BioThreads: A novel VLIW-based Chip-Multi-Processor for accelerating biomedical image processing applications

D. Stevens, A. Echiadis, V. Azorin-Peris, V. Chouliaras, J. Zheng and S. Hu

Abstract: *We discuss BioThreads, a novel, configurable, extensible System-on-Chip multi-processor and its use in accelerating biomedical signal processing applications such as imaging PhotoPlethysmoGraphy (IPPG). BioThreads is derived from the LE1 open-source VLIW core and efficiently handles Instruction, Data and Thread-level parallelism; In addition, it supports a novel mechanism for the dynamic creation, and allocation of software threads to uncommitted processor cores by implementing key POSIX Threads primitives directly in hardware. In this study, the BioThreads core is used to accelerate the calculation of the oxygen saturation map of living tissue in an experimental setup consisting of a high speed image acquisition system, connected to an FPGA board and to a host system. Results demonstrate near-linear acceleration of the core algorithms of the target blood perfusion assessment. with increasing number of cores. The BioThreads processor was implemented on both standard-cell and FPGA technologies; In the first case, full real-time performance is achieved with UU cores whereas on a mid-range Virtex6 device near-real-time performance is achieved with KKK cores*

Introduction and Motivation

Real-time execution of biomedical signal processing codes in *in-vitro* and *in-vivo* assessment is a key capability which allows clinicians to make important decisions and perform medical interventions safely, accurately and quickly as these are based on hard facts, derived in real-time from physiological data [1, 2]. In the area of Biomedical Image Processing, a number of imaging methods have been proposed over the past few years including laser Doppler [3], optical coherent tomography [4] and more recently, imaging PhotoPlethysmoGraphy (IPPG) [5, 6]; However, none of these techniques is able to attain its true potential without a real-time biomedical image processing system based on *Very Large Scale Integration* (VLSI) technology. This is an area where advanced computer architecture concepts, routinely utilized in high-performance consumer and telecoms Systems-on-Chip (SoC), [7] can provide the required data streaming and execution bandwidth to allow for the real-time execution of algorithms that would otherwise be executed not in real time using more established techniques and platforms (sequential execution on a PC host).

Such SoC-based architectures typically include scalar embedded processor cores with a fixed Instruction-Set-Architecture (ISA) which are widely used in standard-cell (ASIC) [8] and reconfigurable (FPGA)-based embedded systems [9]. These processors present a good compromise for the execution of general-purpose code such as the user interface, low-level/bandwidth protocol processing, the embedded Operating System (eOS) and occasionally, low-complexity signal processing tasks. However, they lack considerably in the area of high-throughput execution and high-bandwidth data movement as often is required by the core signal processing algorithms in most application domains. An interesting comparison of the capabilities of three such scalar engines targeting Field-Programmable technologies (FPGAs) is given in [10].

To relieve this constraint, scalar embedded processors have been augmented with DSP coprocessors in both tightly-coupled [11] or loosely-coupled configurations [12] to target performance-critical inner loops of DSP algorithms. A side-effect of this approach is the lack of homogeneity in the SoC platform programmer's model which itself necessitates the use of complex 'mailbox-type' [13] communications and the programmer-managed use of multiple address spaces, coherency issues and DMA-driven data flows, typically under the control of the scalar CPU.

Another architectural alternative is the implementation of the core DSP functionality using custom (hardwired) logic. Using established methodologies (register-transfer-level design, RTL) this task involves long development and verification times and results in systems that are of high performance yet, they are only tuned to the task at hand. Also, these solutions tend to offer little or no programmability making difficult their modification to reflect changes to the input algorithm. In the same architectural domain, the synthesis of such hardwired engines from High Level Languages (ESL synthesis) is an area of active research in academia [14][15] (academic efforts targeting ESL synthesis of Ada and C-descriptions); Industrial tools in this area have matured [16][17][18] (commercial offerings targeting C++, C and UML+C++ respectively) to the point of competing favourably with hand-coded RTL implementations, at least for certain type of designs [19-BDTI_study].

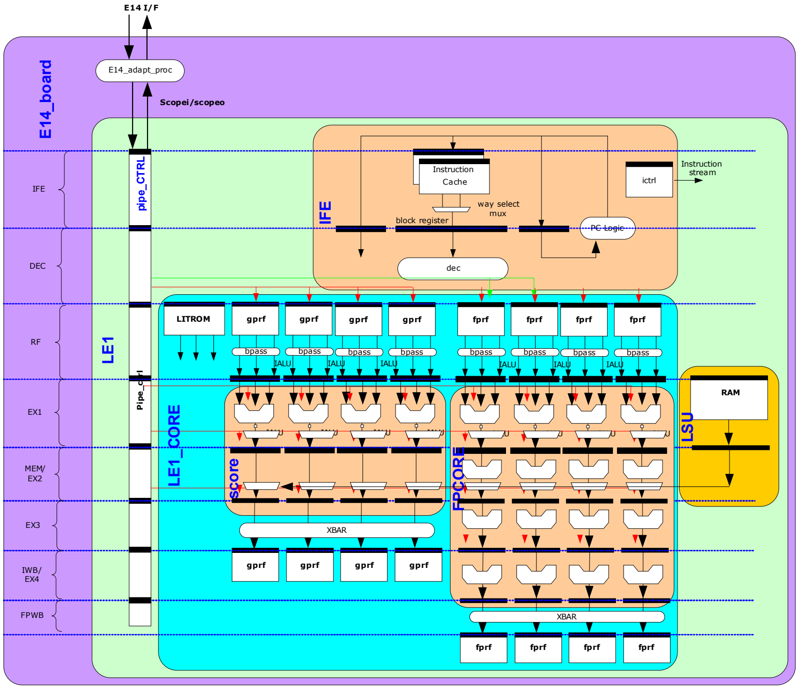
A potent solution to high performance VLSI systems design is provided by *configurable, extensible processors* [20]. These CPUs allow the extension of their architecture (programmers model and ISA), and microarchitecture (execution units, streaming engines, coprocessors, local memories) by the system architect. They typically offer high performance, full programmability and good post-fabrication adaptivity to evolving algorithms through the careful choice of the custom ISA and execution/storage resources prior to committing to silicon. High performance is achieved through the use of custom instructions which collapse Data Flow Graph (DFG) sub-graphs (especially those repeated many times [21]) into a single instruction node. At the same time, these processors deliver better power efficiency compared to non-extensible processors, via the reduction in the dynamic instruction count of the target application and the use of streaming local memories instead of data caches.

All the solutions to developing high-performance digital engines for consumer and in this case, biomedical image processing, mentioned so far suffer from the need to explicitly specify the *software/hardware interface* and schedule communications across that boundary. This work proposes an alternative, all-software solution, based on a novel configurable, extensible VLIW chip-multi-processor (CMP) based on an open-source VLIW core [22, 23] and targeting both FPGA as well as standard-cell (ASIC) silicon. The VLIW paradigm was chosen as VLIW architectures are very potent engines in exploiting instruction-level (ILP)and data-level (DLP) parallelism, the former via the static (compile-time) specification of independent RISC-ops (refered to in this text as a 'syllable' or RISCop) per VLIW instruction and the former, via the automatic unrolling and pipelining of inner loops. Key to this is the use of advanced compilation technologies such as Trimaran [24] for fully-predicated EPIC architectures or VEX [25] for the partially-predicated BioThreads core used in this work. At the same time, the extensibility of the core LE1 engine permits the shifting of the hardware/software interface, exchanign performance for silicon area and always supporting full programmability.

