Toward Robust Integrated Circuits: The Embryonics Approach

Daniel Mange, MEMBER, IEEE, Moshe Sipper, SENIOR MEMBER, IEEE, André Stauffer, MEMBER, IEEE, AND Gianluca Tempesti, MEMBER, IEEE

Invited Paper

The growth and operation of all living beings are directed by the interpretation, in each of their cells, of a chemical program, the DNA string or genome. This process is the source of inspiration for the Embryonics (embryonic electronics) project, whose final objective is the design of highly robust integrated circuits, endowed with properties usually associated with the living world: self-repair (cicatrization) and self-replication. The Embryonics architecture is based on four hierarchical levels of organization. 1) The basic primitive of our system is the molecule, a multiplexer-based element of a novel programmable circuit. 2) A finite set of molecules makes up a cell, essentially a small processor with an associated memory. 3) A finite set of cells makes up an organism, an application-specific multiprocessor system. 4) The organism can itself replicate, giving rise to a population of identical organisms. We begin by describing in detail the implementation of an artificial cell characterized by a fixed architecture, showing that multicellular arrays can realize a variety of different organisms, all capable of self-replication and self-repair. In order to allow for a wide range of applications, we then introduce a flexible architecture, realized using a new type of fine-grained field-programmable gate array whose basic element, our molecule, is essentially a programmable multiplexer. We describe the implementation of such a molecule, with built-in self-test, and illustrate its use in realizing two applications: a modulo-4 reversible counter (a unicellular organism) and a timer (a complex multicellular organism). Last, we describe our ongoing research efforts to meet three challenges: a scientific challenge, that of implementing the original specifications formulated by John von Neumann for the conception of a self-replicating automaton; a technical challenge, that of realizing very robust integrated circuits capable of self-repair and self-replication; and a biological challenge, that of attempting to show that the microscopic architectures of artificial and natural organisms, i.e., their genomes, share common properties.

Keywords*—*Built-in self-test, embryonic electronics, field-programmable gate arrays (FPGA's), multiplexer-based FPGA's, self-repairing FPGA's, self-replicating FPGA's.

Manuscript received April 12, 1999; revised January 12, 2000. This work was supported in part by the Swiss National Foundation under Grant 21-54 113.98, by the Consorzio Ferrara Richerche, Università di Ferrara, Ferrara, Italy, and by the Leenaards Foundation, Lausanne, Switzerland.

The authors are with the Logic Systems Laboratory, Swiss Federal Institute of Technology, Lausanne CH-1015 Switzerland.

Publisher Item Identifier S 0018-9219(00)02876-0.

I. INTRODUCTION

A. Toward Embryonics

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 information: the 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).

Our *Embryonics* project (for *embryonic electronics*) is inspired by the basic processes of molecular biology and by the embryonic development of living beings [1], [43]. 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-replication* and *self-repair*, can also be applied to artificial objects (integrated circuits). We wish however to emphasize that the goal of bio-inspiration is *not* the modelization or the explication of actual biological phenomena.

B. Objectives and Strategy

Our final objective is the development of very large-scale integrated (VLSI) circuits capable of self-repair and self-replication. Self-repair allows partial reconstruction in case of a minor fault, while self-replication allows complete reconstruction of the original device in case of a major fault.

These two properties are particularly desirable for complex artificial systems requiring improved reliability in short-, medium-, or long-term applications.

1) Short-term applications [2]:
   - applications that require very high levels of reliability, such as avionics or medical electronics;
   - applications designed for hostile environments, such as space, where the increased radiation levels reduce the reliability of components;
   - applications that exploit the latest technological advances, and notably the drastic device shrinking, low power-supply levels, and increasing operating speeds, which accompany the technological evolution to deeper submicrometer levels and significantly reduce the noise margins and increase the soft-error rates [33].
2) Medium-term applications, where our aim is to develop very complex integrated circuits capable of on-line self-repair, dispensing with the systematic detection of faults at fabrication, impossible for systems consisting of millions of logic gates [34].
3) Long-term applications, executed on systems built with imperfect components: this is von Neumann's historical idea [8], the basis of all present projects aimed at the realization of complex integrated circuits at the atomic scale (nanotechnologies) [35]–[38].

Self-replication, or "cloning," can be justified independently of self-repair:

- to replicate, within a field-programmable gate array (FPGA), functionally equivalent systems [39];
- to produce the massive quantity of future integrated circuits, implemented using nanotechnologies [30];
- to finally accomplish von Neumann's unachieved dream, that is, the realization of a self-replicating automaton endowed with the properties of universal computation and construction [8].

These emerging needs require the development of a new design paradigm that supports efficient online VLSI testing and self-repair solutions. Inspired by the architecture of living beings, we will show how to implement online testing, self-repair, and self-replication using both hardware and software redundancy. The programmable degree of robustness of our systems, a function of an overhead itself programmable, is one of the major original features of the Embryonics project.

Section II provides a bird's eye view of Embryonics. This section is self-contained so as to give the reader a general overview of our project without delving into the technical details of the following sections. The final Embryonics architecture is based on four hierarchical levels of organization.

- The basic primitive of our system is the *molecule*, the element of a novel programmable circuit, built around a multiplexer.
- A finite set of molecules makes up a *cell*, essentially a small processor with the associated memory.
- A finite set of cells makes up an *organism*, an application-specific multiprocessor system.
- The organism can itself replicate, giving rise to a *population* of identical organisms, the highest level of our hierarchy.

Section III describes in detail the implementation of a prototype of an artificial cell characterized by a *fixed architecture*. We then show that multicellular arrays can realize a variety of different organisms (electronic watch, random number generator, Turing machine), all capable of self-replication and self-repair.

We will see that to meet the requirements of a wide range of applications, we need to develop an architecture characterized by a *flexible architecture*, that is, an architecture that is itself reconfigurable. This architecture will be based on a new type of fine-grained FPGA whose basic element, the *molecule*, is essentially a programmable multiplexer. Section IV describes the implementation of such a molecule, with built-in self-test and shows its use for two applications of very different complexity: a modulo-4 reversible counter and a timer.

The main goal of the Embryonics project is to explore the potential of a novel, robust architecture, rather than to compare such an architecture with existing solutions. After describing the trials and errors of the project, the conclusion of Section V introduces our ongoing research along three axes, each representing a different challenge: a scientific challenge, that of implementing in our architecture the original specifications formulated by von Neumann for the conception of a self-replicating automaton; a technical challenge, that of realizing real integrated circuits, capable of self-repair and of self-replication; and a biological challenge, that of seeking to show that the microscopic architecture of the artificial and natural organisms, that is, the structure of their genomes, share some common properties.

