EMBRYONICS: A NEW FAMILY OF COARSE-GRAINED FIELD-PROGRAMMABLE GATE ARRAY WITH SELF-REPAIR AND SELF-REPRODUCING PROPERTIES

Daniel MANGE, Maxime GOEKE, Dominik MADON, André STAUFFER, Gianluca TEMPESTI
The Swiss Federal Institute of Technology - CH 1015 LAUSANNE (Switzerland)
Phone: (+41 21) 693 26 39 Fax: (+41 21) 693 37 05
e-mail: mange@di.epfl.ch

Serge DURAND, Pierre MARCHAL, Pascal NUSSBAUM
Centre suisse d'électronique et de microtechnique SA - CH 2000 NEUCHATEL (Switzerland)
Phone: (+41 38) 205 662 Fax: (+41 38) 205 630
e-mail: marchal@csemne.ch

ABSTRACT

The growth and the operation of all living beings are directed through the interpretation, in each of their cells, of a chemical program, the DNA. This program, called *genome*, is the blueprint of the organism and consists of a sequence of four discrete characters: A, C, G, and T. This process is the source of inspiration for the Embryonics (embryological electronics) project, whose final objective is the conception of very large scale integrated circuits endowed with properties usually associated with the living world: self-repair (cicatrization) and self-reproduction. Within this framework, we will present a new family of coarse-grained field-programmable gate arrays. Each cell is a binary decision machine whose microprogram represents the genome, and each part of the microprogram is a gene whose execution depends on the physical position of the cell in the array, i.e. on its coordinates. The considerable redundancy introduced by the presence of a genome in each cell has significant advantages: self-reproduction (the automatic production of one or more copies of the original organism) and self-repair (the automatic repair of one or more faulty cells) become relatively simple operations. Even if the described system seems exceedingly complex, we believe that computer architectures inspired by molecular biology will allow the development of new FPGAs endowed with quasi-biological properties extremely useful in environments where human intervention is necessarily limited (nuclear plants, space applications, etc.).

1. INTRODUCTION

A human being consists of approximately 60 trillion (60×10^{12}) cells. At each instant, in each of these 60 trillion cells, the *genome*, a ribbon of 2 billion characters, is decoded to produce the proteins needed for the survival of the organism. This genome contains the ensemble of the genetic inheritance of the individual and, at the same time, the instructions for both the construction and the operation of the organism. The parallel execution of 60 trillion genomes in as many cells occurs ceaselessly from the conception to the death of the individual. Faults are rare and, in the majority of cases, successfully detected and repaired. This process is remarkable for its complexity and its precision. Moreover, it relies on completely discrete processes: the chemical structure of DNA (the chemical substrate of the genome) is a sequence of four bases, usually designated with the letters A (adenine), C (cytosine), G (guanine), and T (thymine). Each group of three bases is decoded in the cell to produce a particular amino acid, a future constituent of the final protein (thus, the triplet ACG will produce threonine [1]).

Our research is inspired by the basic processes of molecular biology [2]. By adopting certain features of cellular organization, and by transposing them to the two-dimensional world of integrated circuits on silicon, we will show that properties unique to the living world, such as self-reproduction and self-repair, can also be applied to artificial objects (integrated circuits). Our final objective is the development of very large scale integrated circuits (wafer scale integration) capable of self-repair and self-reproduction. These two properties, characteristic of the living world, seem particularly desirable for very complex artificial systems meant for hostile (nuclear plants) or inaccessible (space) environments. Self-reproduction allows the complete reconstruction of the original device in case of a major fault, while self-repair allows a partial reconstruction in case of a minor fault.

2. EMBRYONICS

Embryonics, i.e. the quasi-biological development of multicellular automata, is based on a *general hypothesis*, which describes the environment in which the development occurs, and on *three features*, which roughly approximate the biological mechanism of cellular development [3].

2.1 The general hypothesis: the environment

The general hypothesis describes the selected environment in which the quasi-biological development occurs. In the framework of electronics, it consists of a finite (but as large as desired) two-dimensional space of silicon, divided into rows and columns (the third dimension introduced by the physical realization of the system is irrelevant for our purposes). The intersection of a row and a column defines a *cell*, and all cells have an identical physical structure, i.e. an identical network of connections (wires) and an identical set of operators (combinational and sequential logic operators). The physical space or *cellular array* is therefore *homogeneous*, that is, made up of absolutely identical cells: only the *state* of a cell, that is, the combination of the values in its memories, can differentiate it from its neighbors.