The key contributions of this work are summarized as follows: a) we have developed a highly configurable, extensible, Chip-multi-processor based on an open-source VLIW CPU, capable of performing PThread primitives directly in hardware. This is a unique feature of the LE1 (and *BioThreads*) processors and uniquely differentiates them from other key research such as hardware primitives for remote memory access [26] b) We advocate the use of such a complex processing engine in the biomedical signal processing domain such as the real-time blood perfusion and demonstrate the effectiveness of our approach when computing key algorithms. c) We have developed a unified, software-hardware development flow in which all algorithm development takes place in MATLAB, followed by automatic C-code generation and its introduction to the LE1 toolchain. This is a well encapsulated process which ensures that the biomedical engineer is not exposed to the detail of real-time software development for a complex SoC platform; at the same time this methodology results in a working embedded system directly implementing the algorithmic functionality specified in the MATLAB input description. In the process, use of the best computer architecture practices has taken place and a novel hardware mechanism has been researched and developed which allows the exploration of thread-level (TLP) parallelism via supporting key PThread primitives directly in hardware. In that respect, the BioThreads core can be thought of as a combination of OS and processor, delivering services (execution bandwidth and thread allication) to a higher order system and enabling the real-time execution of compute-bound biomedical signal processing codes. The very high scalability of the proposed platform results in near-real-time performance when computing the XYZ.

The BioThreads CMP

The Biothreads CMP extends the LE1 open-source processor with execution primitives to support high speed image processing and dynamic thread allocation and mapping to uncommitted CPU cores. This section discusses briefly these key microarchitectural features.



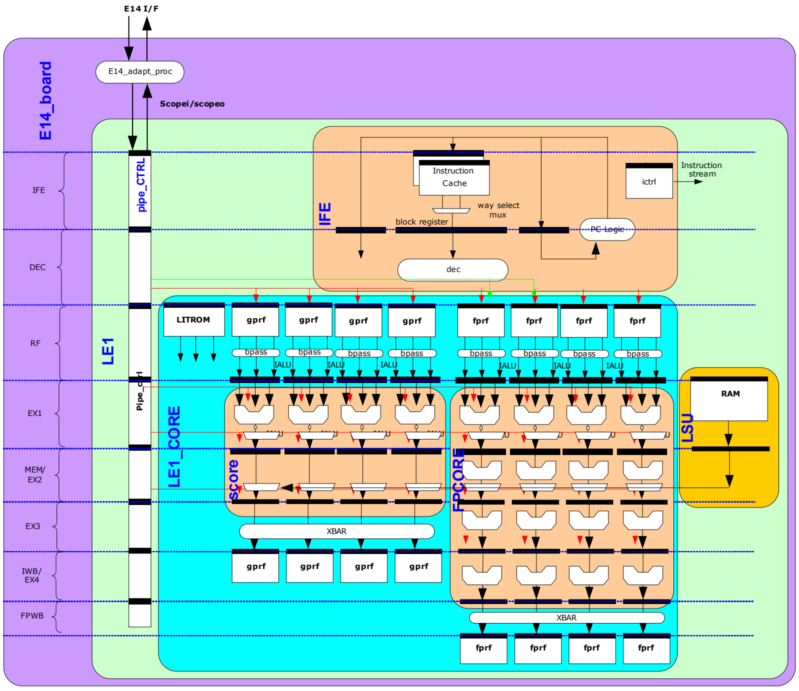


Figure 2: LE1 Core

Fig. 1 depicts the basecase 8-stage pipelined LE1 processor core which is the heart of the BioThreads CMP:

- The CPU consists of the Instruction Fetch Engine (IFE), the execution core (LE1_CORE), the pipeline controller (PCTRL) and the Load/Store Unit (LSU). The IFE can be configured with an instruction cache or alternatively, a closely-coupled instruction RAM (IRAM). These are accessed every cycle and return a long instruction word (LIW) consisting of multiple RISCops for decode and dispatch. The IFE controller handles interfacing to the external memory for ICache refills, and stall control whenever an LIW spans two IRAM locations. It also provides debug capability into the ICache/IRAM. The IFE can also be configured with a branch predictor unit, currently based on the 2-bit saturating counter scheme (Smith predictor).
- The LE1_CORE includes the main execution datapaths of the CPU. There are a configurable number of clusters, each with its own register set. Each cluster includes an integer core (SCORE) and optionally, floating point (FPCORE) core. The figure depicts a single-cluster configuration with a 4-wide SCORE and a 4-wide FPCORE. The integer and floating-point datapaths are of unequal pipeline depth however, they maintain a common exception resolution point.
- PIPE_CTRL is the primary control logic. It is collection of interlocked, pipelined state machines,

which schedule the execution datapaths and monitor the overall instruction flow down the processing and memory pipelines. PIPE_CTRL maintains the control registers of the CPU and handshakes the host during debug operations.

- Finally, the LSU is the primary path of the LE1_CORE to the system memory. It allows for up to ISSUE_WIDTH memory operations and directly communicates with the shared data memory (STRMEM). The latter is a multi-bank, cross-bar solution and scales reasonably well (in terms of speed and area) for up to 32 memory clients. This is depicted in Fig. 2:

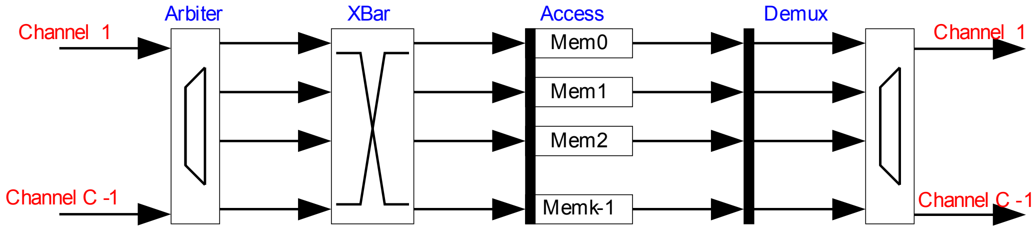


Figure 2: Streaming Memory System of the BioThreads CMP

Finally,

to allow for the exploitation of shared-memory TLP, multiple processing cores can be instantiated in CMP configuration as shown in Fig. 3. The figure depicts a dual-LE1 BioThreads system interfacing to the common streaming data RAM.

