The Tunnel Vision Syndrome: Massively Delaying Progress

Reiner Hartenstein,
Professor, IEEE fellow, SDPS fellow, FPL fellow,
TU Kaiserslautern, Karlsruhe Institute of Technology (KIT), UnB

Introduction. Despite of acceleration factors by up to several orders of magnitude the FPGA share in the IC market is still below 2%. The multicore and supercomputing dilemma massively reduces programmer productivity and the progress of performance improvement. Predictions are warning that, if current trends continue, the total of all electricity bills spiraling out of control could have serious consequences for the overall affordability of computing. The progress in power-efficient high performance solutions, coming from a few isolated areas, is by far too slow to break current trends. A few "silver bullet results" touting a single facet as the complete answer are far from solving the problem. What is the reason of these slow-down effects? Here the **ASAP** conference series was always a platform for critical discussions. The following sections illustrate, how the recognition of systolic arrays and term rewriting as fundamentals of data-stream-based computing was delayed for decades by the tunnel vision syndrome.

The History of Systolic Arrays

Here a very interesting example is the history of systolic arrays: first examples of hardware accelerators. For algebraic operations like matrix computations first systolic arrays have been introduced by algebra specialists. The title of their famous paper claims just only "for special purpose VLSI chips" [1] Who was here going to organize the data stream sequencing ? The algebra experts tell us: "that is not our job!" Some EE guy will come with a soldering iron and connect the array to some signal sources. So the algebra scene missed to define and introduce the data-stream-based machine paradigm. Although it's the (hidden) backbone of running a systolic array, this paradigm has been introduced later by other people: the anti-machine [2, 3] (or Xputer [4] or MoM [2, 5]). What synthesis method did they introduce? Their answer was: "of course algebraic!" This means linear projection yielding only uniform linear pipes, so that only applications with strictly regular data dependencies are supported which is a severe restriction. This problem has been solved later by the EE student Rainer Kress who replaced their algebraic synthesis methods by simulated annealing which supports also any irregular & wild form pipe networks. The consequence is a generalization of the systolic array which we called

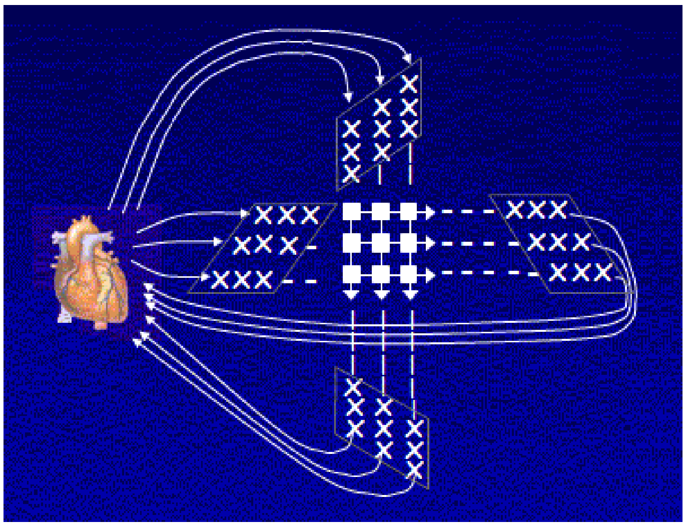

Fig. 1; Who controls data stream sequencing?

“super-systolic array” (or *“Kress Array”*) [6-8]. This was also presented in Killarney, Ireland [8], at the first conference of the series meanwhile called “**ASAP**”. Another synthesis system of that kind came later with a user interface based on a hardware description language came to **ASAP’89** [9]. Systolic arrays on reconfigurable computing platform was covered at **ASAP’94** [10]. Even going a step further discussed at **ASAP’95** and **ASAP’96** means a systolic methodology for programming a data-stream-based general purpose processor [11, 12]. SYS^3 (SYStolic SYnthesiS SYStem) presented in Washington, DC, is an advanced systolic array synthesis system [13, 14]. A data sequencer methodology for avoiding the need for instruction memory cycles at address computation was presented at **ASAP’97** [15].

