# NanoStreams: Advancing the Hardware and Software Stack for Real-Time Analytics on Fast Data Streams

Gillan, C. J., Nikolopoulos, D. S., Bilas, A., & Bekas, C. (2014). NanoStreams: Advancing the Hardware and Software Stack for Real-Time Analytics on Fast Data Streams. 1-8. Paper presented at e-Challenges e-2014 Conference , Belfast, United Kingdom.

**Document Version:**
Peer reviewed version

# NanoStreams: Advancing the Hardware and Software Stack for Real-Time Analytics on Fast Data Streams

Charles J. Gillan[1], Dimitrios S. Nikolopoulos[1], Angelos Bilas[2] and Costas Bekas[3]

[1]*The School of Electronics, Electrical Engineering and Computer Science Queen's University of Belfast, Queen's University, Belfast, BT7 1NN, United Kingdom*
*Tel: +44 2890 971700, Fax: +44 2890 971702, Email:c.gillan@qub.ac.uk*

[2]*FORTH-ICS, Heraklion, GR-70013, Greece*
*Tel: +30 2810391669, Fax: +30 2810391661, Email:bilas@ics.forth.gr*

[3]*IBM Research, Zurich Saumerstrasse 4, 8803 Rueschlikon - Switzerland*
*Tel: +41 44 724 8969, Email: bek@zurich.ibm.com*

**Abstract:** NanoStreams is a consortium project funded by the European Commission under its FP7 programme and is a major effort to address the challenges of processing vast amounts of data in real-time, with a markedly lower carbon footprint than the state of the art. The project addresses both the energy challenge and the high-performance required by emerging applications in real-time streaming data analytics. NanoStreams achieves this goal by designing and building disruptive micro-server solutions incorporating real-silicon prototype micro-servers based on System-on-Chip and reconfigurable hardware technologies.

## 1. Introduction

Real-time data analytics is the process of extracting hidden information and drawing connections between streams of raw, unstructured data that may originate from any sensing, personal computing or enterprise computing device. Analysis of this flood of information, known as Big Data, has become a cornerstone for applications of immense societal and economic impact, including the fields of healthcare, environmental monitoring, telecommunications, cybersecurity, business intelligence, and the capital markets. Real-time analytics is currently conducted in datacentres. Datacentres must store data streams that are generated at unprecedented rates and process these data streams to quantify known properties or discover unknown properties of data.

While striving to meet the high demand for storage and data processing, datacentres incur high operational expenditure costs primarily because they consume inordinate amounts of energy. For example in 2012, Google datacentres were estimated to consume more than 300 MegaWatts and Facebook datacentres more than 60 MegaWatts. Datacentres worldwide use about 30 GigaWatts. In addition to a huge cost and carbon footprint, these centres are notoriously wasteful – often using as little as 10% of the supplied power for actual data storage and processing - and raise major environmental concerns.

The NanoStreams project [1] designs and implements a new architecture for micro-servers and system software that jointly address the performance challenges of real-time data analytics, with a low and sustainable hardware energy footprint. The NanoStreams micro-servers combine the benefits of low-power ARM processors with application-specific, programmable accelerators to achieve high performance and energy-efficiency. The NanoStreams software stack leverages streaming and dynamic execution models in conjunction with data access locality optimisations to sustain high throughput and low latency during analytical processing of event streams.

In this paper we demonstrate the most important aspects of the NanoStreams hardware and software ecosystem using one of the project's use cases, namely pre-trade risk analytics in the financial markets. Processing substantial amounts of price and trade volume data per second with microsecond-level response times, leads to more effective risk management and liquidity provision and thereby to a more efficient financial market. High-end European derivatives markets can, for example, maintain bandwidth saturation for over thirty UDP multicast channels throughout a trading day, requiring more than twenty CPUs dedicated to processing the data feed. In order to reduce network latency, these CPUs are generally sited by financial trading organizations as close to the exchange or other trading venue as possible. This need is served by various managed hosting service companies which offer space, power and high speed network connectivity within their collocated data centres. Data centre provision is in itself a highly profitable business with some analysts predicting a compound annual growth rate within the UK of 14% over the period 2014-18 [2]. Micro-servers can play a significant role in this context by reducing total cost of ownership.