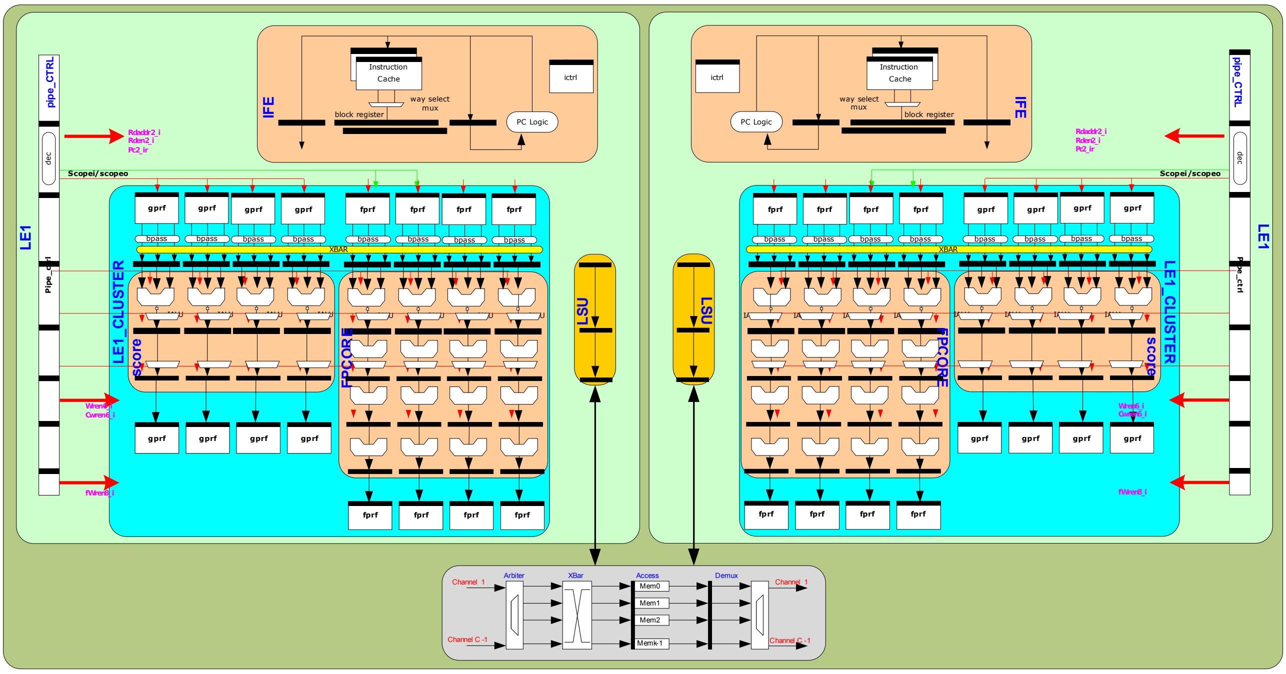


Figure 3: dual shared-memory BioThreads CMP

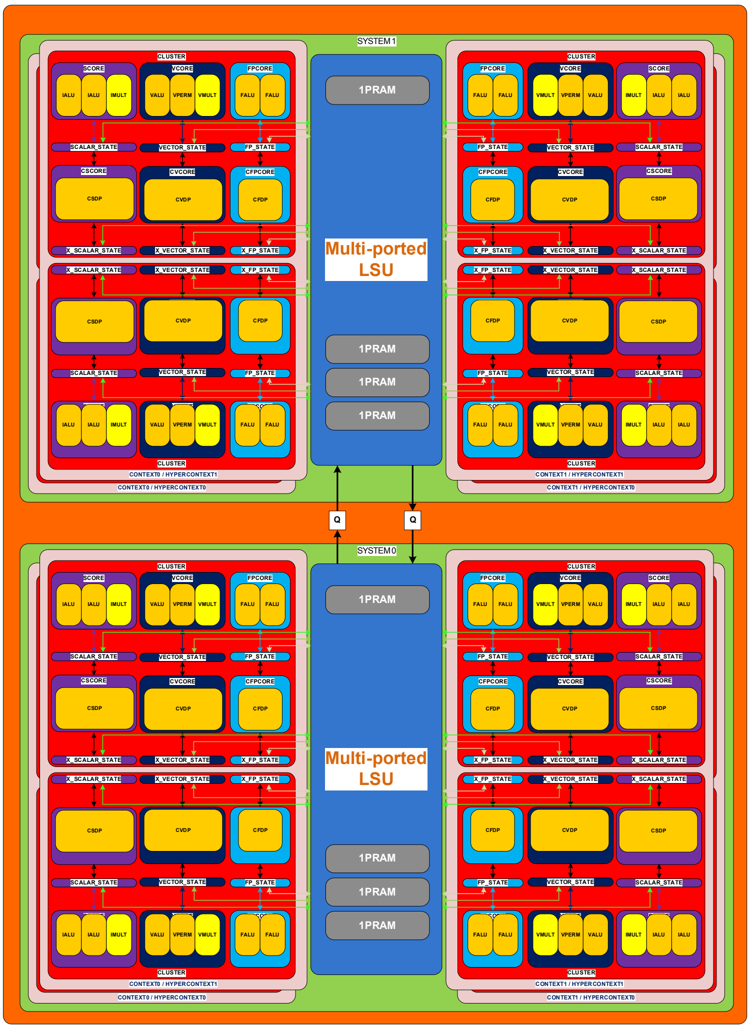


Figure 4: BioThreads engine in hybrid mode

Signal Processing Methodology

The application area selected to investigate the feasibility of deploying a VLIW-based CMP engine for real-time biomedical signal processing was photoplethysmography (PPG), which is the measurement of blood volume changes in living tissue using optical means. PPG is widely used primarily in Pulse Oximetry for the point-measurement of oxygen saturation. In this application, PPG is instead implemented from an area measurement. The basic concept of this implementation, termed imaging PPG (IPPG), is to illuminate the tissue with a homogeneous light source and to detect the light leaving the tissue with a 2D sensor array. This yields a sequence of images from which a map of the blood volume changes, and hence of the physiological function, can be generated for the target tissue. The use of multiple wavelengths in the light source enables the reconstruction of blood volume changes at different depths of the tissue, which can yield a 3D map of the tissue function. This is the principle of operation of the Oximap Real-Time Tomographer [27-OXIMAP]. Such functional maps have numerous applications in clinical diagnostics, including the assessment of the severity of skin burns or wounds, of cardiovascular surgical interventions and of overall cardiovascular function.

