The breaking point of modern processor and platform technology

Sverre Jarp, Alfio Lazzaro, Julien Leduc, Andrzej Nowak

CERN openlab, Geneva, Switzerland

<Andrzej.Nowak@cern.ch>

Abstract. This work is an overview of state of the art processors used in High Energy Physics, their architecture and an extensive outline of the forthcoming technologies. Silicon process science and hardware design are making constant and rapid progress, and a solid grasp of these developments is imperative to the understanding of their possible future applications, which might include software strategy, optimizations, computing center operations and hardware acquisitions. In particular, the current issue of software and platform scalability is becoming more and more noticeable, and will develop in the near future with the growing core count of single chips and the approach of certain x86 architectural limits. Other topics brought forward include the hard, physical limits of innovation, the applicability of tried and tested computing formulas to modern technologies, as well as an analysis of viable alternate choices for continued development.

CERN-IT-2011-005
17/01/2011

1. Introduction

The Worldwide LHC Computing Grid (WLCG) services millions of jobs per month on over one hundred thousand cores and it is the backbone of LHC data processing and a true facilitator of discovery. At its foundation lie the enormous success and the ubiquity of x86 technology [1]. The ability of an x86 core to adapt to high level code is remarkable, and has been an inherent property of nearly all systems bought to run with High Energy Physics (HEP) software in the last 10 years. x86 has given us many other benefits which go far beyond adaptive capacities. The point of view presented in this work is that some of these benefits, certain of which are still being improved and under development, might slowly begin to dissipate and lose their original value, or some of it. If such a scenario were likely to happen, it might jeopardize the efficiency of HEP computing as a whole.

The reason for this is that there is a strong dependency of HEP software on the x86 architecture. HEP has been a benefactor of different architectural enhancements for years but, above all, the driving force behind sustained development has been frequency scaling. It was not unusual to see the frequency double or triple between processor families – e.g. between the Pentium and Pentium 2, or between the Pentium 2 and Pentium 3. These kinds of improvements are not expected today, and jumps in frequency have been replaced by multi-core extensions and microarchitectural enhancements. Faced with such significant changes in mainstream computing technology, one should ask the question whether the pace and benefit of such developments are sustainable in the case of HEP.

The constant increases in computing requirements, quests for finer precision and future prospects – such as that of the Super LHC, with projected data rates an order or two of magnitude higher than the ones today – are all reasons why good performance and efficiency matter.

2. Current state and near future trends

Data processing in HEP is by definition a parallel model. Tracks or events can be processed independently of each other which allows for relatively easy scaling with multi-core technologies by multiplying the amount of independent processes on a single machine. This is the adopted processing scheme for the Grid and it is one that certainly has worked well. One obvious component of the cost of such a strategy is an increased memory requirement per machine and it had been noticed well before its implementation [2]. There are, however, other components of this cost which are not necessarily as apparent.

One seamless effect of "natural" scaling out with multi-core is the reduction of importance of single process performance in favor of a general optimization of the throughput of the whole system. As it is currently standard to have up to 12 cores in a single machine in a computing center, a certain level of overcommitment can often take care of some of the perceived gaps in performance. This might lead to situations where the execution units of the CPU are still underutilized but other resources are under strain.

2.1. Trends in software

A rather common approach to computing in HEP in general is that software is a constant and the hardware should provide a certain level of performance. High level languages, such as C++, are highly valued amongst programmers for their ability to provide a flexible and well organized source. Investigations conducted by CERN openlab [3] and experts at Intel show that such choices contribute directly to the poor efficiency of the code. Recent results obtained by David Levinthal of Intel show that on average, there is a jump every 6-7 instructions and a call every 35-65 instructions [4] – not a surprising picture for well developed C++ applications. Due to the amount of "bubbles" generated in the pipeline by this behavior, typical HEP C++ code will benefit heavily from various facilities present in a modern x86 processor, most notably the out of order engine and simultaneous multi-threading, if available. Despite these gains, the superficial cycles per instruction (CPI) ratio is in many cases unimpressive, between 1 and 2.5. Compared to a theoretical CPI maximum of 0.25, this figure can be approximated to a statement that in such a case the CPU is not doing anything useful during 70% - 95% of the time. In effect, the convenience of C++ is often paid for heavily in efficiency, and the choice of language might well be one of the key points tying HEP to the x86 architecture, which excels at dealing with inefficient code. Unfortunately for HEP, planned improvements in parts of the processor which are today being heavily underutilized will not be very beneficial.

