Architecture and Design

David May

Three new challenges

Computing without power

Working with exabytes

Designing with a billion transistors

Computing without power

We have the potential for a new generation of sealed devices - without the need for wires - or replaceable batteries

A *disruptive* technology - sentient, communicating devices will be everywhere - in environmental control, industrial monitoring, tracking, implants, packaging ... RFID is just the beginning

They depend on

- low-energy computing and communications
- embedded sensors and actuators
- low-cost batteries or scavenged power

Wearables

We can embed technology in clothes - or wear it like jewellery!

Technologies include audio, cameras, accelerometers and gyros, GPS, RFID ...

Applications include sports, healthcare, lifestyle, leisure ...

But the big markets will probably be in fashion!

Will the design houses play a major role in the next wave of electronic devices?

What does low energy involve?

Low voltage circuits and logic; dynamic power control

Minimal operations *including* code and data transfers

Support for compression, signal processing, cryptography ...

Event driven systems *including* software

... some of these systems are high-performance but are idle most of the time - and they need a lot of software

Working with exabytes

The Internet in 2010:

- Exabytes everywhere!
- Petabyte servers, petaflop supercomputers, a billion hosts, a zetabyte online
- Services to ubiquitous and nomadic clients including wearables
- Many different forms of content - visuals, soundscapes ...

Working with exabytes

We will see changes in Internet infrastructure aiming to maximise energy efficiency

These will include

- more, simpler, lower-speed processors in servers
- processors optimised for tasks such as transcoding, cryptography, searching ...

There is a potential short-term *accelerator-in-server* opportunity - helped by the use of Ethernet as a standard interconnect

Architecture for exabytes

We’re going to need

... big addresses

... arithmetic and logic on big numbers

... accurate numerics on big numbers

... efficient data representation

... fast, low-latency communication and input-output

Long arithmetic

To multiply two n-word integers we have to execute n^2 multiply instructions - if we have the right instructions!

carryout, result := MUL op1, op2, carryin

and doubling the wordlength reduces the multiply operation count by factor 4.

So (eg) a 64 bit CPU with the right instructions will be 10-100 times faster than a 32 bit CPU with the wrong ones!

Long wordlengths

Let's think about a machine with 128-bit registers:

... big integers: 2^{128} is around 10^{39}

... 64.64 format gives a range of around 10^{20} at 1 in 10^{20} precision

... enough addresses for every byte - or bit - on the planet

... accurate double precision floating point multiply-accumulate

... powerful SIMD data representation

Designing with a billion transistors

‘State-of-the-art’ design costs - and timescales - are escalating

Few applications can support application specific designs

Design and differentiation must be fast - electronics is fashion

This suggests a move to:

- generic programmable and/or reconfigurable chips
- generic ‘platforms’ customisable at low cost

Potential opportunity for new industry ...

The Fabless, IP-less chip company!

Manufacturing companies will not be able to design a fraction of the ASICs made possible by generic platforms

So there's space for new companies which do ASICs purely by programming and configuring platforms - and by selling the result - *not* by selling the programs and configurations

These companies will have specialised high-value applications and market expertise but will not require huge investment

Processor arrays

Large collections of processors were successfully employed in a few computers in the 1980s but have only just come into widespread use:

... Google search engine - tens of thousands of processors
... digital animation (typically 1,000 processors)
... supercomputers - a few thousand processors

and we are beginning to see

... chips full of processors

Processor arrays

... are relatively easy to design, verify and test

... can exploit local clocks for high speed

... could deliver tera-op performance within 5 years

... can integrate control, media-processing and input-output

... can exploit concurrent languages with compiler optimisations

They are potentially an important generic platform technology

Innovations in Design

We have been making hardware design tools look like programming languages for years

But what we need is a single language which can target software, re-configurable hardware - and hardware *from the same source*

So this is a potential opportunity. It would have a big impact on design efficiency - and on the efficiency of designs!

Concurrent languages are the key!

Key Issues - Architecture and Design

David May

Key issues

Reducing power/energy makes new applications possible

... for decades the main driver has been reducing cost

Power-efficient clusters require more, slower processors

... for decades processor speed *and power* has increased

Applications move on - processors need to move with them

... instead we’re adding on too many hardware gadgets!

Processor arrays and clusters are a potential platform for rapid design of divergent products

Key issues

Architectures should be designed to support tools - *compilers evolve slowly*

Architectures should be designed to support software - *we're writing far too much assembler*

Efficient communications and input-output are essential - *kernels are a big overhead*

There is a potential for much more efficient design methods - and *much more efficient software*

Software

May's Law: *Software efficiency halves every 18 months, compensating Moore's Law*

A mixture of

- shortage of skills
- adding too many features
- copy-paste programming
- massive overuse of windows and mouse-clicks
- reliance on Moore's law to solve inefficiency problems

... and an extreme reluctance to re-write 'legacy' software!

Software

In ubiquitous systems, halving the instructions executed can double the battery life - or enable power-scavenging

And big data sets in servers and supercomputers bring big opportunities for better software and algorithms:

Reducing the number of operations from $N \times N$ to $N \times log(N)$ has a dramatic effect when N is large

... for $N = 30\,billion$ this change is as good as 50 years of technology improvements!

Robotics

... has finally come of age!

we know how to build the control systems

we know how to do the sensors - even vision

we have lightweight materials

there is a market - vacuum cleaners, lawn-mowers, pets ...

... a market for sensors, actuators and embedded intelligence

Robotics

Scalable real-time control - computers in the loop:

“... there are about 650 named muscles in the human body and thousands of unnamed ones ...”

or perhaps:

“... there are around 650 named actuators in a humanoid robot and thousands of unnamed ones ...”

and every one will need a microcontroller!

New technologies

There have been several jumps in computer technology since the 1940s and there are likely to be more ...

... exotic technologies based on molecular structures

... plastic devices which are flexible - and printable

... potential for mechanical-electronic integration

and fundamental technology changes always drive changes in architecture ...