II. A Bird's Eye View of Embryonics

A. From Biology to Hardware

The majority of living beings, with the exception of unicellular organisms such as viruses and bacteria, share three fundamental features.

1) *Multicellular organization* divides the organism into a finite number of *cells*, each realizing a unique function (neuron, muscle, intestine, etc.). The same organism can contain multiple cells of the same kind.
2) *Cellular division* is the process whereby each cell (beginning with the first cell or *zygote*) generates one or two daughter cells. During this division, all of the genetic material of the mother cell, the *genome*, is copied into the daughter cell(s).
3) *Cellular differentiation* defines the role of each cell of the organism, that is, its particular function (neuron, muscle, intestine, etc.). This specialization of the cell is obtained through the expression of part of the genome, consisting of one or more *genes*, and depends essentially on the physical position of the cell in the organism.

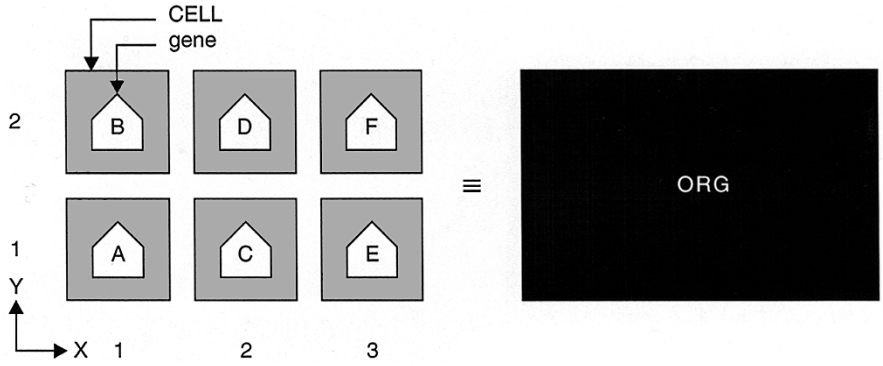


Fig. 1. Multicellular organization of a 6-cell organism ORG.

A consequence of these three features is that each cell is "universal," since it contains the whole of the organism's genetic material, the genome. Should a minor (wound) or major (loss of an organ) trauma occur, living organisms are thus potentially capable of self-repair (cicatrization) or self-replication (cloning or budding) [1].

The two properties of self-repair and self-replication based on a multicellular tissue are unique to the living world. The main goal of the Embryonics project is the implementation of the above three features of living organisms in an integrated circuit in silicon, in order to obtain the properties of self-repair and self-replication.

Our artificial organism will ultimately be divided into cells, themselves decomposed into molecules, a structure that determines the plan of this section: Section II-B describes the three fundamental features of the organism (multicellular organization, cellular differentiation, and cellular division), while in Section II-C we demonstrate that the organism, thanks to these three features, exhibits the two sought properties (self-replication and self-repair). Section II-D describes the cells and presents their essential features (multimolecular organization, molecular configuration, and molecular error detection), while Section II-E shows that the two sought properties (self-replication and self-repair) apply at the cellular level as well as at the organismic level. Section II-F underscores the four organizational levels of the hierarchy of the Embryonics project, which are, from the bottom up, the molecule, the cell, the organism, and the population of organisms.

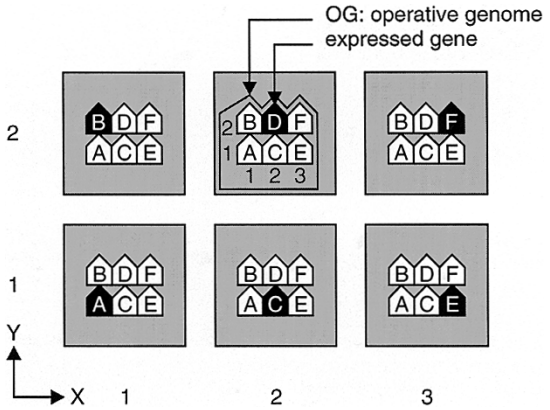
B. The Organism's Features: Multicellular Organization, Cellular Differentiation, and Cellular Division

The environment in which our quasi-biological development occurs is imposed by the structure of electronic circuit and consists of a finite (but arbitrarily large) two-dimensional surface of silicon. This surface is divided into rows and columns, whose intersections define the cells. Since such cells (small processors and their memory) have an identical physical structure (i.e., an identical set of logic operators and of connections), the cellular array is homogeneous. As the program in each cell (our artificial genome) is identical, only the state of a cell (i.e., the contents of its registers) can differentiate it from its neighbors.

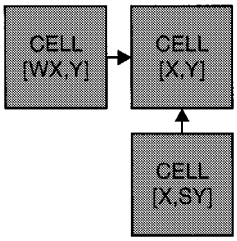
In this section, we first show how to implement in our artificial organisms the three fundamental features of multicellular organization, cellular differentiation, and cellular division, by using a generic and abstract six-cell example. In Sections III and IV, we will propose an actual implementation and various applications. *Multicellular organization* divides the artificial organism (ORG) into a finite number of cells (Fig. 1). Each cell (CELL) realizes a unique function, defined by a subprogram called the *gene* of the cell and selected as a function of the values of both the horizontal (X) and the vertical (Y) coordinates (in Fig. 1, the genes are labeled A to F for coordinates X,Y=1,1 to X,Y=3,2). Our final artificial genome will be divided into three main parts: the *operative genome* (OG), the *ribosomic genome* (RG), and the *polymerase genome* (PG). Let us call operative genome (OG) a program containing all the genes of an artificial organism, where each gene (A to F) is a subprogram characterized by a set of instructions and by the cell's position (coordinates X,Y=1,1 to X,Y=3,2). Fig.1 is then a graphical representation of organism ORG's operative genome.

Let then each cell contain the entire operative genome OG [Fig. 2(a)]: depending on its position in the array, i.e., its place within the organism, each cell can then interpret the operative genome and extract and execute the gene which defines its function. In summary, storing the whole operative genome in each cell makes the cell universal: given the proper coordinates, it can execute any one of the genes of the operative genome and thus implement *cellular differentiation*. In our artificial organism, any cell CELL[X,Y] continuously computes its coordinate X by incrementing the coordinate WX of its neighbor immediately to the west [Fig. 2(b)]. Likewise, it continuously computes its coordinate Y by incrementing the coordinate SY of its neighbor immediately to the south. Taking into consideration these computations, Fig. 3 shows the final operative genome OG of the organism ORG.

