environmental conditions change, it is the continuous adaptation of natural organisms adjusting to their environment, through evolutionary processes, which sustains life. As nature itself has proven its development of robust system design, the biologically inspired evolutionary process is here applied to the minimization of incompletely specified digital hardware circuits in the Generalized Reed–Muller (GRM) form.

Genetic Algorithm (GA) [15] techniques provide a means for applying the theory of evolution within an artificial system. The GA is a system that evolves problem parameters directly. Through a process of emergent intelligence, the GA formulates engineering solutions based on an accumulated knowledge of the problem and the merit of potential solutions. In recent years, the GA, as a machine learning technique, has been successfully applied to a wide range of engineering problems. However, with the realization of computer-designed algorithms, only limited research to date has applied these concepts to digital logic [11,13]. Our past experience has shown the GA application to logic minimization to have limitations of size, computation time, and solution optimality [7,8]. In comparison, several decades of research have contributed to the current human understanding and efficient implementation of systems for logic design and minimization.

Most of the current human-aided design tools utilize AND–OR design implementations for both logic synthesis and minimization, for simplicity, readily available minimization tools, and because of historical reasons in the development of digital systems. However, since AND and EXOR components exist in most ASIC libraries, they can be employed either in a CLB of an FPGA or in a full custom VLSI design. While not as widely utilized for integrated circuit design as the AND–OR Sum-of-Product (SOP) logic, the Exclusive-Or Sum-of-Product (ESOP) form (the most general, unrestricted AND–EXOR logic form) compares favorably [40]:

> Functions realized by such circuits (ESOPs) can have fewer gates, fewer connections, and take up less area in VLSI and especially, FPGA realizations. They are also easily testable [14,28]. It was shown, both theoretically and experimentally [31,33–35,38] that ESOPs have on average smaller numbers of terms for both 'worst case' and 'average' Boolean functions. It was also shown that ESOPs and all their sub-families have their counterparts in logic with multi-valued inputs...

Additionally, recently two-level AND–EXOR realizations (ESOPs) have even been proposed for the combinational logic portions of finite state machines, as they have proven more testable and can yield less area than two-level AND–OR implementations [21]. Thus, it can be concluded that the AND–EXOR implementation is in many applications superior to the AND–OR logic, for both its testable and economical characteristics [21,36,37,39].

AND–EXOR logic already serves as a good alternative basis for FPGA designs, by achieving denser logic than AND–OR implementations. Because any function of a certain number of inputs can be realized in a logic block of an FPGA, for example, the costs of realizing a 5-input OR and a 5-input EXOR in Xilinx technology are the same. (Note that FPGAs may use logic look-up tables to realize AND–EXOR combinations in multi-level circuits, thus avoiding the architectural design restrictions imposed by other AND–OR-based logical devices [44]. However, this is at the expense of not employing the two-level, AND–EXOR physical hardware implementation, which realizes the high testability quality of this class.) AND–EXOR logic has also been used in full custom VLSI chip design of data path logic, arithmetic circuits, controllers with embedded counters and adders, and easily testable circuits. For several industrial applications, specialized AND–EXOR tools serve for such designs as preprocessors for standard factorization and decomposition in EDA tools from commercial companies.

It is interesting to note that binary AND–EXOR logic represents a special case of Galois Field logic [1,41], GF(k), where the radix $k = 2$.