Proceedings of the 2007 IEEE Workshop on
Evolvable and Adaptive Hardware (WEAH 2007)

A novel platform for complex bio-inspired architectures

Gianluca Tempesti
Department of Electronics
University of York
Heslington, York YO10 5DD
gt512@york.ac.uk

Fabien Vannel, Pierre-André Mudry, Daniel Mange
Cellular Architectures Research Group
École Polytechnique Fédérale de Lausanne (EPFL)
CH-1015 Lausanne, Switzerland
name.surname@epfl.ch

Abstract— **Cellular architectures represent the natural approach to apply bio-inspired mechanisms to the world of digital hardware. To derive any useful property (in terms of computation) from these mechanisms, however, it is necessary to examine systems that are large enough to pose problems for conventional design methodologies. Moreover, implementing these mechanisms in actual hardware is the only way to ensure that they are efficient from a computational standpoint.**

The realization of this kind of systems, however, requires resources that are both quantitatively and qualitatively different from conventional, off-the-shelf platforms. In this article, we describe a novel hardware platform aimed at the realization of cellular architectures.

The system is built hierarchically from a very simple computing unit, called *ECell*. Several of these units can then be connected, using a high-speed serial communication protocol, to a more complex structure called the *EStack*. Consisting of four different kinds of interconnected boards (computational, routing, power supply, and display), these stacks can then be joined together to form an arbitrarily large parallel network of programmable circuits.

I. Introduction

Cellular architectures represent the natural approach to apply bio-inspired mechanisms to the world of digital hardware. In particular, they represent a possible avenue to identify ways to cope with the increasing complexity of digital systems: current architectures and design methodologies cannot cope with the expected complexity of the next generation of hardware devices.

To verify this claim, however, it is necessary to analyze these mechanisms for systems that are large enough to pose problems for conventional design methodologies and complex enough to justify the overhead unavoidably associated with approaches inspired by nature. It is undeniable that the application of bio-inspired mechanisms to hardware design has too often been limited to simple problems that cannot be scaled to an appropriate level of complexity.

Obviously, there are very good reasons to limit experimentation to simple systems: the efficiency of bio-inspired mechanisms from a computational standpoint can only be verified through an actual hardware implementation (usually in reconfigurable logic) or a detailed simulation. However, the lack of appropriate platforms and the very high computational demands of low-level simulation effectively limit the size of the systems that can be implemented.

On the hardware side, the realization of bio-inspired systems requires resources that are both quantitatively and qualitatively different from conventional, off-the-shelf platforms. In the case of cellular architectures, the hardware implementation of complex systems requires very large devices (in practice, vast amounts of programmable logic) that can be quickly and easily configured as arrays of computational units communicating in complex patterns.

In our past research, we have tried to combine the study of potential applications of bio-inspired mechanisms to the design of large computing systems with the development of hardware platforms capable of implementing these mechanisms at a useful scale. In this article, we present the latest platform we have designed to implement and verify bio-inspired systems of increasing complexity.

The paper is organized as follows: in the next section, we will introduce some of the motivations that are leading us to investigate biological systems as a possible source of inspiration for the design of complex hardware. In section III we will provide a brief overview of the computational approach that motivated the hardware architecture and describe some of the previous work that led us to the development of particular architecture of our system. The hardware platform will then be described in some detail in section IV. Finally, we will present an overview of some utilization issues and conclude the article with some observations and an outline of future work.

II. Motivations

Until recently, increases in computing power have been achieved mostly by accelerating the operating frequency of processors and by designing architectures capable of exploiting instruction-level parallelism through hardware mechanisms such as super-scalar execution. Both approaches seem to have reached their practical limits, mainly due to design complexity and cost-effectiveness issues.

The current trend in computer design seems to favor a simplification of the hardware units (to ease design and layout) and a switch to thread-level parallelization. Computational

1-4244-0699-4/07/$20.00 ©2007 IEEE

power is achieved not by a single very fast and complex processor, but through the parallel operation of several simpler on-chip processors, each executing a single thread. This kind of approach is currently implemented commercially through multi-core processors and in research through the *Network-on-a-Chip* (NoC) paradigm [1][2].

Extrapolating this trend to take into account the vast amount of on-chip resources that will be available in the next few decades (either by further shrinking the silicon fabrication processes or by the introduction of molecular-scale devices), together with the predicted features of such devices (e.g., their high sensitivity to faults or the lack of global synchronization), this approach comes to resemble another paradigm, commonly known as *cellular computing*.