At startup, the first cell or *zygote* (Fig. 4), arbitrarily defined as having the coordinates X,Y=1,1, holds the one and only copy of the operative genome OG. After time t1, the genome of the zygote (*mother* cell) is copied into the neighboring (*daughter*) cells to the east (CELL[2,1]) and to the north (CELL[1,2]). This process of *cellular division* continues until the six cells of the organism ORG are completely programmed (in our example, the farthest cell is programmed after time t3).



(a)



(b)

Fig. 2. Cellular differentiation. (a) Global organization. (b) A cell `CELL[X,Y]` with its west neighbor `CELL[WX,Y]` and its south neighbor `CELL[X,SY]`; `X=WX+1`; `Y=SY+1`.

```
OG: operative genome
X = WX+1
Y = SY+1
case of X,Y:
  X,Y = 1,1: do gene A
  X,Y = 1,2: do gene B
  X,Y = 2,1: do gene C
  X,Y = 2,2: do gene D
  X,Y = 3,1: do gene E
  X,Y = 3,2: do gene F
```

Fig. 3. The operative genome `OG` of the organism `ORG`.

C. The Organism's Properties: Organismic Self-Replication and Organismic Self-Repair

The *self-replication* or *cloning of the organism*, i.e., the production of an exact copy of the original, rests on two assumptions.

- There exists a sufficient number of spare cells in the array (at least six in the example of Fig. 5) to contain the additional organism.
- The calculation of the coordinates produces a cycle (`X=1` $\rightarrow$ `2` $\rightarrow$ `3` $\rightarrow$ `1...` and `Y=1` $\rightarrow$ `2` $\rightarrow$ `1...` in Fig. 5, implying `X=(WX+1) modulo 3` and `Y=(SY+1) modulo 2`).

As the same pattern of coordinates produces the same pattern of genes, self-replication can be easily accomplished if the program of the operative genome `OG`, associated with the homogeneous array of cells, produces several

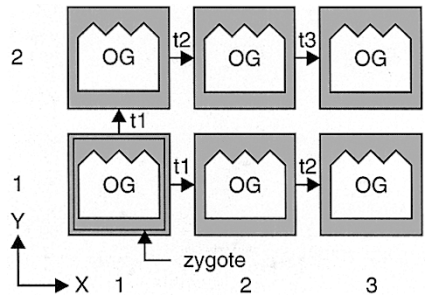


Fig. 4. Cellular division; `OG`: operative genome; `t1...t3`: three cellular divisions.

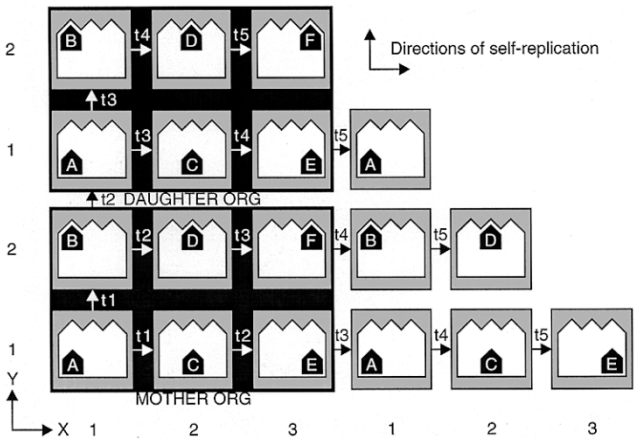


Fig. 5. Self-replication of a six-cell organism ORG in a limited homogeneous array of 6×4 cells (situation at time `t5` after 5 cellular divisions); `MOTHER ORG` = mother organism; `DAUGHTER ORG` = daughter organism.

occurrences of the basic pattern of coordinates. In our example (Fig. 5), the repetition of the vertical coordinate pattern (`Y=1` $\rightarrow$ `2` $\rightarrow$ `1` $\rightarrow$ `2`) in a sufficiently large array of cells produces one copy, the *daughter organism*, of the original *mother organism*. Given a sufficiently large space, the self-replication process can be repeated for any number of specimens in the `X` and/or the `Y` axes.

In order to implement the *self-repair of the organism*, we decided to use spare cells to the right of the original organism (Fig.6). The existence of a fault is detected by a `KILL` signal, which is calculated in each cell by a built-in self-test mechanism realized at the molecular level (see Section II-E). The state `KILL=1` identifies the faulty cell, and the entire column to which the faulty cell belongs is considered faulty and is deactivated (column `X=2` in Fig. 6). All the functions (`X` coordinate and gene) of the cells to the right of the column `X=1` are shifted by one column to the right. Obviously, this process requires as many spare columns to the right of the array as there are faulty cells or columns to repair (two spare columns, tolerating two successive faulty cells, in the example of Fig. 6). It also implies that the cell needs to be able to bypass the faulty column and to divert to the right all the required signals (such as the operative genome and the `X` coordinate, as well as the data buses).

Given a sufficient number of cells, it is obviously possible to combine self-repair in the `X` direction and self-replication in both the `X` and `Y` directions.

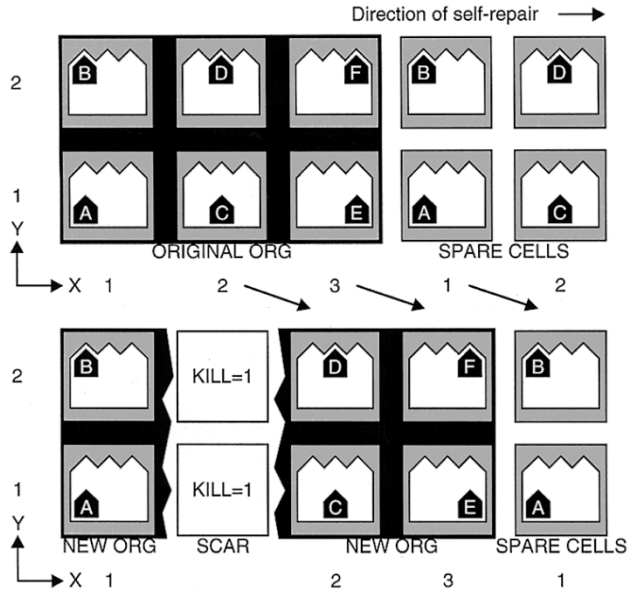


Fig. 6. Organismic self-repair.

D. The Cell's Features: Multimolecular Organization, Molecular Configuration, and Molecular Fault Detection

In each cell of every living being, the genome is translated sequentially by a chemical processor, the *ribosome*, to create the proteins needed for the organism's survival. The ribosome itself consists of molecules, whose description is an important part of the genome.

