The Future of Computer Technology
and its implications for the computer industry

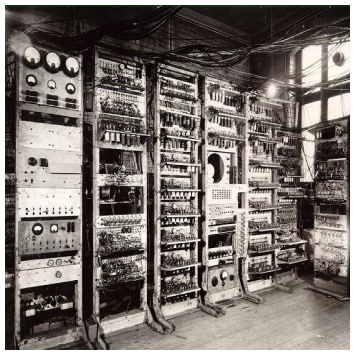
Steve Furber
The University of Manchester

Progress in computer technology over the last four decades has been spectacular, driven by Moore's Law which, though initially an observation, has become a self-fulfilling prophecy and a board-room planning tool. Although Gordon Moore expressed his vision of progress simply in terms of the number of transistors that could be manufactured economically on an integrated circuit, the means of achieving this progress was based principally on shrinking transistor dimensions, and with that came collateral gains in performance, power-efficiency and, last but not least, cost.

The semiconductor industry appears to be confident in its ability to continue to shrink transistors, at least for another decade or so, but the game is already changing. We can no longer assume that smaller circuits will go faster, or be more power-efficient. As we approach atomic limits device variability is beginning to hurt, and design costs are going through the roof. This is impacting the economics of design in ways that will affect the entire computing and communications industries. For example, on the desktop there is a trend away from high-speed uniprocessors towards multi-core processors, despite the fact that general-purpose parallel programming remains one of the great unsolved problems of computer science.

If computers are to benefit from future advances in technology then there major challenges ahead, involving understanding how to build reliable systems on increasingly unreliable technology and how to exploit parallelism increasingly effectively, not only to improve performance, but also to mask the consequences of component failure. Biological systems demonstrate many of the properties we aspire to incorporate into our engineered technology, so perhaps that suggests a possible source of ideas that we could seek to incorporate into future novel computation systems?

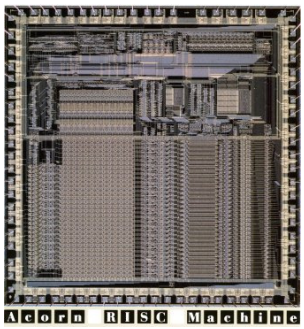
Half a century of progress



When, on June 21 1948, the Manchester 'Baby' computer (photo, right) first executed a program stored in its cathode ray tube memory to produce the correct result, this signalled the start of the modern era of computing. We will be celebrating the 60th anniversary of this singular event later this year. Over those 60 years we have seen many developments in computer architecture that have made machines more flexible and easier to program, but these pale into insignificance alongside the progress in the technology used to build the machines.

To see how far computer technology has progressed over the last 60 years we can compare some of the key characteristics of machines then and now. The 'Baby' (more formally called the SSEM – Small-Scale Experimental Machine) occupied several

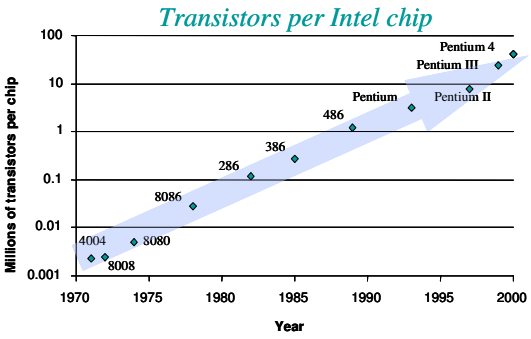
post-office racks of electronics based on thermionic valves – vacuum tubes in American English – and executed 700 instructions per second while consuming around 3.5 kW of electrical power. In 1985, the first ARM processor [1] (ARM1, pictured right) executed 6 million instructions per second and used 0.1 W. Today, a typical power-efficient embedded computer, such as the ARM968 [2] that we will hear more about later, occupies 0.4 mm^2 on the surface of a silicon chip using a 130nm process. The processor has as much capacity in its registers as the Manchester Baby's main memory. The ARM968 delivers about 200 million instructions per second on a power budget of 20 mW.



One way to compare these computing machines is on the basis of their energy-efficiency – the energy consumed to execute one instruction, which is the computer equivalent to the 'miles per gallon' measure for a car. Baby used 5 joules per instruction, ARM1 used 15 nanojoules per instruction, and the ARM968 uses 100 picojoules per instruction. The ratio of the Baby and ARM968 figures points to a staggering improvement in the energy-efficiency of computers over 60 years by a factor of $5x10^{10}$. It is this progress that drives today's explosion in consumer electronics and pervasive computing. One of the ironies of the situation is that the market is growing so fast that the net contribution of electronics to global energy consumption is also growing despite, or arguably as a direct result of, the continuing improvements in energy-efficiency that enable the creation of ever more attractive commodity applications.

