Available online at www.sciencedirect.com
SCIENCE DIRECT®
BioSystems 76 (2004) 191–200
BioSystems
www.elsevier.com/locate/biosystems

POEtic: an electronic tissue for bio-inspired cellular applications

Yann Thoma [a,*], Gianluca Tempesti [a], Eduardo Sanchez [a], Juan-Manuel Moreno Arostegui [b]

[a] *Logic Systems Laboratory, Swiss Federal Institute of Technology in Lausanne, Lausanne 1015, Switzerland*
[b] *Technical University of Catalunya (UPC), Barcelona, Spain*

Received 28 February 2003; received in revised form 11 July 2003; accepted 1 August 2003

Abstract

In this paper, we introduce the general architecture of a new electronic tissue called POEtic. This reconfigurable circuit is designed to ease the implementation of bio-inspired systems that bring cellular applications into play. It contains special features that allow a developer to realize systems that require evolution (Phylogenesis), development (Ontogenesis), and/or learning (Epigenesis). A dynamic routing algorithm has been added to a structure similar to that of common commercial FPGAs, in order to allow the creation of data paths between cells. As the creation of these paths is dynamic, it is possible to add new cells or to repair faulty ones at runtime.

Keywords: POEtic; Evolvable hardware; Bio-inspired systems; Dynamic reconfiguration

1. Introduction

Life is amazing in terms of complexity and adaptability. After the fertilization of an ovule by a spermatozoid, a simple cell is capable of recursively dividing itself to create an entire living being. During its lifetime, an organism is also capable of self-repair in case of external or internal aggressions. Living beings possessing a neural network can learn tasks that allow them to adapt to their environment. And finally, at the population level, evolution allows a population to evolve in order to survive in an ever-changing environment. POEtic machines draw inspiration from these three life axes: Phylogenesis (evolution), Ontogenesis (development), and Epigenesis (learning). On the phylogenetic axis, we find systems inspired by the processes involved in the evolution of species through time. The process of evolution is based on natural selection and alterations to the genetic information. These alterations occur through two basic mechanisms: crossover and mutation. Both these mechanisms are, by nature, non-deterministic, which represents both their strength and their weakness. Evolutionary algorithms, such as genetic algorithms (Holland, 1973) or genetic programming (Koza, 1992), exploit this phylogenetic axis to solve NP-hard problems, or in general, problems that cannot be solved by deterministic approaches. All these algorithms rely on the fact that a potential solution (an individual) may be described by a string of values (a string of binary bits or a string of real values). A population of individuals, described by their genotypes, is built, starting with random ones. The phenotype of each individual, created from its genotype, is evaluated, and a selection mechanism is applied to the population. The new generation is then created by applying crossover and mutations on the parent generation, and the process is repeated with the new individuals. These types of

* Corresponding author.
E-mail address: yann.thoma@epfl.ch (Y. Thoma).

0303-2647/$ – see front matter
doi:10.1016/j.biosystems.2004.05.023

algorithms are not guaranteed to find the best solution to a problem, but can usually find an acceptable solution more rapidly than deterministic approaches. The ontogenetic axis concerns the development of a single multi-cellular organism. This process exploits very specific mechanisms to direct the growth of the organism. Cells divide, each offspring containing a copy of the genome (cellular division) and acquiring a functionality (e.g., liver cell) depending on its position (cellular differentiation). Every cell contains the blueprint for the entire organism (the genome), and thus can potentially replace any other cell (Pearson, 2001), providing the possibility of healing (self-repair). The ontogenetic concept of growth has been largely ignored in the past, for obvious reasons: growth implies the "creation" of new material during the lifetime of an organism, a feature that remains impossible to this day in the world of electronics. However, while the silicon substrate of an electronic circuit cannot grow, a reconfigurable circuit (such as our POEtic tissue, as we shall see later) allows its function to be modified at runtime, and can thus implement a growth process based on information rather than matter. The Embryonics project (Mange et al., 2000; Ortega and Tyrell, 1998) is one of the few research efforts in this area. A part of Ontogenesis, morphogenesis, has been the subject of research (Lindenmayer, 1968), and the concept of development is also attracting growing attention (Kitano, 1998). The epigenetic axis concerns the processes and structures of an organism that are directly affected by the environment, such as the nervous system or the immune system. The size of the genome does not allow it to completely describe the structure of an adult organism, and, interaction between the organism and its environment is therefore the only way for further development. Epigenetic mechanisms have already had considerable impact on hardware design in the form of artificial neural networks (ANNs), two-dimensional arrays of massively-interconnected processing elements (the neural cells). ANNs have been successfully exploited in many different domains, from character recognition to robot control, thanks to their adaptability and learning capabilities. State machine evolution, as presented in (Mesot et al., 2003), also uses a learning process to build a robot controller.