$$P = \sum_{i=0}^{3} 2^{i} \cdot y_{i} \cdot X$$

Fig. 2; Math formula for 4 bit integer multiplication

Using Term Rewriting (TR) for Synthesis

A trailblazing idea for TRS use in microchip design [16] has been delayed by 30 years. All Term Rewriting Systems (TRS) in EDA are used for verification, which means bottom-up approaches. By the year 2002 the still only known example of TRS top-down use in EDA was the automatic generation of an integer multiplier floorplan from the Math formula. This has been investigated in 2001 by the TRS expert Prof. Mauricio Ayala-Rincón from Universidad de Brasilia. Equations in fig. 3. are the set of rules by the TRS system for this 4 bits wide integer multiplier example [17, 18]. Here "shl" is the KARL language's shift left operator and its prefix i or j or (i+j) or (i-j) is the number of bits to be shifted and "shr" stands for "shift right". The equation in Figure 2 shows the design task of a 4 bit integer multiplier. Figure 4 shows the ABL graphic editor's presentation illustrating the result: a 4 bit integer multiplier, where "lsb" selects the least significant bit. The circuit has been manufactured via the MPC service of the E.I.S. Project [19].

$$2^{1} \cdot w = i \text{ shl}(w)$$
$$i \text{ shl}(w) = -i \text{ shr}(w)$$
$$i \text{ shl}(j \text{ shl}(w)) = (i+j) \text{ shl}(w)$$
$$i \text{ shl}(v) + i \text{ shl}(w) = i \text{ shl}(v+w)$$
$$i \text{ shl}(w) = j \text{ shl}((i-j) \text{ shr}(w))$$
$$j \text{ shl}(w) = j \text{ shl}((j-i) \text{ shr}(w))$$
$$X = (x_3, x_2, x_1, x_0)$$
$$Y = (y_3, y_2, y_1, y_0)$$

Fig. 3; Term Re-writing rules.

Fundamentals of this term rewriting example were published in the late 70ies, but were ignored by the TRS expert scene. In 2001 Prof. Ayala-Rincón investigated this example and came to the conclusion [20, 21], that working for verification, all EDA applications of TRS had used only bottom-up methods, and, that this multiplier design example is still new, since it is the only known top-down TRS application in the EDA scene. This quarter decade delay is another example having been caused by the Tunnel Vision Syndrome.

We must rethink all basic assumptions.

It has been demonstrated above, that the discovery of the fundamental significance of very simple ideas mentioned early has been delayed by decades. An example from the early 70ies is the growing popularity of hardware description languages (HDLs) [22, 23]. Mapping from software to HDL turned out to be very simple: “a decision box turns into a demultiplexer“. Why did it take 30 years to find out this extremely simple truth?”

Often special research scenes and their technical committees treat their area as the center of the world. This Aristotelian view point must be replaced by a Copernican model of the R&D world. We cannot afford a further dominance of traditional reductionist approaches suffering from the tunnel vision syndrome. This also results in fundamental misconceptions in algorithmic complexity blocking further progress. We must rethink the basic assumptions. Not only for bridging the hardware/software chasm urgent curriculum revisions have become a massive challenge. We must radically reinvent computing and its education framework. We must introduce connected thinking to bridge the gaps between different paradigms like between instruction-stream-based computing and data-stream-based computing as well as between reconfigurable and hardwired, but urgently also between several abstraction levels. The history of computer engineering proved, that moving to such "connected thinking" approaches can be extremely successful, like within the Mead-&-Conway VLSI design revolution [24,25] (see Fig. 5), which has been the most influential successful project in the history of modern computing science