Loosely based on the observation that biological organisms are highly complex structures realized by the parallel operation of vast numbers of relatively simple elements (the cells), this paradigm tries to draw an analogy between multi-cellular organisms and parallel processing systems. At the base of this analogy lies the observation that organisms, in addition to being extremely tolerant to faults and fully asynchronous, are built through a bottom-up self-assembly process and do not require a complete layout.

The actual interpretations and implementations of this paradigm are extremely varied, ranging from theoretical studies [3][4] to commercial realizations (notably, the *Cell* CPU [5][6] jointly developed by IBM, Sony and Toshiba), through biologically-based systems [7], OS-based mechanisms [8] and amorphous computing approaches [9].

Depending from the authors, the *cells* may comprise different levels of complexity ranging from very simple, locally-connected, logic elements to high-performance computing units endowed with memory and complex network capabilities. However, in every case, the basic idea of two-dimensional systems composed of relatively simple connected elements, remains. Our past research, for example, has approached cellular computing by designing large arrays of custom processing elements and by analyzing how some of the mechanisms involved in the development of organisms can be effectively applied to these arrays in order to achieve useful properties such as fault tolerance or growth.

III. Background

Almost every living being share, with the notable exceptions of viruses and bacteria, the same basic principles for their organization. Based on cell differentiation, the incredible complexity present in organisms is met by multi-cellular organization where cells having a limited function achieve very complex behaviors by assembling into specific structures and operating in parallel. By analogy, in the context of thread-level parallelism in a computing machine, *cellular computing* consists of the replication of similar, relatively simple computing elements that execute in parallel the different parts of a given application.

Our past research in this area [10][11][12][13] has focused on developing a hierarchical approach to developing digital hardware that can efficiently implement some specific aspects of this bio-inspired approach. One of its main contributions to the field is the self-contained representation of a possible mapping between the world of multi-cellular organisms in biology and the world of digital hardware systems, based on 4 levels of complexity, ranging from the population of organisms to the molecule (Fig. 1).

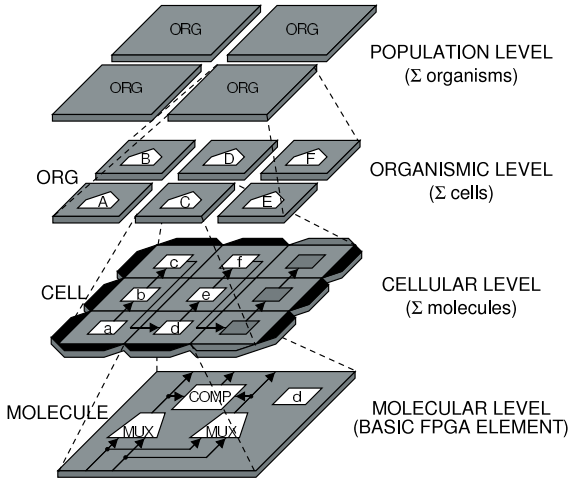

Fig. 1. Hierarchical levels of complexity in a mapping of biological organisms to silicon.

Within this mapping, we define an artificial organism as a parallel array of cells, where each cell is a simple processor that contains the description of the operation of the whole organism in the form of a program (the *genome*). This program, replicated in each cell of the organism as in a living being, is read in parallel in each cell but different parts of it are executed depending on the spatial coordinates of the cell within the organism. The redundancy inherent in this approach is compensated by the added capabilities of the system, such as growth [11] and self-repair [10].

In the currently available technology, the molecules are defined as the basic elements of a programmable logic circuit, and the hardware substrate can then be seen as a uniform surface of programmable logic. To realize our systems, a configuration bitstream is injected into the circuit, causing the molecules to assemble into cells. The cells themselves, after a replication phase analogous to cellular division and growth, self-organize to form the organism.

A key aspect of our project is the need for extremely large prototyping platforms: the implementation of the systems we described requires considerable amounts of programmable logic. This need, along with the non-standard features of our approach, lead us to design and build custom platforms that allow us to implement and test in hardware the mechanisms involved.

The first platform of this kind was realized a few years ago thanks to a grant of the Villa Reuge foundation and was destined mainly to illustrate the features of our approach to the general public. The structure of the platform was centered around the need to clearly display the operation of the system

and, as a consequence, the BioWall ([14], [15]) is a very large machine ($5.3m \times 0.6m \times 0.5m$ for a total of almost $4m^3$). The BioWall is composed of 4000 "molecules", each consisting of a 8 by 8 two color LED matrix, one transparent touch sensor and one Spartan® XCS10XL reconfigurable circuit. Every molecule is connected only to its four cardinal neighbors (with the exception of a few global clock and reset signals).