2. The POEtic project

POEtic machines draw inspiration from these three life axes (Phylogenesis, Ontogenesis, and Epigenesis) (Sanchez et al., 1997; Eriksson et al., 2003; Roggen, 2003; Sipper et al., 1997; Tempesti et al., 2002; Tyrrell et al., 2003) and from the multi-cellular structure of complex biological organisms. The aim of the POEtic project (http://www.poetictissue.org) is the development of a computational substrate optimized for the implementation of digital systems inspired by all three models mentioned above. The POEtic tissue is a self-contained, flexible, and physical substrate designed to interact with the environment through spatially-distributed sensors and actuators, in order to develop and adapt its functionality, through a process of evolution, growth, and learning, to a dynamic and partially unpredictable environment. The tissue must also self-repair parts damaged by ageing or environmental factors in order to remain viable and perform the same functionality. A novel hardware substrate is sorely needed for research in the domain of bio-inspired machines, as conventional circuits are ill-suited to implement adaptation and specialization, key features of all biological organisms: no bio-inspired architecture can be rigid, but must rather be able to adapt and specialize depending on the desired application. This requirement can be satisfied by exploiting reconfigurable logic (e.g., FPGAs). The POEtic tissue is such a reconfigurable circuit, specially adapted for the implementation of bio-inspired cellular systems, as we shall see. POEtic architectures will consist of two-dimensional arrays of cells. Each cell is a small, fully reconfigurable processing element, specialized for the application and for the bio-inspired mechanisms to be implemented. The role of the tissue is then to provide a digital molecular substrate for the implementation of the cells: as organic cells are constituted by molecules, so our artificial cells will be constituted by the programmable logic elements of our POEtic circuit. Each artificial cell in our architecture contains the entire "genome" for the whole tissue (that is, all of the information required by the organism for the execution of its task) but expresses only part of it (its "gene") depending, for instance, on its position within the organism. This differentiation allows multi-cellular organisms, and thus the POEtic tissue, to display behaviors of

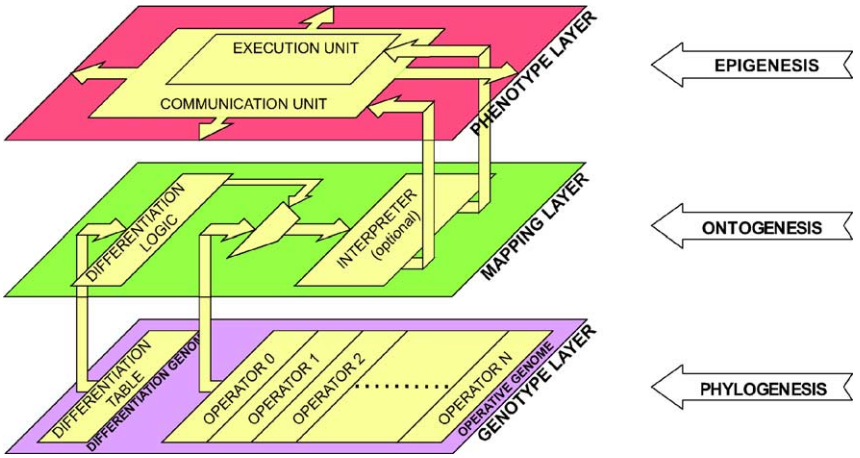

Fig. 1. The three organizational layers of the POEtic project.

a greater complexity than those of single-cell organisms. Moreover, this solution increases the robustness of the system by making it completely local and allowing individual cells to reconfigure their function for growth, self-replication, and self-repair (Ontogenesis). Following the three axes of bio-inspiration, the POEtic cells will be designed as a three-layer structure (Fig. 1):

- The phylogenetic axis acts on the genetic material of a cell. Typically, it could be used to find and select the genes of the cells for the genotype layer, which is conceptually the simplest of the three layers of the tissue, as it is mainly a memory containing the genetic information of the organism.
- Ontogenesis concerns the development of the individual, and thus the mapping or configuration layer of the cell, which implements cellular differentiation and growth. In addition, it has an impact on the system as a whole for self-repair. The configuration layer selects which gene will be expressed depending on a user-defined differentiation algorithm.
- The epigenetic axis modifies the behavior of the organism during its operation, and is therefore best applied to the phenotype layer, probably the most application-dependent layer. If the final application is a neural network, the phenotype layer of the cell will consist of an artificial neuron. In particular, the POEtic project will concentrate on spiking neurons (Eriksson et al., 2003), but this choice does not imply that the tissue will be limited to such a model.