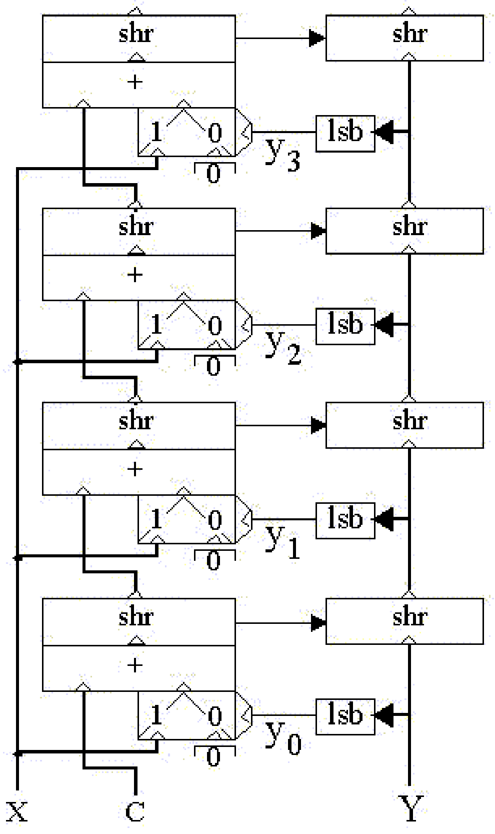

Fig. 4; 4 bit integer multiplier.

We need a massive exploitation of all areas from circuit design and test, over architecture, system design, run time and operating systems, and programming up to an exhaustive coverage of application-specific algorithmic aspects from all application areas. In addition to **ASAP**s (application-specific array processors) a new EDA and compilation framework is coming up for dynamically reconfigurable array processors offering power-efficient performance optimization features.

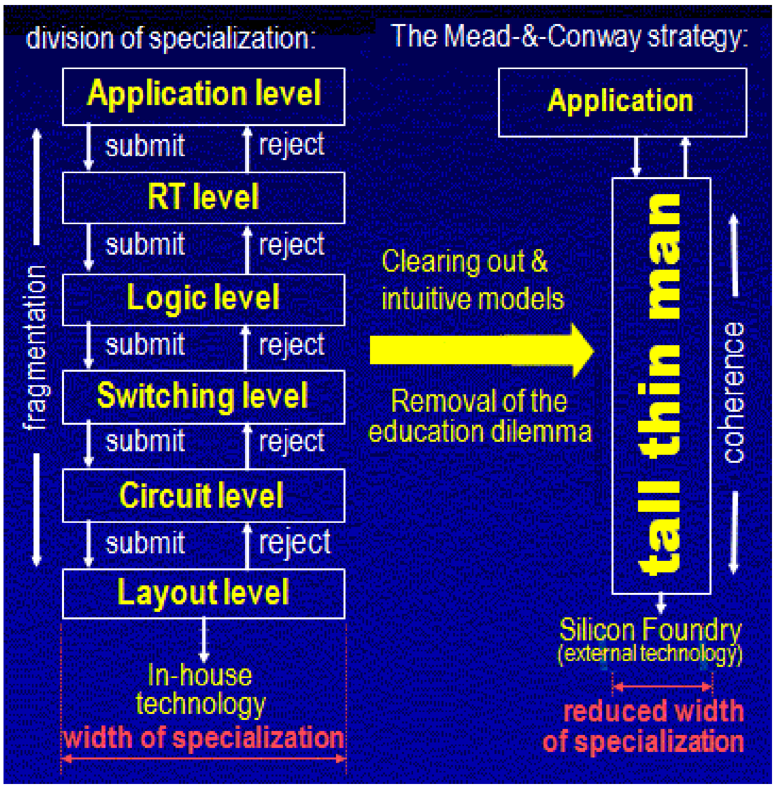

Fig. 5; Perspective by merging abstraction levels.

The R&D landscape requires radically new solutions for design and programming. Also the coming extremely massive parallelism in extreme-scale computing demands an elimination of all obstacles to meet unprecedented demands on data handling by a much deeper integration between applications and data inside all kinds of memory, from CPU caches thru disks. This requires massive changes at all levels from compilers and runtime systems down to storage behavior, challenging all disciplines from circuit design and test, up to architecture, system design, run time and operating systems, and programming. Overcoming the hardware-only mind set would be a fascinating job for computer system architecture colleagues and curriculum designers.

Conclusions

The paper has illustrated, how reductionist philosophies of most specialized research areas tend to prevent or to massively delay to achieve fundamental discoveries. This paper warns, that there are areas, where because of reductionists' tunnel view attitudes the R&D progress is by far too slow to cope with problems, which must be urgently solved, like, for instance, for the long term survival of our ICT infrastructures. We must urgently rethink all basic assumptions and disruptively reform the motivational views and structures of all R&D group, reorganize the cooperation patterns of the entire R&D landscape and drastically reorganize all curricular patterns and their interconnect.