Moore or Less?

The spectacular progress in computer technology has become intimately associated with Gordon Moore's 1965 prediction [3] that the number of transistors on an integrated circuit would continue to grow exponentially for a further 10 years, that is until 1975. That "Moore's Law", as it has become universally known, continues to apply today, more than 30 years after its original 'sell-by' date, is a testament to the transition from its original status as an objective extrapolation based on observation and inside knowledge to its present position as the central boardroom planning tool of the global semiconductor industry, epitomised by its role in the Industry Technology Roadmap for Semiconductors (ITRS) [4]. It has become a self-fulfilling prophecy; industry investment is set at the level required to make it happen. As an illustration of how far this has gone, the 12GB microSD card (picture, right) incorporates of the order of fifty billion transistors in a tiny package smaller than a finger nail and just a millimetre thick. (Using multi-level cell technology, each transistor stores two bits of data.)



Along with the exponential growth in the number of transistors on an integrated circuit have come important benefits. The primary mechanism by which this growth has been achieved is transistor shrinkage – making transistors physically smaller through ever more demanding advances in manufacturing technology. As transistors

were made smaller they became cheaper, switched faster, and used less energy per function. As a result a win-win spiral was established wherein the only restraint on how fast transistors could shrink was the time it took to recover the investment in one technology node before moving on the next (smaller, usually by a factor $\sqrt{2}$ in linear dimension) node.

There are downsides to this exponential progress. The cost of building a manufacturing facility (a 'fab') also grows exponentially, as does the cost of designing a state-of-the-art chip. But the benefits outweigh the drawbacks, and for those with deep enough pockets to fund the enormous up-front investment, the chip business has been highly profitable because of its almost limitless expansion potential as digital products become smaller, lighter, more functional and more affordable.

However, exponential growth is ultimately unsustainable. Sooner or later some limit will be reached, going beyond which will require a technological change of a different order from that which has driven the computer industry over the last half-century. As transistors approach the dimensions of atoms current technology will cease to work. All technologies saturate, following an 'S' curve that starts with exponential growth but ends with asymptotically slow advances. There are many possible reasons why progress in computer technology will slow, and every commentator has their favourite – device physics, economics, power dissipation, process variability – but unless there is an as yet unforeseen breakthrough into a completely new technology, the slowdown is almost upon us now. Over the next decade improvement will be increasingly hard-won, with design and manufacturing costs rising inexorably as the fundamental physics of very small devices renders their characteristics increasingly hard to control. One manifestation of the growing cost of design is a drop-off in the number of design starts for complex Systems-on-Chip as the cost-effectiveness of these devices is increasingly called into question. Another is the slowing in the establishment of new fab-less semiconductor start-up companies (companies established to develop their own chip designs for manufacture through third-party 'foundry' services), where the investment required to break-even has increased the risk beyond the comfort limits of venture capital investors.

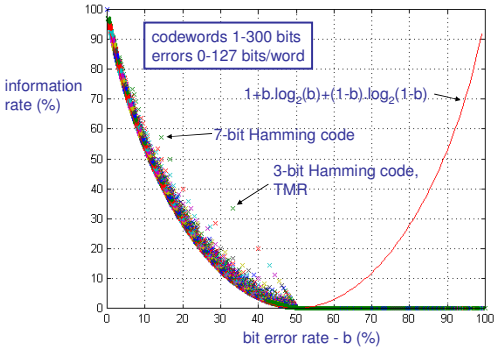
All of these factors suggest that the future will not be simply an extrapolation of the past – it is time for designers to rethink the trade-offs and balances of what constitutes the optimal use of the available technology.

Living with failure

An immediate consequence of the near-atomic scale of near-future transistors is the need for designs to cope with increasing device variability and failure-rates [5]. Models demonstrate that device characteristics will display increasing variability, expressed as the ratio of the variance of a characteristic such as transistor threshold voltage to its mean [6]. The high variance, combined with the statistics of high numbers that come into play as the number of transistors on a chip extends into the tens of billions, means that many devices will be marginal or fail completely, leading to a high incidence of both soft (transient) and hard (permanent) failures.

