Intellectual Challenges in

Energy-Aware COmputing (EACO)*

University of Bristol, Merchant Venturers Building, Room 4.01

EACO Workshop Hosts: Kerstin Eder and David May

Please send any feedback to Kerstin.Eder@bristol.ac.uk.

October 18, 2011

1 Purpose

The purpose of the EACO Initiative at the University of Bristol is to bring together researchers and engineers with interests in *energy-aware computing* for discussions to identify intellectual challenges that can be developed into collaborative research projects. We strive to go significantly beyond the state of the art.

*The EACO workshop series is supported by the Institute for Advanced Studies (http://www.bris.ac.uk/ias) and the Merchant Venturers School of Engineering.

2 Contributors

The list of contributors includes participants from the EACO workshops as follows:

EACO W1, 25 May 2011:

Steve Kerrison	Microelectronics Research Group, University of Bristol
Daniel Schien	Sustainable Computing, University of Bristol
James Hanlon	Microelectronics Research Group, University of Bristol
Leon Atkins	Algorithms Research Group, University of Bristol
Jose Nunez-Yanez	Microelectronics Research Group, University of Bristol
Lorna Colquhoun	RED, University of Bristol
Roger Shepherd	ST-Ericsson
Zuner Bortolotto	School of Physiology and Pharmacology, University of Bristol
Chris Preist	Computer Science, University of Bristol
Rob Hunt	Microelectronics Research Group, University of Bristol
George Papaharalabos	BRL
David May	Microelectronics Research Group, University of Bristol
Kerstin Eder	Microelectronics Research Group, University of Bristol

EACO W2, 13/14 July 2011:

Geza Lore	ARM
Tony Gore	Aspen
George Papaharalabos	Bristol Robotics Lab
Robin Kennedy	NMI
Marcin Hlond	NVIDIA
Rahul Kharche	NVIDIA
Andy Kuligowsky	Picochip
Gajinder Panesar	Picochip
Roger Sheperd	STEricsson
Peter Marwedel	TU Dortmund
Leon Atkins	Computer Science, University of Bristol
Arash Beldachi	Electronics & Electrical Engineering, University of Bristol
Steve Bryant	Microelectronics Research Group, University of Bristol
Lorna Colquhoun	RED, University of Bristol
Kerstin Eder	Microelectronics Research Group, University of Bristol
Corina Grigore	Computer Science, University of Bristol
James Hanlon	Microelectronics Research Group, University of Bristol
Simon Hollis	Microelectronics Research Group, University of Bristol
Rob Hunt	Microelectronics Research Group, University of Bristol
Steve Kerrison	Microelectronics Research Group, University of Bristol
David May	Microelectronics Research Group, University of Bristol
Simon McIntosh-Smith	Microelectronics Research Group, University of Bristol
Atuken Nabina	Electronics & Electrical Engineering, University of Bristol
Jos Nuez-Yaez	Microelectronics Research Group, University of Bristol
Daniel Schien	Computer Science, University of Bristol
Trevor Spiteri	Physics, University of Bristol
Timothy Jones	University of Cambridge
Jeyan Thiyagalingam	e-research Centre, University of Oxford
Henk Muller	XMOS
David May	Microelectronics Research Group, University of Bristol
Kerstin Eder	Microelectronics Research Group, University of Bristol

3 EACO Initiative "Intellectual Challenges"

3.1 Overview

The following is a list of *Intellectual Challenges* (ICs) identified at the first EACO workshop and further refined and extended at the second EACO workshop.

The individual ICs are to be labelled with an estimated timeline, indicating **S** - short, **M** - medium or **L** - long term challenges. The labelling should be established based on discussions in consultation with the EACO community.

An interesting "classification" of ICs would be to distinguish between "incremental" approaches that advance the state of the art in a step by step fashion as opposed to "radically new paradigm shifting" approaches. Again, this classification provides scope for discussion within the EACO community.

3.2 Strategy to take ICs forward

It is expected that student projects or staff secondments can be set up to advance **S** challenges so that we can obtain early results on case studies. These can later be used to underpin research funding applications. We should make every effort to try and get these early results published. Challenges labelled **M** are prime candidates to put forward for internal funding at the Faculty of Engineering or at University of Bristol level. Challenges labelled **L** are much longer term and we need to watch out for funding opportunities at national (EPSRC and TSB - TICs) and international level (EC), gradually building up or profile and track record making use of **S** and **M** challenges. Collaboration with other universities and with industrial partners is key to success.