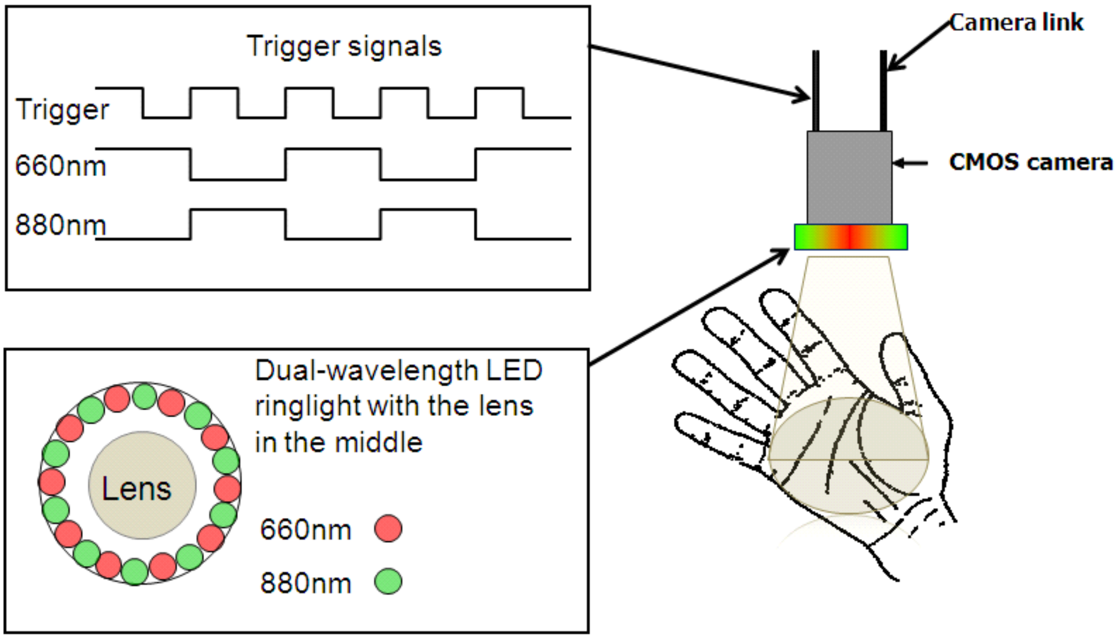


Figure 4: Schematic diagram of IPPG setup including the dual wavelength LED ringlight, lens, CMOS camera and subject hand.

A reflection–mode IPPG setup was put together for the validation experiment in this investigation, the basic elements of which are a ringlight illuminator, a lens and a high sensitivity camera [28-MIKROTRON CAMERA Mikrotron Eosens CL, Germany] as the detecting element, and the target skin tissue as defined by its optical coefficients and geometry. The introduction of the fast digital camera facilitates the non-contact measurement at a sufficiently high sampling rate to allow PPG signal reconstruction from a large and homogeneously illuminated field of view at more than one wavelength, as shown in Fig. 3. The illumination unit is synchronised with the camera acquisition system by means of a micro-controller (PIC16F876A, Microchip) in order to perform multiplexed acquisition of a sequence of images of the area of interest, exposed to red 660nm LED ([29-RED_LED], infra-red 880nm LED[30-IR_LED]. Ambient light is kept to a minimum during acquisition for optimum signal quality, By employing the same principles relating to photoplethysmography (PPG), the ultimate scope of this experiment was to detect blood volume changes in living tissue and to generate a map of blood perfusion using the BioThreads engine as the signal processing platform. This blood perfusion map was generated from the power of the PPG signal in the frequency domain. The acquired image frames were processed as follows:

a. N frames were recorded with the acquisition system (DAVID state No of frames, frame size M x N depth of resolution).

b. The average fundamental frequency of the PPG signal was manually extracted (1.4Hz).

c. Data were streamed to the BioThreads platform and the 64-point Fast Fourier Transform (FFT) of each pixel (or if clusters of pixels were used, please state size [n x m] - **FOR DAVID**) was calculated. This was done by taking the pixel values of all image frames for a particular pixel position to form a pixel value vector in the time domain.

d. The Power of the FFT tap corresponding to the PPG fundamental frequency was copied into a new matrix at the same coordinates of the pixel (or cluster) under processing. In the presence of blood volume variation at that pixel (or cluster) the power would be larger than if there was no blood volume variation.

Repeating (d) for all the remaining pixels (or clusters) provides a new matrix (or image) whose elements (or pixels) values depend on the detected blood volume variation power. This technique allows the generation of a blood perfusion map, as a high PPG power can be attributed to high blood volume variation and ultimately to blood perfusion. Figure 5 illustrates a simplified diagrammatic representation of the algorithm used and Fig. 6 shows the

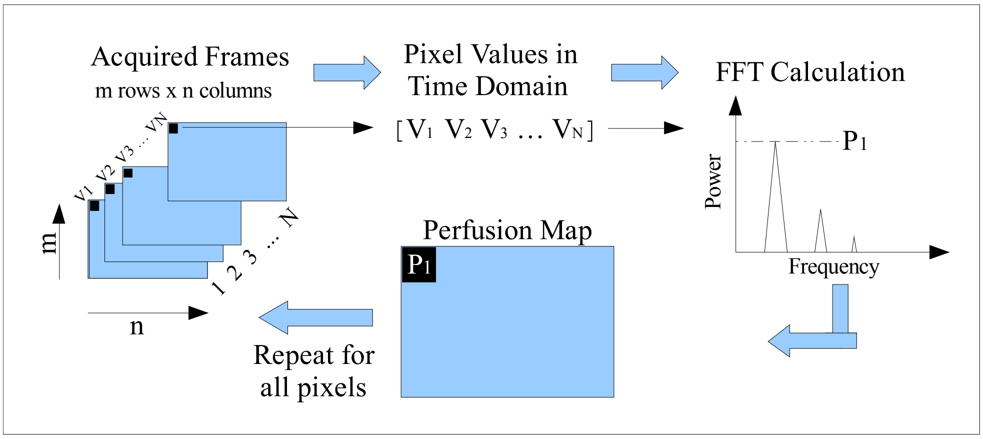


Figure 5: High-level view of the signal processing algorithm

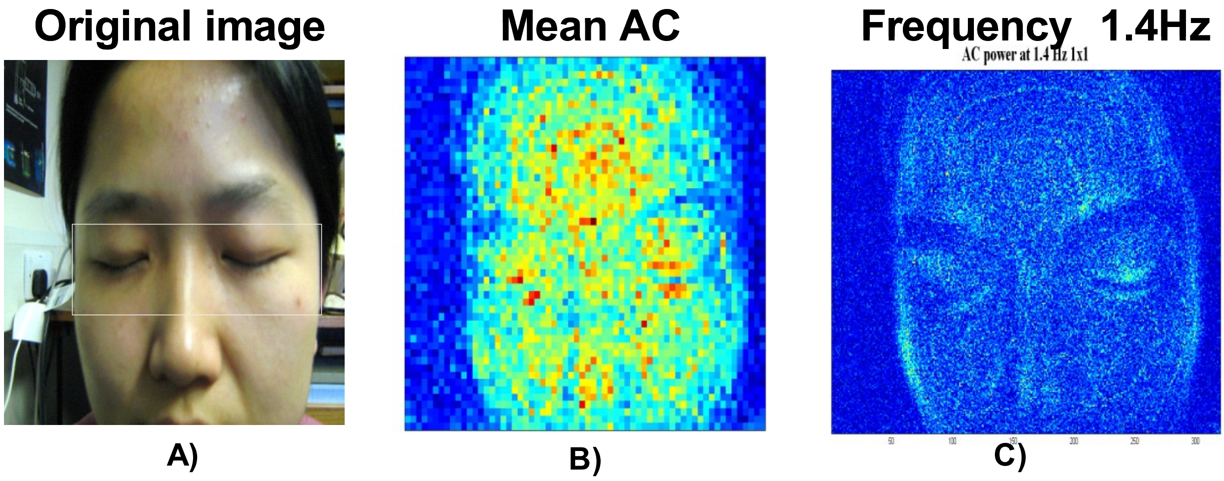


Figure 6: The original image. (b) Corresponding ac map. (c) Corresponding ac power map at HR = 1.3 Hz