This "electronic tissue" has been successfully used for developing bio-inspired computing machines [16], and has served as a basis for the development of a second bio-inspired architecture, the POEtic tissue ([17] and [18]). In both cases, the same idea of highly parallel interconnected simple cells has served as the background idea for the realization of the architecture.

Despite the fact the BioWall has fulfilled its role and has been successfully used during several years, it suffers from several limitations which hinder the development of new applications. Firstly, all the FPGAs present on every unit can only be programmed with the same configuration (limited to the 10000 equivalent logic gates of the Spartan XCS10XL), while the considerable delays inherent in propagating a global signal over distances measured in meters limit the clock speed to about one megahertz (adequate for applications that require human interaction, the intended target of the platform). Secondly, the BioWall acts as slave to a host PC: this limitation prevents the machine from being fully autonomous and introduces a functional bottleneck at the interface between the PC and the reconfigurable logic.

These drawbacks, along with the evolution of programmable logic devices, have led us to to the development of *CONFETTI* (for *CONFigurable ElecTronic TIssue*), a novel platform for the implementation of our systems.

IV. A NOVEL HARDWARE PLATFORM FOR CELLULAR COMPUTING

The *CONFETTI* platform tries to avoid the different drawbacks described above by proposing an increased amount of versatility and interchangeability in the different constituting elements of the hardware system. Moreover, the system is built hierarchically by connecting elements of increasing complexity. The main hardware unit is the *EStack* (Fig. 2), a stack of four layers of PCBs:

- The *ECell* boards (18 per *EStack*) represent the computational part of the system and are composed of an FPGA and some static memory. Each *ECell* is directly connected to a corresponding routing FPGA in the subjacent *ERouting* board.
- The *ERouting* board (1 per *EStack*) implements the communication layer of the system. Articulated around 18 FPGAs, the board implements a routing network based on a regular grid topology, which provides inter-FPGA communication but also communication to other routing boards.
- Above the routing layer lies a board called *EPower* that generates the different required power supplies.

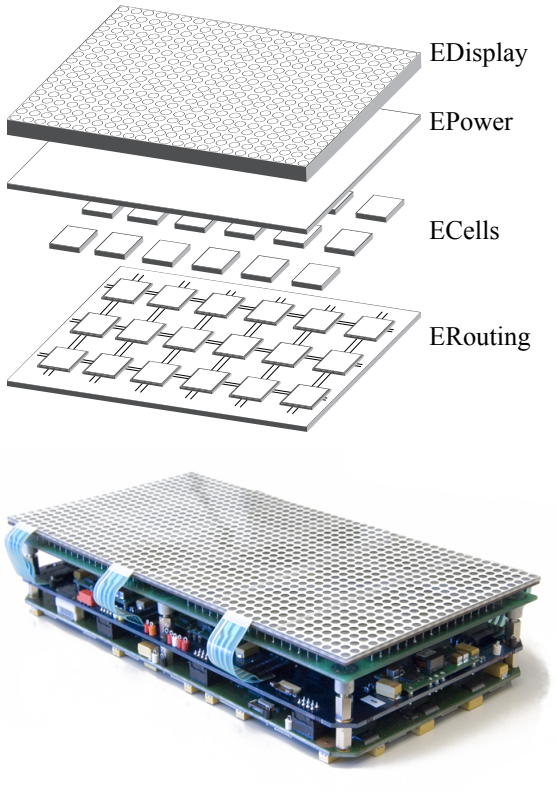

Fig. 2. *EStack* - schematic and photo.

- The topmost layer of the *EStack*, the *EDisplay* board, consists of a RGB LED display on which a touch sensitive matrix has been added.

A complete *CONFETTI* system consists of an arbitrary number of *EStack*s seamlessly joined together (through border connectors in the *ERouting* board) in a two-dimensional array. The connection of several boards together potentially allows the creation of arbitrarily large surfaces of programmable logic. The current test machine, for example, consists of six *EStack*s in a 3 by 2 array (Fig. 3).

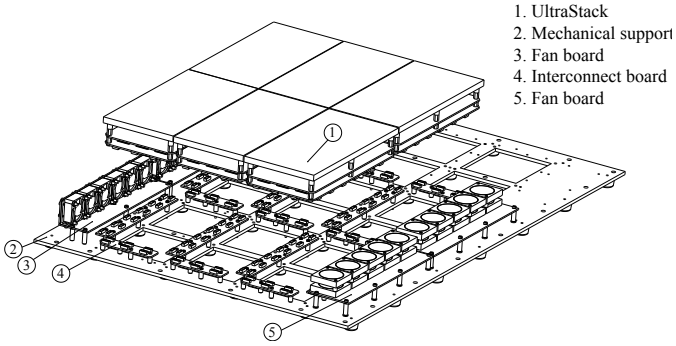

Fig. 3. Schematic of a 3x2 *EStack* system.