The challenges are listed in no particular order.

IC1 **Learning from Biology: More Pre-Processing.** An approach based on approximate pre-processing with on demand more detailed focused computations should be more energy efficient than traditional fully accurate computations. Where can this be exploited?

In this context: **Selected Connectivity.** Can we design systems that "know" what processing will be required and that adjust accordingly?

Can we design systems that "know" how much energy will be required/permitted and adjust accordingly? This motivates "Energy Budgets" as non-functional requirements.

IC2 **Learning from Biology: The Dark Silicon Challenge.** Radically new approaches are required to address the "Dark Silicon Challenge", i.e. the fact that engineers can design systems so dense that power supply only permits supplying small fractions of them with power at a time. Can we turn the "Dark Silicon Challenge" into a feature inspired by biology? After all, the the brain only uses selected parts at a time.

One potential solution is increased heterogeneity in the design at the SoC level. With a surplus of transistors that can not all be powered at the same time, it will increasingly make sense to use some of the transistor budget for specialised accelerators. These accelerators may only be used a fraction of the time, but they can be highly optimised for performance and energy efficiency

for these specific tasks. Increased heterogeneity of design presents significant challenges in terms of design trade-offs and for system-level programming.

IC3 **Learning from Biology: Massively Parallel Extremely Low Power Processing.** Build a board with many extremely low power processors and experiment with ways to use them inspired by biology (brain). XMOS devices may be ideal to start this locally, but there may be other options to be explored with industrial collaborators who come forward.

In this context: What connectivity does such a system really need and how can we write software to exploit this?

Two key aspects to consider: Introducing massive parallelism for energy efficiency/low power and introducing it to improve computational performance. How can we make more informed decisions about the tradeoff between both?

IC4 **From "Always On" To "By Default Off".** A paradigm shift is needed over the entire system stack: from the gate and logic level up to algorithm and application level. It is also important that these work together (and not against each other) to produce systems with this behaviour.

A starting point is a study of how energy efficient event-driven computation is compared to traditional polling loops; from event-driven software to event-driven hardware. A paradigm shift is needed towards a more energy efficient concurrent programming model.

The paradigm shift will require educating engineers e.g. in terms of programming models. Translating HPC code into an event-driven style may provide an interesting case study.

The impact of moving from "always on" to "by default off". It would be interesting to estimate the savings this could bring on a big scale, i.e. re-calculate existing energy consumption predictions for the next 10 to 20 years into the future based on the assumption that systems are "by default off, but always available" rather than based on what we currently have.

What is the impact of shifting from desk top PCs to tablets for consumption of electronic media? Is there a net gain or reduction in energy consumption?

IC5 **Power/Energy Characterization of Processor Designs.** Early estimation of power or energy consumption of processors/SoCs is needed to enable hardware design engineers to better understand the properties of the design as it is being developed.

The key challenges to achieve this are:

- **Early Provision of Use Cases.** These provide the typical workload for which a power or energy profile is required. There is a potential for tools to synthesize representative use cases.
- **Fast Estimation of Power/Energy Profile of Hardware.** Current techniques are computationally expensive and rely on information that only becomes available late in the hardware design flow. Moving to higher levels of abstraction should help provide earlier profiles at the cost of losing accuracy.

There is a need to re-evaluate published approaches for multi-core designs. A key question also is how to use the power/energy characterization for optimization. This is particularly challenging for multi-core designs.

IC6 **Power/Energy Characterization of Software.** What is the power or energy behaviour of key software use cases running on selected embedded systems, e.g. mobile platforms? Can a power "linter" be developed (a -Opower compile option), exploiting state-of-the-art static analysis and a suitably high-level characterization of the target hardware for execution?

Can software power modelling be done independent of the target processor, i.e. without having to model the target processor (in detail)? Are there characteristic parameters of a processor that would have to be included into such a generic software power modelling technique? If so, which ones?

UoB has already contributed towards the Carbon Trust supplement document to the Greehouse Gas Protocol on impact assessment of ICT. In particular we have been working with Microsoft UK on the methodology of software energy consumption assessment.