## 2. Objectives

Micro-servers aim at reducing the carbon footprint of datacentres by replacing high-performance server-class processors and memories with low-cost and low-power alternatives that are that widely used in mobile embedded systems such as smartphones and tablets. Back-of-the-envelope calculations suggest that micro-server processors such as the Samsung Exynos and Intel's Ivy Bridge consume 5 to 30 times less power than server-class counterparts, such as Intel's Sandy Bridge, and are 5 to 15 times more energy-efficient in integer performance per Watt[3]. However, these micro-server processors are also limited in floating point performance and SIMD/SIMT acceleration capabilities.

NanoStreams bridges the performance gap between low-power smartphone-class processors and high-end server processors by enhancing a micro-server SoC based on the ARM Cortex multi-core architecture with application-specific, energy-efficient, and programmable accelerators. Specifically, the project is building a heterogeneous micro-server with a host ARM-based SoC node and an analytics accelerator SoC node, the latter built on reconfigurable fabric. The two nodes can match the performance of a 170-Watt server-class processor, in a power envelope of less than 10 Watts. The analytics accelerator SoC is based on nanocores, an extremely lightweight and power-efficient core design template developed by Analytics Engines Ltd. Nanocores are based on a dataflow design methodology for reconfigurable fabrics, which can pack up to 250 64-bit nanocores or 650 32-bit nanocores, within the FPGA space available in the Xilinx Zynq SoC platform. Further performance scaling to match the performance of high-end servers is achieved by adopting a scale-out approach where multiple micro-servers are densely replicated.

The research and development activity in NanoStreams is driven at all stages by input from the community of end users who deploy server solutions today and who would ultimately deploy the proposed micro-server architectures. Our software substrate is co-designed with the micro-server SoCs and enables multiple case studies within the lifetime of the project, including in addition to the risk analytics described in this paper: high-performance implementation of sparse linear algebra operations (unlike the dense linear algebra routine in the well-known LAPACK and MAGMA libraries), multi-threaded query execution for in-memory column databases and a real-time platform for correlating ICU patient respiratory and ECG data and predicting near time clinical events. By offering faster streaming analytics, we hope to unlock some of the hidden patterns in the vast datasets generated by each application domain.

## 3. Pricing Financial Options

Figure 1 illustrates the components in our stack referencing a number application domains. Our novel, holistic hardware-software co-design methodology combines and also links optimizations within each of the boxes in Figure 1 to deliver a system solution. In this paper, we illustrate the NanoStreams stack for real-time analytics by describing in detail the implementation of pricing European vanilla options with it.

A financial option is a derivative product which is a contract giving the holder the right to either buy (Call) or to sell (Put) one or more underlying assets, such as a fixed number of shares in a company, for a defined price and either on or before a contract end date. Options are distinguished by the different terms in their contracts. In this paper we consider only the European vanilla type, which means that the contract can only be exercised on the end date.

The price of an option on any given date is commonly modelled using stochastic calculus but in general analytical solutions for the resultant equations are not possible. A variety of computational numerical solution methods have been developed instead. Monte Carlo techniques are widely used in this context [4].

The European vanilla option is an exception because its price can be expressed by an analytical solution to a partial differential equation, known as the Black-Scholes model [5], which assumes that the underlying asset price (spot price) follows a log normal distribution and furthermore that both the volatility of the underlying spot price and the risk free rate of return are constant. Monte Carlo methods can still be used for, and tested with, European vanilla options so that an implementation then forms the basis for moving beyond the restrictions of the Black-Scholes model and can be developed further to handle the so-called exotic option types. In this work, we choose options on the stock of a single company and compute prices for put contracts and for call contracts using the Monte Carlo solution method. This problem is embarrassingly parallel in the sense that each option contract on each stock is an independent mathematical calculation.

