IET Computers & Digital Techniques

Review Article

IET Journals
The Institution of Engineering and Technology

Brain-inspired computing

ISSN 1751-8601
Received on 1st October 2015
Revised on 14th November 2015
Accepted on 7th December 2015
doi: 10.1049/iet-cdt.2015.0171
www.ietdl.org

Steve B. Furber ✉

School of Computer Science, The University of Manchester, Manchester M13 9PL, UK
✉ *E-mail: steve.furber@manchester.ac.uk*

Abstract: The inner workings of the brain as a biological information processing system remain largely a mystery to science. Yet there is a growing interest in applying what is known about the brain to the design of novel computing systems, in part to explore hypotheses of brain function, but also to see if brain-inspired approaches can point to novel computational systems capable of circumventing the limitations of conventional approaches, particularly in the light of the slowing of the historical exponential progress resulting from Moore's Law. Although there are, as yet, few compelling demonstrations of the advantages of such approaches in engineered systems, a number of large-scale platforms have been developed recently that promise to accelerate progress both in understanding the biology and in supporting engineering applications. SpiNNaker (Spiking Neural Network Architecture) is one such large-scale example, and much has been learnt in the design, development and commissioning of this machine that will inform future developments in this area.

1 Introduction

There has recently been a significant increase worldwide in interest in, and funding for, research into brain function, exemplified by the European €1B ICT Flagship Human Brain Project and the US White House $300M BRAIN (Brain Research through Advancing Innovative Neurotechnologies) initiative. Why is there such a consensus that the time is right for new initiatives in what remains a very challenging frontier of science?

Two complementary answers to this question suggest themselves:

- Computer technology is now (and only now) approaching the capability required to contemplate constructing large-scale computer models of the brain.
- At the same time, computer technology is approaching fundamental physical limits, motivating the quest for alternative approaches to the historic reliance on making transistors ever smaller. The brain is seen as a potential source of such alternative approaches to computation, but this is impeded by our partial understanding of how the brain works.

Of course, alongside these advances in computer technology there have been advances in neuroscience and a growing appreciation of how to build computer and electronic models of neural systems. A greater understanding of the brain should also facilitate progress in the development of treatments for the many debilitating diseases of the brain, but this is not a new issue, and while it may provide further support for the various international initiatives it does not explain the timing of them. Likewise, the natural human desire to expand our understanding of ourselves motivates research in this area – what could be more fundamental to understanding humanity than understanding the organ that embodies our personalities and memories, and determines our every action? – but this motivation is long-standing and does not explain 'why now?'.

Research into the brain is, of course, not new. Neuroscientists have been engaged in the very demanding work of understanding the brain from the bottom up for more than a century, while psychologists have been pursuing a top-down approach to the problem for even longer, and some of the world's great religions have been exploring consciousness and the nature of mind for millennia. More recently, brain-imaging machines have been added to the toolset. However, the brain spans many orders of magnitude in scale, and there is a very large gulf between the scales that are tractable from the bottom up, even with today's advancing multi-electrode array technology, and those that can be resolved from the top down with imaging techniques. Somewhere in this gulf are the most important scales for understanding information processing in the brain – how is information represented, communicated, processed and stored? So far the only tools available to explore these intermediate scales are computer models, and computational neuroscientists have been exploring this space since the very earliest days of computers.

Computational neuroscience has been able to take advantage of the exponential progress in the capabilities of computer technology, informed, of course, by progress in neuroscience and psychology. However, the scale of the problem is daunting even for today's most advanced machines. Scale is important. There are many examples of artificial neural systems (that may or may not bear some relationship to biological brains) that depend critically on scale. The key concepts are deeply rooted in the counter-intuitive geometric properties of high-dimensional spaces and, if these models are scaled down to accommodate the limitations of the computers they run on, their functionality will be compromised, if not totally lost – an example of such a model is Kanerva's sparse distributed memory [1]. Thus we have seen a growth in interest in the design and construction of specialised – brain-inspired – computer systems built both to explore the benefits of deploying our partial knowledge of brain function and to push back the boundaries that constrain computational neuroscience models on conventional machines.

The key contributions of this paper are as follows:

- a discussion of the major challenges impeding progress in computer technology (Section 2);
- an introduction to the brain from a computer engineer's perspective (Section 3);
- metrics for comparing computers with brains (Section 4);
- the major challenges in building brain-inspired machines (Section 5) and an overview of current large-scale projects building brain-inspired machines (Section 6);

• details on the SpiNNaker (Spiking Neural Network Architecture) project (Section 7), and lessons from early user experience on SpiNNaker (Section 8).