The hardware structure of each layer is specific to its role (e.g., the genotype layer will be mainly storage). The explicit separation in three layers will allow the study and implementation of complex genotype-to-phenotype mappings that can exploit the growth and reconfiguration properties of the tissue. Finally, the three distinct hardware layers allow the implementation of different bio-inspired mechanisms of self-organization, allowing, for example, to implement neuronal structures without growth or evolution, or conventional fault-tolerant systems deprived of adaptation. The main features of the POEtic tissue are listed below:

- An on-chip μ-processor takes care of the evolution process. It is a custom 32-bit RISC, with special random number generation operations.
- Cellular individuals are implemented on a molecular structure that is flexible enough to realize any desired digital circuit, but specialized enough to implement bio-inspired systems efficiently.
- The tissue is capable of self-configuration, useful for growing and/or self-repairing systems.
- An auto-routing mechanism for inter-cellular communication is hard-coded to allow changes of the cellular network at runtime.
- Self-repair at the cellular level can be implemented easily, using dedicated features at the molecular level.

In the next section we describe the global architecture of the POEtic circuit. In Section 4, the molecular plane

structure is detailed, while in Section 5, we explain the inter-cellular dynamic routing algorithm implemented in the second plane.

3. Structural architecture

The POEtic circuit comprises two parts (Fig. 2): the organic subsystem, which is the functional part of the circuit, and the environmental subsystem. Cells, and thus organisms, are implemented in the organic subsystem. It is composed of a grid of small molecules and of a cellular routing layer. Molecules are the smallest unit of programmable hardware and can be configured by software, while dedicated routing units are responsible for the inter-cellular communication. The main role of the environmental subsystem is to configure the molecules. It is also responsible for the evolution process, and can therefore access and change the state of every molecule in order to evaluate the fitness of an organism.

3.1. Environmental subsystem

The environmental subsystem is mainly composed of a micro-controller (a 32-bit RISC processor). Its function is to configure the molecules and to run the evolutionary mechanisms. In order to speed up evolutionary processes, a random number generator has been added directly in hardware. An AMBA bus (ARM, 1999) is used to connect the processor to a system interface that takes care of the communication between the processor and the organic subsystem. This bus is also connected to external pins in order to allow multi-chip communication as well as the use of an external RAM.

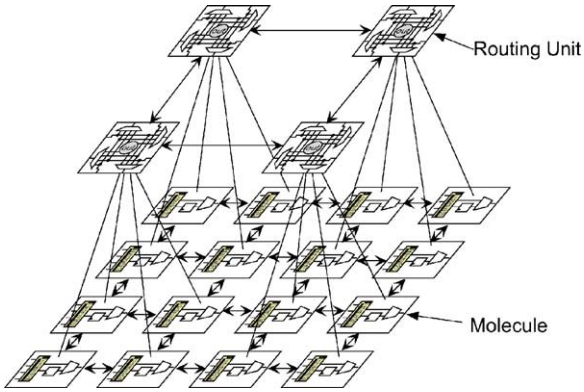

Fig. 3. The organic subsystem is composed of two layers: the molecules and the routing units.

3.2. Organic subsystem

The organic subsystem is made up of two layers (cf. Fig. 3): a two-dimensional array of basic elements, called molecules, and a two-dimensional array of routing units. Each molecule is connected to its four neighbors in a regular structure. It mainly contains a 16-bit look-up table (LUT) and a flip-flop (DFF), and has the capability of accessing the routing layer that is used for inter-cellular communication. This second layer implements a dynamic routing algorithm

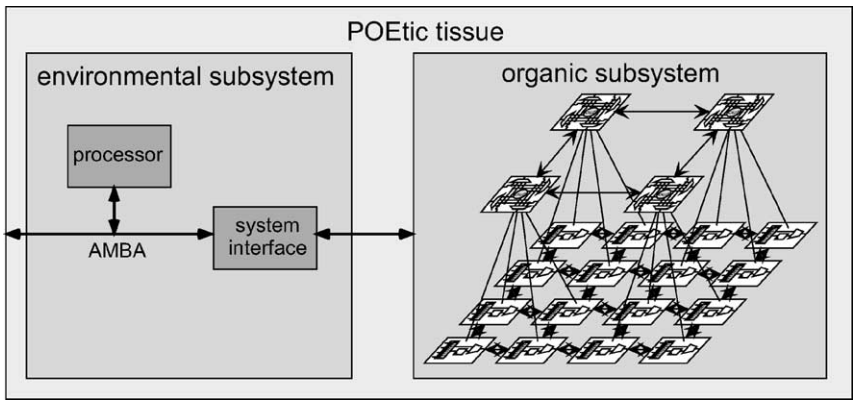

Fig. 2. The POEtic chip, composed of an environmental and an organic subsystems.

allowing the creation of data paths between cells at runtime. All three layers of the POEtic cells, described in the previous section, will be implemented on this subsystem. The molecular structure is described in the next section, while the inter-cellular routing algorithm is explained in Section 5.