## 4. Technology Description

The Analytics-on-Chip (AoC) substrate, represented by the lowest layer boxes in Figure 1, is implemented in our prototype system by a heterogeneous host system, a Boston Viridis server, composed of Calxeda EnergyCore cards and a Xilinx Zynq SoC board for instantiation of the accelerated compute kernels, which we call *nanocores*. The chosen operating system is Linux and Ethernet networking is used between the AoC engines allowing sharing of accelerators in scale-out systems. The financial risk computation discussed in this paper consists of several executable binaries, representing the involvement of the layer labelled “C” and the layer labelled “ACE Cosy parameterized compiler”, as well as all boxes below these, on both sides of the figure. The left hand side corresponds to the running on the host while the right hand side is for the accelerated nanocores. For clarity, we point out that in this paper there is no engagement with the top three layers of Figure 1.

In the NanoStreams prototype system stock price data is first extracted over the Internet from the Yahoo finance server and then distributed among our application code over a pair of UDP multicast channels. The OptionPricer program is the computational workhorse of the system. It attaches to the multicast channels and receives stock price data. It then executes a Monte Carlo pricing model for all contracts defined on each stock. The OptionPricer implements a multithreaded work queue model of processing enabled by the

POSIX pthreads library. Since a multicast group is used to distribute the stock prices, multiple instances of OptionPricer can easily be executed on the same or on different nodes, thereby providing a scalable computation system.

A complex scenario arises when two or more programming components (e.g. third party libraries), based on threads, are used simultaneously, whether within the same binary or in concurrently executing binaries on one OS. In this scenario, the components would independently create thread pools and attempt to maximise thread-level parallelism to handle

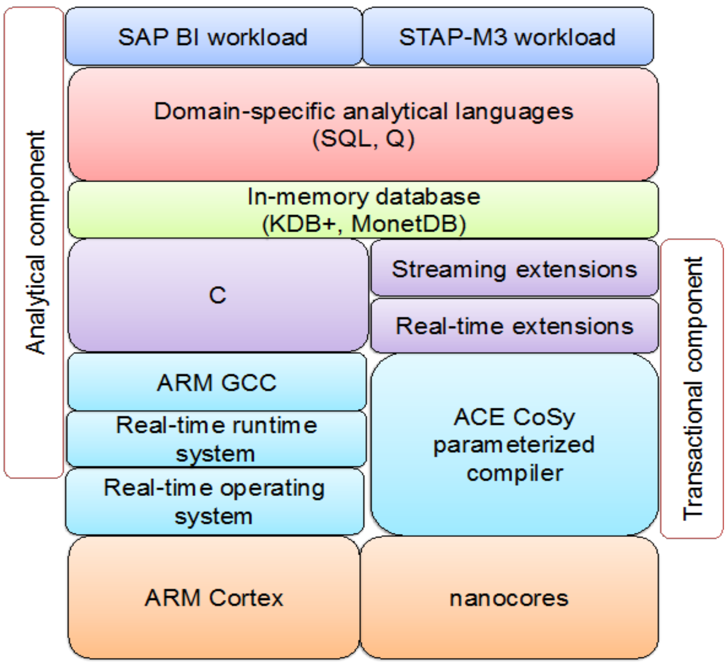

*Figure 1. The NanoStreams software stack*

their respective workloads. This behavior is likely to incur oversubscription of resources and contention. In reality, only a subset of the active threads can be executed simultaneously without incurring unnecessary overhead. This problem arises due to the fact that each library independently requests threads to optimise its individual performance, while overlooking the requirements of other libraries for hardware resources. Various approaches, such as Lithe[6], Concert [7] and Manticore[8], propose solutions to this problem but which generally involve modifying source code, an approach which is unattractive with large legacy code suites. NanoStreams has addressed this by incorporating a new runtime scheduler based on interposition techniques, as indicated on the left hand side of Figure 1.

## 5. Developments

The NanoStreams prototype system embraces state of the art aspects, such as the use of UDP multicast channels and the libevent library [9], but also has a number of novel features which are described briefly in the following sections.