2 The end of Moore's law

Progress in computer technology since the first electronic stored-program computer ran its first program on 21st June 1948 in Manchester, England, has been nothing short of formidable. The Manchester 'Baby' executed around 700 instructions per second, while a modern processor – even a fairly simple mobile phone processor of the sort used in SpiNNaker – is around a million times faster. 'Baby' consumed 3.5 kW of electrical power, while the SpiNNaker processor consumes a hundred thousand times less.

Most of this progress has resulted from Moore's Law [2] – the observation made by Gordon Moore in 1965 that the number of transistors that could be manufactured on a single silicon chip doubled every 18 months to two years. This started as an observation, but rapidly became the major planning tool for the entire semi-conductor industry (and hence a self-fulfilling prophecy!). The principal means for delivering Moore's Law has been to make transistors ever smaller. As CMOS transistors get smaller they become cheaper to make, faster, and more energy-efficient. This win-win scenario has driven the industry forward for five decades and has led to the prevalence of computer technology in every walk and aspect of life.

However, no exponential growth pattern can continue forever, and in the case of shrinking transistors, there is the obvious physical limit of the size of the atom. Many seemingly insuperable obstacles have been overcome getting to where we are today, but this one really does seem to be insuperable! We are rapidly approaching this limit, at least in the sense that the statistics of the bulk properties of the semi-conductor material in the active region of a transistor are being compromised by the small number of atoms in the region.

Alongside this approach to physical limits, economic limits are also slowing progress. The cost of designing a billion-transistor chip and the cost of building a 'fab' (fabrication facility) are growing alarmingly. Furthermore, as we approach the physical limit, transistors are becoming less reliable and less predictable in their performance characteristics. Energy-density is also increasing to infeasible levels, leading to the concept of 'dark silicon' – the idea that it will not be feasible to have all of the regions on a chip active at the same time. Software will have to make decisions as to which chip functions are most vital at any time, and switch other functions off to keep power consumption within acceptable limits. All these factors suggest that Moore's Law progress is slowing, and we need to seek alternative ways forward if historic rates of progress are to be maintained into the future.

The first tremor in this impending technology quake was the transition made in the early years of this century by all of the major microprocessor manufacturers away from ever-faster clock speeds and to multicore processors. Instead of using the additional transistors delivered by Moore's Law to deliver a faster single processor, they are now used to deliver more processors (on a single chip). This move, first to multicore and then to many-core architectures, is a result of power constraints. As a consequence, most computer systems now operate with multiple processor cores, and understanding how to deploy these resources – how to write parallel computer programs – is a problem that cannot be ignored, though it has been the 'Holy Grail' of computer science for half a century and remains very challenging for general-purpose software.

Computer engineers are therefore facing a range of new issues concerned with designing energy-efficient parallel systems that are resilient to component variability and failure. This is hard enough, but application domains are also spreading into new areas that create even greater challenges: cognitive systems that sense and respond to their environment are emerging in many areas, from driverless cars and driver-assist technologies through to robot lawn mowers and domestic vacuum cleaners, many increasingly dependent on very challenging forms of computer vision. Such applications require the computer-controlled systems to have flexible capabilities that move them ever closer to the capabilities of biological systems. Maybe biology can help find new solutions to these challenges?

3 Brains

The nearest biological analogue to the computer is the brain. The brain is an information processing system that accepts inputs from a wide range of sensors, from vision through audition, taste, smell, touch through to the many proprioceptive sensors that provide feedback on the state of the biological machine. The brain processes these inputs in the light of stored memories of past experience and 'hard-wired' instinctive knowledge to move through and interact with its environment using its many actuators (muscles). How the brain does this is far from fully understood, though a lot is known about the physiology of the brain and about its construction. What follows is a computer engineer's perspective on the key features of the biological brain.

The fundamental building block of the brain is the *neuron*, or brain cell. From a computer engineer's perspective, neurons are a bit like logic gates in that they have multiple inputs and a single output. The typical fan-in/fan-out (or in neuroscience terminology, convergence/divergence) numbers are quite different, however: whereas logic gates typically have two, three or four inputs, neurons have thousands or tens of thousands of inputs, and some have hundreds of thousands. The connections between neurons are *synapses*. Neurons communicate principally by issuing electro-chemical *spikes* that are pure impulses – no information is conveyed in the shape of the spike, the information is simply in its timing. Biological networks learn primarily through *synaptic plasticity* – whereby a synapse adjusts is efficacy in response to local activity – and through *structural plasticity* – whereby ineffective synapses may be removed and new synaptic connections formed.

Thus a thought is simply a spatio-temporal pattern of spikes in the brain, as are sensory inputs and motor command outputs. Memories are somehow formed through the plasticity mechanisms, though this is far from fully understood.