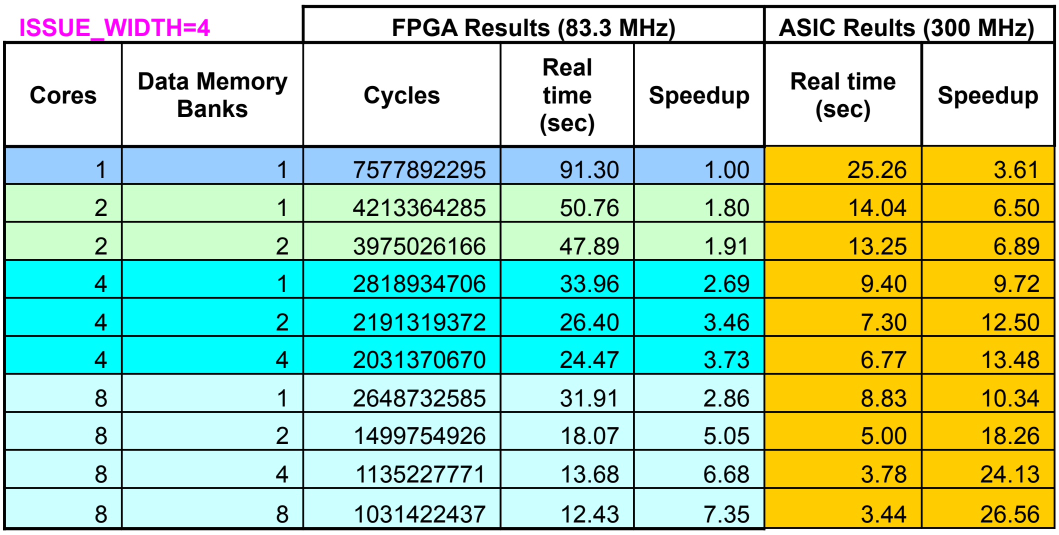
Results and discussions

This study set out to investigate the plausibility of using advanced SoC platforms for real-time biomedical assessment. This section presents the results of our study and is split into two major sections: a) Performance (real-time) results, pertaining to the real-time calculation of the oxygen map and b) SoC platform results. The latter include data such as area, maximum frequency, when targeting a Xilinx Virtex6 LX240T FG1156 [31]

FPGA and a mainsteam 0.13 um standard-cell process .

Perfromance Results

Table 1: Performance evaluation

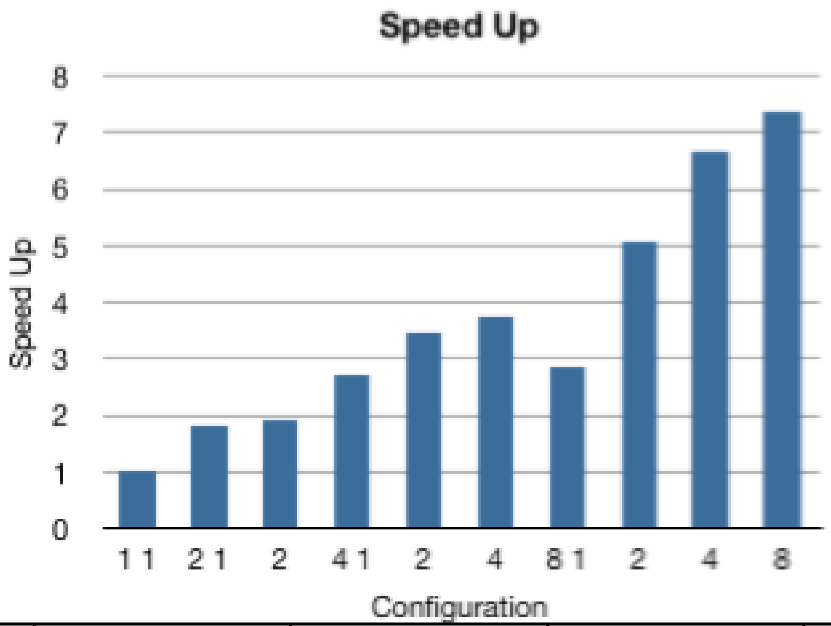


ISSUE_WIDTH=4		FPGA Results (83.3 MHz)			ASIC Reults (300 MHz)	
Cores	Data Memory Banks	Cycles	Real time (sec)	Speedup	Real time (sec)	Speedup
1	1	7577892295	91.30	1.00	25.26	3.61
2	1	4213364285	50.76	1.80	14.04	6.50
2	2	3975026166	47.89	1.91	13.25	6.89
4	1	2818934706	33.96	2.69	9.40	9.72
4	2	2191319372	26.40	3.46	7.30	12.50
4	4	2031370670	24.47	3.73	6.77	13.48
8	1	2648732585	31.91	2.86	8.83	10.34
8	2	1499754926	18.07	5.05	5.00	18.26
8	4	1135227771	13.68	6.68	3.78	24.13
8	8	1031422437	12.43	7.35	3.44	26.56

To be able to confirm or deny this, two outcomes will be examined. Firstly the execution time of the code, and finally how the use of threads has affected execution time.

The figures of interest from the simulation runs which are important to this study are the ‘Total Cycles’. This value is the number of cycles, including stalls and waits, for the entire code to run. As mentioned previously the simulator is cycle-accurate and all values output are considered valid as if the code were running on the LE1 hardware.

The execution time which is of interest is calculated using the ‘Total Cycle’ count and the theoretical speed of the LE1 processor. For this study it has been assumed that the LE1 will run at a speed of 200MHz. (The execution time is calculated by dividing the ‘Total Cycle’ count by the speed of the processor.)



[Table showing the results from the simulations.]

The data input for the study is 2 seconds worth of data, so the acquisition time for the data is 2 seconds. For the output to be considered ‘real-time’ it is required that the execution of the code can be performed in a 2 second window also. This so that with each acquisition there is an output image available for display. As can be seen in the table above [#] the execution times get faster, but none are near the 2 seconds which is the target. In order for the execution time to be 2 seconds the total cycles would need to be in the range of 400 million cycles.

It is possible to analyse the data above and extrapolate the data to work out how many threads/memory banks would be required to perform the task in 2 seconds.

The table above [#] displays the the total cycles for each configuration along with the speed up values, compared to a 1 thread with 1 memory bank configuration. As can be seen the speed up of adding extra hardware (thread number with an equal number or memory banks) shows a near linear speed up. The reason for the speed up being not as positive when there it not maximum memory banks is due to the memory conflicts which result in stall cycles and therefore greater total cycles.

[The graph above displays the speed up values.]

As can be seen in the above graph [#] and discussed earlier, the speed up when viewing memory banks equal to the number of threads is near linear. Due to this near liner speed up it is possible to extrapolate the data collected to consider the hardware required to execute this code in ‘real-time’. To calculate this configuration we take the ‘Total Cycles’ taken to execute the code with 1 Thread and 1 Memory Bank and divide by the required number of cycles. As mentioned earlier this required number of cycles is 400 Million.

7,577,892,295 / 400,000,000 = 18.944...