C++ is likely to remain the language of choice for HEP for the near future. However, even higher level managed languages, such as Python and Java, are making advances as well – mostly as scripting and interface languages. At this time, neither the former nor the latter is fit for high performance computation.

2.2. Trends in hardware

Moore's Law has long been the driving idea behind silicon innovation, which allows chipmakers to produce products giving customers (such as physicists) more performance with every new generation. However, an increased number of transistors on a die is not tied in any direct way to better performance. Recent products and roadmaps announced by chip manufacturers suggest that the emphasis will be not on further core enhancements, or even adding numerous cores, but on more functional units. In recent months we've seen the addition of graphics, cryptography and enhanced vector/SIMD units to mainstream processor lines [5]. Unless an unlikely scenario takes place, a natural consequence of this trend is that HEP will not be the primary customer of these developments and will get only minor benefits coming from microarchitectural changes, incomparable to the improvements that were observed until now.

A particular example in this case, one that the HEP community is influenced by directly, is that of the vector/SIMD unit. In a typical case, HEP programs make very limited, if any, use of vector

computing. Vectorization by hand isn't always feasible but modern compilers (such as the Intel compiler) often offer the functionality of automatic loop vectorization and effective reporting on vectorization. This topic is gaining vastly in importance given that vector widths on the x86 are growing in size. Given the 256 bit SIMD width of AVX today, one is using only 25% of compute capability when processing scalar doubles, and less than 13% when processing scalar floats. There is a clear indication that vector widths might grow to 1024 bits and AVX has been designed to support such a potential expansion [6].

It has already been highlighted that so far HEP is making successful use of multi-core systems through independent processing. The increase in performance, and subsequently throughput, is provided through the number of cores which increases from generation to generation. It should be noted however, that this is not a sustainable option for growth as the increase in the number of cores in future processors is arithmetic, not geometric. For example, the maximum number of cores for Intel's families of processors was 2, 4, 8 and 10 for Core 2, Penryn, Nehalem and Westmere respectively.

In addition, recent investigations by CERN openlab [7][8] show that there exist noticeable drops in efficiency on many-core systems. In particular, in the case of the Nehalem-EX, HEPSPEC06 [9] scaling on 32 cores that is close to expected is obtained only after enabling hardware threading (SMT). The throughput for 32 cores (no SMT) loaded with HEPSPEC06 is only around 75% of the throughput of 1 process multiplied by the number of cores [7].

Unsurprisingly, developments at both Intel and AMD suggest that hardware threading is the processor manufacturers' response to the high expectations of customers with respect to CPU efficiency [9]. The benefit of this technology for HEP is unquestionable, with observed throughput improvements being in the 30% range [7][8]. Nevertheless, due to the high memory requirements of WLCG jobs, SMT typically remains switched off on production machines. In order for the HEP community to be able to benefit from hardware threading on a large scale, some simple yet fundamental changes to both software and job scheduling need to be made. In addition, the problem of relatively high memory consumption per process would need to be addressed as well.

3. Further future trends and the hard limits of innovation

In addition to the constraints already mentioned, there is a group of hard limits for innovation which should be kept in mind when considering future activities.

