Neural Networks 24 (2011) 961–978

Contents lists available at SciVerse ScienceDirect

Neural Networks

journal homepage: www.elsevier.com/locate/neunet

2011 Special Issue

Concurrent heterogeneous neural model simulation on real-time neuromimetic hardware

Alexander Rast*, Francesco Galluppi, Sergio Davies, Luis Plana, Cameron Patterson, Thomas Sharp, David Lester, Steve Furber

School of Computer Science, University of Manchester, Manchester, M13 9PL, UK

ARTICLE INFO

Keywords:
Hardware
Multimodel
Multiscale
Universal
Asynchronous
Real time
Neuromimetic

ABSTRACT

Dedicated hardware is becoming increasingly essential to simulate emerging very-large-scale neural models. Equally, however, it needs to be able to support multiple models of the neural dynamics, possibly operating simultaneously within the same system. This may be necessary either to simulate large models with heterogeneous neural types, or to simplify simulation and analysis of detailed, complex models in a large simulation by isolating the new model to a small subpopulation of a larger overall network. The SpiNNaker neuromimetic chip is a dedicated neural processor able to support such heterogeneous simulations. Implementing these models on-chip uses an integrated library-based tool chain incorporating the emerging PyNN interface that allows a modeller to input a high-level description and use an automated process to generate an on-chip simulation. Simulations using both LIF and Izhikevich models demonstrate the ability of the SpiNNaker system to generate and simulate heterogeneous networks on-chip, while illustrating, through the network-scale effects of wavefront synchronisation and burst gating, methods that can provide effective behavioural abstractions for large-scale hardware modelling. SpiNNaker's asynchronous virtual architecture permits greater scope for model exploration, with scalable levels of functional and temporal abstraction, than conventional (or neuromorphic) computing platforms. The complete system illustrates a potential path to understanding the neural model of computation, by building (and *breaking*) neural models at various scales, connecting the blocks, then comparing them against the biology: computational cognitive neuroscience.

1. Dedicated hardware, dedicated model?

Neural network modelling is transitioning from relatively small-scale models that simulate relatively small "subcircuits" or specific behavioural functions to large-scale models attempting realistic simulation of large areas of the brain and complex interactions between behavioural regions. The challenges of this scale of modelling are formidable. Much depends on what the level of process abstraction is: there is as yet no clear consensus on whether detailed behaviour at the level of ion channels is critical (Ananthanarayanan & Modha, 2007; Migliore, Cannia, Lytton, Markram, & Hines, 2006). There is, furthermore, a lurking data-analysis problem: even if the system could perform a detailed simulation at the channel level of a large-scale area, the potential volume of data on the simulation could extend to the terabyte-level or even larger (with, say, a network containing 10^{12} synaptic connections). Quite aside from the storage requirement, the question becomes one of arranging and displaying the data in a way that can be made intelligible to the human researcher. Simulating large-scale neural networks thus requires various levels of abstraction. The research community also accepts, in the main, that large-scale neural modelling needs dedicated hardware (Johansson & Lansner, 2007). Dedicated neural hardware, however, has its own problem: what neural model to implement?

The intuitively obvious approach is to make the hardware be a literal translation of a given model into circuitry (Indiveri, Chicca, & Douglas, 2006). Modern "neuromorphic" architectures have been more intimately concerned with biological plausibility than previous designs, usually implementing spiking networks with full dynamics (Pelayo, Ros, Arreguit, & Prieto, 1996). An important motivation for using neuromorphic chips has always been that it is possible to fabricate analogue devices whose characteristics have real similarities with biological neurons (Hynna & Boahen, 2006). An analogue chip can not merely "simulate" a neural network, it can *be* a neural network, physically implemented in silicon (Zaghloul & Boahen, 2006). However, this intuitively elegant direct-implementation capability has also been conceptually limiting, because as a result most neuromorphic chips directly implement a specific model in hardware (Pelayo et al., 1996), with some general-purpose reconfigurability to tailor the system for different

* Corresponding author. Tel.: +44 07747 807599.
E-mail address: rasta@cs.man.ac.uk (A. Rast).

0893-6080/$ – see front matter © 2011 Elsevier Ltd. All rights reserved.
doi:10.1016/j.neunet.2011.06.014

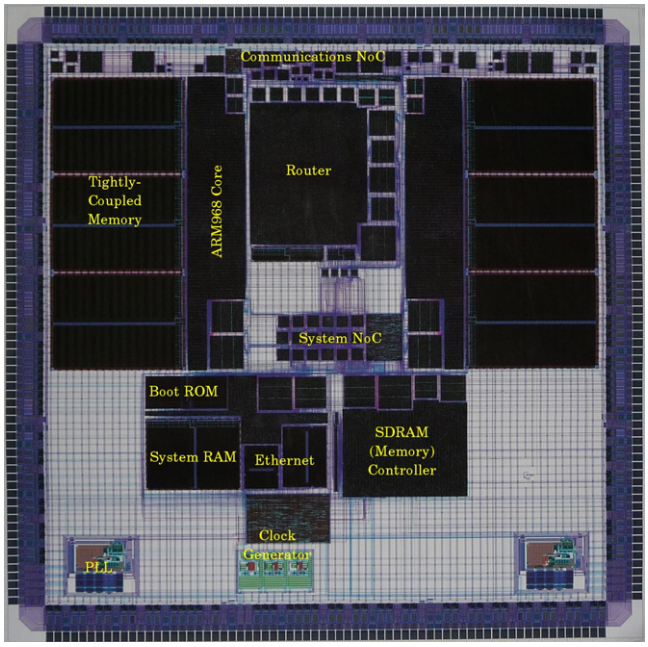

Fig. 1. SpiNNaker test chip. Image taken from the GDS2 plot sent to manufacture. The chip was fabricated at UMC using their 130e-llsp low-power process. Die size is 5 × 5 mm.

parameter values (Vogelstein, Mallik, Vogelstein, & Cauwenberghs, 2007), eliminating the possibility of implementing different neural models with the same hardware. "Direct implementation" is in fact a somewhat misleading concept, because the "real" model of neural processing is unknown (outside certain well-studied areas such as the retina), providing no certainties a chip, whether analogue or digital, is directly implementing *anything* (Furber & Temple, 2007; Thomas & Luk, 2009).