A. *The* ECell *board*

The *ECell* (Fig. 4) constitutes the basic computational building block of our hardware platform. It is articulated around a Xilinx® SPARTAN® 3 XC3S200 FPGA coupled with 8 Mbits of 10 ns SRAM memory and a temperature measurement chip. Equivalent to 200000 logic gates, the FPGA includes features such as hardware multipliers, 216 Kb of internal dual-port memory and four digital clock managers (DCM) that allow to obtain, from a local 50 MHz clock, working frequencies of up to 300 MHz. All these components are set on a small (26×26 mm) 8-layer PCB.

The *ECell* possesses various connections, including differential high-speed connections lines with the subjacent FPGA on the *ERouting* board (3 pairs in each direction, 500 Mbit per pair), configuration lines, a communication bus to the *EDisplay* board, as well as power supply lines.

Compared to the BioWall, these boards present some interesting features. Besides being based on much larger FPGAs, the *ECell*s is not limited to neighbor-to-neighbor connections, since the hardware present on the *ERouting* board allows the creation of complex communication patterns without losing computational resources. Also, the presence of a local clock generator allows the implementation of GALS (Globally-Asynchronous, Locally-Asynchronous) systems, a very interesting paradigm for bio-inspired approaches. Finally, the fact the *ECell* boards are plugged into the system (rather then being an integral part) implies the possibility of replacing these boards, either with more powerful ones as FPGA technology improves or with more "exotic" systems that could be used to experiment other bio-inspired approaches.

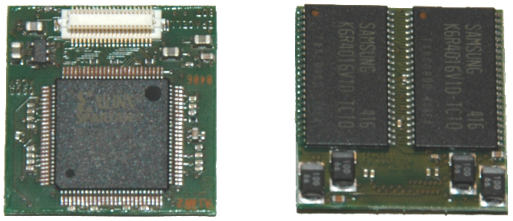

Fig. 4. The *ECell* board (two sides).

B. *The* ERouting *board*

To reduce as much as possible the load on the *ECell* board, where the computational power resides, communication in our system is implemented within a separate *ERouting* board, which also handles tasks such as system configuration and display management.

Measuring 192×96 mm, this highly complex board (twelve layers) has components soldered on both sides and can host as many as eighteen *ECell* boards. Based on a six-by-three regular grid topology (Fig. 5), this board is composed of a set of eighteen reconfigurable circuits (the *ERouting* FPGAs) and eighteen Flash memories.

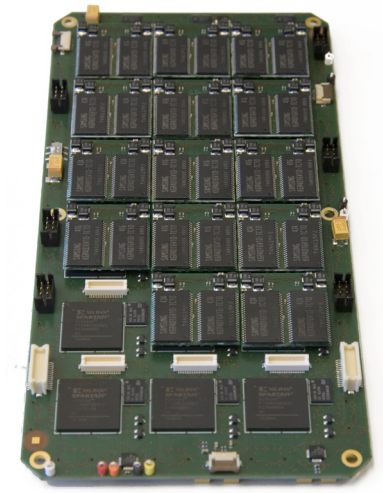

Fig. 5. The *ERouting* board with four *ECell* boards removed on the bottom left corner.

Physically, every *ERouting* FPGA is linked to its four cardinal neighbors and to the *ECell* board above it (Fig. 6). This setup was selected for modularity and scalability purposes (it avoids long and global communication lines that could cause bandwidth degradation in a big *CONFETTI* configuration) and because it is the kind of layout typically used in cellular computing applications.

The links between each FPGA are implemented using the built-in SPARTAN® 3 LVDS (Low Voltage Differential Signaling) I/O drivers that allow, in our case, data rates up to 500 Mbit/s. As depicted on Fig. 6, two communication buses (one for each direction) are present for each neighboring pair. Also, since there is no global clock in the system, a clock signal is transmitted on one differential pair to synchronize the data transmitted on the two others pairs. Thus, at *ERouting* level, a bandwidth of 1 Gbit/s is available on each *ERouting* FPGA for every direction. Moreover, the same type of bus exists between each *ERouting* FPGA and its corresponding *ECell*.