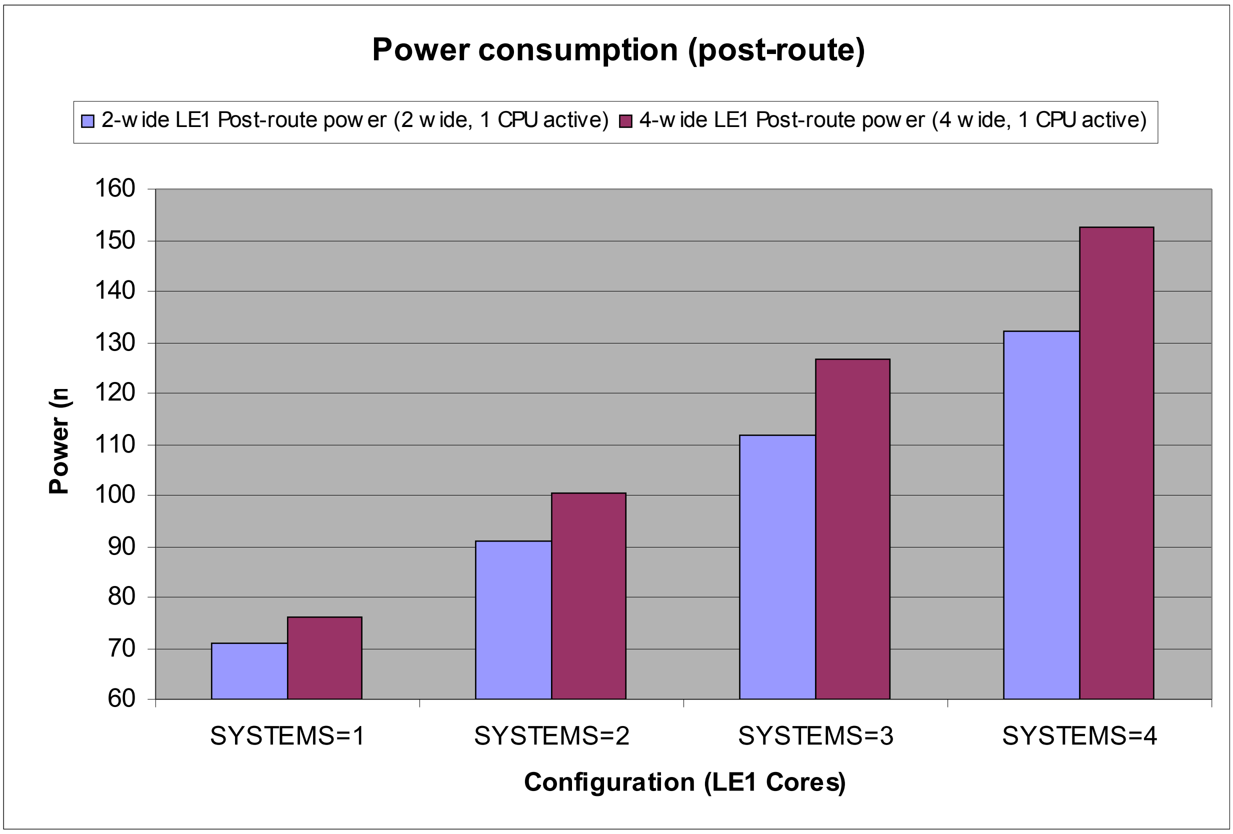
The results in the theoretical number of threads needed to be 19. Following the trend seen above this would also require 19 memory banks. This is only a theoretical value as this number of threads has yet to be examined.

VLSI Platform results

This section details the VLSI implementation of a number of configurations on both standard-cell (TSMC0.13 um 1P8M) and FPGA (Xilinx Virtex6 LX240T-FF1156-1).

Standard-cell (TSMC0.13LV process)

For the TSMC runs, 2-wide (2 ALUs, 1 Mult, 1 LSU channel) and 4-wide (4 ALUs, 1 Mult, 1 LSU channel) configurations were used as the basic building blocks of single, 2-way, 4-way and 8-way shared-memory multiprocessors. Each CPU included a private IRAM block (64K) with all configurations sharing a 128K single-bank data block.



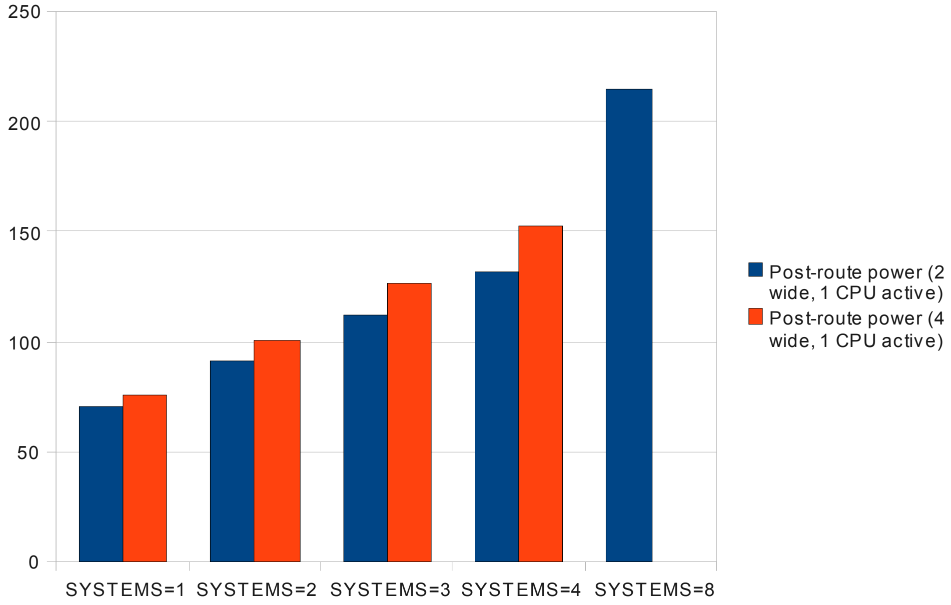


Figure 1. VLSI power consumption (statistical and post-route)

A scripting mechanism automatically modified the RTL configuration, executed RTL (pre-synthesis) regression tests, performed front-end synthesis and executed post-synthesis simulations. Following that, the internal node activity file was re-imported into the synthesis tool to produce a final post-synthesis power figure and statistical power consumption was also recorded. These steps were re-run after the synthesized configuration was automatically placed-and-routed on Cadence Encounter, with full post-route simulations (including timing and parasitics information) to produce a final power figure. In the post-route case, only one CPU was executing the application code.

Fig. 8 depicts the power consumption (both statistical and post-route) of the CPU configurations used in the campaign. The 4-wide single-CPU system consumes 6.7% more power with that number reaching 13.2% for 4-CPU systems. This asymptotic increase in power consumption is attributed to the crossbar complexity with increasing number of processors in the single-bank shared data memory. These figures are obtained with only CPU0 executing the benchmark code. No data exist for an 8-processor 8-wide multiprocessor due to synthesis issues.