4. Molecular structure

The first layer of the POEtic tissue is a two-dimensional array of small programmable units, called molecules. Each molecule is connected to its four neighbors and to a routing unit (four molecules for one routing unit), and mainly contains a 16-bit look-up table (LUT) and a flip-flop (DFF) (Fig. 4). This structure, while seemingly very similar to standard FPGAs (Brown et al., 1992), is however specially designed for POEtic applications: different running modes let the molecule act like a memory, a serial address comparator, a cell input, a cell output, or others. With a total of eight modes, these molecules allow a developer to build cells that are capable of communicating with each other, of storing a genome, of healing, and so on. The eight modes of operation of the molecule are the following:

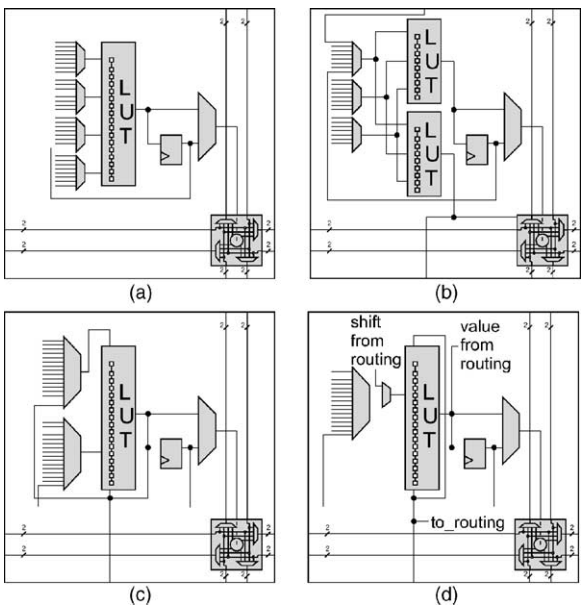

Fig. 4. Structure of a molecule in four different modes: (a) normal, (b) 3-LUT, (c) shift memory, and (d) cellular input.

- Normal: the LUT is a simple 16-bit look-up table.
- 3-LUT: the LUT is split into two 8-bit look-up tables. Two outputs can be used, of which one can be sequential. One of the outputs can be considered as a carry and is directly sent to the south neighbor, allowing fast arithmetic operations.
- Communication: the LUT is split into one 8-bit shift register and one 8-bit look-up table. This mode can be used to implement a packet routing algorithm that will not be presented in this paper.
- Shift memory: the LUT is considered as a 16-bit shift register. This mode is very useful to efficiently store the genome in every cell. Shift memories can be chained in order to create memories of depth 32, 48, etc.
- Configure: the molecule has the capability of reconfiguring its neighbor. Combined with shift memory molecules, this mode can be used to differentiate the cells. A selected part of the genome, stored in the memory molecules, can be shifted to configure the LUT of other molecules (for instance to assign weights to neural synapses).
- Input address: the LUT is a 16-bit shift register and is connected to the routing unit. The 16 bits represent the address of the cell from where the information arrives. The molecule's output is the value coming from the inter-cellular routing layer (this mechanism will be detailed in the next section).
- Output address: the LUT is a 16-bit shift register and is connected to the routing unit. The 16 bits represent the address of the cell, and the molecule sends the value of one of its inputs to the routing unit (this mechanism will be detailed in the next section).
- Trigger: the LUT is a 16-bit shift register, and is connected to the routing unit. Its task is to supply a trigger every n clock cycles (where n is the number of bits of the addresses), needed by the routing algorithm for synchronization.

