Artificial Cellular Division by Self-Inspection

Enrico Petraglio, Daniel Mange, André Stauffer, and Gianluca Tempesti

Swiss Federal Institute of Technology
Logic Systems Laboratory
CH-1015 Lausanne, Switzerland
enrico.petraglio@epfl.ch

Abstract. This article describes a novel approach to the implementation on an electronic substrate of a process analogous to the cellular division of biological organisms. Cellular division is one of the two processes that allow the multicellular organization of complex living beings, and is therefore a key mechanism for the implementation of bio-inspired features such as development (growth) and self-repair (cicatrization). In particular, we shall describe the architecture and operation of a new kind of programmable logic device capable of realizing, in silicon, a cellular division process.

1 Introduction

The majority of living beings, with the exception of unicellular organisms like viruses and bacteria, share a common *multicellular organization*: the organism is divided into a finite number of cells, each realizing a single function (skin, neuron, muscle, etc.). This architecture relies on two mechanisms:

- *Cellular division* is the process through which each cell achieves its duplication. During this phase, a cell copies its genetic material (the *genome*) and splits into two identical daughter cells.
- *Cellular differentiation* defines which function a cell has to realize. This specialization, which essentially depends of the cell position in the organism, is obtained through the expression of a part of the genome.

In a multicellular organism, each cell contains the whole of the organism's genetic material (the genome) and is therefore "universal", i.e. potentially capable of replacing any other cell. In presence of a physical degradation, each living organism is then potentially capable of self-repair (cicatrization).

Of these two mechanisms, the most difficult to realize in silicon-based systems is cellular division. In biological organisms, in fact, this feature implies, if not the creation, at least the formation of new physical entities (cells). Such manipulation of the material substrate is not possible in today's electronic circuits, which cannot be physically altered after fabrication. Luckily for bio-inspired research, programmable logic devices (FPGAs) [9,11,3] can be used to approximate this process by allowing the manipulation not of the physical substrate,

A.J. Ijspeert et al. (Eds.): BioADIT 2004, LNCS 3141, pp. 217–232, 2004.

but of the *logical structure* of a circuit. This article describes our approach to the implementation of a process analogous to biological cellular division in such a programmable circuit.

Our approach is based on the Embryonics project (Section 2) for all that concerns multi-cellular organization and cellular differentiation, and on the Cell Matrix system (Section 3) for self-inspection based replication. Section 4 will then introduce the system we have developed. An example of our system will presented in section 5.

2 The Embryonics Project

The main goal of the Embryonics (embryonic electronics) project [8,10] is to implement, in an integrated circuit, a system inspired by the development of multi-cellular organisms and capable of self-test and self-repair.

In Embryonics (Figure 1), an artificial organism (ORG)[1] is realized by a set of *cells*, distributed at the nodes of a regular two-dimensional grid. Each cell contains a small processor coupled with a memory used to store the program (identical for all the cells) that represents the organism's genome. In the organism, each cell realizes a unique function, defined by a sub-program called the *gene*, which is a part of the genome. The gene to be executed in a cell is selected depending on the cell's position within the organism, defined by a set of X and Y coordinates. In Figure 1, the genes are labeled A to F for coordinates $(X, Y) = (1, 1)$ to $(X, Y) = (3, 2)$.

The first kind of self-replication in Embryonics systems is that of the organism: an artificial organism is capable of replicating itself when there is enough free space in the silicon circuit (at least six cells in the example of Figure 1) to contain the new daughter organism and if the calculation of the coordinates produces a cycle. In fact, as each cell is configured with the same information (the genome), the repetition of the vertical coordinate pattern ($Y = 1 \rightarrow 2 \rightarrow 1 \rightarrow 2$) causes the repetition of the same pattern of genes and therefore, in a sufficiently large array, the self-replication of the organism for any number of specimens in the X and/or the Y axes.

In Embryonics, however, there is a second replication process, which corresponds to the cellular division process in biological entities, used to put in place the initial array of cells that will then be differentiated to obtain the organism.

The need to build cells of different size and structure depending on the application naturally led to the use of programmable logic (FPGAs) as a physical substrate in Embryonics. Each of the computational elements of the FPGA can then be seen as *molecules*, assembled in a precise configuration to form a cell. As all cells are identical, the development process is analogous to the replication of this configuration for as many times as there are cells in the organism.