3.1. The scalability of Moore's Law

One of the most important problems the industry will face in several years is that of silicon scalability. The scaling principles defined by Moore's Law have been relentlessly pursued over the years, but physical limits exist and they are already a concern. Already today there are products where a layer of metal can be only 5 atoms thick [11]. Assuming that the right materials and processes are developed to bring feature sizes down even further, there is a hard limit which cannot be overcome with traditional methods. Projected feature sizes in 10 years are around 5-7nm [12], which would be a layer of only 50-70 silicon atoms (roughly 1Å each) that would need to be accurately placed in mass production. In the meantime, a 1nm (10Å) limit reachable in 2020 is the hard limit envisioned by Intel [13], which is the world's leading semiconductor manufacturer. While certain technologies under development (such as 3D stacking and nanowires) might allow for such a feature size, it is by no means as guaranteed, as current feature sizes were 10 years ago. Thus the promises of keeping Moore's Law alive are diminishing, and the consequence of that might be that the problem of computing might need to be revisited on a rather fundamental level.

3.2. Power consumption limits

In direct relation to the issues mentioned above is the aspect of Thermal Design Power (TDP). Higher frequency, and thus higher performance of a CPU, can theoretically be delivered but at a heavy cost defined by the following formula: $P \sim C \cdot f \cdot V_{cc}^{2}$, where P is the TDP, C is the capacitance of the

system, f is the frequency and V_{cc} is the supply voltage. V_{cc} needs to be kept as low as possible, it cannot be reduced by large margins, since a low voltage level slows down the switching speed and imposes limits on the maximum frequency. In consequence, major chipmakers strongly opt for keeping the TDP constant across their product lines and this trend is unlikely to change, despite some relaxations in TDP limits introduced through Turbo Mode [5]. The most recent implementation of Turbo Mode proposed in the Sandy Bridge architecture from Intel defines a more granular approach to dynamic frequency scaling and is an attempt to further reduce the energy switching quantum. The benefit of this technology, especially with respect to power/performance efficiency, remains to be investigated. To sum up, it does not seem likely that in the next 5-10 years there will be a return of frequency scaling as witnessed in the Pentium era.

3.3. Inter-core interactions

Furthermore, a modern x86 core is not ready to co-exist with many (double digits and more) other cores of the same type. A lot of the fundamental evolution around x86 is based on a shared memory concept and on the concept of coherent caches. Relatively recent Non-Uniform Memory Access (NUMA) implementations are only an evolution of these ideas and already today we can appreciate the strong influence of NUMA on our workloads [14]. Already with the core counts of modern platforms, e.g. 40 for Intel and 48 for AMD, keeping cache coherency and cache traffic under control prompts the introduction of novel solutions. Examples are HyperTransport, QPI (CSI) or ring buses within the CPU, such as the one in Sandy Bridge designs from Intel [5]. It is likely that these issues will either limit the multi-core push, or at least force a redesign and architectural, inter-core communication constraints.

3.4. Trends in experimental x86 designs

Experimental designs such as Intel's Knights Ferry [15] or the Single-chip Cloud Computer [16] are a testbed for different avenues, all of which begin with the issues described above. The Knights family (also known as Many Integrated Core - MIC) prototype is an x86 32-core, 128-thread, in-order co-processor with wide 512-bit vectors. It features a ring bus for inter-core communication and a familiar x86 cache architecture. As a very noteworthy development vehicle, it allows select programmers to evaluate the behaviour and performance of their code on an architecture incorporating many possible future technologies. It is not unreasonable to expect that the result of this experiment will influence the plans for the whole x86 family.

On the other hand, the SCC is a cluster of 48 Intel Architecture cores, tiled and interlinked only with their neighbors, where packet-based message passing is the basic means of communication. In the case of SCC there is no coherence support in the hardware for memory access (for simplicity) and the chip features advanced power management options, with variable frequencies and voltages per tile (2 cores) and tile group (8 cores) respectively. This is yet another example of how experimental development vehicles are meant to test technology and shape the future of x86 processors and it is also a clear indication of the directions in which x86 computing is going.