The above rather simple picture of brain function is, of course, over-simplified. Some neurons do not emit spikes but rather emit *neuromodulators* – chemicals that have a global effect on neurons and especially synapses within their sphere of influence. One of these is *dopamine*, which is known to represent a reward mechanism that can confirm otherwise tentative plasticity changes. Some neurons make direct electrical connections through *gap junctions*, and there is debate about the active role of other cells in the brain – the glial cells that form the scaffolding around which the neurons assemble their complex wiring structures. Then we can go inside the cells and look at the role of the DNA that defines the resident protein mix, the mix of ion channels that generate the spiking behaviour, or even consider the ~1,500 proteins at work in each synapse, small variations in which have significant effects on the synapse's operation. Some argue that quantum effects in the neuron may be vital to understanding deep phenomena such as consciousness. Overall this is a very complex picture, much of which has yet to be unravelled and understood.

What is clear is that brains are highly parallel, very energy-efficient, and highly resilient to component failure. They are also extremely accomplished in performing complex cognitive tasks involving sensing and interacting with their environment. Biology has thus addressed many of the major issues now facing computer engineers and has much to teach us, if only we could understand how it works.

4 Computers against brains

There have been many attempts to compare computers with brains, for example on the basis of energy efficiency, but as we still do not understand the information processing principles at work in

the brain all such comparisons are at best provisional. Most comparisons look at the energy used by computers modelling brain components and compare this with the ~25 W consumed by the brain itself. On this basis computers come out very badly! The best estimates of the computing power required to model a full human brain come out in the exascale region – 10^{18} operations per second. Exascale is the next target in high-performance computing, and it is proving formidably difficult to deliver this level of performance on a 20 MW power budget, but even if this is achieved that is still a million time less energy-efficient than biology.

On the other hand, it might be argued that an equally valid comparison would be to consider a brain modelling a computer. A 32-bit binary number represents a 10-digit decimal integer. A human might take perhaps 5 s to add two such 10-digit numbers written one above the other on a sheet of paper, consuming 100 J. An efficient microprocessor system running at 40 mW could perform 200 M such additions per second, hence each addition consumes 200 pJ, around $10^{12}\times$ less energy than the human. Of course, a brain performing 10-digit addition is not just doing the addition, it is also carrying out a complex vision task to recognise the numbers on the sheet of paper, and also maintaining body functions such as breathing and circulation.

Brains use a number of recognisable techniques to achieve efficient operation and robustness. Spikes are ~100 mV pulses, around one tenth of the ~1 V signals typically used on microchips. On the usual capacitive model of energy consumption this represents a 100× energy saving. The very low signal swing leads to poorer noise immunity, but the robustness mechanisms at work in the biology are highly tolerant of such noise. Computers use dense binary codes which lead to high levels of activity, whereas brains use sparse codes wherein fewer than 10% of neurons are highly active at any time, giving a ~10× energy saving. The computations in a neuron do not carry the overhead of instruction fetch and decode, saving ~10× energy, and are carried out by analogue processes in the synapses, dendrites and soma, perhaps representing a further 100× energy saving over the digital processes in computers. Overall these simple engineering considerations suggest that brain technology could be six orders of magnitude more efficient than digital computers, so perhaps the first comparison above is closer to the truth?

To achieve robustness to component failure brains deploy several techniques worthy of consideration:

- Parameter representation: computers use binary numbers, which are dense but non-uniform – as the names suggest, the most significant bit (MSB) carries far more significance than the least significant bit (LSB). Thus a failure of the MSB is far more catastrophic than is a failure of the LSB. Brains use a far more evenly distributed representation that is far less skewed, and a failure of a single neuron has little detrimental effect irrespective of which neuron it is.
- The use of pulse communication (spikes) in neurons makes fault-tolerance more straightforward, as neurons fail silent, in a known state. This is much easier to accommodate within a fault-tolerance framework than is the conventional level signalling in a computer, where a component may fail stuck at 0 or 1 and failure is indistinguishable from a valid signal – components effectively fail noisily and in an unknown state.
- The plasticity of a biological neural network allows it to compensate for a component failure – nearby neurons will retune their representations to minimise the loss of representational capacity of the population.

Thus we see that whereas in a computer a component failure may be severe (such as an MSB), is hard to ignore, and is permanent, in a brain a component failure is minor, minimally interfering, and will be further compensated for through plasticity.

Can any of these advantages of the biological system be transferred across to the engineered system? This is not straightforward – it is not obvious how you would build an efficient adder for two population-coded parameters, for example – but it might be easier if we understood more deeply how the brain achieves its superiority, particularly in resilience.

Some of the differences between brains and computers are more understandable if you look at their origins and their intent. Computers are designed to meet the need for exact symbolic operation. In the Turing Machine concept there is no scope for approximation or error – the result is either exactly correct or it is irrelevant. A brain, on the other hand, just has to be right enough, or perhaps a little bit faster or cleverer than another brain that plans to eat it! However, computers are increasingly being used for more brain-like tasks, such as understanding complex visual scenes. Here precision may not be of the essence, and biological principles may be applicable. Certainly, there is increasing interest worldwide in trying to understand how to build machines that operate on more brain-like principles than do conventional computers.