To implement this replication, Embryonics splits the process into two phases:

- the *structural* phase, where a "skeleton" is created in order to divide the physical space in a collection of groups of molecules which are empty cells;

[1] Group of artificial cells that executes a given task

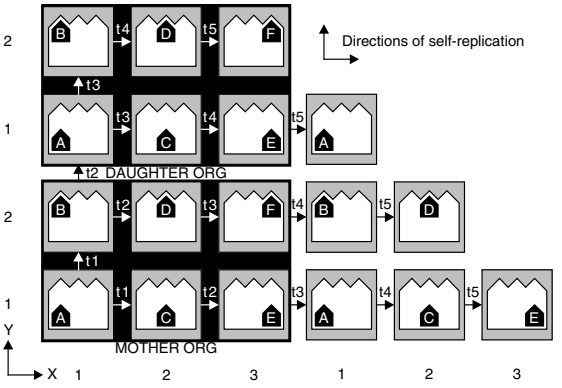

Fig. 1. Self-replication of a 6-cell organism in a limited homogeneous array of 6x4 cells. Only the expressed gene is shown in each cell.

– the *configuration* phase, where the configuration is sent in parallel into all the empty cells created during the structural phase.

The structural phase is implemented by a small cellular automaton (CA) [1], integrated in the molecular grid, capable of transforming a one-dimensional string of states (analogous to a configuration bitstream stored in a memory chip) into a two-dimensional structure (the blocks that will host the cells).

Figure 2 shows that the CA elements are placed in the spaces between the molecules of the FPGA. With an appropriate sequence of states, the automaton will be able to partition the array into identical blocks of variable size. The "skeleton" created by the automaton during the structural phase can be seen as the *membrane* of the artificial cells. Once the membrane is in place, the second part of the cellular self-replication begins: a bitstream containing the genome is sent to all the blocks in parallel (Figure 2), automatically creating multiple copies of the same artificial cell. At the end of this phase, the coordinate-based differentiation mechanism defines the structure of the organism.

This replication process is quite different from biological cellular division, as all cells are created in parallel. Our new approach, designed to be integrated with Embryonics, implements a cellular division much closer to reality.

3 Cell Matrix

Developed by N. Macias [4], Cell Matrix is a fine-grained reconfigurable architecture, composed of a two-dimensional grid of identical elements (referred to as cells[2]). Each cell in the grid is interconnected with its four cardinal neighbors and contains a lookup table (LUT) used as a truth table to define its ouputs.

[2] A cell in the Cell Matix architecture can be compared to a molecule in the Embryonics tissue

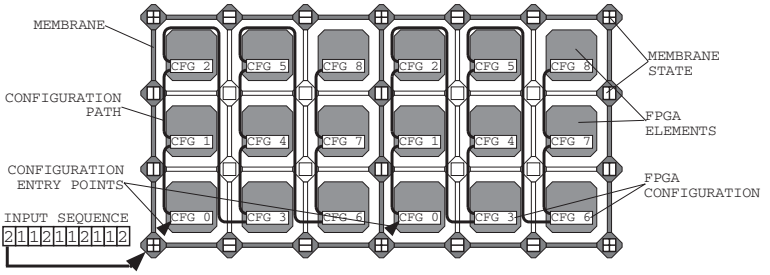

Fig. 2. First, an input sequence is sent to the CA (represented by the lozenge network) in order to set up the membrane of each cell. Then, the genome is sent in parallel to each cell composing the ORG.

Cells of Cell Matrix circuits are capable of self-replication using a self-inspection mechanism: each cell is at the same time configurable and able to configure other cells without any external command. This feature is called *self-duality*, and requires two different modes of operation for the cell: *D-mode* (Data mode) and *C-mode* (Control mode). In D-mode, the cell's LUT processes the four input signals in order to generate output signals, whereas in C-mode, the input data is used to fill up (configure) or re-write (re-configure) the LUT of the cell.

The self-duality feature is the key to the self-replication mechanism of a Cell Matrix cell. Using this mechanism, a cell is able to produce an identical copy of itself, inspecting its own truth table and copying it to another cell in the circuit.

Figure 3 shows a 2x2 Cell Matrix grid, in which each cell has two inputs and two outputs per side. Each cell contains an internal 16-row by 8-column truth table, in which four 4-variable universal functions can be realized. This internal memory governs the combinational behavior of the cell.

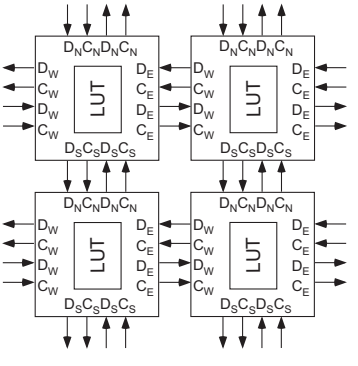

Fig. 3. 2x2 Cell Matrix grid. Each cell, which has two inputs and two outputs per edge, is connected with its four direct neighbors.

A cell uses its D lines to exchange information with its neighbors; the nature of this information is defined by the mode in which the cell is operating. In D-mode, a cell reads its D input values and uses them to select one of the 16 lines of its LUT, generating the eight outputs of the cell. In C-mode, the information sent to the cell trough the D lines is serially shifted into its LUT, while the previous contents of the LUT are shifted out to the D outputs. Using this mechanism, a cell is able to configure its neighbors. The C inputs are used to define the operating mode of the cell. If any of the cell's C inputs is 1, the cell is in C-mode, otherwise it is in D-mode.