References

[1] M. J. Foster and H. T. Kung: The Design of Special-Purpose VLSI Chips; 7th ISCA, La Baule, France, May 6-8, 1980

[2] R. Hartenstein, A. G. Hirschbiel, M. Weber: MOM - a Partly Custom-Designed Architecture Compared to Standard Hardware, Proc. IEEE Compeuro, Hamburg, Germany, May 8 – 12, 1989,

[3] http://anti-machine.org/

[4] http://xputer.de/

[5] R. Hartenstein, A. G. Hirschbiel, M. Weber: Mapping Systolic Arrays onto the Map-Oriented Machine (MoM); **ASAP'89**, Kilarney, 1989.

[6] R. Hartenstein, R. Kress: A Datapath Synthesis System for the Reconfigurable Datapath Architecture; Asia and South Pacific Design Automation Conference (ASP-DAC'95), Nippon Convention Center, Makuhari, Chiba, Japan, Aug./Sept. 1995

[7] Rainer Kress: A Fast Reconfigurable ALU for Xputers, Ph.D. Dissertation, 1996, University of Kaiserslautern, Germany

[8] http://kressarray.de/

[9] R. Hartenstein, K. Lemmert: CHDL-Based CAD System for the Synthesis of Systolic Architectures; in: **ASAP'89,** Kilarney, 1989.

[10] R. Hartenstein, H. Reinig, et al.: A Dynamically Reconfigurable Wavefront Array Architecture for Evaluation of Expressions; **ASAP'94**, San Francisco

[11] R. Hartenstein, J. Becker, et al.: A Parallelizing Compilation Method for the Map-oriented Machine; **ASAP'95**, Strasbourg, France, July 1995

[12] R. Hartenstein, J. Becker, M. Herz, et al.: A Synthesis System for Bus-based Wavefront Array Architectures; **ASAP'96** Chicago, Ill., USA

[13] R. Hartenstein, K. Lemmert: SYS3 - A CHDL-Based Systolic Synthesis System, Proc. CHDL'89, Washington, D.C., U.S.A., Elsevier, 1989

[14] Karin Lemmert: SYS3- A Systolic Synthesis System around KARL, Ph. D. Dissertation, University of Kaiserslautern, Germany,198

[15] R. Hartenstein, J. Becker, M. Herz, U. Nageldinger: A Novel Sequencer Hardware for Application Specific Computing; **ASAP`97**, Zurich, Switzerland

[16] R. Hartenstein: Fundamentals of Structured Hardware Design; Elsevier, 1977

[17] http://xputers.informatik.uni-kl.de/karl/karl_history_fbi.html#3.1 algebraic structured design

[18] R. Hartenstein: The History of KARL and ABL; in: .J. Mermet (editor): Fundamentals and Standards in Hardware Description Languages; Kluwer Academic Publishers, September 1993

[19] http://www.fpl.uni-kl.de/staff/hartenstein/eishistory_en.html

[20] M. Ayala-Rincón, R. Maya Neto, R. P. Jacobi, C. Llanos, R. Hartenstein: Architectural Specification and Simulation Through Rewriting-Logic; Colombian Journal of Computation, Vol 3(2):20-34, 2003.

[21] M. Ayala-Rincón, R. P. Jacobi, R. Hartenstein, C. Llanos: Modeling Reconfigurable Systolic Arrays for Computing Algebraic Operations via Rewriting-Logic; Seminar on Dynamically Reconfigurable Architectures, Dagstuhl, Germany, July 20-25, 2003

[22] W. A. Clark: Macromodular Computer Systems; 1967 SJCC, AFIPS Conf. Proc.

[23] C. G. Bell et al: The Description and Use of Register-Transfer Modules (RTM's); IEEE Trans-C21/5, May 1972

[24] Reiner Hartenstein: Early EDA Innovations in Europe driven by Lynn Conway's Lambda Notation; http://hartenstein.de/Hartenstein-EDA-Innov-Europe-3.pdf/

[25] http://xputer.de/EIS