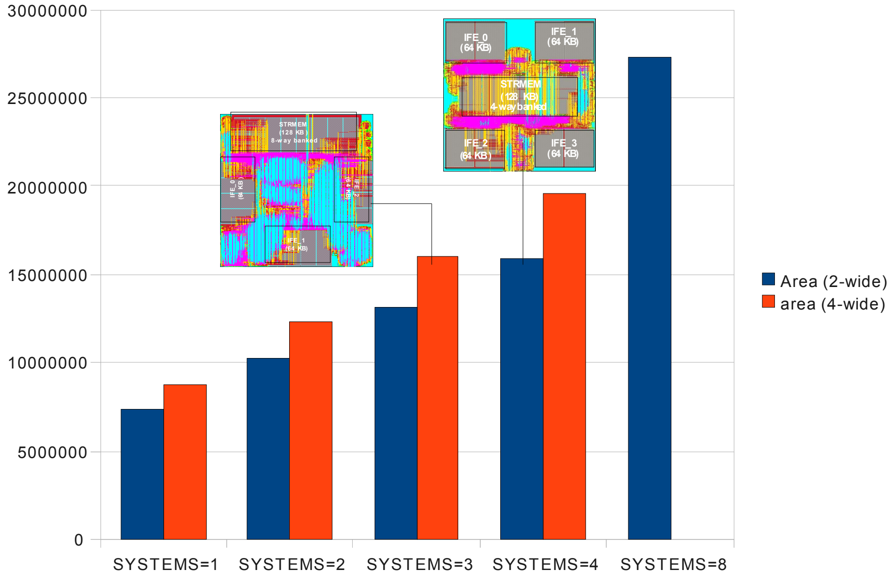


Figure 2. VLSI campaign area (um sq.)

Fig. 9 shows the post-route (real silicon) area of the configurations studied. In this case, the 4-wide is only 15.6% larger than the 2-wide processor with that number increasing to 19%. The non-linear increase in the area is due primarily to the use of the single shared memory in the multiprocessor.

Xilinx Virtex6 LX240T-FF1156-1

To provide further insight on the use of very advanced system FPGAs [VIRTEX6] when implementing a CMP platform, the BioThreads processor was re-targeted to a mid-range device of the latest Virtex6 devices from Xilinx Inc. For these configurations, the following systems were studied: 2-wide configurations: X1, X2, X4 and X8 BioThread cores, each with a private 128KB IRAM and a shared 256KB DRAM, for a total memory of 384KB, 512KB and 1024KB respectively. 4-wide configurations implemented a X1, X2 and X4 BioThread cores with the same amount of IRAM and DRAM, for the same total on-processor memory. Both 2-wide and 4-wide configurations included a Microblaze 32-bit scalar processor system, with a 32-bit single PLB backbone, to interface to the camera, stream-in frames, initiate execution on the BioThreads processor and extract the proceesed results (Oxygen map) from the CMP, upon completion. Finally, the calculated Oxygen map was returned to the host system for display purposes.

Both 2 and 4-wide configurations achieved the requested opertaing frequnecy of 83 MHz (this is the limit at which both the Microblaze platform and the BioThreads core operate with the external DDR3 – For the case where only the on-board block RAM is used, this is significantly increased).

Targeting

Conclusions

This work discussed the methodology and evaluated the feasibility of using high performance Chip-multi-processors for accelerating biomedical signal processing codes. Experts in the Biomedical Signal Processing domain require advanced processing capabilities without having to resort to the expertise routinely utilized in the consumer electronics and telecomunications domains were the silicon platform is designed by one team, benchmarked and optimized by a second team and programmed by a third. We have made a deliberate choice to stay within the reach of tools routinely utilized by Biomedical Signal Processing experts, such as MATLAB, and designed our infrastructure to directly interface to this tool. In this way, we demonstrated the capability of computing, in near-real-time the blood perfusion of living tissue, using algorithms developed in the Embedded MATLAB subset of that tool; The autogenerated C code was passed on to our toolchain which compiled it into an application binary and performed architectrure space evaluation to idenfity the best BioThreads configurations. Following that, the code was loaded onto the CPU, residing on the FPGA

References

[1] K. Rajan, L.M. Patnaik, *"CBP and ART image reconstruction algorithms on media and DSP processors"*, Microprocessors and Microsystems 25 (2001) 233-238

[2] Omkar Dandekar, R. Shekhar,*"FPGA-Accelerated Deformable Image Registration for Improved Target-Delineation During CT-Guided Interventions"*, IEEE Transactions On Biomedical Circuits and Systems, Vol. 1, No. 2, June 2007

[3] Wardell K. and Nilsson G.E., "Duplex Laser Doppler Perfusion Imaging," Microvasc. Res., 52, pp. 171-182 (1996).

[4] Srinivasan S., Pogue B. W., Jiang S. D. Dehghani H., Kogel C., Soho S., Gibson J. J., Tosteson T. D., Poplack S. P. and Paulsen K. D., "Interpreting haemoglobin and water concentration, oxygen saturation, and scattering measured in vivo by near infrared breast tomography," PNAS, 100(21), 12349-12354 (2003).

[5] S. Hu, J. Zheng, V. Chouliaras, and R. Summers, *"Feasibility of imaging photoplethysmography"* Proceedings of the International Conference on BioMedical Engineering and Informatics, (Institute of Electrical and Electronics Engineers, NewYork, 2008), pp. 72-75.

[6] P. Shi, V. Azorin Peris A. Echiadis, J. Zheng, Y. Zhu, P. Y. S. Cheang, S. Hu, "*Non-contact Reflection Photoplethysmography towards Effective Human Physiological Monitoring"*, J Med. Biol. Eng. J Med. Biolog. Eng., Vol.30 Iss 30 (2010): 161-167

[7] V. A. Chouliaras J. L. Nunez, D. J. Mulvaney, F. Rovati, D. Alfonso, *'A Multi-standard Video accelerator based on a vector architecture'*, IEEE Transactions on Consumer Electronics, Vol. 51, Issue 1, Feb 2005, pp 160-167, ISSN:0098-3063

[8] www.arm.com ARM1022ETM Technical Reference Manual. Retrieved June 2007

[9] www.xilinx.com, *"MicroBlaze Processor Reference Guide"*, Doc. UG081 (v10.3). Accessed January 2010

[10] D. Mattson, M. Christensson, *"Evaluation of synthesizable CPU cores"*, Master Thesis, Chalmers University of Technology, Department of Computer Engineering, 2004. Accessed

February 2010

[11] V. A. Chouliaras, J. L. Nunez, "*Scalar Coprocessors for accelerating the G723.1 and G729A Speech Coders"* IEEE Transactions on Consumer Electronics, Vol. 49, Issue 3, Aug. 2003, pg. 703-710

[12] N. Theodoridis, G. Nikolaidis, S., *"The ARISE Reconfigurable Instruction Set Extensions Framework"*, Proceedings of the International Conference on Embedded Computer Systems: Architectures, Modeling and Simulation, 2007. IC-SAMOS 2007. 16-19 July, 2007

[13] Xilinx XPS Mailbox V1.0a data sheet. Available from www.xilinx.com Accessed February 2010

[14] M. Dossis, *"Custom Coprocessor Compiler"*, Private Communication.