A cell, using its LUT, can change its C output values and therefore can control the mode of any neighboring cell. By changing the mode of its neighbor cells, a single cell can reach and control any cell placed in the circuit. Figure 4 shows a typical programming sequence. First, cell X configures the LUT of the Y cell in order to transform it in a simple wire, which has to bypass C and D data from X to Z. After this step the D and C outputs of X are directly connected to the D and C inputs of Z, hence cell X is able to configure the LUT of Z by sending directly its information through the D line.

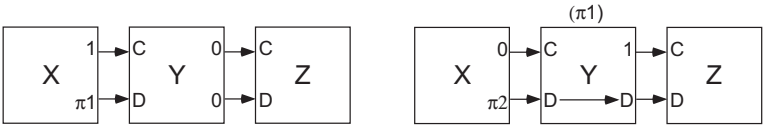

Fig. 4. Cell X configuring non-adjacent cell Z. On the left, cell X, by using truth table $\pi 1$, configures cell Y (its direct neighbor) as a wire. On the right, cell Y is configured with $\pi 1$ truth table and cell X can directly configure cell Z using $\pi 2$ truth table.

Our approach will combine the self-replication capabilities of a Cell Matrix architecture with the Embryonics approach in order to create a new molecule which will better approximate the biological cellular division process in Embryonics systems.

4 Self-Inspection on Embryonics Tissues

In our approach, we will redefine the molecules of the Embryonics project in a form similar to the cells of the Cell Matrix circuit (a LUT and two types of input/output connections) in order to realize a cellular division mechanism based on self-inspection [2] in Embryonics systems. Moreover, we will present the design of a finite state machine (FSM), which is placed in each molecule: during the configuration phase, this FSM will control the hardware configuration and the communication protocol of the molecule.

By definition, in self-inspection based self-replication, a replicating system has to generate its description by examining its own structure. Such a description is then used to recreate an identical copy of the original system [7]. In

our case, the system which performs the self-replication is realized on an integrated circuit which consists of a surface of silicon divided in identical elements (i.e., molecules) distributed at the nodes of a regular two-dimensional grid. Each molecule contains the same hardware components and only the contents of its registers can differentiate it from its neighbors. Therefore, a self-replicating cell creates its description by reading the contents of its register. We will say that our system performs a simplified kind of self-inspection because it achieves self-replication by copying the contents of its registers and not by generating the description of the new copy.

4.1 Molecular Self-Replication

The first step of the design is the realization of a molecule capable of replicating its content in one of its four neighbors. It has to be noted that this phase of self-replication draws inspiration from the Cell Matrix mechanism presented in figure 4. However, the design of our molecule is totally original as the molecule has to fit the requirements of Embryonics.

Let us consider that the information that a molecule has to send to its neighbor in order to replicate itself is fully contained in its LUT. Since the molecule, as in the Cell Matrix cell, has one D output line per edge, it has to read and send the content of its LUT serially. To implement this behavior, the molecule can use its LUT as a shift-register in which input and output are connected together. The molecule can then create a rotating bitstream and read (self-inspect) and send out (self-replicate) its information through the D output line, as shown in Figure 5.

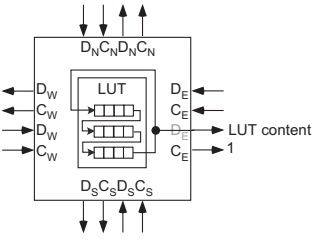

Fig. 5. A shift register implementing the rotating memory.

The molecule as in Cell Matrix, also has to set its C output line to 1 in order to change the operation mode of its target neighbor. The latter is now able to receive the information arriving form the source molecule. The end of the replication process is signaled by a one-bit register (called *overflow*) that is set when the LUT has been completely filled (the mechanism to detect the end of the replication process in Cell Matrix is not described in the publications). By testing the register, the target molecule knows that the replication process is finished and sends an an acknowledgment signal through its C output line to the

source molecule (Figure 6). Once the configuration is over, the source molecule returns in a quiescent state while the target molecule becomes a source molecule and starts to self-replicate.

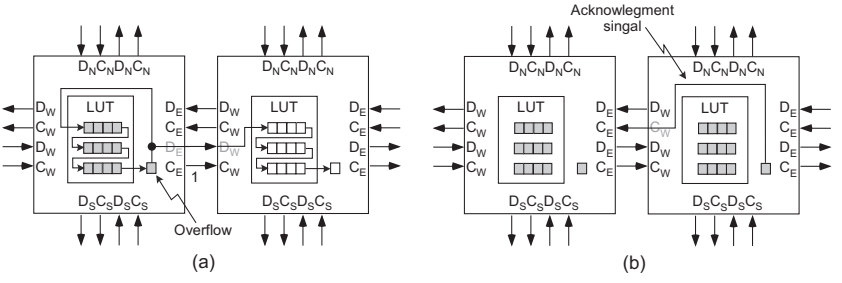

Fig. 6. (a) First, the two molecules are in C-mode. The source molecule sends its memory content, through the D line, into the target cell. (b) When the target molecule is configured (i.e. the overflow bit is set), it sends back an acknowledgment signal, which will stop the configuration process.