### *5.1 Run-time thread scheduling*

The NanoRuntime implements the NanoStreams user-level scheduler, as a thin user-level runtime layer sitting between application libraries and the operating system. The NanoRuntime intercepts calls to the pthreads library and is invisible to the application programmer. The runtime, which assumes only vanilla Linux support for user contexts as a baseline, follows a strict 1:1 mapping of active application threads to hardware contexts and enforces spatial isolation of each group of threads that originates from the same library, to achieve good data access locality. Proportional allocation of resources to libraries is achieved through a combination of proportional allocation of cores and time epochs for execution on those cores, to each library.

### *5.2 Programming the accelerator fabric*

One of the barriers to the widespread adoption of accelerator hardware in HPC applications remains the issue of programmability. Each type of accelerator has a unique programming environment requiring investment from users to rewrite code. Nanocores are a new class of programmable, reconfigurable accelerators, which aim to improve analytical processing throughput by exposing the parallel computational capabilities of the underlying hardware to the software domain, allowing rapid, run-time reconfiguration. The possibility for run-time reconfiguration will extend from the software executing on an individual core to the interconnection between cores.

To facilitate the creation of nanocores, the project has developed a methodology to identify hotspot kernels, down to the level of sets of instructions, with a view to migrating only these instructions to a network of customised nanocores implemented in the off-load accelerator. An assembly language has been defined and an assembler and linker tool created. A novel feature of the assembly language is that it is defined using Haskell terms to allow rapid prototyping. Using the ACE Cosy parameterized compiler tool chain, a compiler for the C language has been created targeting the assembly code of each nanocore.

### *5.3 Scalable low latency communications channels to the accelerators*

Fundamental to realising the performance gain of the nanocores has been the provision of a low latency communications channel between the host and accelerator. The project has pursued a loosely-coupled approach that offers certain important advantages by allowing:

- addition of an accelerator to any server without affecting the host processor architecture.
- creation of customized accelerators for narrow application domains as they do not affect processor manufacturing volume.

When connecting an accelerator to the host processor via a system level interconnect, the typical path today is to place the accelerator on the PCI-E interconnect, such as with GPUs. PCI-E has an important role for systems where a large number of third-party peripherals may be connected to general purpose servers. However with energy efficient systems this may not be necessary or possible due to the fact that such systems are more tightly packaged, eg. by placing all I/O on the same chip as the processor. With this in mind, the project has chosen to explore an Ethernet-based physical layer and protocol, as the interconnect between the host and the accelerator. In particular, data packets are moved from user space to the physical layer without the need to go via layer 3 or 4 of a conventional IP stack effectively amounting to kernel by-pass. We argue that an Ethernet-based approach exhibits a number of advantages:

- *Market scale*: It is mature and widely used in the datacenter.
- *Transparency*: It supports diverse topologies and physical media.
- *Performance*: It can achieve high throughput.
- *Elasticity*: It allows building racks that consist of general-purpose servers and accelerators, where resources are provisioned dynamically and thereby can achieve high

utilization. Depending on the needs of the workload, it allows a single server to use multiple accelerators and a single accelerator to be shared by multiple servers.

## 6. Results

We present initial results on the evaluation of power consumption for one part of the NanoStreams system in this section. The rate limiting part of the option pricing computation is a lengthy for-loop, an aspect that it shares in common with HPC applications in many fields. For a Put option the formula is [4]

$$\text{Option price} = \frac{e^{-rT}}{N} \sum_{i=1}^{N} \text{Max}\left(0, P - S\, e^{\left(r-\frac{\sigma^2}{2}\right)T + \sigma\sqrt{T}\, x_i}\right) \quad (1)$$

where P is the strike price in the option contract, S is the current (spot) price of the underlying stock, r is the annual risk free rate, σ the volatility and T is the time in years to expiry. The $x_i$ are a set of N normally distributed random numbers. The Monte Carlo method exhibits slow convergence, O(1/sqrt(N)) thereby requiring the use of large values for N in general.