The challenge of designing reliable systems on unreliable technologies is not new – John von Neumann wrote an early paper on the subject [7], perhaps not surprisingly when you consider the unreliability of the thermionic valves in use at the time – but today's engineers are used to the integrated circuit medium that has offered extremely high levels of reliability for several decades. Furthermore, techniques in use today to cope with rare failures in high-reliability applications simply will not scale to address the problems looming over the next decade or two. Triple Modular Redundancy is

fine if the reliability of an individual subsystem such as a microprocessor is very high, but if there is even a 0.1% probability of transistor failure then none of the three redundant microprocessors is at all likely to work, and having three of them vote on the result when they are all malfunctioning is not going to work at all well! Forecasts for future technologies suggest that component failure rates will be much higher than 0.1%.



To illustrate the problems that arise in dealing with high rates of sort error, the figure to the right shows the percentage information rate (in corrected bits of data per hundred bits of raw data) against the percentage bit error rate (bit errors per hundred bits of data) for a range of redundant encodings. For example, triple modular redundancy (TMR) requires each bit of data to be repeated three times, and delivers a 33% information rate (one bit of information for every three bits used) while coping with a 33% bit error rate (it can correct a single-bit error in each group of three bits).

The envelope of points corresponds to Shannon's information 'entropy' measure [8], which represents the limit of what is achievable. At first it may seem odd that all of the points plotted in the figure lie inside the envelope, which suggests that they perform better than the theoretical optimum. However, this is easily explained. TMR, for example, can cope with a 33% bit error rate, but only if exactly one error falls in each 3-bit codeword. A random 33% bit error rate would give two errors within a 3-bit codeword with a high probability, causing uncorrectable errors and, since it is not possible for the received to work out where the uncorrectable errors have occurred, another layer of redundancy and error correction is required which further reduces the information rate.

We can see that any system designed to cope with a 30% component failure rate requires a 10x redundancy overhead, and 10% failure tolerance requires a 100% redundancy overhead (at a minimum). A conclusion is that if continuing to shrink transistors moves us into a domain where component failures become too frequent, the overhead of adding redundancy to accommodate those failures could easily outweigh the benefits of the increased transistor resource made available by the shrinkage. There is therefore a limit to how far it is technically advantageous to continue in this direction.

Of course, this analysis is only applicable if failures and errors are truly random. Hard errors are consistent from one data word to the next, and their location can be learnt and allowed for. As a result the hardware overheads for coping with hard errors and component failures are potentially much lower than those for soft errors.

One area where techniques exist to cope with quite high failure rates is in memories. Devices such as the 12Gbyte microSD card shown earlier incorporate sophisticated error detection and correction schemes to cope with soft errors, and can internally test areas of memory and map out those that fail at unusable rates due to hard errors (in a similar way to the way that bad sectors are mapped out of use on a hard disk drive). This is one of the reasons why such devices can achieve such impressively high transistor counts; the regularity of the physical layout is another. Another area is communications, where in particular radio communications must cope robustly with high bit error rates. However, none of these approaches seems applicable to complex logic structures such as a microprocessor.

One question to be determined is the level of failure rate that is acceptable. In media systems such as digital TV and portable music players is it necessary to guarantee that no errors ever get through? Very occasional picture glitches arise in digital TV due to uncorrectable communication errors, but because of the picture encoding these glitches tend to be highly visible and intrusive. Surely it must be possible to ensure that minor uncorrectable errors in communication lead to imperceptible errors in the picture, perhaps affecting only the least significant bits (LSBs) of a colour value?

The same argument can be applied to numerical computing. Current approaches apply as much resource to protecting the LSBs as to the most significant bits (MSBs). Surely, if resource is at a premium more should be used to protect the MSB than the LSB? The concept of unequal error correction has been applied to communications protocols (protecting the routing information more strongly than the data payload) [9], but there seems to be considerably more scope here than is currently exploited.

In the limit, where the technology constrains all aspects of a system to display occasional errors, might it be possible to design a microprocessor where a small error in the instruction stream leads to a commensurately small error in the program's execution?

Many cores make light work

A consequence of the way computer technology is changing is the recent paradigm-shift in high-performance microprocessors. For several decades every ounce of accessible single-thread performance was squeezed out of high-end processors, delivered at the cost of ferocious architectural complexity. Features were added that were well past the point of diminishing returns on local cost-effectiveness because all of the software depended on a particularly simple single-threaded programming model. Half a century of research into parallel computing has yet to yield any general-purpose approach to parallelism, so the uniprocessor model dominated the general-purpose market and the inefficiencies of the over-complex processors made sense in the context of the overall system.