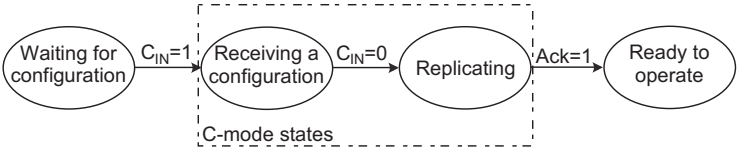

Fig. 7. Sequence of states describing the behavior of a molecule during the molecular self-replication.

As a Cell Matrix cell, our molecule can operate in two different modes (C or D). The example of Figure 6, however, defines new switching rules between these two modes. This behavior can be represented by the state graph shown in Figure 7.

In the *Waiting for configuration* state, the molecule is not configured and quiescent. When a C input becomes 1, the molecule passes into the *Receiving configuration* state, switches to C-mode, and starts filling its LUT with the incoming configuration. When the configuration process is over, the molecule generates an acknowledgment signal and waits for its C input value to return to 0. At this point, the molecule changes its internal state to *Replicating* but is still operating in C-mode, ready to configure one of its neighbors. Once the molecule has sent out its LUT content and received the acknowledgment signal, it returns into D-mode and reaches the last state (*Ready to operate*) on the graph, where it will remain during the normal operation of the circuit.

4.2 Cellular Self-Replication

The next step of the design is the insertion of new hardware that will allow the molecules to assemble in an artificial cell. It has to be noted that, in recent publications [5,6], the Cell Matrix system has been extended to include the possibility of grouping cells in order to compose a scructure called *super-cell*[3]. However, we choose to develop our own system capable of creating artificial cells. Such a system will be expressly conceived to work with our new molecules and will perfectly be compatible with the requirements of the Embryonics project (self-replication and local fault-tolerance). As a consequence, the mechanisms introduced in this section for the creation of cells are very different from those exploited by Cell Matrix's super-cells.

An artificial membrane, as in Embryonics cells, will be used to define which molecules belong to which cell. This membrane will be realized with a 4-bit register (*MembReg*) which will store the relative position of the molecule in the cell (Figure 8), thus defining its state.

In a dividing cell, each molecule can be in one of the following states:

- *ready to operate*: the molecule has already executed its replication;
- *replicating*: the molecule is configuring one of its neighbors;
- *ready to replicate*: the molecule is waiting to start its self-replication.

Therefore, the FSM introduced above must be modified in order to handle the new molecular states.

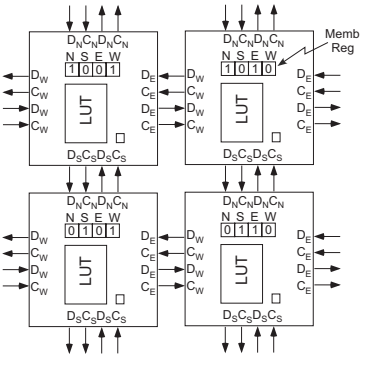

Fig. 8. A cell composed of 2x2 molecules. Each *MembReg* is filled in order to define the position of the molecule in the cell. N.B: If the molecule is placed in the center of the cell the MembReg contents is equal to zero.

As an example, Figure 8 shows a set of 2x2 molecules, which will represent the source cell that will use self-inspection to self-replicate its contents (i.e. the

[3] A super-cell in the Cell Matix architecture can be compared to a cell in the Embryonics tissue

artificial genome) in order to create a daughter cell on its right. First, each molecule in the cell will use the membrane information to know its position. In this example, a molecule can start to replicate (*replicating* state) if it belongs to the left part of the membrane; otherwise it has to wait (*ready to replicate* state).

The molecules in the leftmost column send out their LUT contents on the east D output lines. The information travels through the columns of molecules in the *ready to replicate* state and reaches the empty target molecules (Figure 9). The source molecules configure the targets until they receive the acknowledgment signals (Figure 10). At this point, the target molecules are totally configured and in the *ready to replicate* state, the first column stops sending data, switches its state from *replicating* to *ready to operate* and changes the values of its C lines from 1 to 0. This change is picked up by the second column of molecules, which switches its state from *ready to replicate* to *replicating* and starts to replicate. The first cellular division is over when the second column of molecules ends its replication.