As mentioned, in the Embryonics project each cell is a small processor, sequentially executing the instructions of a first part of the artificial genome, the operative genome OG. The need to realize organisms of varying degrees of complexity has led us to design an artificial cell characterized by a flexible architecture that is itself configurable. It will therefore be implemented using a new kind of fine-grained FPGA.

Each element of this FPGA (consisting essentially of a multiplexer associated with a programmable connection network) is then equivalent to a *molecule*, and an appropriate number of these artificial molecules allows us to realize application-specific processors. We will call *multimolecular organization* the use of many molecules to realize one cell. The configuration string of the FPGA (that is, the information required to assign the logic function of each molecule) constitutes the second part of our artificial genome: the *ribosomic genome* RG. Fig.7(a) shows a generic and abstract example of an extremely simple cell (CELL) consisting of six molecules, each defined by a *molecular code* or MOLCODE (a to f). The set of these six MOLCODE's constitutes the ribosomic genome RG of the cell.

The information contained in the ribosomic genome RG thus defines the logic function of each molecule by assigning a molecular code MOLCODE to it. To obtain a functional cell, we require two additional pieces of information:

- the *physical position* of each molecule in the cellular space;
- the presence of one or more *spare columns*, composed of *spare molecules*, required for the self-repair described below (Section II-E).

The definition of these pieces of information is the *molecular configuration* [Fig.7(b)]. Their injection into the FPGA will allow:

1) the creation of a border surrounding the molecules of a given cell;
2) the insertion of one or more spare columns;
3) the definition of the connections between the molecules, required for the propagation of the ribosomic genome RG.

The information needed for the molecular configuration (essentially, the height and width of the cell in number of molecules and the position of the spare columns) makes up the third and last part of our artificial genome: the *polymerase genome* PG (a terminology that will be justified in Section II-E).

Last, it is imperative to be able to automatically detect the presence of faults at the molecular level and to relay this information to the cellular level. Moreover, if we consider that the death of a column of cells is quite expensive in terms of wasted resources, the ability to repair at least some of these faults at the molecular level (that is, without invoking the organismic self-repair mechanism) becomes highly desirable. The biological inspiration for this process derives from the DNA's double helix, the physical support of natural genomes, which provides complete redundancy of the genomic information though the presence of complementary bases in the opposing branches of the helix. By duplicating the material of each molecule (essentially the multiplexer MUX) and by continuously comparing the signals produced by each of the two copies [Fig. 7(c)], it is possible to detect a faulty molecule and to generate a signal FAULTY=1, realizing the *molecular fault* detection that will make possible cellular self-repair (described below in Section II-E).

E. The Cell's Properties: Cellular Self-Replication and Cellular Self-Repair

A consequence of the multimolecular organization and of the molecular configuration of the FPGA [Section II-D and Fig.7(b)] is the ability, for any given cell, to propagate its polymerase genome PG and its ribosomic genome RG in order to automatically configure two daughter cells, architecturally identical to the mother cell, to the east and to the north (Fig. 8), thus implementing *cellular self-replication*.

Cellular self-replication is a prerequisite for cellular division at the organismic level described above [Section II-B and Fig.4], during which the operative genome is copied from the mother cell into the daughter cells. In living systems, a specific molecule, the *polymerase enzyme*, allows cellular replication through the duplication of the genome. It is by analogy to this enzyme that the third part of our artificial genome is called polymerase genome.

The presence of spare columns, defined by the molecular configuration, and the automatic detection of faulty molecules (Section II-D, Fig.7(b) and (c) allow *cellular*

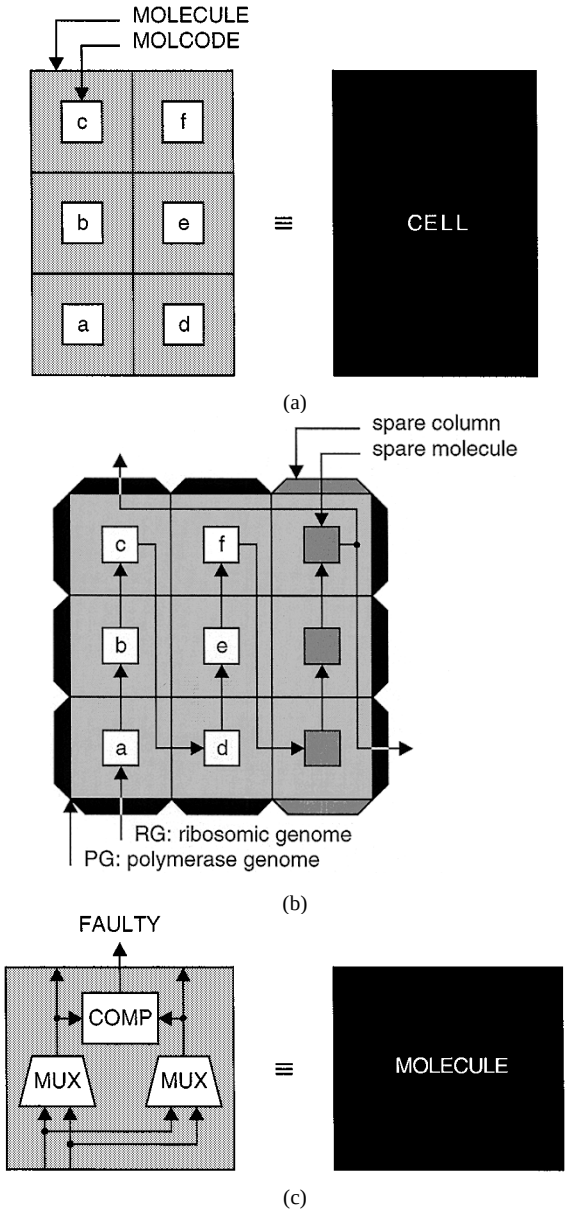


Fig. 7. The cell's features. (a) Multimolecular organization; RG: ribosomic genome: a, b, c, d, e, f. (b) Molecular configuration; PG: polymerase genome: height × width = 3 × 3; 001 = spare column. (c) Molecular fault detection; MUX: multiplexer; COMP: comparator.

self-repair: each faulty molecule is deactivated, isolated from the network, and replaced by a neighboring molecule, which will itself be replaced by a neighbor, and so on until a spare molecule (SM) is reached [Fig. 9(a)].

The number of faulty molecules handled by the molecular self-repair mechanism is necessarily limited: in the example of Fig.9(a), we tolerate at most one faulty molecule per row. If more than one molecule is faulty in one or more rows [Fig. 9(b)], molecular self-repair is impossible, in which case a global signal KILL=1 is generated to activate the organismic self-repair described above (Section II-C and Fig. 6).