Now, suddenly, everything has changed. Dual-core processors are standard, quad-cores are emerging, and the industry speaks of future growth in terms of ever-more processor cores on a chip. What has happened? General-purpose parallelism certainly hasn't been solved and, until it is, the utility of the future many-core processors remains questionable (for general-purpose desk-top applications; there is no issue with using them in many server applications where multiple independent transactions offer easily enough inherent parallelism to keep all of the cores busy). The reality is that diminishing returns from additional complexity, design costs, and the shifting balance between logic and wire delays on a chip combined to render the uniprocessor roadmap very unattractive. Cut-and-paste is as easy on silicon as anywhere else, so putting two or four cores on a chip isn't much harder than one (though maintaining a coherent memory model and balancing bandwidth requirements is non-trivial). The industry has simply abandoned the uniprocessor route as too hard and taken the line of least resistance. They can market ever more processor power through the multi-core route; whether or not you can use that power is your problem, not theirs.

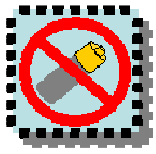
There is now considerably greater motivation to make progress on general-purpose parallelism, because there is no longer any other way forward. An interesting consequence here is that, when a solution does emerge, this could cause another seismic shift in the balance of forces that determines the optimal point for a computer architecture. Processors can be assessed in terms of their manufactured cost (which

relates to performance per unit area of silicon) and their running cost (which amounts to their energy-efficiency, as discussed earlier). On the first of these, high-performance processors, as used in desk-top machines, and embedded processors such as the ARM, as used in mobile phones and music players, are broadly equivalent. On the second, the embedded processors win hands down. Where the embedded processor loses out is in its single-thread performance, but if/when parallelism is readily available it will be much more power-efficient to use a large number of simple processors rather than a small number of high-end processors. Many simple cores could indeed make light work of a computing task, in a sense that contributes directly to significantly improved energy-efficiency.

Grand Challenges in microelectronic design

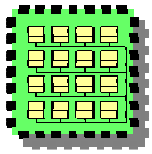
Much is changing in computer technology, as we have seen. This demands new and more visionary approaches from the microelectronics design community if the challenges presented by the technology are to be addressed and the potential for new types of design and product exploited to the full. The UK microelectronics design research community has identified a set of four Grand Challenges for work in this area that create an agenda for future progress:

- Batteries Not Included – minimising the energy demands of electronics.
  As electronics becomes increasingly pervasive it is simply impractical to power it from batteries that constantly need changing. Can we use scavenged energy, or get power requirements so low that a single battery will power the product throughout its life?



- Silicon meets Life – interfacing electronics to biology.
  Retinal prostheses, implanted medical diagnostics, brain-machine interfaces – these are all promising life-enhancing technologies that require a much closer integration between electronics and biology.

- Moore for Less – performance-driven design for net-generation chip technology.
  The drive for ever-higher computing power will continue, but much more attention must be paid to the costs of so-doing: costs to the environment, and design costs.

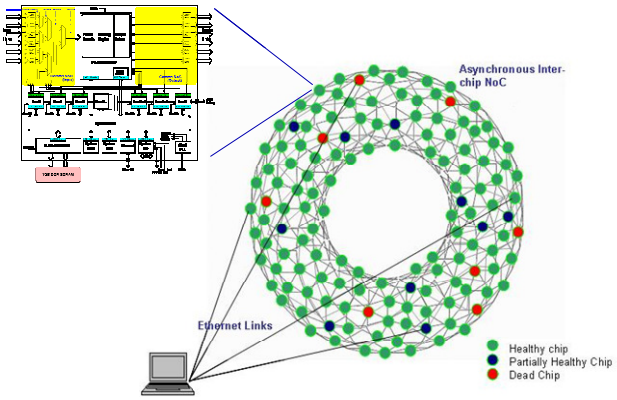


- Building Brains – neurologically-inspired electronic systems.
  Our brains are much more power-efficient than electronics, and much more tolerant of component failure. If we could gain insights into how the biological system functions we might learn how apply those lessons to novel computational systems, and how to build reliable systems on unreliable technologies. We might also learn something interesting about ourselves in the process!

These Challenges say something about how the research community sees the future development of computer technology, and our ability to exploit it through the creation of useful designed artefacts. They are all, of course, multi-disciplinary, and electronic design is only one aspect of them, but they indicate a long-term research agenda based upon the research community's insights into where the technology will allow us to go over the next decade or two.

Biology knows best

To find an example of a system that copes with component failure, exploits very high levels of parallelism and demonstrates excellent energy-efficiency, we can turn to biology. The information-processing principles upon which the brain operates are poorly understood, but the underlying technology has been studied in great detail. We lose about one neuron a second throughout our adult life, but suffer little evident loss of functionality as a result. The hundred billion neurons operate slowly and use minimal energy, but together perform tasks beyond the capabilities of our most powerful computers. If we could understand how the brain delivers this functionality we might learn how to build more resilient and energy-efficient machines.