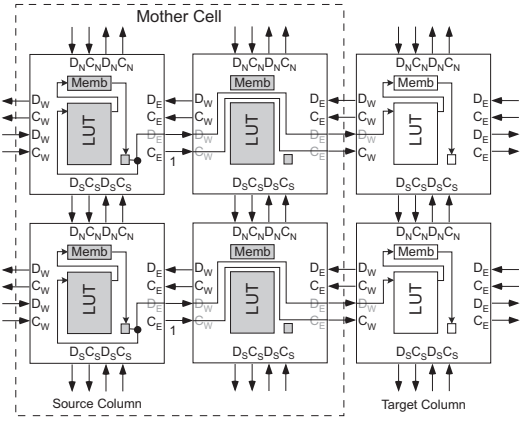

Fig. 9. The molecules composing the source column are in the *replicating* state, while the other molecules in the mother cell are in the *ready to replicate* state. The grey boxes represent the configured elements, while the blank boxes represent empty elements.

The state graph for this new cellular behavior is shown in Figure 11.

It should be noted that, according to the new state graph, a cell is able to self-replicate in one direction only. That is, the self-replication mechanism presented above is not able to fill up an FPGA circuit, which is a two-dimensional structure. Therefore, the molecule and the states graph have to be further modified in order to perform a two-dimensional cellular division. A possible solution is to use a second acknowledgment signal, *Ack Cell*, which will inform all the molecules in the cell that the cellular division is over and instruct the cell has to restart its replication in a different direction.

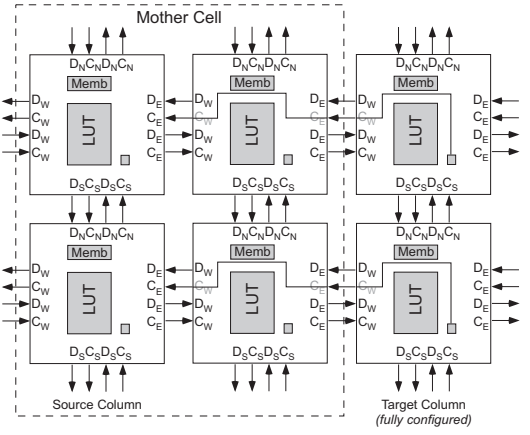

Fig. 10. The molecule composing the target column are totally configured. Each molecule, using the overflow register, generates an acknowledgment signal in order to stop the molecular self-replication.

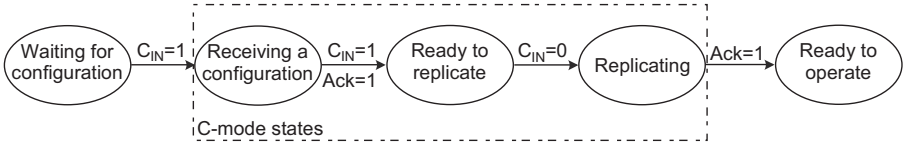

Fig. 11. Sequence of states describing the behavior of a molecule during the cellular self-replication.

For example, if a cell is created in the bottom left corner of an FPGA, this cell has to replicate itself to the north and then to the east. In order to fill all the available space in the circuit, each new cell has to replicate exactly as its mother. It should be noted that, with these growth rules, two source cells could be configuring the same set of molecules at the same time. In order to avoid such data collisions, only the cells on the east side of the FPGA are able to replicate in the two dimensions. Figure 12 shows the two-dimensional growth behavior.

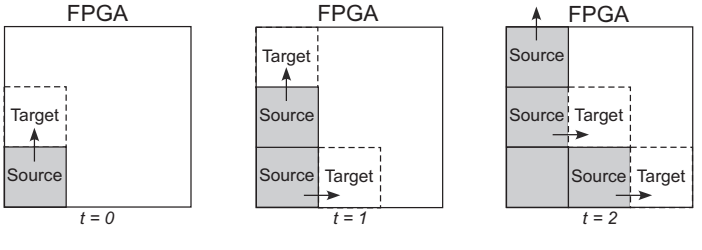

Fig. 12. Cellular division, two-dimensional growing behavior.

5 An Example: The LSL Organism

This section will present an extremely simplified example, the display of the acronym "LSL", for Logic Systems Laboratory. This acronym is considered as a one-dimensional artificial organism composed of three cells (Figure 13a). Each cell is located by a X coordinate, ranging from 1 to 3 in decimal or from 01 to 11 in binary. For coordinate values $X = 1$ and $X = 3$, the cell should implement the L character, while for $X = 2$, it should implement the S character. A totipotent cell (Figure 13b) comprises then $6 \times 7 = 42$ molecules, 36 of which being invariant, five displaying the S character, and one displaying the L character. A modulo-3 counter is embedded in the final organism and implements the truth table shown in table 1.

Table 1. The modulo-3 counter.