In the short term an initial project could investigate the power consumed by std libraries (e.g. glibc's math.h and so on) for various architectures/systems.

In the medium term we could evaluate the cost of the different levels of abstraction in the system stack, from instruction sets, via operating systems to libraries. Can any layers be "removed"?

In the longer term: How do we validate the models we are fitting to data? They appear empirical, rather than rigorously derived. Can symbolic computation (i.e. computer algebra) help to build models from first principles that can be both rigorous and computationally tractable.

If software developers could understand the relative savings/costs of different software design options and algorithms then that would help them explore the design space wrt energy efficiency of software applications which adds a new dimension to the traditional software design space.

Should a "power fault" be introduced to complement "segmentation faults"? This may be useful for software that has a "Power Budget" as a non-functional requirement.

IC7 **Parallel Design of Complex Software Systems.** Re-design complex applications such as a web browser targeted at parallel execution with energy efficiency as a primary design goal. How does it differ from current solutions?

Also, how do different software paradigms affect power consumption for equivalent tasks? We should investigate representative programming languages and compilers to establish a "base line".

Strong links to IC3 "Learning from Biology: Massively Parallel Extremely Low Power Processing.": The software needs a hardware context, e.g. to determine granularity.

In this context: **Can we design efficient methods to distribute computation over resources?** How?

The HPC/Exascale community should become involved in this challenge.

IC8 **Low Power Multi-Voltage Design and Verification.** Voltage domain crossing impacts functional design and hierarchy. Better understanding of the impact of design decisions is required to enable more informed decisions and to ease verification.

Verification of the complex control for increasingly sophisticated power management is challenging.

IC9 **Dynamic Power Control of Mobile Computer Systems.** When and by how much should a computer system change its voltage and frequency? This is currently a hard challenge. Prediction is needed to enable more informed decisions, particularly in environments where voltage/frequency scaling impacts real-time behaviour and timing constraints.

An initial study could focus on the optimal hardware power features exposed to software.

IC10 **More "Power" to Programmers.** Programmers need to be able to design software with energy efficiency as a primary design goal. This will become mandatory when "Power/Energy Budgets" are an integrated part of the system requirements.

One approach is to provide constructs in programming languages to explicitly express energy budgets within code e.g. in terms of assertions similar to how functional properties are being expressed in hardware design.

Another approach is to provide programmers with constructs that enable them to control the degree of accuracy of the computation at program level, explicitly trading accuracy for energy efficiency.

Static analysis should enable more informed decisions during design space exploration so that programmers better understand the tradeoffs and consequences of their decisions.

One starting point is the MilePost extension to gcc. It employs machine learning to drive optimisation heuristics for compilers? Can a similar approach be taken to optimize energy usage?

The programmer has to have a top down view of the system for this to work. What about adaptive code? Power critical sometimes performance critical otherwise.

IC11 **Raising Energy Consumption Awareness.** Can we provide user feedback for transactions or computations? Raising end-user awareness may help change end-user behaviour towards reducing energy consumption.

Initial projects could focus on the quantification of energy for computation and network transport; feedback to user in webpage, netflix, etc.

Energy profilers (e.g. gprof) in languages and static tools such as "power linters" need to be developed.

At what stage does it cost more to run software than to obtain the licenses? Increasing amounts of data are being analyzed in data centres that consume large amounts of energy. If the specification for the data analysis software contained non-functional requirements such as "Energy Budgets" then end users could make more informed decisions when purchasing software. This

would also give a good rationale to invest into researching/developing techniques that help software engineers design to meet these budgets.

IC12 **Energy Autonomous Robots.** Some robots, e.g. the EcoBot, generate their own energy. It is essential that accurate estimates become available so that such systems can assess the feasibility of a task in term of there being enough energy to complete it before engaging in its execution. These estimates go beyond estimating the cost of computation and require energy consumption modelling at system level including mechanical parts of the robot.

IC13 **Communication vs Computation**. For different architectures and applications, show the trade-off between communication/storage vs computation in terms of energy consumption and other design goals such as performance and code size. This can be explored at all levels of abstraction in the system stack, from algorithms down to architecture and micro architecture.

As a starting point, we need to understand how communication, memory and computation is changing over time and application domains. For this we need metrics and benchmarks for comparison.