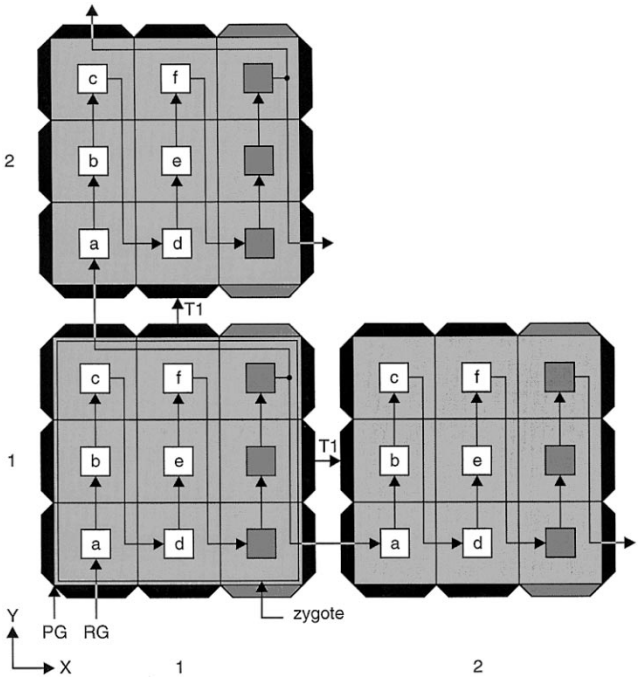


Fig. 8. Cellular self-replication; T1: one instance of cellular self-replication; RG: ribosomic genome; PG: polymerase genome.

F. The Embryonics Landscape

The final architecture of the Embryonics project is based on four hierarchical levels of organization which, described from the bottom up, are the following (Fig. 10).

- The basic primitive of our system is the *molecule*, the element of our new FPGA, consisting essentially of a multiplexer associated with a programmable connection network. The multiplexer is duplicated to allow the detection of faults. The logic function of each molecule is defined by its molecular code or MOLCODE.
- A finite set of molecules makes up a *cell*, essentially a processor with the associated memory. In a first programming step of the FPGA, the polymerase genome PG defines the topology of the cell, that is, its width, height, and the presence and positions of columns of spare molecules. In a second step, the ribosomic genome RG defines the logic function of each molecule by assigning its molecular code or MOLCODE.
- A finite set of cells makes up an *organism*, an application-specific multiprocessor system. In a third and last programming step, the operative genome OG is copied into the memory of each cell to define the particular application executed by the organism (electronic watch, random number generator, and Turing machine being examples shown by us to date).
- The organism can itself self-replicate, giving rise to a *population* of identical organisms, the highest level of our hierarchy.

III. A Cellular Implementation and Its Applications

In the Embryonics project, each cell of our artificial organism will be implemented by a small processor exe-

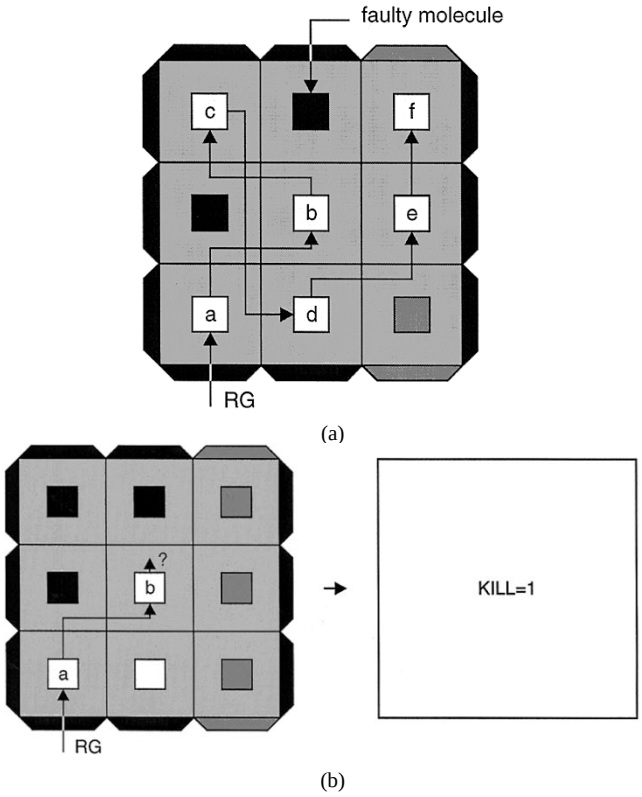


Fig. 9. Cellular self-repair. (a) Possible self-repair (at most one faulty molecule per row). (b) Impossible self-repair (more than one faulty molecule per row): KILL=1 (self-repair at the organismic level).

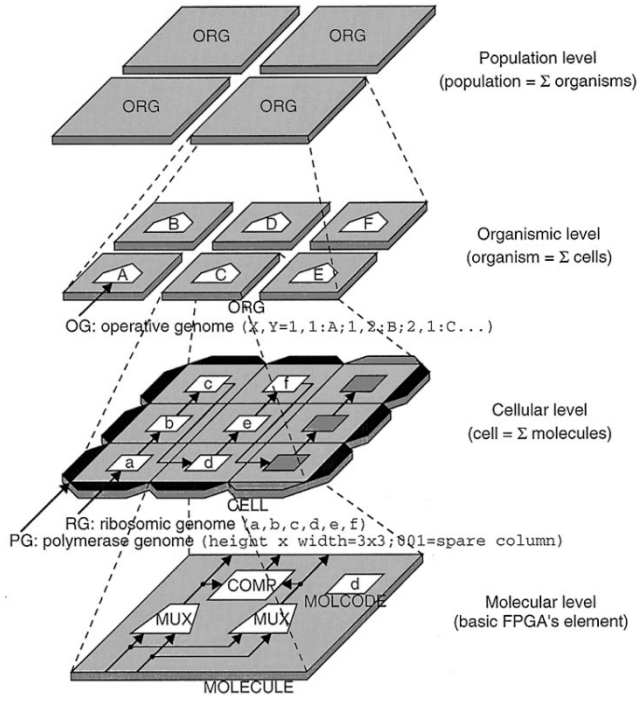


Fig. 10. The Embryonics landscape: a four-level hierarchy.

cuting sequentially the instructions of the operative genome OG. While our final objective is the conception of a new FPGA (the molecular FPGA described in Section II-D), which will allow us to adapt the architecture of our cells to a given problem, we decided to test our approach on a demonstration system in which each artificial cell is characterized by a *fixed architecture*. Section III-A describes in detail our artificial cell, while in Sections III-B–D, we show that multicellular arrays can realize a variety of very different organisms (electronic watch, random number generator, Turing machine), all capable of self-replication and self-repair. Last, Section III-E deals with the range of applications for our cell and its limitations.