Character	X	X1	X0	X+	X1+	X0+
L	1	0	1	2	1	0
S	2	1	0	3	1	1
L	3	1	1	1	0	1

The modulo-3 counter is represented by the logic diagram and symbol of figure 15. According to the table, the value of the binary variable $X0$ is sufficient to distinguish the display of character L ($X0 = 1$) from the display of character S ($X0 = 0$ or $\overline{X0} = 1$). It is important to note that the modulo-3 counter is designed to produce a cycle in the calculation of the coordinates, in our example: $X = 1 \rightarrow 2 \rightarrow 3 \rightarrow 1...$ and $Y = 1 \rightarrow 1 \rightarrow 1....$ This coordinate calculation will differentiate the cells in order to produce as artificial organisms composed of three cells as it is possible (Figure 14).

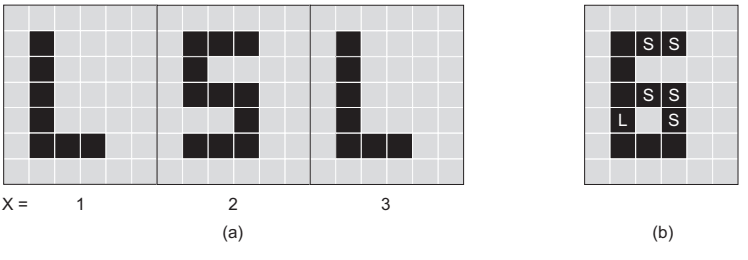

Fig. 13. (a) The three cells of the artificial organism. (b) The totipotent cell.

These specifications are sufficient to design the final architecture of the totipotent cell, which is shown in figure 16. The final cell is composed of seven different types of molecules, each of which realizes one of the following tasks:

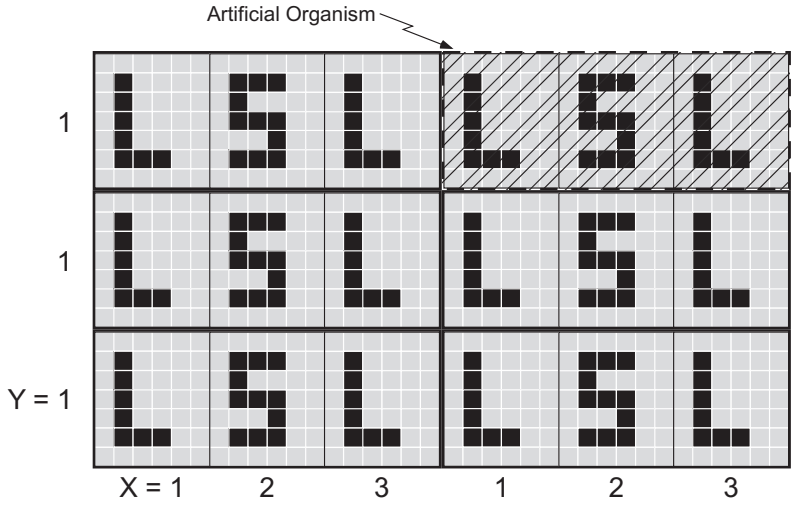

Fig. 14. A population of artificial organisms obtained by the repetition of the cellular coordinate $X = 1 \rightarrow 2 \rightarrow 3 \rightarrow 1...$ and $Y = 1 \rightarrow 1 \rightarrow 1....$

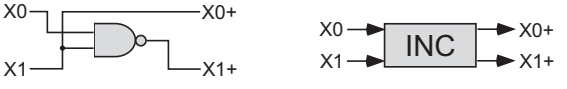

Fig. 15. Logic diagram ans symbol of the incrementer.

- **0**: Function-less molecule.
- **1**: Two horizontal busses, which carry $X0$ and $X1$ signals.
- **2**: Modulo 3 intcrementation.
- **3**: One vertical bus, which carries the $X0$ signal.
- **4**: Permanent switch-on display.
- **5**: Display of S character only ($X0 = 0$ or $\overline{X0} = 1$).
- **6**: Display of L character only ($X0 = 1$).

Figure 16 shows that each molecule composing the totipotent cell has to memorize two characters. The first character, which is in brackets, represents the hexadecimal value memorized in the MembReg and used to construct the artificial membrane (Table 2), while the second character, which is in bold font, controls the molecular behavior and can be coded using three bits. Therefore, in order to implement this example, the logic core of our new molecule is composed by a 4×1 LUT, which is used to memorize the second character controlling the task that the molecule has to realize.