IC14 **Re-evaluating Algorithms for Energy Efficiency.** Finding the best "hardware fit" for an algorithm is a challenging task. This is further complicated by the performance focused optimizations of compilers and at microarchitectural level. Some of these are counter-productive when it comes to optimizing for energy consumption. Research questions like *Can we find an architecture or hardware configuration where Bubblesort outperforms Quicksort wrt energy consumption?* provide interesting material for discussion and for student projects.

In this context we also need to invest effort into the: **Re-evaluation of Traditional Metrics for Computational Complexity.** The metrics traditionally used to determine the computational complexity of an algorithm in terms of the input data size are not precise enough to capture energy consumption. For instance, the number of data items does not capture the amount of switching in the hardware caused by the data stream.

Overall, an interesting research programme may be a general **"RE-ASSESSMENT OF SYSTEM DESIGN METHODS FOR ENERGY EFFICIENCY"** covering all aspects of system design from applications / algorithms down to the hardware.

IC15 **Education.** Software developers need to gain an insight into what consumes energy when a program is being executed. They need to develop an intuition to be able to anticipate the impact their design decisions have on energy consumption during execution. This requires tools that enable energy-aware design space exploration and enhanced curricula that promote energy efficiency to a first class design goal for software developers. HPC is a good testbed for this - can energy consumption be included into the assessment of assignments? Awareness needs to be raised across the subjects from computer architecture to software engineering.

4 Rationale for the EACO Initiative

Energy efficiency is now a major (if not the major) concern in electronic systems engineering. The European Commission's 2011 Work Programme on ICT has a strong focus on making systems energy efficient. In the UK, the government and the research councils have made Energy Efficiency one of the top priorities of their programmes, for details see http://www.rcukenergy.org.uk. This is the right time to identify and address the intellectual challenges that lead to energy-aware computing. With a modest amount of investment at this stage we could be in a very strong position to take a lead in this field.

5 Background

Traditionally, researchers and engineers work within one or perhaps two layers of the *system stack* with very limited overlap, e.g. software engineers, computer architects or hardware designers. However, energy-aware computing is a challenge that requires investigating the entire system stack from application software and algorithms, via programming languages, compilers, instruction sets and micro architectures, down to the design and manufacture of the hardware, or alternatively bottom up. This is because energy is consumed by the hardware performing computations, but the control over the computation ultimately lies within the software and algorithms, i.e. the applications running on the hardware. It has recently become clear that, while hardware can be designed to save a modest amount of energy, the potential for savings is far greater at the higher levels of abstraction in the system stack. The greatest savings are expected from energy-consumption-aware software. Hence, addressing the challenge of energy-aware computing requires collaboration between researchers from all the above named areas and a good understanding of the applications that will drive software and hardware development in the future.

6 Opportunity

The current activities on energy efficient computing within the Microelectronics research group already cover almost all layers traditionally related to microelectronics in the system stack. This is a unique strength of the Microelectronics research group and puts us into an excellent position to make significant advances by collaborating. However, we must reach out to the upper layers of the system stack, including algorithms and applications, ranging from pervasive computing in health care, via robotics to the large scale complex IT systems that enable cloud computing. We must understand the wider impact on sustainability and end-user behavior, and also alternative models for energy efficient computation such as those inspired by biology.

7 Aim and Objectives

A series of three workshops on Energy-Aware COmputing has been funded by the Institute for Advanced Studies in collaboration with the Merchant Venturers School of Engineering to bring together researchers from diverse backgrounds with a common interest in pushing the boundaries of Energy-Aware COmputing. The aim of the EACO workshop series is to develop a new research initiative on Energy-Aware COmputing at the University of Bristol. Our initial and longer term objectives are to:

- bring together researchers for discussions with the aim of finding common ground,
- take stock and position ourselves,
  - Where are we now? What is required in the future?
  - What is the state of the art? Where are the barriers?
  - What are the intellectual challenges?
- formulate a joint vision,
  - How can we advance the state of the art?
  - How can we address the intellectual challenges?
- decide on a research and collaboration strategy,
- network nationally and internationally (both within the EU and further abroad) with leading experts in the field (both in academia and in industry) to investigate the potential for research collaboration, and
- prepare to compete for research funding to kick-start this new initiative e.g. by
  - addressing funding bodies at national (EPSRC and TSB - TICs) and international (EC) levels as well as
  - raising industrial sponsorship for our research.