A. A Cell Based on a Binary Decision Machine

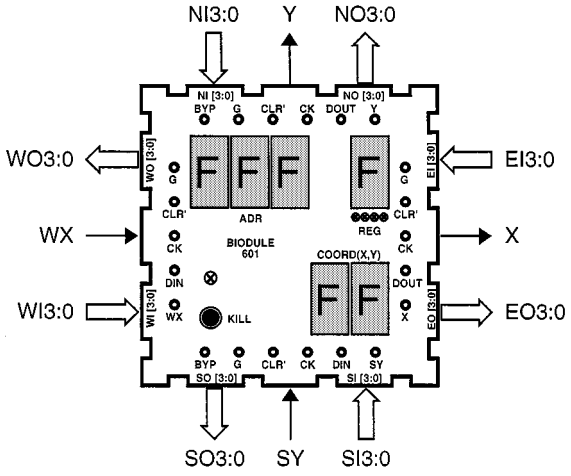
Our artificial cell, called *MICTREE* (for *microinstruction tree*) is basically a *binary decision machine* [3]–[5] implemented using standard electronic components and embedded into a plastic container [Fig. 11(a)]. These containers can easily be joined to obtain two-dimensional arrays as large as desired [Fig. 14(b)].

The MICTREE cell sequentially executes microprograms written using the following set of instructions [Fig. 11(b)]:

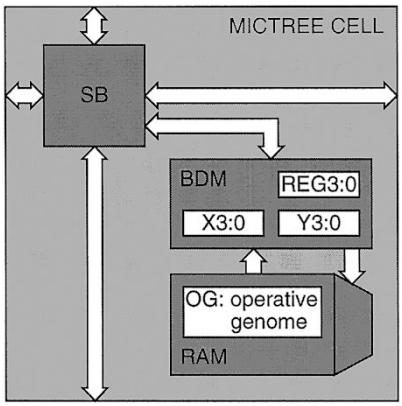
- **do** REG = DATA;
- **do** X = DATA;
- **do** Y = DATA;
- **do** VAROUT = VARIN;
- **if** VAR **else** LABEL;
- **goto** LABEL.

The state register REG and both coordinate registers X and Y are 4 bits wide (REG3:0, X3:0, and Y3:0). The variable VAROUT designates any one of the 4-bit-wide output buses in the four cardinal directions (VAROUT $\in$ {SO3:0,WO3:0,NO3:0,EO3:0}), while the variables VARIN designates any one of the 4-bit-wide input buses or the state register REG (VARIN $\in$ {SI3:0,WI3:0,NI3:0,EI3:0,REG3:0}). Thus, the instruction "**do** VAROUT = VARIN" allows the value of any of the input buses or of the state register REG to be sent to any of the cell's four cardinal neighbors. The test variables VAR include the set VARIN and the following additional variables: WX3:0 (the X coordinate sent by the cell's western neighbor), SY3:0 (the Y coordinates sent by the cells' southern neighbor), and G (a global variable, usually reserved for the synchronization clock).

The coordinates are transmitted from cell to cell serially, but are computed in parallel within the cell. Therefore, each cell performs a series-to-parallel conversion on the incoming coordinates WX and SY, and a parallel-to-series conversion of the coordinates X and Y it computes and propagates to the east and north. The genome microprogram is also coded serially: it enters the cells through the DIN pin(s) and is then propagated through the DOUT pin(s). The pins CK and CLR' are used, respectively, for the propagation of the clock signal and to reset the binary decision machines, while the signal BYP (bypass), connecting all the cells in a column, is used for self-repair. In a MICTREE cell, pressing the KILL button alerts a cell to the presence of a fault. The KILL button therefore replaces, in the demonstration system, the molecular fault



(a)



(b)

Fig. 11. The MICTREE artificial cell. (a) Front panel of a demonstration module implementing the cell. (b) Block diagram; SB: switch block; BDM: binary decision machine; RAM: random access memory.

detection mechanism mentioned in Section II. The effect of the KILL button is to deactivate the column containing the faulty cell: all the cells in the column "disappear" from the array, that is, become transparent with respect to all horizontal signals. Since the computation of the coordinates occurs locally depending on the neighbors' coordinates, such disappearance automatically engenders a recomputation of the coordinates of all the cells to right of the deactivated column, completing the reconfiguration of the array.

The size of the artificial organism embedded into an array of MICTREE 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, a limit imposed by the size of the coordinate registers) and then by the size of the memory of the binary decision machine storing the genome microprogram (1024 instructions). An editor, a compiler for the assembly language, and a loader simplify the task of writing and debugging the microprograms and generating the genome's binary code, charged serially through the DIN input of the mother cell.

B. The StopWatch and the BioWatch

For clarity's sake, we will begin with a simple example of a one-dimensional artificial organism: a *timer*, called *StopWatch*, designed to count seconds (from 00 to 59) and minutes (from 00 to 59) [44], [45]. This organism is implemented with four cells [Fig.12(a)] and is characterized by two distinct genes: "Countmod10", which counts modulo 10 the units of seconds or minutes, and "Countmod6," which counts modulo 6 tens of seconds or tens of minutes. The conception of these genes using the instructions of Section III-A is described in detail elsewhere [4].

Fig. 12(a) shows the operative genome OG of StopWatch (i.e., the set of all the genes with the corresponding X coordinate). By storing in each cell the entire operative genome OG, we implement cellular differentiation [Fig.12(b)]. In order to verify the property of self-replication of the organism [Fig. 13(a)], the computation of the X coordinate occurs modulo 4. The final program, the operative genome OG, is shown in Fig. 12(c).

The self-replication of StopWatch can be accomplished if:

1) there exists a sufficient number of spare cells [four cells to the right of the original organism in Fig. 13(a)];
2) the calculation of the X coordinate produces a cycle (X=1 → 2 → 3 → 4 → 1...).

If both these conditions are satisfied, the *mother organism* produces an exact copy of itself, the *daughter organism*.