4. Exotic options

In addition to mainstream trends which are independent of the developments in HEP and at CERN, there are certain less conventional options which might help to sustain the computing space and thus might be considered.

One "natural" option is to pursue software multi-threading to its fullest extent, where applicable. With a clean and thorough implementation, this method can yield substantial memory savings which can easily translate into direct financial savings as it is a low cost solution to a high cost problem. A successful example is the multi-threaded Geant4 prototype constructed by researchers at Northeastern University [17], where the memory usage of a complex Geant4 simulation has been reduced to around 25 megabytes per thread from an original figure of hundreds of megabytes. This option seems to be logical and appropriate, especially given the fact that core counts within a single machine are bound to

grow over the next several years, and memory requirements of HEP software are independently growing with its complexity.

Another option is to deepen the divide between independent processes today and – for instance – to place every process on its own machine (or, alternatively, together with a small number of other similar processes). CERN openlab has already begun examining such an approach and it has concentrated on the low-cost, power efficient Atom processor produced by Intel for server-like deployments [18]. Initial results have been encouraging, although since then the Atom processor hasn't developed as hoped. Nevertheless, one could envision very low-cost, low power PC farms or arrays of microblades based on highly efficient implementations of x86 technology. As it is a promising and cost-effective option honoring the peculiarities of HEP software, it will be revisited in the future.

Considering the two options above, a "middle point", hybrid architecture could be developed. In such a case, hardware systems would run "clusters" or "batches" of jobs, wherein jobs within a single cluster would be similar, but would differ across clusters. This scenario could be a much larger scale version of the experimental Single-chip Cloud Computer, or of what NVIDIA has implemented in their G80 architecture [19]. Many of the issues related to this kind of architecture have already been extensively explored on a different scale using various communication technologies, for example the MESI protocol, NUMA or MPI.

Yet another option is to consider different requirements of jobs and, in order to minimize cost, to move to pipelined processing. This idea however, is not likely to be implemented in a Grid environment where, by definition, one machine should be able to serve any submitted job.

Finally, a rather more serious undertaking would be a complete switch to a different microarchitecture. This is a highly disruptive scenario but it is one that has been implemented several times in the past – most notably when moving from mainframe to RISC systems and then to PC (x86) computing. In the near and medium future however, the chances of such a shift happening seem very low. This is stated considering the overhead of such an operation and given that x86 will most likely remain the best performing option due to the complexity and characteristics of HEP software and the share of the total market. The role of other architectures might be explored in an "accelerator" format. This question however, raises the complex issue of the triggering and cost of a complex transition. The trust that the HEP community bestowed on the commodity industry is remarkable but should not be taken for granted.

5. Recommendations and conclusions

For the reasons outlined in this paper, as well as in order to succeed in the quest for optimal, or even sustained throughput, every effort is recommended to be made to ensure that HEP software and x86 developments are not progressing in orthogonal directions.

In order to address the case where there are many unused transistors available, but the frequency needs to remain fixed, semiconductor manufacturers had to look for applications for the surplus silicon. As a thorough rearrangement of the delicate innards of x86 processors is out of the question, the extra transistors have to be assigned to caches or functional units. Both the former and the latter have been developing heavily in the past years but the operating frequency of the CPU has remained largely unchanged. There are several technologies which can serve as examples of such improvements: most notably hardware threading, vector/SIMD units, added functional units and further platform integration along with an incremental core count increase.

It becomes clear that it is up to the client to benefit from these technologies as their proliferation and expansion is guaranteed. It is imperative that hardware enhancements which have a future are considered seriously and utilized with good effect. One prominent example of such a neglected feature is vectorization – the scope of applicability of this technique is much wider than the scope of its usage. Secondly, software threading is a technique which can be applied to many of the frameworks used in HEP without the need to invest significant resources. The return on this investment is very high: a memory consumption decrease of an order of magnitude in the case of the multi-threaded Geant4

prototype and a 100x speedup in the case of the ALICE track fitter example (including vectorization) pioneered by GSI researchers [20], with the assistance of CERN openlab.