The SpiNNaker project, a collaboration between the Universities of Manchester and Southampton, with industry partners ARM Ltd and Silistix Ltd, aims to deploy a million ARM processor cores in a massively-parallel computer with the objective of modelling large systems of spiking neurons in biological real time [10]. The machine (illustrated right) is based upon a specially-designed multi-core processor chip incorporating 20 ARM968 processors, connected by an intra- and inter-chip communications fabric conceived to support the very high levels of connectivity found between neurons in the brain. Each multi-core chip is connected to a local 128Mbyte memory chip. The total system has 8 terabytes of memory and can execute 256 tera (10^{12}) instructions per second. Even this amount of computing power is capable of modelling only a billion fairly simple spiking neurons in real time, which is perhaps approaching 1% of the human brain.

A system on this scale would have been inconceivable, or at least unrealistically expensive, only a decade ago. Today's technology renders it feasible within a relatively modest research budget; tomorrow's technology may depend upon some of the lessons we learn from it about biological redundancy and fault-tolerance if it is to continue the remarkable progress that we have seen in the capabilities of computer technology over the last 60 years.

Conclusions

The first 60 years of computer technology has seen spectacular progress, exemplified by the ten orders of magnitude improvement in computer energy-efficiency. This progress underpins the explosion in consumer electronics products that we see today. Continuing progress is by no means guaranteed, however, as the technology approaches atomic scale and a range of problems ranging from fundamental physics to design complexity and economics threaten to obstruct the way forward.

The stresses are already beginning to show, with visible changes in business practice and the shift to multi-core processors (ahead of the software to exploit them) evident as early manifestations of the problems ahead. Much less reliable technology will follow, forcing further changes in architecture, design practice and, if designers are unsuccessful in fully containing these problem, discernable changes in system robustness and performance.

There are many research challenges in the road ahead, and one promising avenue is to increase our understanding of how biology delivers reliable systems on unreliable platforms. We could also learn something about energy-efficiency from biology. Although we have come a long way since the first computers, we still have about a factor one million to catch up before our machines are competitive with nature in this respect, a gap the closing of which would be a major contribution to a sustainable future for our planet!

Acknowledgments

The SpiNNaker project is supported by the UK Engineering and Physical Sciences Research Council (EPSRC), in part through the Advanced Processor Technologies Portfolio Partnership Award at the University of Manchester. Steve Furber holds a Royal Society-Wolfson Research Merit Award. The support of these sponsors and the project partners is gratefully acknowledged.

References

[1] Furber, S. B. & Wilson, A. R. The Acorn RISC Machine - an architectural view. *IEE Journal of Electronics and Power* **33**(6), June 1987, 402-504.

[2] http://www.arm.com/products/CPUs/ARM968E-S.html

[3] Moore, G. E. Cramming more components onto integrated circuits. *Electronics* **38**(8), 1965, 114-117.

[4] Semiconductor Industry Associations of Europe (ESIA), Japan (JEITA), Korea (KSIA), Taiwan (TSIA) and the USA (SIA). *International Technology Roadmap for Semiconductors*, 2007. www.itrs.net

[5] Borkar, S. Designing reliable systems from unreliable components: the challenges of transistor variability and degredation. *IEEE Micro* **25**(6), November-December 2005, 10-16.

[6] Roy, G., Brown, A. R., Adamu-Lema, F., Roy S. & Asenov, A. Simulation study of individual and combined sources of intrinsic parameter fluctuations in conventional nano-MOSFETs. *IEEE Trans. Electron. Dev.* **52**, 2006, 3063-3070.

[7] von Neumann, J. Probabilistic logics and the synthesis of reliable organisms from unreliable components. In eds. C. E. Shannon, C. E. & McCarthy, J., *Automata studies*, Princeton University Press, 1956, 43-98.

[8] Shannon, C. E. A mathematical theory of communication. *Bell System Technical Journal* **27**, July 1948 379-423 & October 1948 623-656.

[9] Namba, K. & Fujiwara, E. Unequal error protection codes with two level burst and bit error correcting capabilities. *Proc. IEEE Intl. Symp. Defect and Fault Tolerance in VLSI Systems*, 2001, 299–307.

[10] Plana, L. A., Furber, S. B., Temple, S., Khan, M., Shi, Y., Wu J. & Yang, S. A GALS infrastructure for a massively parallel multiprocessor. *IEEE Design & Test of Computers* **24**(5), Sept.-Oct. 2007. 454-463.