[15] NISC Toolset User Guide Version. 2007.07, Centre for Embedded Computer Systems, University of California at Irvine, http://www.cecs.uci.edu/~nisc. Accessed February 2009

[16] Y. Guo, J. R. Cavallaro, *"A Low Complexity and Low Power SoC Design Architecture for Adaptive MAI Suppression in CDMA Systems"*, Journal of VLSI Signal Processing 44, 195–217, 2006

[17] V. Kathail, S. Aditya, R. Schreiber, B. Ramakrishna Rau, D. Cronquist, M. Sivaraman, *"PICO: Automatically designing custom computers"*, IEEE Computer, September 2002, pg. 39-47

[18] Robert Thomson, Scott Moyers, David Mulvaney and V. Chouliaras, *"The UML-based design of a hardware H.264/MPEG 4 AVC video decompression core"*. Proceedings of the 5th International UML-SoC Workshop (in conjunction with 45th DAC, Anaheim Convention Center, USA, June 7th, 2008

[19-BDTI_STUDY]

[20] Leibson, S.; Kim, J, *"Configurable processors: a new era in chip design"*, IEEE Computer, July 2005, vol.38, no.7, pp. 51-59

[21] N. Clark et al., *"Automated custom instruction generation for domain-specific processor acceleration"*. IEEE Transactions on Computers Vol. 54, Issue 10. pp 1258 – 1270 Oct 2005.

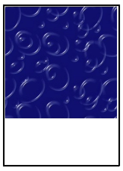
[22] D. Stevens, V. A. Chouliaras, *"LE1: A parameterizable VLIW Chip-Multiprocessor with hardware PThreads support"*, Proceedings of the 2010 IEEE Symposium in VLSI, ISVLSI 2010, Kefalonia, Greece

[23] www.vassilios-chouliaras.com/le1 , accessed June 2010
[24] J. A. Fischer, P. Faraboschi, *"Embedded Computing. A VLIW approach to architectures, compilers and tools"*. Morgan Kaufmann publishers, 2005
[25-TRIMARAN]

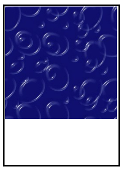
[26] Sotirios G. Ziavrasa,Ã, Alexandros V. Gerbessiotisb, Rohan Bafnaa, *"Coprocessor design to support MPI primitives in configurable multiprocessors"*, INTEGRATION, the VLSI journal 40 (2007) 235–252
[27-OXIMAP]
[28-MIKROTRON CAMERA]
[29-RED_LED]
[30-IR_LED]

[31-VIRTEX6]

David Stevens was born in Athens, Greece in 1969. He received a B.Sc. in Physics and Laser Science from Heriot-Watt University, Edinburgh in 1993 and a M.Sc. in VLSI Systems Engineering from UMIST in 1995. He worked as an ASIC design engineer for INTRACOM SA and as a senior R&D engineer/processor architect for ARC International. Currently, he is a senior lecturer in microelectronics in the Department of Electronic and Electrical Engineering at the University of Loughborough, UK where he's leading the research in CPU architecture and microarchitecture, SoC modeling and software parallelization. His research interests include CPU microarchitecture, high-performance embedded CPU implementations, performance modeling, custom instruction set design and Electronic System Level (ESL) design methodologies. He is the architect of the SS_SPARC multicore platform and a founder of Axilica Ltd.

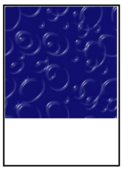


Angelos Echiadis was born in Athens, Greece in 1969. He received a B.Sc. in Physics and Laser Science from Heriot-Watt University, Edinburgh in 1993 and a M.Sc. in VLSI Systems Engineering from UMIST in 1995. He worked as an ASIC design engineer for INTRACOM SA and as a senior R&D engineer/processor architect for ARC International. Currently, he is a senior lecturer in microelectronics in the Department of Electronic and Electrical Engineering at the University of Loughborough, UK where he's leading the research in CPU architecture and microarchitecture, SoC modeling and software parallelization. His research interests include CPU microarchitecture, high-performance embedded CPU implementations, performance modeling, custom instruction set design and Electronic System Level (ESL) design methodologies. He is the architect of the SS_SPARC multicore platform and a founder of Axilica Ltd.

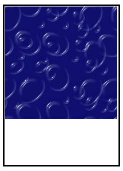


Vincent Azoris Peris was born in Athens, Greece in 1969. He received a B.Sc. in Physics and Laser Science from Heriot-Watt University, Edinburgh in 1993 and a M.Sc. in VLSI Systems Engineering from UMIST in 1995. He worked as an ASIC design engineer for INTRACOM SA and as a senior R&D engineer/processor architect for ARC International. Currently, he is a senior lecturer in microelectronics in the Department of Electronic and Electrical Engineering at the University of Loughborough, UK where he's leading the research in CPU architecture and microarchitecture, SoC modeling and software parallelization. His research interests include CPU microarchitecture, high-performance embedded CPU implementations, performance modeling, custom instruction set design and Electronic System Level (ESL) design methodologies. He is the architect of the SS_SPARC multicore platform and a founder of Axilica Ltd.

Vassilios Chouliaras was born in Athens, Greece in 1969. He received a B.Sc. in Physics and Laser Science from Heriot-Watt University, Edinburgh in 1993 and a M.Sc. in VLSI Systems Engineering from UMIST in 1995. He worked as an ASIC design engineer for INTRACOM SA and as a senior R&D engineer/processor architect for ARC International. Currently, he is a senior lecturer in microelectronics in the Department of Electronic and Electrical Engineering at the University of Loughborough, UK where he's leading the research in CPU architecture and microarchitecture, SoC modeling and software parallelization. His research interests include CPU microarchitecture, high-performance embedded CPU implementations, performance modeling, custom instruction set design and Electronic System Level (ESL) design methodologies. He is the architect of the *BioThreads* platform and a founder of Axilica Ltd.



J. Zeng was born in Athens, Greece in 1969. He received a B.Sc. in Physics and Laser Science from Heriot-Watt University, Edinburgh in 1993 and a M.Sc. in VLSI Systems Engineering from UMIST in 1995. He worked as an ASIC design engineer for INTRACOM SA and as a senior R&D engineer/processor architect for ARC International. Currently, he is a senior lecturer in microelectronics in the Department of Electronic and Electrical Engineering at the University of Loughborough, UK where he's leading the research in CPU architecture and microarchitecture, SoC modeling and software parallelization. His research interests include CPU microarchitecture, high-performance embedded CPU implementations, performance modeling, custom instruction set design and Electronic System Level (ESL) design methodologies. He is the architect of the SS_SPARC multicore platform and a founder of Axilica Ltd.



Sijung Hu was born in Athens, Greece in 1969. He received a B.Sc. in Physics and Laser Science from Heriot-Watt University, Edinburgh in 1993 and a M.Sc. in VLSI Systems Engineering from UMIST in 1995. He worked as an ASIC design engineer for INTRACOM SA and as a senior R&D engineer/processor architect for ARC International. Currently, he is a senior lecturer in microelectronics in the Department of Electronic and Electrical Engineering at the University of Loughborough, UK where he's leading the research in CPU architecture and microarchitecture, SoC modeling and software parallelization. His research interests include CPU microarchitecture, high-performance embedded CPU implementations, performance modeling, custom instruction set design and Electronic System Level (ESL) design methodologies. He is the architect of the SS_SPARC multicore platform and a founder of Axilica Ltd.