4.1. Molecular communication

Inter-molecular communication in the POEtic circuit is implemented, as in most conventional FPGAs, by a mix of short-distance connections between

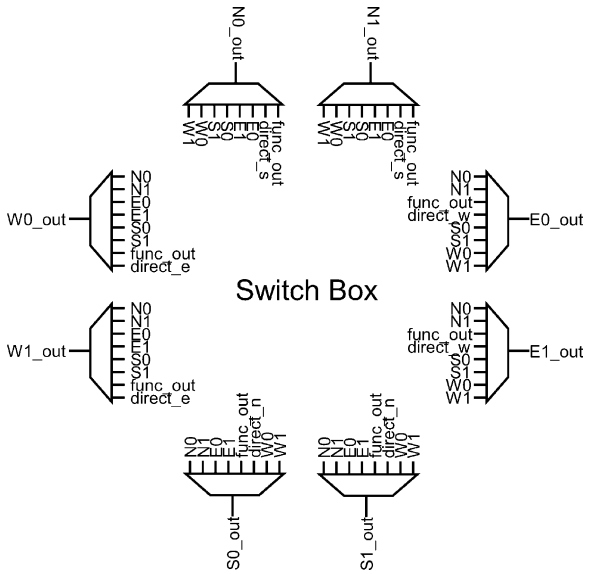

Fig. 5. The switch box present in every molecule.

immediate neighbors and of long-distance connections for more complex routing patterns. Unlike most commercial FPGAs, however, the long-distance connection network is implemented by a switch box (identical in all molecules) that prevents the possibility of short circuits in the network by using multiplexers and directional lines (Fig. 5). This marginally more expensive solution was imposed by the requirements of bio-inspired applications: on one hand, evolutionary mechanisms, when applied to the configuration bitstream of a programmable circuit, generate random solutions that can potentially generate short circuits when two-directional lines are used; on the other hand, the self-configuration capabilities of the circuit can potentially give rise to transient states that, once again, could set up short circuits in the system. Each switch box consists of eight input lines (two from each cardinal direction) and eight corresponding output lines, a size dictated by empirical evidence collected within the Embryonics project (Mange et al., 2000). Each output line can be connected to one of the six input lines from the other cardinal directions (no u-turns allowed) or to one of the two possible outputs of the molecules (the output or the inverted output).

4.2. Self-configuration

To be capable of self-repair and growth, an organism needs to be able to create new cells and to configure them. The configuration system of the molecules can be seen as a shift register of approximately 80 bits split into five blocks: the LUT, the selection of the LUT's input, the switch box, the mode of operation, and an extra block for all other configuration bits. Each block contains, as shown in Fig. 6, together with its configuration, one bit indicating, in the case of a reconfiguration coming from a neighbor, if the block has to be bypassed. This bit can only be loaded from the micro-processor, and remains stable during the entire lifetime of the organism. The special configure mode allows a molecule to partially reconfigure its neighborhood. It sends bits coming from another molecule to the configuration of one of its neighbors. By chaining the configurations of neighboring molecules, it is possible to modify multiple molecules at the same time, allowing, for example, the synaptic weights in a neuron to be changed.

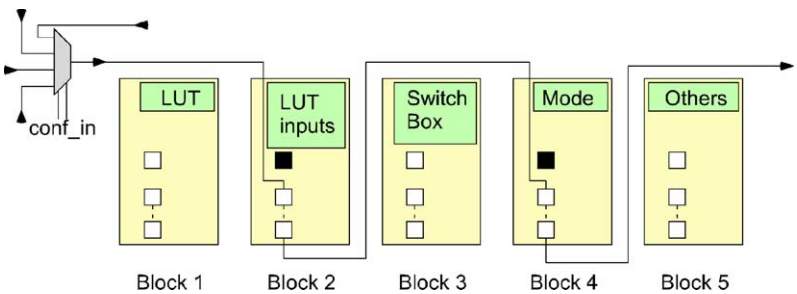

Fig. 6. All configuration bits of a molecule, split up into 5 blocks. The partial configuration bits of blocks 2 and 4 are set, enabling the reconfiguration of the LUT inputs and of the mode of operation by a neighboring molecule.

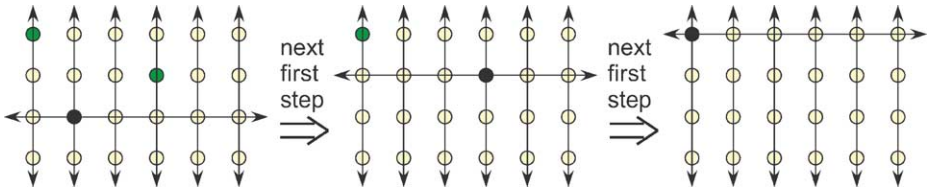

Fig. 7. Three consecutive first steps of the algorithm. The black routing unit will be the master, and therefore perform its routing.

