Higher Performance by less CPUs

Reiner Hartenstein, TU Kaiserslautern, http://hartenstein.de

The end of Moore's Law. Increasing the architectural complexity and increasing the clock frequency of single-core microprocessors has come to an end (e. g. see, what has happened to the intel Pentium 4 successor project). Instead, multi-core microprocessor chips are emerging from the same vendors. But just more CPUs on the chip is not the way to go for very high performance. This lesson we have learnt from the supercomputing community having followed the wrong road map for decades. For much more successful efforts we need a paradigm shift, over to a new fundamental model, such as available from the Reconfigurable Computing community.

Classical parallelism does not scale. However, classical parallelism does not scale well - a very expensive lesson which we have already learnt from the supercomputing community with massively increasing the number of processors when going cheap COTS (commodity off the shelf). With the growing degree of parallelism, the programmer productivity goes down drastically („The Law of More"). It is an illusion to believe, that scalability would get massively better, when all these processors will be resident on a single chip - as long as the monopoly of the von Neumann mind set will not be relieved, where the classical fundamental paradigm is still based on concurrent sequential processes and message passing through shared memory, both being massively overhead-prone and memory-cycle-hungry. Rescue should not be expected from threads, although intel pre-announced some tools intended to avoid, that the programmers shy away. In his cover feature article [1] Edward A. Lee from UC Berkeley claims, that for concurrent programming to become mainstream, we must discard threads as a programming model. Nondeterminism is the overhead-prone problem, not only hidden behind methodologies attributed „speculative".

Escape the Software Development Paradigm Trap, said IRIS director Mark Bereit [2], who refutes the assumption that software development will always be difficult and bug-ridden, noting that this is due „solely to the software development paradigm that we've followed, unchallenged, for decades", also the reason of bad scalability and bad programmer productivity in classical parallelism. Bereit proposes reworking the model and studying other engineering disciplines for inspiration. He proposes to study mechanical engineering. But much better is studying Reconfigurable Computing. Tensilica senior vice president Beatrice Fu said, that reconfigurable computing offers the option of direct processor-to-processor communications without going through memory nor through a bus. This paradigm shift is an old hat, but until recently mostly ignored, not only by the supercomputing community. Buses cause multiplexing overhead [3] and the dominating instruction-stream-based-only fundamental model is extremely memory-cycle-hungry [4]: the reason of the *„memory wall"*. The alternative offered by reconfigurable computing is data stream parallelism by highly parallel distributed local memory. This memory parallelism is more simple and more straight forward than e. g. interleaved memory access known from vector computers.

Crooked Labelling. The difference between Parallel Computing and Reconfigurable Computing is often blurred by projects labelled "reconfigurable", which, in fact are based on classical concurrency on a single chip. To avoid confusion: switching multiplexers or addressing registers at run time is not „reconfiguration". At run time, real Reconfigurable Computing never has an instruction fetch: only data streams are moving around.

FPGAs have become mainstream years ago in embedded systems. This is a new computing paradigm based on configware instead of software. Configware is not instruction-stream-based and needs compilation methods which are simple, but fundamentally different from compiling software. Compared to software solutions speed-up factors up to 4 orders of magnitude have been obtained, due to a different form of parallelism, which is drastically less overhead-prone than classical parallelism from concurrency by communicating (von Neumann) sequential processes. A few of the many embedded FPGA application examples are: control, signal processing, automotive, multimedia, video, wireless, music, vision, coding, defense, image processing, crypto, pattern recognition, HDTV, manufacturing, aerospace, computer graphics, and many others.

FPGA-based Scientific Computing. More recently FPGAs are also highly popular for scientific computing many application areas: A few of these application examples are: medical, physics, defense, environmental, chemical, evolution, mathematics, fluid dynamics, astrophysics, bio, genetic, weather, chemistry, molecular, mechanics, neural network, DNA, pattern recognition, computer graphics, materials science, atomic bomb, data mining, combustion, crash simulation, black hole, petroleum, oil and gas, and many others.

CPUs outperformed by FPGAs. The world-wide total running compute power of FPGAs outperforms that of CPUs. Most total MIPS running worldwide have been migrated from CPUs to accelerators, often onto FPGAs. The FPGA market with more than 6 billion US-Dollars is the fastest growing segment of the integrated circuit market. The rapidly growing number of FPGA-based design starts is now an an order of magnitude higher than the shrinking number of ASIC-based design starts. By this software to configware migration, enormous

speed-up factors are obtained, ranging from one and more orders of magnitude (OoM), for instance in astrophysics, to two and more OoM in molecular biology and other bioinformatics applications, for instance, over to 3 OoM in cryptography, and more than 3 OoM in DSP and wireless communication, up to almost 4 OoM in pattern matching and image processing: higher performance by less CPUs.

Massively slashing the electricity bill and equipment cost. Also major supercomputer vendors went to Reconfigurable Computing, where a highly welcome side effect of migration to FPGAs is the massive reduction of the electricity bill amount (by one order of magnitude) and of the equipment cost (already one supercomputer vendor, sgi, has filed for Chapter 11 bankruptcy protection). Saving electricity by software to configware migration might become a strategic issue at national level or global level. The press tells us, that 25% of Amsterdam's electricity consumption goes into server farms (Amsterdam is an internet hub), and, that Google's yearly electricity bill amounts to 50,000,000 US-dollars - more than the value of its equipment.