The transformations, in particular the development of an artificial multicellular organism from a single cell, the mother cell, occur then without supply of material (the network of silicon is present at the start), with a trivial supply of energy (the classical power supply of an electronic system), and with a crucial supply of information.

2.2 First feature: multicellular organization

The first feature is that of *multicellular organization:* the artificial organism is divided into a finite number of cells, where each cell realizes a unique function, described by a sub-program called the *gene* of the cell. The same organism can contain multiple cells of the same kind (in the same way as a living being can contain a large number of cells with the same function: nervous cells, skin cells, liver cells, etc.). In this presentation, for clarity's sake, we will confine ourselves to a simple example of a 1-dimensional artificial organism (Fig. 1): a *random number generator* implemented with five cells and featuring two distinct genes (rules 90 and 150 of Wolfram's cellular automaton [4]).

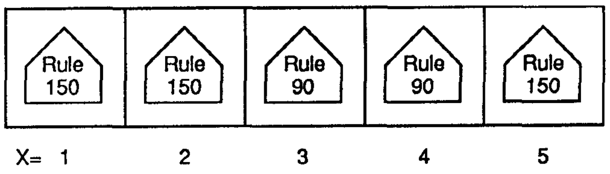

Fig. 1. A 1-dimensional non-uniform multicellular organism.

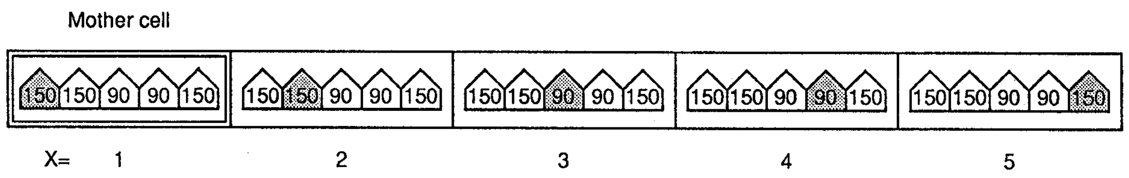

Fig. 2. The 1-dimensional organism with a genome in every cell.

2.3 Second feature: cellular differentiation

Let us call *genome* the set of all the genes of an artificial organism, where each gene is a sub-program characterized by a set of instructions and by its position (its coordinates X,Y). Fig. 1 then shows the genome of the random number generator, with the corresponding horizontal (X) coordinate. Let then each cell contain the entire genome (Fig. 2): depending on its position in the array, i.e., its place in the organism, each cell can interpret the genome and extract and execute the gene which configures it. In summary, storing the whole genome in each cell makes the cell universal: it can realize any gene of the genome, given the proper coordinates.

2.4 Third feature: cellular division

At startup, the mother cell or *zygote* (Fig. 2), arbitrarily defined as having the coordinate X = 1, holds the one and only copy of the genome. At time t+1, the genome of the mother cell is copied into the neighboring (daughter) cell to the east. The process then continues until the 1-dimensional space is completely programmed. In our example, the furthest cell is programmed at time t+4. In the more general case of 2-dimensional arrays, the genome of the mother cell is copied into the two neighboring daughter cells to the north and to the east, and so on until the 2-dimensional space is completely programmed.

3. EXAMPLE: A RANDOM NUMBER GENERATOR

Wolfram [4] exhaustively studied 1-dimensional cellular automata consisting of identical cells where the future state Q+ of a cell is a function (*rule*) of the state Q of the cell itself, the state QW of its left-hand (western) neighbor, and the state QE of its right-hand (eastern) neighbor. Among all the possible rules, we are interested in the types 90 and 150 defined by the following boolean equations:

$$Q+ = QW \oplus QE$$

$$Q+ = QW \oplus Q \oplus QE$$

Hortensius et al. [5] have shown that a well-chosen arrangement of Wolfram cells of type 90 and 150 produces a *non-uniform cellular automaton* which is, in fact, a random number generator. For a 5-cell automaton, the final arrangement is shown in Fig. 3.

3.1 Computing the gene for rule 150

In all human beings, the chain of characters which makes up the DNA is executed sequentially by a chemical processor, the *ribosome*. Drawing inspiration from this biological mechanism, we use a microprogram to compute first each of the genes of the artificial organism, then its coordinates, and finally its complete genome.