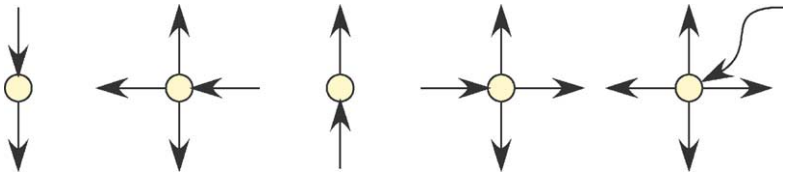

Fig. 8. The propagation direction of the address: north → south ‖ east → south, west and north ‖ south → north ‖ west → north, east, and south ‖ routing unit → north, east, south, and west.

5. Inter-cellular routing

As presented above, our circuit is composed of a grid of molecules in which cells are implemented. In a multi-cellular system, cells need to communicate with each other: a neural network, for example, often shows a very high density of connections between neurons. Commercial FPGAs have trouble dealing with these kinds of applications because of their poor routing capacity. Given the purpose of the POEtic tissue, special attention was given to this problem. Therefore, a second layer was added on top of the molecules, implementing a dynamic routing algorithm inspired by (Moreno Arostegui et al., 2001).

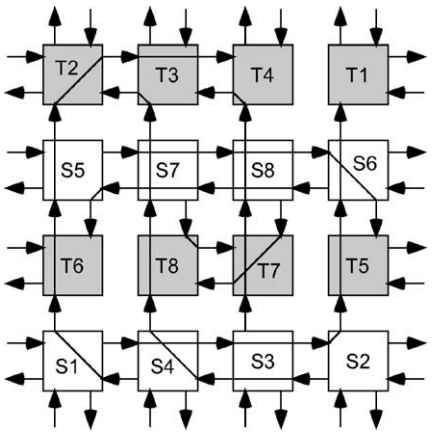

Fig. 9. Test case with a densely connected network.

5.1. Principle

The dynamic routing system is designed to automatically connect the cells' inputs and outputs. Each output of a cell has a unique identifier at the organism level. For each of its inputs, the cell stores the identifier of the source from which it needs information. A non-connected input (target) or output (source) can initiate the creation of a path by broadcasting its identifier, in case of an output, or the identifier of its source, in case of an input. The path is then created using a parallel implementation of the breadth-first search algorithm[1]. When all the paths have been created, the organism can start operation, and execute its task, until a new routing is launched, for example after a cell addition or a cellular self-repair. Our approach has many advantages compared to a static routing process. First of all, a software implementation of a shortest path algorithm, such as Dijkstra's, is very time-consuming for a processor, while our parallel implementation

[1] This algorithm can be seen as a particular case of Dijkstra's shortest path algorithm (Dijkstra, 1959), where all edges have a weight of 1.

requires a very small number of clock cycles to finalize a path. Secondly, when a new cell is created it can start a routing process without the need of recalculating all paths already created. Thirdly, a cell has the possibility of restarting the routing process of the entire organism if needed (for instance after self-repair). Finally, our approach is totally distributed, without any global control over the routing process, so that the algorithm can work without the need of the central micro-processor. The routing algorithm is based on three phases:

5.1.1. Phase 1: finding a master

In this phase, every target or source that is not connected to its correspondent partner tries to become master of the routing process. A simple priority mechanism chooses the most bottom-left routing unit to be the master, as shown in Fig. 7. Note that there is no global control for this priority, every routing unit knowing whether or not it is the master. This phase is over in one clock cycle, as the propagation of signals is combinational.

5.1.2. Phase 2: broadcasting the address

Once a master has been selected, it sends its address in case of a source, or the address of the needed source in case of a target. As shown in Section 4, the address is stored in a molecule connected to the routing unit. It is sent serially, in n clock cycles, where n is the size of the address. The same path as in the first phase is used to broadcast the address, as shown in Fig. 8. Every routing unit, except the one that sends the address, compares the incoming value with its own address (stored in the molecule underneath). At the end of this phase, that is, after n clock cycles, each routing unit knows if it is involved in this path. In practice, there has to be one and only one source, and at least one target.