The artificial genome, which has to be injected in the FPGA in order to create the first totipotent cell is represented in figure 17. It has to be noted that each row represents the configuration stream of a row of molecules, and can be splitted in several groups of 8-bits each group representing the configuration of

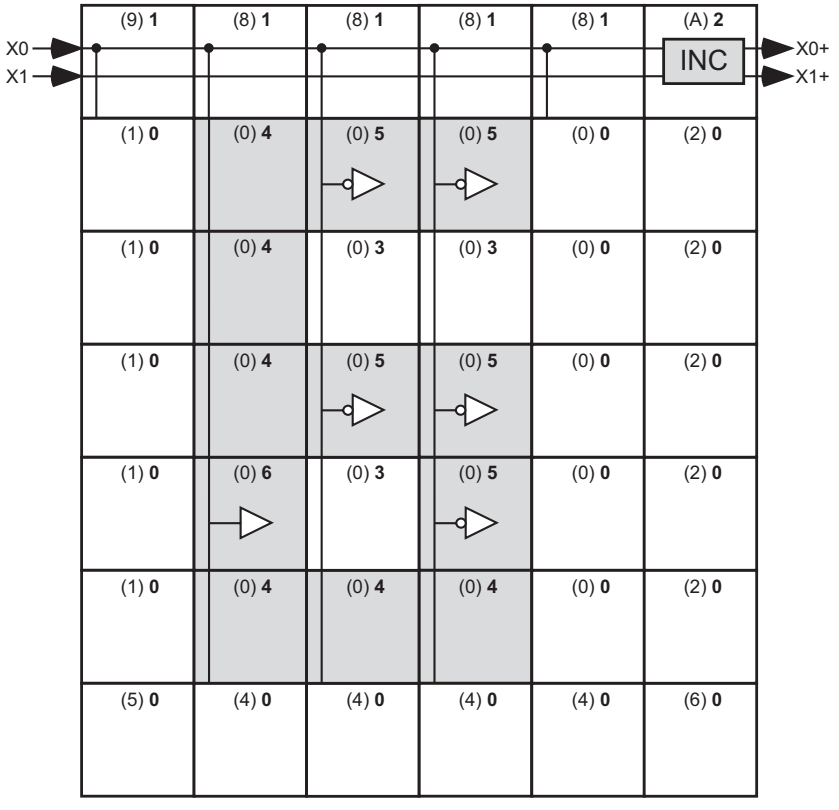

Fig. 16. The final totipotent cell made up of $6 \times 7 = 42$ molecules

a single molecule (i.e. membrane information + molecular task). Analyzing the membrane information of the first row we can see that this flow of information will configure the south row of the cell.

Last, it was possible to embed the basic molecule in each of the 2000 field-programmable gate arrays of the BioWall [12] and to show the growth of the the multicellular artificial organism, followed by its selfreplication in both vertical and horizontal dimensions (Figure 18). We therefore, obtain a population of identical organisms (i.e. clones) thus creating a fourth level in our hierarchy, a population of organisms.

In figure 18 it is interesting to note that two cells are displaying a corrupted line (or column). This is the characteristic behavior of a cell during its replication phase. In fact, a replicating cell, inspects its content in order to sent it to its daughter. Therefore, during this phase, the cell is not able to correctly perform its task (in our example: display a L or a S).

Table 2. Possible contents of the MembReg register

HEX value	N	S	E	W	position in the cell
0	0	0	0	0	molecule placed in the center of the cell
1	0	0	0	1	west molecule
2	0	0	1	0	east molecule
4	0	1	0	0	south molecule
5	0	1	0	1	southwest molecule
6	0	1	1	0	southeast molecule
8	1	0	0	0	north molecule
9	1	0	0	1	northwest molecule
A	1	0	1	0	northeast molecule
others	-	-	-	-	invalid combination

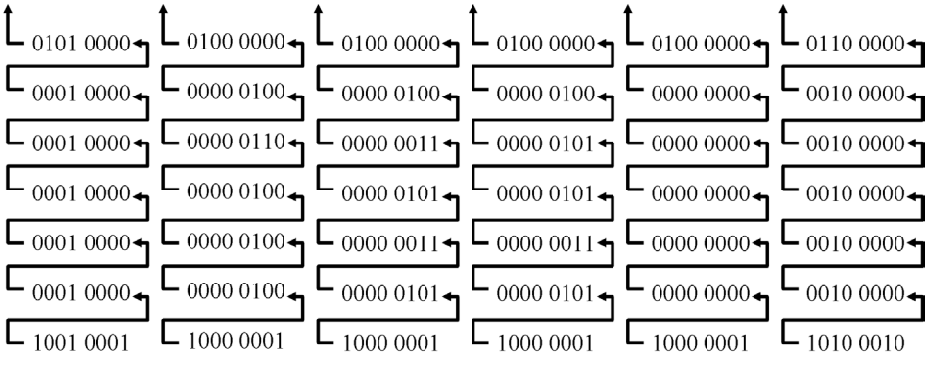

Fig. 17. The artificial genome of the totipotent cell.

6 Conclusion

This paper presented a novel approach for self-replication for Embryonics systems, closer to biological cellular division than past mechanisms. The new molecule thus introduced will be the basic block of a new generation of Embryonics circuits, and is designed to be integrated into a more complex system to implement a full developmental (growth) process in digital hardware.