*Table 1. Measured times and node power for option pricing on the Calxeda node.*

| Mode | N | Elapsed seconds | Mean time per option (s) | Options priced/sec | Avg.system power (W) |
|---|---|---|---|---|---|
| Performance | 500000 | 41.22 | 0.067 | 14.97 | 6.8 |
| | 1000000 | 82.64 | 0.134 | 7.46 | 6.8 |
| | 2000000 | 163.67 | 0.265 | 3.77 | 7.4 |
| Powersave | 500000 | 342.96 | 0.556 | 1.79 | 4.3 |
| | 1000000 | 663.46 | 1.075 | 0.93 | 4.5 |
| | 2000000 | 1322.86 | 2.144 | 0.47 | 4.4 |

We executed the code implementing equation 1, on a 4 CPU (1.4 GHz) Calxeda node, to price 617 option contracts defined on the Facebook stock, using four compute threads. The instantaneous power consumption was obtained from on-node sensors thereby allowing computation of the average power consumption for the full computation. This was repeated in two operational modes. In Powersave mode the active CPUs run at the lowest frequency, while in Performance mode all CPUs run at maximum frequency, Table 1 presents the data for different values of N in equation 1. We repeated the computations on an Intel Xeon based server (2.5 GHz) and Table 2 presents the corresponding results.

Multiplying the average system power by the elapsed time, for each result, gives an estimate for the total energy in Joules consumed in the computation. In both tables it is apparent that increasing the value of N from 500,000 to 2,000,000 (which would improve the numerical convergence by a factor of two) requires four times the energy (eg. in Table 1 performance mode - 280 J versus 1211 J). The Powersave mode is not energy efficient for this problem, on either platform, due to the lengthy computation times. For example in Table 2, for the largest N, Powersave mode consumes 6881J while Performance mode uses 3582J to complete the same calculation.

Using profiling tools we established that the most of the computation time was spent in the pseudo-random number generator closely followed by the exponential function. A highly tuned nanocore generating random numbers, consuming approximately 150mW, was subsequently created and instantiated on the attached accelerator. This led to an approximate reduction of 20% in the compute time on the Calxeda node.

*Table 2 Measured times and system power for option pricing on an Intel Xeon server using four threads*

| **Mode** | **N** | **Elapsed seconds** | **Mean time per option (s)** | **Options priced/sec** | **Avg.system power (W)** |
|---|---|---|---|---|---|
| Performance | 500000 | 8.602 | 0.014 | 71.724 | 99.4 |
| | 1000000 | 17.404 | 0.028 | 35.450 | 102.8 |
| | 2000000 | 34.409 | 0.056 | 17.931 | 104.4 |
| Powersave | 500000 | 18.408 | 0.030 | 33.518 | 91.4 |
| | 1000000 | 37.010 | 0.060 | 16.671 | 92.2 |
| | 2000000 | 74.230 | 0.120 | 8.312 | 92.7 |

An individual Calxeda node on the low-power Viridis micro-server is not designed to outperform one Intel server node and this is clear from comparing the two tables above, where an approximate 5:1 speed-up ratio is evident in the elapsed timings in. However, the embarrassingly parallel nature of contract pricing with one underlying stock permits several Calxeda nodes to be applied concurrently to the problem. An amalgam of Calxeda nodes could reduce the elapsed compute time to match or better the performance of the Intel server node reported in Table 2 and, even allowing for some additional energy consumption for inter-node networking, remain within a similar 100W power budget.

## 7. Business Benefits

Efficient computation of option contract prices enables an efficient market. The embarrassingly parallel nature of option pricing workloads lends itself to massive hardware scalability. However even leaving aside the quite considerable challenges in server resource provisioning, scheduling and management inherent in building ever larger data centres, satisfying the demand for compute resource is constrained by the operational constraints on power consumption.

The outputs from NanoStreams will lead to improved understanding of the characteristics of heterogeneous computing systems processing big data in real-time, not just in computational finance. The Big-Data problem is now unquestioned [10] and it is widely recognized that successfully managing big data can be a game-changing asset in any industry [11]. NanoStreams already catalyses specific product development efforts in trading risk analytics and in other application domains, driven by three SMEs (ACE, Neueda, Analytics Engines Ltd) involved which are already developing new products and services which will be offered to service the market for handling big-data.

More generally, a significant output from Framework programmes run by the European Commission has been the creation of a world leading ecosystem for high performance embedded system technology in Europe. Our project builds upon this and offers the industrial partners within the project, including the three SMEs that provider disruptive technology to the market, and two larger global corporates (IBM, Credit Suisse), to gain significant competitive advantage. The project is both sustaining and creating new jobs within all the partner organizations involved.