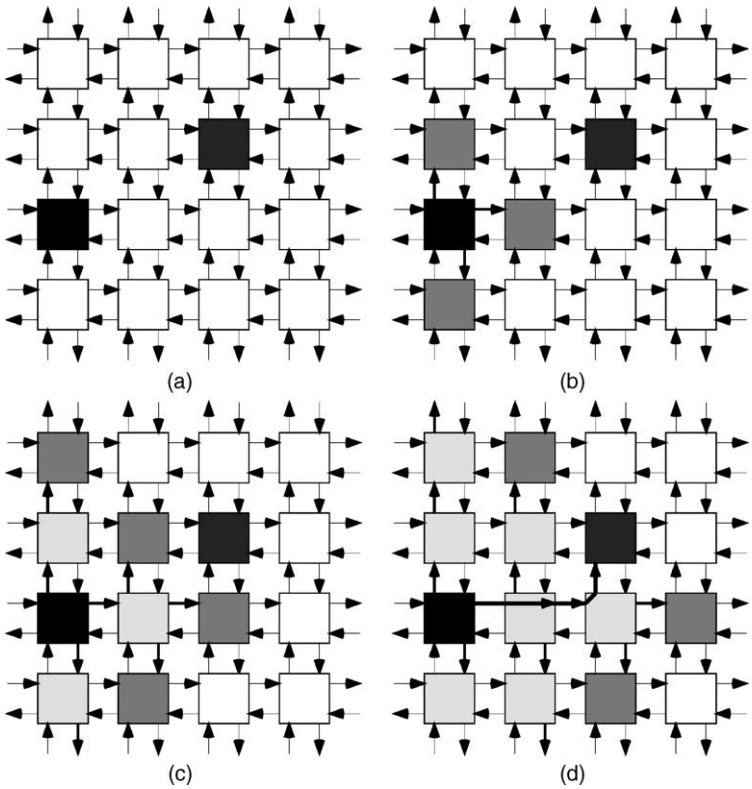

Fig. 10. Step (a) one, (b) two, (c) three and (d) four of the path construction process between the source placed in column 1, row 2 and target cell placed in column 3, row 3.

5.1.3. Phase 3: building the shortest path

The last phase, largely inspired by (Moreno Arostegui et al., 2001), creates a shortest path between the selected source and the selected targets. An example involving eight sources and eight targets is shown in Fig. 9, for a densely connected network. A parallel implementation of the breadth-first search algorithm allows the routing units to find the shortest path between a source and many targets. Starting from the source, an expansion process tries to find targets. When one is reached, the path is fixed, and all the routing resources used for the path will not be available for the next successive iterations of the algorithm. Fig. 10 shows the development of the algorithm, building a path between a source placed in column 1, row 2 and a target cell placed in column 3, row 3. After 3 clock cycles of expansion, the target is reached, and the path is fixed, prohibiting the use of the same path for a successive routing.

6. Conclusion

In this paper we presented a new electronic circuit dedicated to the implementation of bio-inspired cellular applications. It is composed of a RISC micro-processor and of two planes of functional and routing units. The first plane, a two-dimensional array of molecules, is similar to standard FPGAs, and makes the circuit general enough to implement any digital circuit. However, molecules have self-configuration capabilities that are not present in commercial FPGAs and that are important for the growth of an organism and for self-repair at the cellular level. The second plane is a two-dimensional array of routing units that implement a dynamic routing algorithm. It is used for the inter-cellular communication, letting the tissue dynamically create paths between cells. This algorithm is totally distributed, and hence does not need the control of a micro-processor. The first prototype of the POEtic chip will only contain approximately 500 000 equivalent gates. This size would not allow to have more than 1000 molecules in one chip. It will be only possible to implement a very simple organism on a such small number of molecules. Therefore, we included in the design the possibility of implementing a multi-chip organism by seamlessly joining together any number of chips (Fig. 11). This circuit has been tested with a simulation based on the VHDL files describing the entire system. The next step of the project, which is currently under way and which will be completed by the time the conference will take place, is to develop, from the VHDL files, the VLSI layout to realize a testchip to validate the design of our circuit.

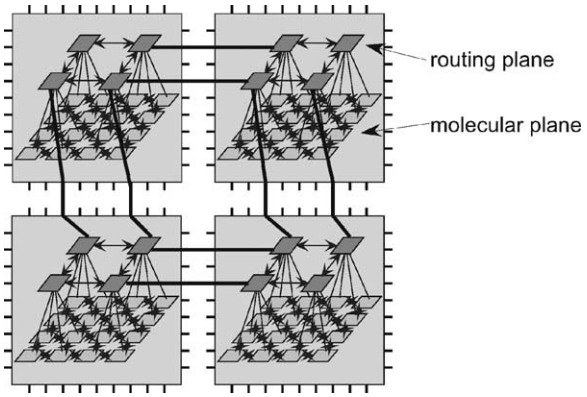

Fig. 11. A multi-chip organism shows the inter-cellular connections.

Acknowledgements

This project is funded by the Future and Emerging Technologies programme (IST-FET) of the European Community, under grant IST-2000-28027 (POETIC). The information provided is the sole responsibility of the authors and does not reflect the Community's opinion. The Community is not responsible for any use that might be made of data appearing in this publication. The Swiss participants to this project are supported under grant 00.0529-1 by the Swiss government.