The von Neumann paradigm is loosing its dominance. Not only our curricula still mostly ignore, that the de facto current model is not von Neumann, but a symbiosis of CPU (or multiple CPUs) and non-von-Neumann accelerators - in fact, a dual-paradigm model. To-day, by far most compute power comes from non-von-Neumann accelerators attached to the CPU: a mixture of (a) hardwired accelerators, and (b) reconfigurable accelerators, like, for instance, FPGAs. The CPU (central processing unit) looses this central role by becoming an auxiliary processing unit mainly to run legacy code. Much more MIPS than running software are running configware, although most engineers do not yet know how to spell „configware".

The Reconfigurable Computing Paradox. FPGAs have bad technology parameters. Because of massive overhead like wiring overhead, reconfigurability overhead, and routing congestion their effective integration density (transistors per chip really serving the application: DeHon's first law) is less than 0.01% of that of the Gordon Moore curve. More negative factors are contributed, since FPGAs are very power-hungry (compared to hardwired accelerators) and their clock frequency substantially less than 1 GHz) is massively lower that of microprocessors or hardwired accelerators. There are more negative factors on FPGAs, such as very poor application development support, implementation languages and tools unacceptable for software people, and extremely poor reconfigurable computing education, or none at all, since it is ignored by CS curricula. What is the reason of the rapid market growth and of these massive speed-up factors, although parameters and other factors around FPGAs are so massively bad? Reasons of this paradox are: the paradigm trap (the wrong mind set using the wrong model), severe educational deficits (graduates having the wrong educational background: missing the de facto job market), and management deficits. Removing these educational deficits will help to remove this paradox.

Educational Deficits. A less welcome side effect of the paradigm shift are educational deficits needing a training on the job, since typical CS or CE curricula ignore Reconfigurable Computing - still driving the dead road of the von-Neumann-only mind set [4]. A new IEEE international workshop series on Reconfigurable Computing Education has been founded to cope with this problem [5].

Advantages of Coarse-grained Reconfigurability. Coming along with a more convenient abstraction level than from FPGAs, coarse-grained reconfigurability makes the educational gap smaller. Another advantage of coarse-grained reconfigurability is the much higher computational density than coming with FPGAs. A computational density by 4 orders of magnitude higher than with a FPGAs is obtained by using a rDPA (reconfigurable DataPath Array) with rDPUs instead of CPUs. For software people this is much more easy to understand than FPGAs because DPUs share the same abstraction level with CPUs. To software people the configuration of FPGAs looked more like logic design on a strange platform. But in contrast to a CPU, a DPU is not instruction-driven and has no program counter and its operation is transport-triggered by the arrival of operand data. This new machine paradigm (the counterpart of von Neumann) is based on free form large pipe networks of rDPUs (without memory wall & compilation is easy), but not on concurrent sequential processes. There is no instruction fetch overhead at run time since these pipe networks, generalizations of the systolic array, are configured before run time. This new paradigm is based on data-streams generated by highly parallel distributed on-chip local small but fast memory which consists of auto-sequencing memory (ASM) blocks [6] using reconfigurable generic address generators (GAG), providing even complex address computations not needing memory cycles [6]. This kind of memory parallelism is more simple and more straight forward than interleaved memory access known from vector computers.

The personal supercomputer is near. The Munich-based startup PACT [7] has demonstrated, that a 56 core 16-bit rDPA running at less than 500 MHz can host simultaneously everything needed for a world TV controller, like multiple standards, all types of conversions, (de)compaction, image improvements and repair, all sizes and technologies of screens, and all kinds of communication including wireless. More high performance by less CPUs, by reconfigurable units instead of CPUs. By this methodology also a single-chip

game console is feasible on such a coarse-grained reconfigurable platform. A highly promising vision would be a super pentium with multiple dual-mode PUs, which could individually run in CPU mode or in rDPU mode (not using the program counter). Choosing the right distributed on-chip memory and the tight reconfigurable interconnect between these PUs is the key issue.

Management Deficits. Because of educational background problems the impact of a paradigm shift can hardly be explained to managers by a one page executive summary. A well known historical example is the windows methodology of OS user interfaces having been pioneered on all the Altos computers running at Xerox PARC in the late 70ies. But the at that time very rich Xerox Corp. failed in launching this as a product, so that a startup called Apple Computer was needed to open this market by the Macintosh. So it is not very likely, that one of the major microprocessor vendors now going toward multi core microprocessor chips will adopt the coarse-grained reconfigurable array methodology. Another vision would be, that e. g. Xilinx inserts a rDPA onto a new platform FPGA targeting the scientific computing market. The RAMP project [8] having proposed to run the operating system on an FPGA sounds like: „Xilinx inside“ instead of „intel inside“.

Literature

[1] Edward A. Lee: The Problem with Threads; COMPUTER, May 2006

[2] Mark Bereit: Escape the Software Development Paradigm Trap; Dr. Dobb's Journal (05/29/06)

[3] R. Hartenstein, G. Koch: The Universal Bus considered harmful; in: R. Hartenstein R. Zaks (editors) Microarchitecture of Computer Systems; American Elsevier, New York 1975

[4] R. Hartenstein: Why we need Reconfigurable Computing Education; http://hartenstein.de/RCedu.html

[5] http://helios.informatik.uni-kl.de/RCeducation/

[6] M. Herz et al.; Memory Organization for Stream-based Reconfigurable Computing; 9th IEEE International Conference on Electronics, Circuits and Systems (ICECS 2002), September 15-18, 2002, Dubrovnik, Croatia

[7] http://pactcorp.com

[8] http://ramp.eecs.berkeley.edu/