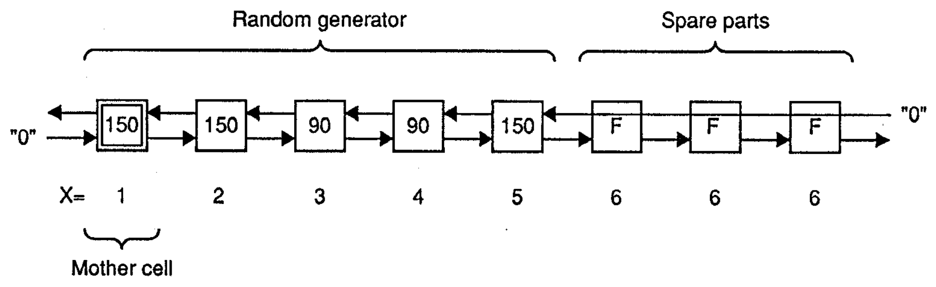

Fig. 3. A 5-cell (X=1...5) random number generator with 3 spare cells (X=6).

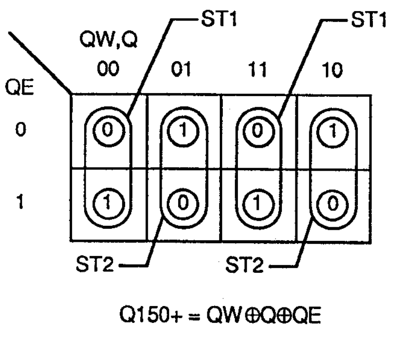

Fig. 4. Gene for rule 150: Karnaugh map.

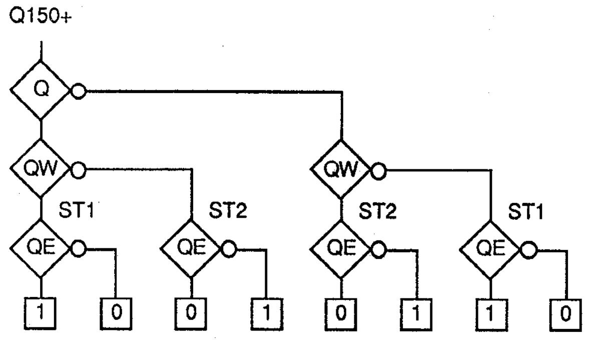

Fig. 5. Gene for rule 150: complete binary decision tree.

In this paper, we will only demonstrate how to establish the sub-program corresponding to the gene for rule 150. The use of Karnaugh map for simplifying trees [6] shows that no simplification of Q150+ is possible (Fig. 4: there is no *block*, i.e. no pattern formed by 2^m adjacent 0s or 1s). We define therefore a *complete* (or *canonical) binary decision tree* having 8 branches corresponding to the 8 possible input states of Q150+ (Fig. 5). However, we can identify four sub-maps in the form of outlined *blocks of blocks* labelled ST1 and ST2 (Fig. 4). Transforming the complete binary decision tree of Fig. 5 in *joining first the corresponding sub-trees* ST1 and ST2, then the output elements 0 and 1, we obtain a new representation: the *binary decision diagram* (BDD) (Fig. 6). For a microprogrammed realization, the binary decision diagram of Fig. 6 is the flowchart for the gene of rule 150. The software implementation of this flowchart (Fig. 7) requires *test instructions* and *assignment operators* whose *mnemonics* are:

```
if VAR else LABEL
do REG = DATA
```

The *non-conditional jump* is a particular case of the test instruction where the test variable is the logic constant 0. Its mnemonic is simply:

```
goto LABEL
```

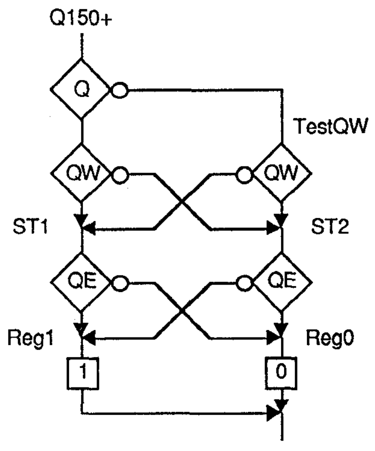

Fig. 6. Gene for rule 150: binary decision diagram.

```
Q150+:   if Q else TestQW
         if QW else ST2
ST1:     if QE else Reg0
Reg1:    do REG =1
         goto End
TestQW:  if QW else ST1
ST2:     if QE else Reg1
Reg0:    do REG =0
End:     ...
```

Fig. 7. Gene for rule 150: assembly language sub-program.