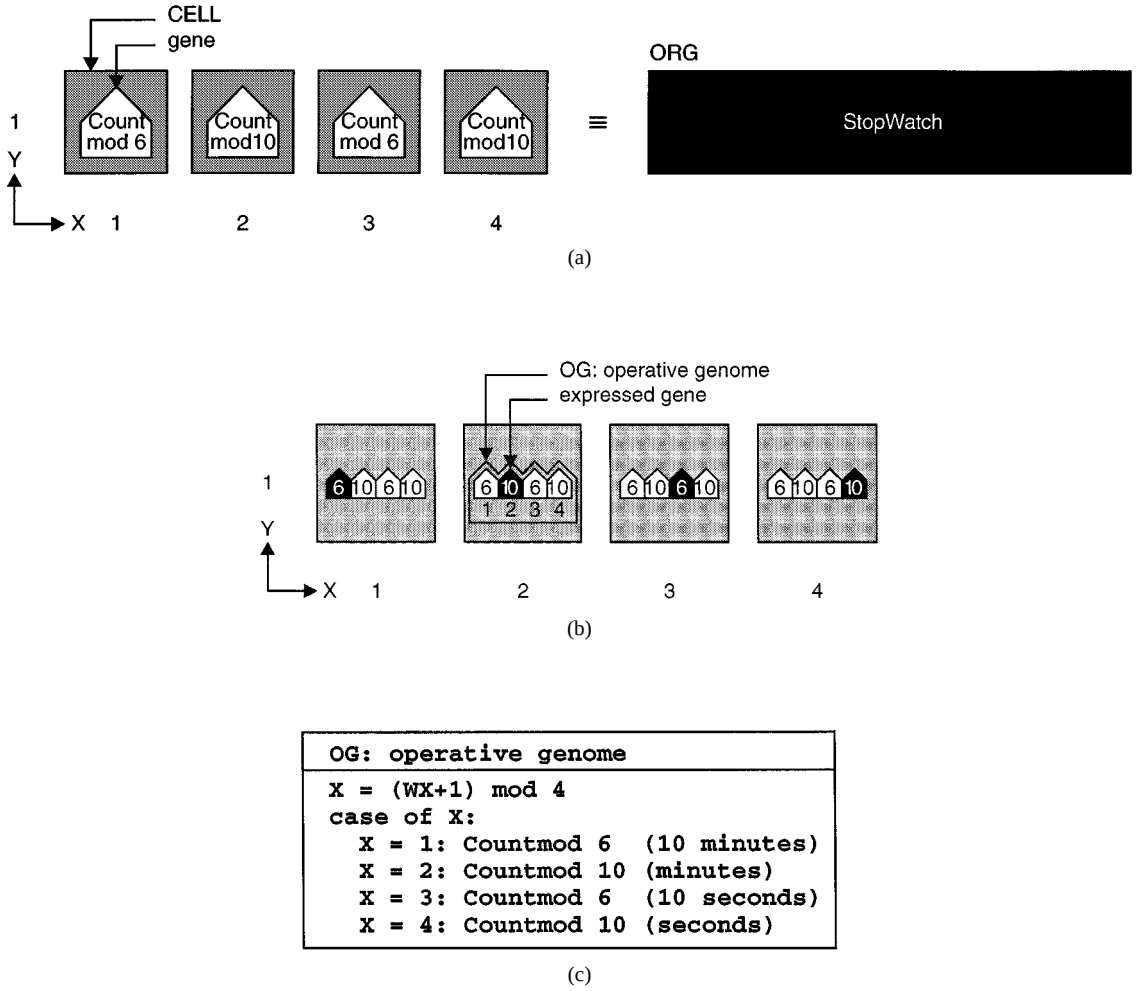
To demonstrate the self-repair of StopWatch, we will use the four cells to the right of the original organism [Fig. 13(b)] as spare cells. Once a faulty cell has been identified [state KILL=1 in cell X=3 for the original organism of Fig.13(b)], all the functions (X coordinate and gene) of the cells on the right of column X=2 are shifted by one column to the right. In the one-dimensional example of StopWatch, the presence of four spare cells allows the organism to tolerate four successive faulty cells.

By adding a modulo-24 counter for counting the hours (from 00 to 23) to our modulo-3600 counter (the StopWatch), we can easily realize a *full digital watch*, called *BioWatch*, (Fig. 14) [4]. The modulo-24 counter is the composition of two partial counters, one for the units of hours, the other for the tens of hours. The final genome of the full digital watch thus consists of four distinct genes, distributed among six cells identified by the horizontal coordinates X=1 to X=6.

With a greater number of MICTREE cells, it would be easy to introduce additional features to our electronic watch, that is, functions other than the counting of seconds, minutes, and hours. For example, computing the date, keeping track of the day of the week, or handling leap years. In any case, the genomic design of the BioWatch guarantees extreme flexibility through genome reprogramming, as well as considerable reliability, thanks to the self-repair and self-replication properties.

C. A Random Number Generator

Wolfram [6] exhaustively studied uniform *one-dimensional cellular automata* consisting of identical cells defined by a binary state and a neighborhood with a connectivity




```
OG: operative genome
X = (WX+1) mod 4
case of X:
  X = 1: Countmod 6  (10 minutes)
  X = 2: Countmod 10 (minutes)
  X = 3: Countmod 6  (10 seconds)
  X = 4: Countmod 10 (seconds)
```

(c)

Fig. 12. StopWatch, a digital timer. (a) Multicellular organization. (b) Cellular differentiation. (c) The operative genome OG.

radius equal to one (i.e., the future state of a given cell depends on the present state Q of the cell itself and on that of its immediate neighbors to the west QW and to the east QE). Such a cell is defined by a truth table of $2^3 = 8$ lines, and there exist $2^8 = 256$ such truth tables. Each of these functions is called the *rule* of the automaton and is identified by a decimal number between 0 and 255. Hortensius *et al.* [7] have shown that a well-chosen arrangement of Wolfram cells of type 90 and 150 produces a *nonuniform cellular automaton* which is, in fact, a *random number generator*. For a five-cell automaton, the final arrangement is shown in Fig. 15. In such an arrangement, the *periodic condition* (i.e., the values of the inputs of the leftmost and rightmost cells) is equal to zero. The global state 00 000 is a fixed point of the generator, while the remaining $2^5 - 1 = 31$ states form a cycle of maximum length.

The properties of self-replication and self-repair of the random number generator were demonstrated and have been described elsewhere [5].

D. A Specialized Turing Machine

With a more theoretical goal in mind, in order to compare the capabilities of the Embryonics project with those of von Neumann's self-replicating automaton [8], we wanted to show that an artificial multicellular organism can implement a specialized Turing machine and exhibit the properties of self-replication and self-repair [9].

The example we settled upon is that of a *parenthesis checker*, as described by Minsky [10]. The function of the machine is to decide whether a sequence of left (open) and right (closed) parentheses is well-formed (i.e., if for every open parenthesis in the sequence there exists a corresponding closed parenthesis). A specialized Turing machine for checking parentheses consists of a *tape*, decomposed into squares, and a *finite-state machine* with a *read/write head*.