References

AMBA, 1999. Specification, Rev 2.0. Advanced RISC Machines Ltd (ARM). http://www.arm.com/armtech/AMBA_Spec.

Brown, S., Francis, R., Rose, J., Vranesic, Z., 1992. Field Programmable Gate Arrays. Kluwer Academic Publishers.

Dijkstra, E.W., 1959. A note on two problems in connexion with graphs. Numerische Mathematik 1, 269–271.

Eriksson, J., Torres, O., Mitchell, A., Tucker, G., Lindsay, K., Halliday, D., Rosenberg, J., Moreno, J.-M., Villa, A.E.P., 2003. Spiking neural networks for reconfigurable POEtic tissue, evolvable systems: from biology to hardware. In: Tyrrell, A.M., Haddow, P.C., Torresen, J. (Eds.), Proceedings of the 5th

International Conference on Evolvable Hardware (ICES'03). pp. 165–173

Holland, J., 1973. Genetic algoritms and the optimal allocation of trails. SIAM J. Comput. 2.2, 88–105.

Kitano, H., 1998. Building complex systems using developmental process: an engineering approach. In: Proceedings of the 2nd International Conference on Evolvable Systems (ICES 98), vol. 1478 of LNCS. Springer-Verlag, Berlin, pp. 218–229.

Koza, J.R., Genetic Programming. MIT Press, 1992.

Lindenmayer, A., 1968. Mathematical models for cellular interaction in development, Parts i and ii. J. Theor. Biol. 18, 280–315

Mange, D., Sipper, M., Stauffer, A., Tempesti, G., April 2000. Towards robust integrated circuits: the embryonics approach. In: Proceedings of the IEEE, vol. 88.4. pp. 516–541.

Mesot, B., Sanchez, E., Peña, C.-A., Perez-Uribe, A., 2003. SOS++: finding smart behaviors using learning and evolution. In: Proceedings of the 8th International Conference on Artificial Life, Artificial Life VIII. Bradfford Books. The MIT Press, Cambridge, MA, pp. 264–273.

Moreno Arostegui, J.M., Sanchez, E., Cabestany, J., 2001. An in-system routing strategy for evolvable hardware programmable platforms. In: Proceedings of the 3rd NASA/DoD Workshop on Evolvable Hardware. IEEE Computer Society Press.

Ortega, C., Tyrell, A., 1998. MUXTREE revisited: embryonics as a reconfiguration strategy in fault-tolerant processor arrays. In: Proceedings of the 2nd International Conference on Evolvable Systems (ICES 98), vol. 1478 of LNCS. Springer-Verlag, Berlin, pp. 206–217.

Pearson, H., 2001. The regeneration gap. Nature 414, 388–390.

Roggen, D., Floreano, D., Mattiussi, C., 2003. A morphogenetic system as the phylogenetic mechanism of the POEtic tissue. Evolvable systems: from biology to hardware. In: Proceedings of the 5th International Conference on Evolvable Hardware (ICES 03), vol. 2606 of LNCS. Springer-Verlag, Berlin, pp. 153–164.

Sanchez, E., Mange, D., Sipper, M., Tomassini, M., Perez-Uribe, A., Stauffer, A., 1997. Phylogeny, ontogeny, and epigenesis: three sources of biological inspiration for softening hardware, evolvable systems: from biology to hardware. In: Higuchi, T., Iwata, M., Liu, W. (Eds.), Evolvable Systems: From Biology to Hardware, vol. 1259 of LCNS. Sprinter-Verlag, Berlin, pp. 33–54.

Sipper, M., Sanchez, E., Mange, D., Tomassini, M., Perez-Uribe, A., 1997. A phylogenetic, ontogenetic, and epigenetic view of bio-inspired hardware systems. IEEE Trans. Evolut. Comput. 1.1, 83–97.

Tempesti, G., Roggen, D., Sanchez, E., Thoma, Y., Canham, R., Tyrrell, A., Moreno Arostegui, J.-M., December 2002. A POEtic architecture for bioinspired systems. In: Proceedings of the 8th International Conference on the Simulation and Synthesis of Living Systems (Artifical Life VIII). Sydney, Australia, pp. 111–115.

Tyrrell, A.M., Sanchez, E., Floreano, D., Tempesti, G., Mange, D., Moreno Arostegui, J.-M., Rosenberg, J., 2003. POEtic tissue: an integrated architecture for bio-inspired hardware, evolvable systems: from biology to hardware. In: Proceedings of the 5th International Conference on Evolvable Hardware (ICES 03), vol. 2606 of LNCS. Springer-Verlag, Berlin, pp. 129–140.