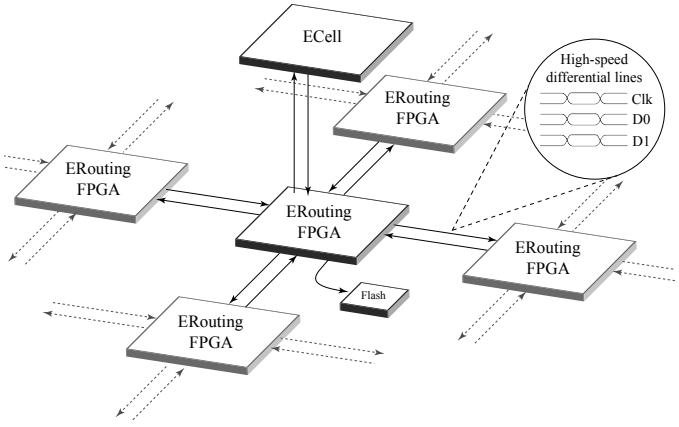

Fig. 6. Detail of one *ERouting* FPGA and link with its *ECell* module.

As the *ERouting* boards constitute the communicating backplane of the whole *CONFETTI*, connections between the different *EStack* boards are also implemented here. External connectors are present on the four sides of the board and provide the same connectivity as the links between the FPGAs: two adjacent *ERouting* boards then represent effectively a single uniform surface of FPGAs. This setup allows the creation of systems consisting of several *EStack*s that behave as a single, larger *EStack*.

C. *The* EPower *board*

The SPARTAN® 3 FPGAs used in *CONFETTI* are very fast and have several interesting embedded features but have the disadvantage of needing several well-stabilized low-power voltages: the FPGA core is powered by a 1.2 V voltage but also needs 2.5 V for the LVDS interface and configuration purposes. Finally, a 3.3 V voltage is needed to interface the Flash and SRAM memories.

To cope with all these requirements and the fact that the eighteen *ECell*s and the *ERouting* board are not only very complex but also power-hungry, an *EPower* board (Fig. 7) was added on top of *ERouting* and *ECell* layers.

This six-layer board, mainly responsible for supplying the correct voltage to the components on the other boards, has the same size as the *ERouting* board. Articulated around six DC / DC converters, it generates from a global 5 V the three mentioned voltages of 1.2 V, 2.5 V and 3.3 V that are then sent to the *ERouting* board via six 8-pins connectors.

Obviously, for a system of this kind, considerations of power consumption, thermal management, and system monitoring come into play. A set of mechanisms, including temperature sensors and fans to cool the system should a rise in temperature be detected, have been put in place to monitor the operation and ensure the safety of the system without compromising its scalability.

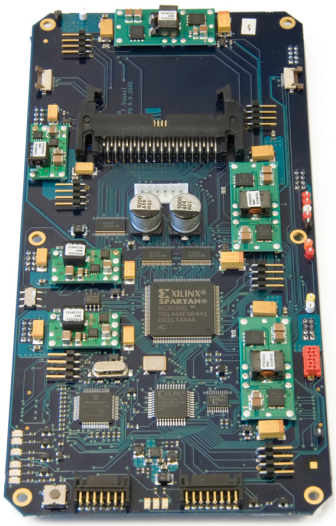

Fig. 7. The *EPower* board.

D. *The* EDisplay *board*

Unlike the BioWall, which was primarily a demonstrator, the main purpose of *CONFETTI* is the high-speed prototyping of complex multi-cell systems. Nevertheless, our experience with the earlier machine led us to integrate in the new one a relatively simple display. On the very top of the *EStack* (see Fig. 2) lies then a 24-bit RGB LED display capable of displaying 48×24 pixels. The displays can be put side by side without any gap, a necessary feature in view of building large systems consisting of several *EStack*s side by side. The purpose of this display is to provide an overview of the operation of the system (for example, to display long-term patterns such as network behavior or thermal buildup) or a compact representation of the state of the units in a cellular architecture. Each *ECell*has access to only part of the screen, a square of eight by eight pixels directly above it. To provide a direct input channel, a touch-sensitive surface is associated to each square.

V. USING THE *CONFETTI* PLATFORM

The *CONFETTI* platform is a very powerful prototyping tool, but also a very complex one. Its open architecture allows great versatility but also requires careful design. To simplify the implementation of the systems to be prototyped, we are working on a set of software tools and reusable hardware (VHDL) components that will allow the user to exploit the capabilities of the system.

In this subsection, we discuss some of the issues related to three key aspects in the design of a system: its configuration, its routing network, and finally its synchronization.