For the very simple example of Fig. 16, which checks the sequence A⟨⟨⟩⟩A (where A is a symbol delimiting the sequence of parentheses), we can implement the Turing machine as a multicellular organism with the following structure.

- For Y=1, we use five cells to display the sequence A⟨⟨⟩⟩A. These five cells correspond to the tape of the Turing machine.
- For Y=2, we use five cells to implement the read head of the Turing machine. In fact, the head is realized by only one of the cells (X=2 in Fig. 16), the others being

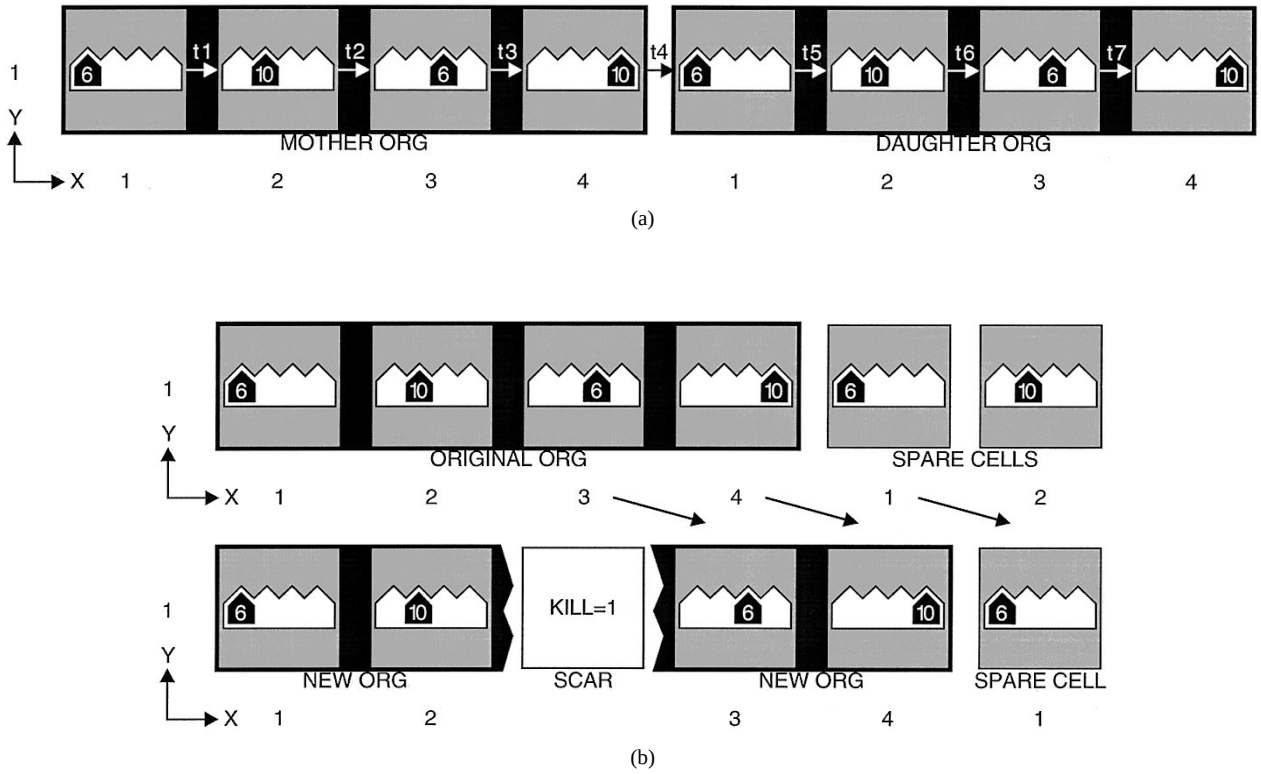


Fig. 13. The four-cell StopWatch organism. (a) Self-replication in an array of cells (`t1` . . . `t7`: seven cellular divisions). (b) Self-repair with two spare cells and one faulty cell.

inactive. The head is mobile: depending on its current state ($\rightarrow$, $\leftarrow$, or $\uparrow$), it can move, in the next clock cycle, to the right, to the left, or stay in place.

There exist, finally, only two distinct genes: the *Head* gene, which implements the algorithm for checking the parentheses and is executed by the cells of row `Y=2`, and the *Tape* gene, which describes the state of the tape and is executed by the cells of row `Y=1`. In addition, each cell is defined by an *initial condition*, that is, a specific value of the register `REG` at the beginning of the algorithm (`REG` $\in \{$`A`,$\langle$,$\rangle\}$ for `Y=1`, `REG` $\in \{$`0`, $\rightarrow\}$ for `Y=2` in Fig. 16).

The properties of self-replication and self-repair of this application are demonstrated elsewhere [4], [9].

E. Final Remarks

The main focus of this section was the description of an artificial cell, called MICTREE, based on a binary decision machine capable of executing a microprogram of up to 1024 instructions. Any organism realized using MICTREE cells satisfies the three features of the Embryonics project (Section II-B): multicellular organization, cellular differentiation, and cellular division. The MICTREE cell, itself realized with a commercial FPGA and a RAM, was finally embedded into a demonstration module, and we showed that an array of these modules exhibits the two desired properties of self-repair and self-replication.

The trivial applications of the MICTREE 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. One-dimensional (e.g., Wolfram's) [6] and two-dimensional (e.g., Conway's Life) [11] uniform cellular automata are natural candidates for this kind of realization. The nontrivial applications are those in which the cells in an array have different genes: the genome is then a collections of genes, and the coordinates become necessary. The cellular array is then limited by the coordinate space ($16 \times 16 = 256$ cells in the proposed realization). One-dimensional (like the examples of the StopWatch, BioWatch, and the random number generator) and two-dimensional (specialized Turing machine) cellular automata fall into this category. Let us also mention that the realization of uniform cellular automata with the automatic calculation of an initial condition (realized by setting the internal register `REG` to a predetermined value in each cell of the organism at the start) is an important special case which also requires separate genes and a coordinate system.

IV. A Molecular Implementation and Its Applications

In Section III, we introduced the implementation of an artificial cell, called MICTREE. Its architecture is *fixed*, and it is thus easy to find an application whose requirements exceed the capabilities of the MICTREE cell: a number of instructions greater then 1024, horizontal or vertical coordinates superior to 16, or a register of more than 4 bits are all demands which would require a redesign of the original cell. To meet the requirements of all possible applications, we want to develop an artificial cell endowed with a *flexible architecture*,