Thirdly, hardware threading deserves the fullest attention. In order for SMT to be feasible in practice, standard memory allocations need to be significantly lowered – a requirement which applies also in the case of threading.

The fourth point is that software needs to be assessed both from the source and algorithmic angles. Significant advances can be made through algorithmic optimization, as the ALICE track fitter example demonstrates through a speedup of ~5 orders of magnitude [20] (~4 orders of magnitude for the ALICE track finder).

Finally, the choice of a compiler and its options is paramount to achieving efficient processing on modern x86 platforms. With many of CERN's software frameworks still running in 32-bit mode (e.g. because of memory issues), many of the improvements brought after the Pentium are forfeited in. Advanced compiler optimization may not guarantee bit-precision in floating point, but well designed algorithms will permit heavy optimization and yield code which can run an order of magnitude faster than non-optimized code. Performance Monitoring Unit (PMU) based performance assessment is also helpful when investigating compiler related changes and software performance in general.

If implemented properly, these key points will help to implement sustainable scalability on future flavors of x86 and will prepare HEP software for other challenges that might lie ahead.

References

[1] Jarp S, Tang H, Simmins A and Yaari Refael 1995 PC as a Physics Computer for LHC? *CHEP 2010*

[2] Panzer B 2005 CPU, disk and memory issues *LCG Computing fabric presentations*

[3] CERN openlab webpage *http://cern.ch/openlab*

[4] Levinthal D 2010 Performance Analysis and SW optimization lab for CERN *CERN openlab*

[5] Kahn O, Piazza T and Valentine B 2010 Intel Next Generation Microarchitecture Codename Sandy Bridge *Intel Developer Forum 2010*

[6] Firasta N, Buxton M, Jinbo P, Nasri K and Kuo S 2009 Intel AVX: New Frontiers in Performance Improvements and Energy Efficiency *Intel Software Network*

[7] Jarp S, Lazzaro A, Leduc J and Nowak A 2010 Evaluation of the Intel Nehalem-EX server processor *CERN openlab*

[8] Jarp S, Lazzaro A, Leduc J and Nowak A 2010 Evaluation of the Intel Westmere-EP server processor *CERN openlab*

[9] Micheloto M et al 2009 A comparison of HEP code with SPEC benchmark on multicore worker nodes *CHEP 2009*

[10] Drysdale G, Valles A and Gillespie M 2009 Performance Insights to Intel Hyper-Threading Technology *Intel Software Network*

[11] Bohr M, Chau R, Ghani T and Mistry K 2007 The High-k Solution *IEEE Spectrum*

[12] Holt W 2010 Increasing Our Technology Lead *Intel Investor Meeting 2010*

[13] Mayberry M 2010 Emerging Technologies and Research Focus *Moore's Law Roundtable*

[14] Leduc J 2010 NUMA memory systems *CERN openlab workshops*

[15] Skaugen K 2010 Petascale to Exascale *Intel MIC Launch*

[16] Intel 2010 Single-chip Cloud Computer *Intel SCC Symposium*

[17] Apostolakis J, Cooperman G and Dong X 2010 Multithreaded Geant4: Semi-Automatic Transformation into Scalable Thread-Parallel Software *Europar 2010*

[18] Balazs G, Jarp S and Nowak A 2008 Is the Atom processor ready for High Energy Physics? An initial analysis of the dual core Atom N330 processor *CERN openlab*

[19] NVIDIA 2006 NVIDIA GeForce 8800 GPU Architecture Overview

[20] Kisel I 2009 Online Event Reconstruction in HEP Experiments *FIAS Colloquim, Frankfurt*