A. *Configuration*

The reconfigurable circuits used in *CONFETTI* are all based on SRAM technology: they can reconfigured an unlimited amount of times and in a relatively short time (typically 20 ms). Unlike the BioWall, in *CONFETTI* every FPGA can be configured with a different bitstream. While this solution obviously affords a much greater flexibility, it also introduces issues related to the need to store and retrieve a large number of configurations.

Notably, because every *ECell* FPGA could have a different configuration and even be reconfigured dynamically, one of the problems that need to be addressed by the system is how to direct the correct configuration to each of the FPGAs in the system. This task is performed within the *ERouting* board, where each FPGA can access an adjacent 16 Mbit Flash memory, typically used to store as many as sixteen different configurations for the *ECell* FPGAs or serve as non-volatile memory available for applications.

The contents of these memories can be modified via an external interface to a computer. Currently, this constitutes the only way to store new applications for the *ECell*s. However, in view of future developments, the configurability of the *ERouting* FPGAs allows almost unlimited versatility in the configuration scheme, allowing for example the implementation of applications that would update the Flash contents

using external memories, Ethernet or WiFi connection, etc., or retrieve the *ECell* configurations from sources other than the local Flash memory. This kind of versatility is a key issue for the implementation of complex mechanisms inspired by heterogeneous natural processes.

B. Routing

One of the main challenges in today's hardware architectures resides in implementing versatile communication capabilities that are able to provide a sufficient bandwidth whilst remaining cost- and size-efficient, as evidenced for example in research for *Network-On-Chip* [19][1] and other [20] systems. For our platform, we opted for a solution based on high-speed serial connections able to sustain different kinds of routing algorithms.

While the 500 Mbit/s links between FPGAs in the *ERouting* board (Fig. 6) provide only local *physical* communication capabilities, the configurability of the FPGAs obviously allows the implementation of more complex routing mechanisms. While neighbor-to-neighbor communication is sufficient for many cellular computing applications, more complex communication schemes such as broadcasting or point-to-point communication can nevertheless be realized by configuring the FPGAs to implement the required switch logic. This kind of configurability is necessary to be able to implement applications that require complex data transfer patterns between the *ECell*s.

Of course, many different types of networking paradigms exist and could be implemented in our system (for example [21], [22] or [23]). As a first example, we decided to use the *Hermes* framework [20], a powerful packet routing system (available as a VHDL core), which provides many interesting functionalities for a relatively low hardware overhead. In the current implementation, five *Hermes* switches are implemented in each *ERouting* FPGA, providing a bandwidth of 500 Mbits/sec in each cardinal direction.

C. Synchronization

As mentioned in section IV-A, an important feature of the *CONFETTI* platform is the presence of multiple clock generators (one for each *ECell*), each running at approximately 50 MHz but out of phase with each other. The presence of these elements reflects the need to emulate with some degree of accuracy very large surfaces of computational elements where the distribution of global signals (including a global clock) becomes very difficult. The lack of global synchronization is (or should be) at the basis of many cellular computing and Network-on-a-Chip approaches.

On the other hand, globally synchronous systems provide some undeniable advantages and are the support of choice for the study of certain complex phenomena using, for example, cellular automata. Though not designed for this kind of systems, the configurability of the FPGAs in the *ERouting* board allows the propagation of global signals throughout the machine, albeit with a performance penalty due to the sheer amount of logic that has to be traversed.

To test the synchronization capabilities of the system and to begin to quantify this penalty, we used the well-known Game of Life cellular automaton [24]. Each *ECell* FPGA implements an 8x8 array of cells (this corresponds to one cell per LED on the *EDisplay* boards), resulting in 6912 cells for the 3x2 *EStack* setup. This solution requires 37% of the *ECell* FPGAs. A 12x12 array (88% of the FPGA) cannot be visualized directly by the *EDisplay* board but increases the total size of the implementation to 15552 cells.

Communication across the *ECell*s is instantiated by a VHDL module that provides 64-bit input and output buses in the four cardinal directions (multiplexed on the available wires). Synchronization is provided by a global signal generated by one of the clock generators, initially propagated across the machine via the *ERouting* FPGAs. This solution, however, incurred in a heavy penalty due to the serialization of transmission, and the update frequency of the automaton was limited to 50 KHz (37 KHz for the 12x12 array). Removing the Hermes framework to exploit the fast 500 Mbit/s for the multiplexed buses increased the operating frequency to 2MHz (1.6 for the 12x12 array).