## 8. Conclusions

We have demonstrated key aspects of the novel stack for real-time analytics being developed by the NanoStreams consortium project and have reported preliminary results for the pricing of European vanilla options. There is nothing in the design of our system that prevents its extension to price so-called exotic options, such as American options or multi-asset barrier options. Our co-designed software stack provides elastic and scale-free co-execution of parallel computational financial application components ameliorating, in part, several of the challenges involved. We envisage in the longer term that the technology that we have developed would provide an engine sitting behind various domain specific languages, interfaces and tools that that are being currently being developed [12].

More generally, by analogy to the era of killer micros that replaced mainframes, our project envisages killer micro-servers that may be used in groups with scale-out topologies to replace larger, less energy-efficient servers. We aim to build micro-servers that are immediately programmable by the analytics domain experts, and run legacy codes with little or no intervention from software developers. We aim to demonstrate a micro-server implemented in real silicon, running realistic workloads for credit risk, ICU patient monitoring and business intelligence analytics.

NanoStreams delivers world leading multi-disciplinary research in server computing technology to the European Community, combining this with the latest research developments in high performance computing and embedded systems. Furthermore, our project resonates strongly with the wider aims and objectives of the FP7 and Horizon2020 programmes. The exploitation opportunities from the project are numerous, embracing new product and service development that is not only innovative, but which can be brought to wider markets in faster waves than previously possible.

## Acknowledgements

The NanoStreams project has received funding from the European Union’s Seventh Framework Programme for research, technological development and demonstration under grant agreement no. 610509.

## References

[1] For full details on NanoStreams see http://www.nanostreams.eu

[2] The Data Center Colocation Market in the UK 2014-18, TechNavio, available on the web at http://www.technavio.com

[3] E. Blem, J. Menon, and K. Sankaralingam, Power struggles: Revisiting the RISC vs. CISC debate on contemporary ARM and x86 architectures, *High Performance Computer Architecture (HPCA2013), 2013 IEEE 19th International Symposium on Computer Architecture*, pp.1-12, 23-27 Feb. 2013 doi: 10.1109/HPCA.2013.6522302

[4] P. Boyle, M. Broadie and P Glasserman, Monte Carlo methods for security pricing, J. Econ Dynamics and Control, vol 21 (1997), 1267-1321

[5] Fischer Black and Myron Scholes, The Pricing of Options and Corporate Liabilities, J. Political Economy, vol. 81, no 3, pp 637-54, 1973.

[6] H. Pan,B. Hindman and K Assanoviċ, Lithe: Enabling Efficient Composition of Parallel Libraries, Proceedings of the First ISENIX Workshop on Hot Topics in Parallelism, HotPAR 09, available on the web at: https://www.usenix.org/legacy/events/hotpar09/tech/

[7] V. Karamcheti and A. A. Chien. A hierarchical loadbalancing framework for dynamic multithreaded computations. In *Supercomputing*, Orlando, FL, November 2002

[8] M. Fluet, M. Rainey, and J. Reppy. A scheduling framework for general-purpose parallel languages. In *International Conference on Functional Programming*, Victoria, British Columbia, Canada, 2008.

[9] N. Matthewson, Fast portable non-blocking network programming with Libevent, available on the web at http://www.wangafu.net/~nickm/libevent-book/TOC.html

[10] J. Manyika, M. Chui, B. Brown, J. Bughin, R. Dobbs, C. Roxburgh, and A.H. Byers. *Big data: The next frontier for innovation, competition and productivity*. Technical report, Global Institute, June 2011.

[11] The Economist Intelligence Unit, Harnessing Big Data, A game Changing Asset, 2011, available on the web at http://www.sas.com/resources/asset/SAS_BigData_final.pdf

[12] S.S Goth, M. Siering and P. Gomber, *How to enable automated trading engines to cope with news-related liquidity stocks*, Decision Support Systems, Vol 62, p 32-42, 2014, doi: 10.1016/j.dss.2014.03.002