5 Building brains

There are a number of current projects aimed at building large-scale brain models, though none is currently aiming at models as large as the human brain. Many complex issues must be taken into consideration in these undertakings, and many unknowns accommodated.

The biggest challenge in any brain-modelling project is to achieve the very high degree of connectivity found in the brain. There are in the region of 10^{15} synapses in the human brain, and 10^{11} 'wires' making those connections. This is clearly beyond the capacity of any electrical wiring technology, but we can exploit the fact that electrical wires are much faster than their biological counterparts to multiplex many biological spikes through far fewer electrical wires. The all-or-nothing nature of the neural spike also helps here, as its nature as a pure asynchronous event allows it to be conveyed efficiently in digital form using address event representation (AER) [3]. In AER systems each neuron is given a unique numerical identifier or 'address', and this address is transmitted to other neurons whenever this neuron spikes. Multiplexing AER events through a single bus or channel will clearly introduce some timing errors as simultaneous events must be sent sequentially, but these errors can be a small fraction of a microsecond, which is negligible in comparison with the time constants at work in neurons, which are of the order of milliseconds. Of course, multiplexing also has its limits, and a single shared channel will not scale to arbitrarily large networks. As a result, a major challenge in large-scale systems is the design of the overall communication fabric and protocols that must distribute and manage the AER event traffic across many channels.

Once the communication has been sorted out, there is then the question of the level of abstraction at which the neural and synaptic processes should be modelled, and what technology should be used to support those models. The full details of the biological cell are very complex, and it is unclear which of these details can safely be abstracted away before there is a risk of losing some vital function. Many large-scale models use 'point neuron' models, such as the leaky integrate-and-fire model and the Izhikevich model [4], which abstract away all of the cell's physical details such as the disposition of its dendritic trees. More complex models based on the Hodgkin–Huxley equations [5] and Rall cable equations [6] may also be used, though these incur several orders of magnitude more computational cost in the ordinary differential equation (ODE) solvers if these are implemented in conventional digital processing hardware.

Similar considerations come to bear on the implementation of synaptic processes, where many different learning rules are of interest to the neural modelling community. The original idea – due to Donald Hebb – is that 'neurons that fire together wire together' [7]. Today there are many variations on this theme, with strong interest in spike timing dependent plasticity (STDP) [8] where a causal relationship between a presynaptic spike and a postsynaptic spike leads to a strengthening of that synapse,

whereas an anti-causal relationship leads to weakening of the synapse. Dopamine-reinforced STDP allows a subsequent 'reward' to confirm a tentative reinforcement, and so on. This is a very rich and diverse area of research (e.g. [9]), and a brain-inspired machine should allow for this diversity.

Implementation technologies range from conventional computers though special purpose many-core machines, GPUs, FPGAs, ASICs to custom analogue circuits that implement the differential equations directly. Analogue circuits may use above-threshold transistors, when it is common to operate significantly faster than biological speeds, or sub-threshold devices – which we digital designers simply describe as 'off' – where the computations are carried out on tiny leakage currents.

All brain-inspired systems, whether designed primarily to support models of the brain itself or to support engineering applications, represent design compromises. There are trade-offs to be made, for example, between energy-efficiency, where sub-threshold analogue circuits represent the optimum technology, and modelling flexibility, where programmable digital systems offer the greatest potential. Similarly, there are trade-offs between synaptic resolution and integration density, since the synaptic weight matrices are the largest data structures to be accommodated. This is perhaps best illustrated by looking at some examples of large-scale brain-inspired systems.

6 Large-scale brain-inspired systems

Not all brain-inspired systems are based on novel hardware. It is possible to apply ideas from the brain in software systems that run on conventional computers. Perhaps the highest-profile example of this is the rather recent interest shown in *deep networks* [10], which have become the leading machine learning technology for a range of applications since Geoff Hinton revisited the issues around training deep multi-layer perceptrons (MLPs) and came up with a new approach that has transformed the cost of training these networks. There have been huge investments in deep networks and their close relative, the convolutional neural network (CNN), the latter being particularly well-suited to automated image classification.

MLPs are not especially brain-like: they do not communicate with spikes (though spiking versions of deep networks are under investigation [11]) and they are trained using error back-propagation, which is not directly biologically plausible. However, they are neural networks, hence their original inspiration is biological. CNNs are a little more biological in that the early convolutional layers resemble the regular patterns of filters found in the retina and early visual processing in the cortex, but they have taken this principle and extended it a long way into the engineering application domain.