3.2 A new field-programmable gate array based on a binary decision machine

While our long-term objective is the conception of very large scale integrated circuits, we started by realizing a demonstration system in which each cell, called *MICROTREE* (for tree of microinstructions), is embedded into a plastic container called *BIODULE 601* (Fig. 8) [7], [8].

The MICROTREE cell consists essentially of a *binary decision machine* [6], executing the microprograms written using the following set of instructions:

```
if VAR else LABEL
goto LABEL
do REG = DATA
do X = DATA
do Y = DATA
```

The first three instructions were introduced above in the computing of the gene 90 and the two last are dedicated to the computing of the coordinates of the cell. In this implementation, the state register REG and both coordinate registers are 4 bits wide (REG3:0, X3:0, AND Y3:0). The size of the artificial organism embedded in an array of MICROTREE cells is limited in the first place by the coordinate space (X=0...15, Y=0...15, that is, a maximum of 256 cells in our current implementation), and then by the size of the memory of the binary decision machine storing the genome microprogram (1024 instructions).

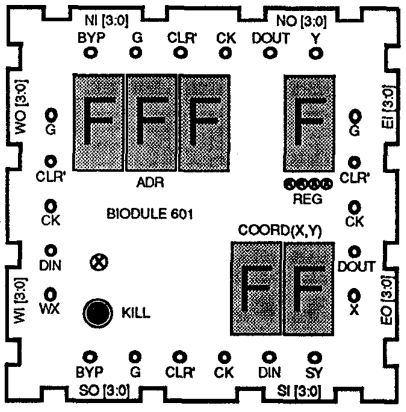

Fig. 8. BIODULE 601: demonstration module including a MICROTREE cell.

3.3 Self-repair

In the BIODULES 601 (Fig. 8), the existence of a fault is decided by the human user by pressing the KILL button of a cell. Therefore, fault detection and fault location, two features which will be indispensable in the final system, where they will be implemented using BIST (Built-In Self-Test) techniques [9] are not present in the BIODULES 601. In order to implement self-repair, we have chosen, favoring simplicity, to shift all the functions of the MICROTREE cell by one cell (or, in the general case, by one column) to the right. Obviously, this process requires as many spare cells (or columns), to the right of the array, as there are faulty cells to repair (three spare cells in the example of Fig. 9). It also implies some modifications to the MICROTREE cell, so as to add the capability of bypassing the faulty cell and shifting to the right all or part of the original cellular array.

3.4 Self-reproduction

The self-reproduction of an artificial organism, for example the random number generator of Fig. 1, rests on two hypotheses: (1) there exists a sufficient number of spare cells (unused cells at the right hand side of the array, at least five for our example) and (2) the calculation of the coordinates produces a cycle (X=1->2->3->4->5 ->1 in Fig. 10). As the same pattern of coordinates produces the same pattern of genes, self-reproduction can be easily accomplished if the microprogram of the genome, associated to the homogeneous network of cells, produces several occurrences of the basic pattern of coordinates (X=1->2->3->4->5 in Fig. 1). In our example, the

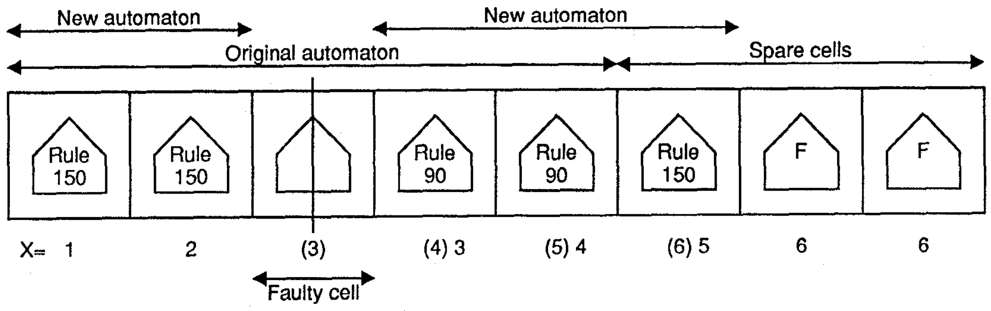

Fig. 9. Self-repair of the 5-cell random number generator in a 8-BIODULE array.