Further optimizations (both in terms of array size and of operating frequency) are indeed possible, but this experience was sufficient to illustrate the kind of penalty required to achieve global synchronization in a system as large as the *CONFETTI* platform. This penalty is not spurious: it reflects the conflict between the need to design scalable systems and the requirements of global synchronization and clearly illustrates the need for asynchronous approaches once the size of the systems increases beyond a certain limit. Once again, it is interesting to note how biological systems have evolved to operate without the need for any sort of global synchronization between their components.

VI. Conclusions and Future Work

In this paper, we described a novel hardware platform aimed at the realization of cellular computing applications ranging from massively parallel computing through the exploration of various routing paradigms to bio-inspired computing. The versatility of the platform along with the potential computational power it can provide offer very interesting perspectives for future developments.

For example, should more processing power be needed, the *ECell* boards could easily be replaced by a bigger reconfigurable circuit or a different kind of circuit. Similarly, from the software perspective, the system's modularity implies that few changes would be required, for example, to allow different types of *ECell* boards on the same *ERouting* substrate.

The latter option is of particular interest in the larger perspective of a prototyping board for unconventional computing: in the current implementation, the *ECell* boards consist of conventional programmable logic, but nothing prevents their replacement with more "exotic" or non-standard units that could potentially be of interest for research in bio-inspired mechanisms. The interest of a modular approach such as the one used for *CONFETTI* would then be in the option of

exploiting the routing resources of the *ERouting* substrate to connect the units and, probably more importantly, in the possibility of taking advantage of an existing design environment.

In fact, work on the development of the *CONFETTI* platform, whose hardware is near completion (with the probable exception of expanded I/O capabilities), is currently focusing on the development of a set of design and routing tools. Often missing from this kind of academic platforms, we believe that the design of at least a basic set of tools to allow an easy prototyping of different systems is a crucial aspect of our platform and is indispensable to be able to exploit the undeniably massive amount of resources it provides.

Once a basic set of tools is in place, we plan to test the capabilities of the platform on several bio-inspired applications and exploit its computational power to explore complex systems. In particular, the size of the reconfigurable substrate will allow to test the kind of MOVE processor arrays [13] that we are currently using as the basis for our Embryonics project. The features of the *CONFETTI* platform (cellular structure, asynchronous operation, complex routing, etc.) are ideally suited to test the performance and the evolvability of large processor arrays. On another axis, we will use the platform to test in real hardware some of the algorithms we have developed to implement self-replication in programmable logic [25]. Too complex to fit within the BioWall's FPGAs, these algorithms are a necessary step to realize systems capable of growth and structural adaptation.

In a longer perspective, we would like to integrate these two research axes into a single system. We hope then to show that it is indeed possible, given current technology, to design scalable hardware systems that are sufficiently powerful to demonstrate the interest of bio-inspired approaches and possibly open up collaborations with other groups that might be interested in using *CONFETTI* as a platform for their own experimentation in this domain.

ACKNOWLEDGEMENTS

This project was funded by the Swiss National Science Foundation Grant number PP002-68674 and by the Leenaards Foundation, Lausanne, Switzerland.

REFERENCES

[1] G. de Micheli and L. Benini, "Networks on chip: A new paradigm for systems on chip design," in *Proc. Conf. on Design, Automation and Test in Europe (DATE02)*. IEEE Computer Society, 2002, p. 418.

[2] W. J. Dally and B. Towles, "Route packets, net wires: on-chip inteconnectoin networks," in *Proc. 38th Conf. on Design Automation (DAC01)*. ACM Press, 2001, pp. 684–689.

[3] M. Sipper, "The emergence of cellular computing," *Computer*, vol. 32, no. 7, pp. 18–26, July 1999.

[4] M. Sipper and E. Sanchez, "Configurable chips meld software and hardware," *Computer*, vol. 33, no. 1, pp. 120–121, January 2000.

[5] D. Pham, T. Aipperspach, and D. B. et al., "Overview of the architecture, circuit design, and physical implementation of a first-generation CELL processor," *IEEE Solid-State Circuits*, vol. 41, no. 1, pp. 179–196, 2006.

[6] D. Pham, E. Behnen, M. Bolliger, H. Hostee, C. Johns, J. Kalhe, A. Kameyama, and J. Keaty, "The design methodology and implementation of a first-generation CELL processor: a multi-core SoC," in *Proc. Custom Integrated Circuits Conference*. IEEE Computer Society Press, September 2005, pp. 45–49.

[7] M. Amos, *Cellular computing*. New York: Oxford University Press, 2004.

[8] K. Govil, D. Teodosiu, Y. Huang, and M. Rosenblum, "Cellular Disco: resource management using virtual clusters on shared-memory multiprocessors," in *Proc. 17th ACM Symposium on Operating Systems Principles (SOSP99)*, 1999, pp. 154–169.