In a similar vein, Jeff Hawkins has developed concepts based on a different perspective on cortical processing into his hierarchical temporal memory model [12]. This model is based on viewing dendrite branches as separate processes, and exploits the properties of high-dimensional binary spaces through the use of sparse distributed representations of information. The model runs on conventional computers, using long binary strings to represent information, and the system is especially effective at recognising anomalous inputs in temporal data sequences.

Looking now at brain-inspired hardware, the IBM TrueNorth chip [13] is a very impressive silicon implementation of a neural network. TrueNorth implements 4,096 neurosynaptic cores each of which models 256 neurons with 256 synaptic inputs. The chip uses 5.4 billion transistors but runs at only 70 mW, using an event-driven hardware style to minimise activity (and thereby save power). The neuron model is fixed, and each synapse is binary with a per-input strength modulation shared across the 256 neurons in a core. A feature of the design is the deterministic behaviour of the system corresponding exactly to a software model that can be used for development and network training.

Other approaches are represented by the Stanford Neurogrid [14], which employs sub-threshold analogue circuits for real-time performance, the Heidelberg HiCANN system [15], which uses wafer-scale above-threshold analogue circuits to run 10,000× faster than biology, and the Cambridge Bluehive system [16], which uses digital circuits on FPGAs to deliver real-time performance. Our own contribution is the SpiNNaker machine, summarised in the next section.

7 SpiNNaker

SpiNNaker, a contraction of "Spiking Neural Network Architecture", is a massively-parallel computer designed specifically to accelerate our understanding of the brain through its ability to support very large-scale systems of spiking neurons in biological real time. The goal is to integrate a million small mobile phone processors in a single system capable of modelling up to a billion neurons and a trillion synapses. A billion neurons is only 1% of the human brain, or ten whole mouse brains! There is nothing magical about the million-core objective, except that it requires scalability to be a first-class consideration from the outset. Why not more than a million? The answer here is really economics – a million processors is about the limit of what can be achieved with a (generous) academic research budget. We aimed from the outset for a build cost of £1 per processor, and despite many individual costs deviating significantly from the planned budget, these deviations approximately cancelled each other out in the final reckoning. The design costs – principally manpower – were significantly higher than the build costs, but again achievable within academic research funding constraints.

The design of the machine has been covered extensively elsewhere, from the silicon [17] to the architecture [18] and the software philosophy [19, 20]. The key 'brain-inspired' feature of the machine is the mechanism it uses to deliver biological levels of connectivity between the spiking neuron models, which is to map AER into a very lightweight packet-switched communication fabric [21] with a packet router at the heart of each SpiNNaker chip. A neural spike represents of the order of one bit of information, and in SpiNNaker this is conveyed in a 40-or 72-bit packet with a 32-bit AER 'key', 8 bits of management data, and an optional 32-bit data payload (which is typically not used for pure spike packets).

A SpiNNaker package contains a processing chip with 18 ARM processor cores together with a 128 Mbyte memory chip, access to which is shared between the processors. 48 of these packages are mounted on an extended double-height Eurocard circuit board (Fig. 1), giving 864 cores per board. 24 boards can be mounted in a 19-inch card frame, and five card frames stacked in a standard 19-inch cabinet (Fig. 2). This cabinet then contains over 100,000 processors and is capable of modelling a spiking neural network of the scale of a mouse brain – 100 million neurons – in biological real time, though of course using simplified point-neuron models. The full million-core machine will require ten of these cabinets.

The machine is supported by software tools that compile a neural network description written in a standard language such as PyNN or Nengo into a form suitable for loading onto the machine, thereby removing from the user any requirement for a detailed understanding of its principles of operation. The model can interface to external AER sensors and actuators for real-time robotics applications.

The use of general-purpose processors in SpiNNaker allows for very flexible modelling of the neuronal and synaptic equations as these are implemented in software. Although the machine is optimised in terms of architectural balance for the simpler 'point neuron' models on the principle that complex networks of simple neurons are probably more interesting than simple networks of complex neurons, nothing here is set in stone. Synaptic plasticity is clearly a vital aspect of neuronal learning, and providing for a range of plasticity mechanisms is key to the usefulness of a research platform such as SpiNNaker. There are trade-offs, of course, as more complex neuron or synapse models will take more compute cycles and therefore each core will be able to model fewer neurons/synapses, but most things turn out to be possible