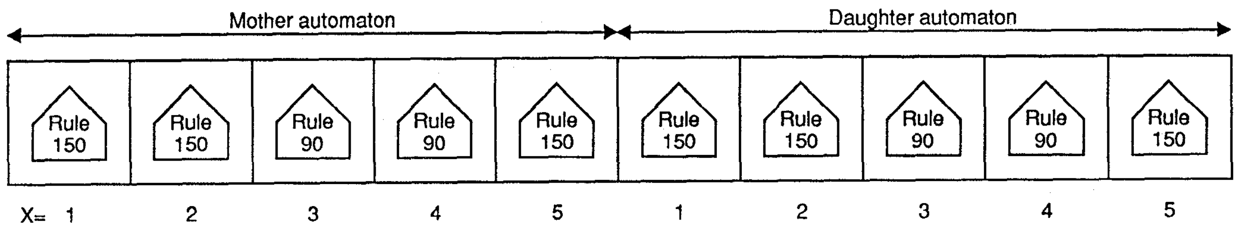

Fig. 10. Self-reproduction of the 5-cell random number generator in a 10-BIODULE array.

repetition of the horizontal coordinate pattern, i.e. the production of the pattern X=1->2->3->4->5->1->2->3->4->5 (Fig. 10), produces one copy, the *daughter automaton*, of the original or *mother automaton*. Given a sufficiently large space, the self-reproduction process can be repeated for any number of specimens, both in the X and the Y axes.

4. CONCLUSION

The main result of our research is the development of a new family of coarse-grained FPGAs called MICROTREE and based on a binary decision machine capable of executing a microprogram of up to 1024 instructions. The original features of this FPGA are essentially: (1) a completely homogenous organization of the cellular array and (2) a sequential execution of microprograms methodically derived from a chosen representation, the binary decision diagram.

Our FPGA satisfies the general hypothesis, as well as the three features of the Embryonics project: multicellular organization, cellular differentiation, and cellular division. The MICROTREE cell, itself realized with a commercial FPGA and a RAM, was finally embedded into a demonstration module called BIODULE 601, and we showed that an array of BIODULES 601 is capable of self-repair and self-reproduction.

The trivial applications of the MICROTREE family are those in which all the cells in the array contain the same gene: the genome and the gene then become indistinguishable and the calculation of the coordinates is superfluous. In this case, the cellular array is not limited in space. 1-dimensional (Wolfram's) and 2-dimensional (life, Langton's loop, etc.) uniform cellular automata are natural candidates for this kind of realization. The non-trivial applications are those in which the cells of an array have different genes: the genome is then a collection of genes, and the coordinates become necessary. The cellular array is then limited by the coordinate space (16x16=256 cells in the proposed realization). 1-dimensional (like the example of the random number generator) and 2-dimensional non-uniform cellular automata fall within this category. Let us also mention that the realization of uniform cellular automata with a predetermined initial state is an important special case which also requires separate genes and a coordinate system. The classic example of the cellular realization of a Turing machine, with a program stored on a tape, represents an application of this kind.

REFERENCES

[1] J. D. Watson, N. H. Hopkins, J. W. Roberts, J. Argetsinger Steitz and A. M. Weiner, *Molecular Biology of the Gene, Fourth Edition*. Menlo Park: The Benjamin/Cummings Publishing Company, 1987.

[2] R. Ransom, *Computers and Embryos*. Chichester: John Wiley, 1981.

[3] D.Mange, A.Stauffer, "Introduction to Embryonics: Towards New Self-repairing and Self-reproducing Hardware Based on Biological-like Properties", *Artificial Life and Virtual Reality*, John Wiley, 1994.

[4] S. Wolfram, *Theory and Applications of Cellular Automata*. Singapore: World Scientific, 1986.

[5] P. D. Hortensius, R. D. McLeod and B. W. Podaima, "Cellular automata circuits for built-in self-test", *IBM J. Res. Develop.*, vol. 34, no. 2/3, pp. 389-405, 1990.

[6] D. Mange, *Microprogrammed Systems: an Introduction to Firmware Theory*. London: Chapman & Hall, 1992.

[7] M. Goeke, "BIODULE 2: documentation technique", Tech. Rep., Logic Systems Laboratory, Swiss Federal Institute of Technology, Lausanne, 1995.

[8] D. Madon, "BIODULE 2: description et utilisation", Tech. Rep., Logic Systems Laboratory, Swiss Federal Institute of Technology, Lausanne, 1995.

[9] M. Abramovici and C. Stroud, "No-overhead BIST for FPGAs," in *Proc. 1st IEEE International On-Line Testing Workshop*, July 1995, pp. 90-92.