[9] H. Abelson, D. Allen, D. Coore, C. Hanson, G. Homsy, J. Thomas F. Knight, R. Nagpal, E. Rauch, G. J. Sussman, and R. Weiss, "Amorphous computing," *Communications of the ACM*, vol. 43, no. 5, pp. 74–82, 2000.

[10] D. Mange, M. Sipper, A. Stauffer, and G. Tempesti, "Toward Robust Integrated Circuits: The Embryonics Approach," *Proceedings of the IEEE*, vol. 88, no. 4, pp. 516–541, 2000, invited Paper.

[11] D. Mange, A. Stauffer, L. Peparolo, and G. Tempesti, "A Macroscopic View of Self-Replication," *Proceedings of the IEEE*, vol. 92, no. 12, pp. 1929–1945, 2004.

[12] A. Tyrrell, E. Sanchez, D. Floreano, G. Tempesti, D. Mange, J.-M. Moreno, J. Rosenberg, and A. Villa, "Poetic tissue: An integrated architecture for bio-inspired hardware," in *From Biology to Hardware: Proc. 5th Int. Conf. on Evolvable Systems (ICES03)*. LNCS2606, Springer-Verlag, 2003, pp. 129–140.

[13] G. Tempesti, P.-A. Mudry, and G. Zufferey, "Hardware/software coevolution of genome programs and cellular processors," in *Proc. 1st NASA/ESA Conf. on Adaptive Hardware and Systems (AHS'06)*. IEEE Computer Society, 2006, pp. 129–136.

[14] G. Tempesti, D. Mange, A. Stauffer, and C. Teuscher, "The BioWall: an electronic tissue for prototyping bio-inspired systems," in *Proc. 3rd Nasa/DoD Workshop on Evolvable Hardware*. IEEE Computer Society.

[15] G. Tempesti and C. Teuscher, "Biology Goes Digital: An array of 5,700 Spartan FPGAs brings the BioWall to "life"," *XCell Journal*, pp. 40–45, Fall 2003.

[16] C. Teuscher, D. Mange, A. Stauffer, and G. Tempesti, "Bio-inspired computing tissues: Towards machines that evolve, grow, and learn," *BioSystems*, vol. 68, no. 2–3, pp. 235–244, February–March 2003.

[17] Y. Thoma, G. Tempesti, E. Sanchez, and J.-M. Moreno Arostegui, "POEtic: An electronic tissue for bio-inspired cellular applications," *BioSystems*, vol. 74, no. 1-3, pp. 191–200, Aug.-Oct. 2004.

[18] Y. Thoma, "Tissu numérique cellulaire à routage et configuration dynamiques," Ph.D. dissertation, EPFL, 1015 Lausanne, Apr. 2005, thesis 3226.

[19] T. Bjerregaard and S. Mahadevan, "A survey of research and practices of Network-on-chip," *ACM Comput. Surv.*, vol. 38, no. 1, p. 1, 2006.

[20] F. Moraes, N. Calazans, A. Mello, L. Möller, and L. Ost, "HERMES: an infrastructure for low area overhead packet-switching networks on chip," *Integrated VLSI Journal*, vol. 38, no. 1, pp. 69–93, 2004.

[21] A. Ngouanga, G. Sassatelli, L. Torres, T. Gil, A. Soares, and A. Susin, "A contextual resources use: a proof of concept through the APACHES' platform," in *Proc. 2006 IEEE Workshop on Design and Diagnostics of Electronic Circuits and Systems (DDECS)*. IEEE Computer Society Press, April 2006, pp. 44–49.

[22] D. Wiklund and D. Liu, "SoCBUS: Switched network on chip for hard real time embedded systems," in *Proc. 17th Int. Symposium on Parallel and Distributed Processing (IPDPS03)*. IEEE Computer Society, 2003, p. 78.1.

[23] M. Amde, T. Felicijan, A. Efthymiou, D. Edwards, and L. Lavagno, "Asynchronous on-chip networks," *IEE Proc. Computers and Digital Techniques*, vol. 152, no. 02, March 2005.

[24] E. R. Berlekamp, J. H. Conway, and R. K. Guy, *Winning Ways for your Mathematical Plays*. Academic Press, 1982.

[25] J. Rossier, Y. Thoma, P. Mudry, and G. Tempesti, "Move processors that self-replicate and differentiate," in *Proc. 2nd Int. Workshop on Biologically-Inspired Approaches to Advanced Information Technology (Bio-ADIT 06)*. LNCS3853, Springer-Verlag, 2006, pp. 328–343.