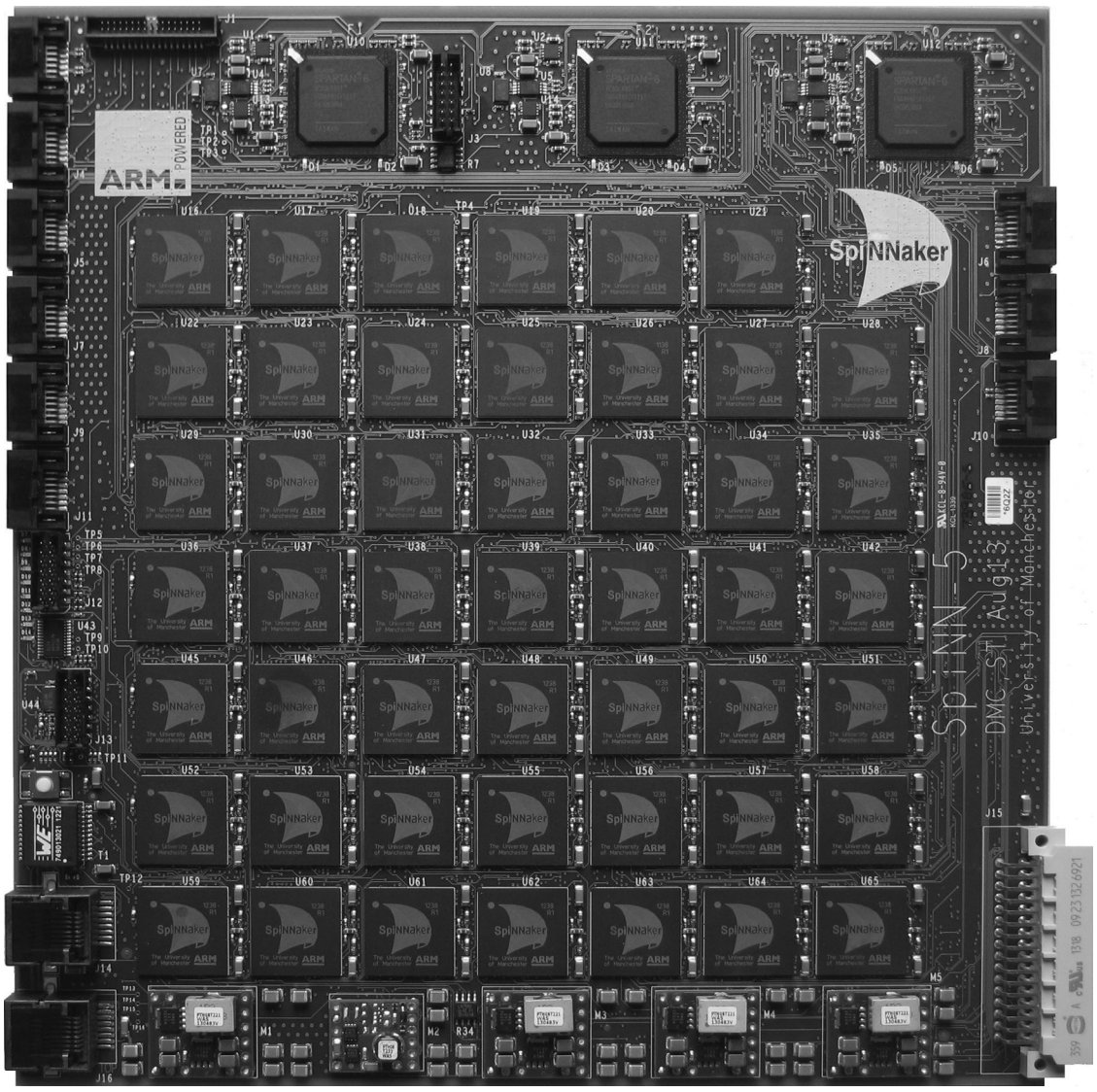

Fig. 1 *48-node SpiNNaker circuit board with 48 packages incorporating 864 ARM processor cores in total*

with a little (sometimes a lot of) thought. The flexibility of the software approach has already been richly demonstrated in our early encounters with real users.

8 Real users

With any generic platform such as SpiNNaker, the designers make certain assumptions about how the machine will be used. Subsequent encounters with real users quickly put those assumptions to the test!

In SpiNNaker's case, the major assumptions were based around the typical characteristics of biological neural systems – variable connectivity in the region of thousands to tens of thousands of inputs per neuron; sparse connectivity whereby each neuron in a population connects to perhaps 10% of the neurons in a population onto which this population projects; mean firing rates in the region of 5 to 10 Hz, with peak firing rates in the hundreds of Hz; and so on. Many users, it turns out, are interested in networks that operate outside these biological settings. Of particular interest is our ability to support the neural engineering framework (NEF) [22] from the University of Waterloo, Canada. The NEF, and its associated language Nengo, is used to support the most advanced multi-behavioural neurocognitive system built to date, Spaun [23]. A major strength of the NEF is that it is built on a solid theoretical foundation, control theory. Currently Spaun runs on a large cluster computer, but this takes 2.5 h to compute each second of real time. A major early goal for SpiNNaker is to run Spaun in real time. However, the NEF generates neural populations with mean firing rates in the hundreds of Hz, and population-to-population projections are generally dense – fully connected. This has led the Nengo SpiNNaker back-end to use techniques rather different from those originally conceived [24]. Instead of communicating individual spikes, the Nengo system decodes the population firing patterns to recover the parameter(s) that they represent, and it is that parameter that is communicated in every 1 ms time-step. Note that this is only possible because the underpinning control theoretic model defines such a decoding.