Complicating the problem is a gap in mutual understanding between the divergent goals of the hardware designers, who typically are interested in the *nature* of the neural model of computation (Westerman, Northmore, & Elias, 1997), and the neurobiologists, who tend to be more interested in the *causes* and *effects* of neural computation (Yu & Cauwenberghs, 2010) Thus the two groups need very different types of models and chips, and a single, "hardwired" chip is not likely to be suitable for both. A better hardware platform would be one that permitted experimentation between various models, at different levels of abstraction, so that modellers could use detail where it was necessary for characterising a neural model, abstract descriptions when understanding system behaviour was paramount or for parts of a larger network model not under intense study.

Rather than make *a priori* decisions about which model is appropriate, we propose an architecture that facilitates the exploration of different models, at various different levels of abstraction. Such a system permits different research groups, with different goals, to simulate neural networks with hardware acceleration, adding model detail where it is relevant for the research problem under study, while abstracting parts of the model necessary for functionality, but either not central to the research question, or important to simplify in order to uncover the fundamental dynamics. This "neuromimetic" architecture (Rast et al., 2010b) uses a universal neural chip, SpiNNaker (Fig. 1), to demonstrate how to implement different neural models at various scales and simulate them simultaneously on the same system, using an integrated tool chain to convert model-level representations into device-level configurations. This system recasts the purpose of neural hardware from that of a model-specific simulation accelerator to that of a tool for real-time, multiscale model exploration.

2. Requirements for scalable multimodel simulation

Simulation of neural networks with multiple, heterogeneous neural and synaptic models tends to be done mostly in software. Much of this is due to the intrinsic assumptions of the field: most researchers expect that the hardware will "hard-wire" the neural model into the silicon, effectively preventing multimodel simulation. Even software attempts have been relatively limited, however, and as in Lange (1990) conclude that existing general-purpose hardware is usually inadequate because of architectural mismatch. In the case of spiking neural networks, there is also a tendency to regard a model with different parameter values between neurons as an "heterogeneous" model (Merolla & Boahen, 2006; White, Chow, Ritt, Soto-Treviño, & Kopell, 1998), even though the actual dynamic equations may remain the same. While the effect of parameter variation is an interesting research area, it does not reveal the effects of varying the actual equations themselves; and the different possible dynamics of neurons with different equations can vary dramatically, often with significant consequences for the network's computational capability or biological fidelity (Izhikevich, 2004).

Realistically, a truly scalable neural network hardware platform will involve dedicated, full-custom chips (Jahnke, Schönauer, Roth, Mohraz, & Klar, 1997). If heterogeneous model simulation is a goal, there is a tradeoff between architectures: the programmability of digital or the low power and efficiency of analogue. Digital designs are readily programmable and can take advantage of aggressive process technology roadmaps, making them a seemingly natural choice. However, they need a high transistor count per neuron, are the most power-hungry, and have the least natural fit with biological prototypes (Joseph & Gupta, 2010). Caught between these opposing poles of equally powerful advantages and disadvantages, digital chips have therefore been exercises in design tradeoff (Kaulmann, Dikmen, & Rückert, 2007; Mahoney & Elhanany, 2008). With analogue designs, the introduction of programmable analogue memories, chiefly floating-gate based (Diorio, Hasler, Minch, & Mead, 1997; Holler, Tam, Castro, & Benson, 1989), has eliminated what was historically the most vexing barrier: weight programmability, but programmable analogue *circuitry* is still very much in its infancy (Basu et al., 2010; Lehtonen & Laiho, 2010), a long way off standardisation. However, in both analogue and digital, the most limiting factor may be conceptual: if the architecture assumes one hardware block per neuron, wire density and power are inevitably going to become factors at large scales.

An emerging neural data communications standard: Address-Event Representation (AER) (Lazzaro, Wawrzynek, Mahowald, Silviotti, & Gillespie, 1993), has the potential to eliminate this conceptual barrier. AER uses packets that encode the source of the spike as an address and is a proven, efficient way to serialise and then multiplex multiple neural signals onto the same series of lines (Boahen, 2000). An AER device can be made configurable, using the same packet-switched interconnect to send the configuration data, and by virtue of being asynchronous mitigates the power problems that arise with synchronous parallel bus architectures (Schemmel, Fieres, & Meier, 2008). AER is well established on the way to becoming a defined standard (Chicca et al., 2007), thus making it the signalling method of choice for future neural designs.

There remains a major gap in tool support: neuromorphic architectures tend not to integrate seamlessly into existing simulators (Brüderle et al., 2009) thus using such chips has, historically, been difficult (Steinkraus, Buck, & Simard, 2005). Recently, PyNN (Davison et al., 2009) has emerged as a common, cross-platform standard for defining and simulating neural networks. PyNN contains plug-in modules for a wide variety of common neural simulators, and extending it is a matter simply of writing a