However, self-replication, as described, is not a very useful feature for current real-world applications, as self-replication is completely deterministic. In other words, its result is a predictable FPGA configuration, which could be realized by conventional FPGA place and route tools.

This argument, however, is not valid if the self-replication process embeds fault-detection and self-repair mechanisms to, for example, grow artificial organisms on an FPGA with fabrication flaws. In fact, standard FPGA place and route tools are not capable of generating bitstream configuration files for FPGA circuits which are not fault-free. A cell composed of molecules capable of au-

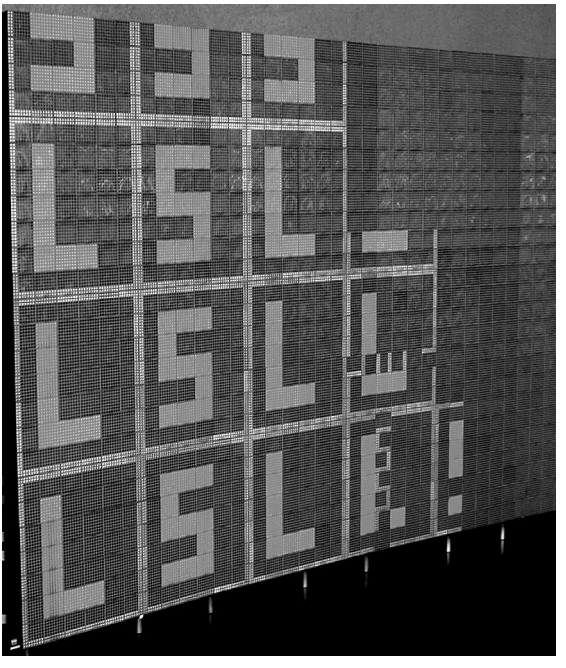

Fig. 18. The BioWall implementation

tonomous self-replication, fault-detection and self-repair, however, will be able to handle flawed FPGAs by adapting its structure so as to avoid faulty areas of the circuit. The next step in the design of the new Embryonics molecule will address this issue and realize autonomous self-replication in faulty circuits.

Moreover, the predicted development of molecular-level electronics (such as nanotechnologies) implies a forthcoming need for mechanisms that will allow a circuit to structure itself, rather than having a structure imposed at fabrication. A cellular division approach such as the one described in this article could be a useful tool to achieve this kind of self-organization.

References

1. Codd, E., F.: Cellular Automata. Academic Press, New York (1968)
2. Ibáñez, J., Anabitarte, D., Azpeitia, I., Barrera, O., Barrutieta, A., Blanco, H., and Echarte, F. Self-inspection based reproduction in cellular automata. Proc. 3rd Eur. Conf. on Artificial Life (ECAL95), LNCS **929**, Springer, Berlin (1995) 564–576
3. Jenkins, J.: Designing with FPGAs and CPLDs. Prentice Hall, Englewood Cliffs, NJ (1994)
4. Macias, N.: The PIG Paradigm: The Design and Use of a Massively Parallel Fine Grained Self-Reconfigurable Infinitely Scalable Architecture. Proc. 1st NASA/DOD Workshop on Evolvable Hardware (EH1999), IEEE Comp. Soc. (1999) 175–180

5. Durbeck, L., Macias N.: Defect-tolerant, fine-grained parallel testing of a Cell Matrix. Proc. SPIE ITCom 2002, Series 4867, (2002) 71–85
6. Durbeck, L., Macias N.: Self-Assembling Circuits with Autonomous Fault Handling. Proc. 4th NASA/DOD Conference on Evolvable Hardware (EH'02), IEEE Comp. Soc. (1999) 46–55
7. Laing R.: Automaton Models of Reproduction by Self-inspection. Journal of Theoretical Biology, volume 66, (1977) 437-456
8. Mange, D., Sipper, M., Stauffer A., Tempesti, G.: Towards Robust Integrated Circuits: The Embryonics Approach. Proceedings of the IEEE **88** (2000) 516–541
9. Sanchez, E.: Field Programmable Gate Array (FPGA) Circuits. Towards Evolvable Hardware, LNCS **1062**, Springer-Verlag, Heidelberg (1996) 1–18
10. Tempesti, G., Mange, D., Stauffer A.: The Embryonics Project: a Machine Made of Artificial Cells. Rivista di Biologia-Biology Forum **92** (1999) 143–188
11. Trimberger, S.: Field-Programmable Gate Array Technology. Kluwer Academic, Boston, MA (1994)
12. Tempesti G., Mange D., Stauffer A., Teuscher C.: The BioWall: an Electronic Tissue for Prototyping Bio-Inspired Systems. Proc. 4th NASA/DOD Conference on Evolvable Hardware (EH'02), IEEE Comp. Soc. (2002) 221–230