At a more detailed level, the assumptions regarding computational requirements were based on published examples of neural ODE solvers using basic Euler integration with 1 ms time steps. However, many users want more accurate ODE solvers [25] particularly for the stiffer models, and these inevitably shift the architectural balance towards greater computational complexity. Some users also want greater accuracy through the use of 0.1 ms time steps; on SpiNNaker this may be best achieved by running 10× slower than real time.

The major computational load in SpiNNaker is processing synaptic connections, where a single core handles up to 5 million connections per second (compared with up to one million neuron ODE time steps per second). Adding plasticity to the synapse model increases the computational cost per synapse significantly,

Fig. 2 *SpiNNaker cabinet incorporating over 100,000 ARM processor cores*

and this has led to considerations of using more than one core per neural population – for example one to handle the ODE solvers and one to handle the synaptic plasticity [26]. A similar approach is under consideration to employ multiple cores to process individual dendritic branches, thereby improving the overall efficiency of the synaptic processing algorithms and, perhaps, allowing each branch to support local non-linear processes.

All these new considerations have emerged as users have attempted to use SpiNNaker to support models that were not anticipated at design time, and perhaps underline the advantages inherent in basing all aspects of the model support apart from the communication primitives in software rather than hardware. The universality of software has its costs in terms of both performance and energy-efficiency, but it offers a flexibility that is hard to achieve any other way, and is particularly valuable in a research platform such as SpiNNaker.

9 Conclusions

Although brains and computers are both primarily information processing systems, there are very great differences in their principles of operation. There are many aspects of the operation of brains that are little understood, yet there is evidence that there are lessons to be learnt that could be applied in the design of machines and that address impending challenges in the design of those machines. These lessons are in areas such as the efficient use of massive parallelism, energy-efficiency and resilience to component failure, none of which requires us to solve the Grand Challenge problem of how the brain supports intelligence and consciousness.

SpiNNaker is one of several projects worldwide addressing the challenge of building large-scale brain-inspired computing platforms. The primary goal of SpiNNaker is to offer a generic platform for modelling large-scale brain models based on spiking neurons, and a bespoke communications infrastructure enables the machine to deliver real-time performance that is unachievable on computing platforms with conventional communication fabrics. Other aspects of the machine are based on conventional (if rather constrained) software mechanisms, which give the machine a degree of flexibility that is proving to be its major strength. It is also enabling users to exploit the machine in ways unanticipated during its design that stretch the initial design assumptions in many ways, all of which will feed into the design of the next generation machine.

Understanding the brain remains as one of the great frontiers of science. Major progress depends primarily on the emergence of theories offering explanations of how the brain encodes, stores and processes information, but those theories need testing, and computers such as SpiNNaker are configured to offer platforms for such hypothesis testing, as well as supporting more experimental explorations from which the data to inspire new theories can emerge. After 15 years in conception and ten in development, SpiNNaker is now available and in use. Where it will lead, and what we will learn from it, remains to be seen!

10 Acknowledgments

The design and construction of the SpiNNaker machine was supported by EPSRC (the UK Engineering and Physical Sciences Research Council) under grants EP/D07908X/1 and EP/G015740/1, in collaboration with the universities of Southampton, Cambridge and Sheffield and with industry partners ARM Ltd, Silistix Ltd and Thales. Ongoing development of the software is supported by the EU ICT Flagship Human Brain Project (FP7-604102), in collaboration with many university and industry partners across the EU and beyond, and our own exploration of the capabilities of the machine is supported by the European Research Council under the European Union's Seventh Framework Programme (FP7/2007-2013) / ERC grant agreement 320689.

SpiNNaker has been 15 years in conception and 10 years in construction, and many folk in Manchester and in our various collaborating groups around the world have contributed to get the project to its current state. We gratefully acknowledge all of these contributions.

11 References

1 Kanerva, P.: 'Sparse distributed memory' (The MIT Press, 1988)
2 Moore, G.E.: 'Cramming more components onto integrated circuits', *Electronics*, 1965, **38**, (8), pp. 114–117
3 Mahowald, M.: 'VLSI analogs of neuronal visual processing: a synthesis of form and function'. Ph.D. dissertation, California Inst. Tech., Pasadena, CA, 1992
4 Izhikevich, E.M.: 'Which model to use for cortical spiking neurons?', *IEEE Trans. Neural Netw.*, 2004, **15**, pp. 1063–1070
5 Hodgkin, A., Huxley, A.F.: 'A quantitative description of membrane current and its application to conduction and excitation in nerve', *J. Physiol.*, 1952, **117**, pp. 500–544
6 Rall, W.: 'Branching dendritic trees and motoneuron membrane resistivity', *Exp. Neurol.*, 1959, **1**, pp. 491–527
7 Hebb, D.O.: 'The organization of behavior: a neuropsychological theory' (Wiley, New York, NY, 1949)
8 Bi, G.Q., Poo, M.M.: 'Synaptic modifications in cultured hippocampal neurons: dependence on spike timing, synaptic strength, and postsynaptic cell type', *J. Neurosci.*, 1998, **18**, pp. 10464–10472

9 Tully, P.J., Hennig, M.H., Lansner, A.: 'Synaptic and nonsynaptic plasticity approximating probabilistic inference', *Front. Synaptic Neurosci.*, 2014, **6**, (8), pp. 1–16
10 LeCun, Y., Bengio, Y., Hinton, G.: 'Deep learning', *Nature*, 2015, **521**, pp. 436–444
11 Stromatias, E., Neil, D., Pfeiffer, M., *et al.*: 'Robustness of spiking deep nelief networks to noise and reduced bit precision of neuro-inspired hardware platforms', *Front. Neurosci.*, 2015, **9**, (222), doi: 10.3389/fnins.2015.00222
12 Ahmad, S., Hawkins, J.: 'Properties of sparse distributed representations and their application to hierarchical temporal memory', *CoRR*, 2015, **abs/1503.07469**, pp. 1–18
13 Merolla, P.A., Arthur, J.V., Alvarez-Icaza, R., *et al.*: 'A million spiking-neuron integrated circuit with a scalable communication network and interface', *Science*, 2014, **345**, (6197), pp. 668–673
14 Silver, R., Boahen, K., Grillner, S., *et al.*: 'Neurotech for neuroscience: unifying concepts, organizing principles, and emerging tools', *J. Neurosci.*, 2007, **27**, (44), pp. 11807–11819
15 Schemmel, J., Bruderle, D., Grubl, A., *et al.*: 'A wafer-scale neuromorphic hardware system for large-scale neural modeling'. Proc. Int. Symp. Circuits System, 2010, pp. 1947–1950
16 Fox, P.J., Moore, S.W., Marsh, S.J.T., *et al.*: 'BluehiveVA field-programmable custom computing machine for extreme-scale real-time neural network simulation'. Proc. IEEE 20th Int. Symp. Field-Programmable Custom Comput., March 2012, pp. 133–140
17 Painkras, E., Plana, L.A., Garside, J.D., *et al.*: 'SpiNNaker: a 1W 18-core system-on-chip for massively-parallel neural network simulation', *IEEE J. Solid-State Circuits*, 2013, **48**, (8), pp. 1943–1953
18 Furber, S.B., Lester, D.R., Plana, L.A., *et al.*: 'Overview of the SpiNNaker system architecture', *IEEE Trans. Comput.*, 2013, **62**, (12), pp. 2454–2467
19 Furber, S.B., Galluppi, F., Temple, S., *et al.*: 'The SpiNNaker project', *Proc. IEEE*, 2014, **102**, (5), pp. 652–665
20 Brown, A.D., Furber, S., Reeve, J.S., *et al.*: 'SpiNNaker – programming model', *IEEE Trans. Comput.*, 2015, **64**, (6), pp. 1769–1782
21 Plana, L.A., Clark, D., Davidson, S., *et al.*: 'SpiNNaker: design and implementation of a GALS multi-core system-on-chip', *ACM J. Emerg. Technol. Comput. Syst.*, 2011, **7**, (4), pp. 17:1–17:18
22 Eliasmith, C., Anderson, C.H.: 'Neural engineering: computation, representation, and dynamics in neurobiological systems' (MIT Press, Cambridge, MA, 2003)
23 Eliasmith, C., Stewart, T., Choo, X., *et al.*: 'A large-scale model of the functioning brain', *Science*, 2012, **338**, (6111), pp. 1202–1205
24 Mundy, A., Knight, J., Stewart, T., *et al.*: 'An efficient SpiNNaker implementation of the neural engineering framework'. Proc. IJCNN 2015, Killarney, Ireland, 2015
25 Hopkins, M., Furber, S.: 'Accuracy and efficiency in fixed-point neural ODE solvers', *Neural Comput.*, 2015, **27**, (10), pp. 2148–2182
26 Galluppi, F., Lagorce, X., Stromatias, E., *et al.*: 'A framework for plasticity implementation on the SpiNNaker neural architecture', *Front. Neurosci.*, 2014, **8**, (429), doi: 10.3389/fnins.2014.00429