A Reconfigurable Architecture for Implementing Locally Connected Neural Arrays

J. E. Graham-Harper-Cater, C. T. Clarke, B. W. Metcalfe and P. R. Wilson
Department of Electronic & Electrical Engineering
University of Bath
Bath, England
Email: J.E.Graham-Harper-Cater@bath.ac.uk

***Abstract*—Moores law is rapidly approaching a long-predicted decline, and with it the performance gains of conventional processors are becoming ever more marginal. Cognitive computing systems based on neural networks have the potential to provide a solution to the decline of Moores law. Identifying common traits in neural systems can lead to the design of more efficient, robust and adaptable processors. Despite the potentials, large-scale neural systems remain difficult to implement due to constraints on scalability. Here we introduce a new hardware architecture for implementing locally connected neural networks that can model biological systems with a high level of scalability. We validate our architecture using a full model of the locomotion system of the *Caenorhabditis elegans*. Further, we show that our proposed architecture archives a 9 fold increase in clock speed over existing hardware models. Importantly the clock speed for our architecture is found to be independent of system size, providing an unparalleled level of scalability. Our approach can be applied to the modelling of large neural networks, with greater performance, easier configuration and a high level of scalability.**

***Keywords*—neural network; neuromorphic; PNAA; Caenorhabditis Elegans; reconfigurable hardware**

I. INTRODUCTION

Moore's Law has, thus-far, provided a clear path for improving computational solutions and their associated power efficiency through the shrinking of transistor scales. However Moore's Law is fast approaching its long-predicted end and novel ways to continue the trend must therefore be developed [1]. Carver Mead drew parallels between the operation of neurons and that of transistors, concluding that greater efficiency may be achieved in transistor based systems by fundamentally changing the way the devices are used [2]. Mead went on to coin the term 'Neuromorphic Systems' to refer to computing platforms that are in some way inspired by naturally occurring neural processing systems, and since this original work there have been many attempts to implement various types of neuromorphic systems. In the last two decades, there has been an increase in research efforts to advance our understanding of brain behaviour using massively parallel processing architectures [1]. Many such systems use communications principles, such as packet switched networks, to produce a fully connected and therefore highly configurable neural architecture. Full connectivity between all neurons provides maximum flexibility, at the potential price of redundancy in unused connections. While beneficial for investigating new topologies or training regimes, this exhaustive architecture can result in large hardware redundancies. The internal routing often presents design challenges, and contributes significantly to the scale or power consumption of the final device. Drawing inspiration from nature, this work proposes a new reconfigurable neural architecture that is optimised for predominantly *locally connected* network implementations. Section III outlines existing hardware neural arrays, comparing their power consumption and implementations. The relevant approaches for modelling neurons are discussed in section II, where the key aspects, features and limitations of the various techniques are reviewed. These concepts are then encapsulated in a proposed programmable neuron array architecture (PNAA) in section IV. Finally Section V details a *C. elegans* locomotive model implementation using the PNAA architecture provided in Section IV, with the validation of the results against previous work and a discussion given in section VI.

II. MODELLING OF NEURONS

Representing the basic actions of a neural network in hardware requires the selection of an appropriate and representative model for the particular action or mechanism of interest. There are a significant number of models available, and a large subset of these see frequent use in neuromorphic systems. It is important to note that neuromorphic hardware is using a representation of a neuron (and synaptic behaviour) to provide a suitably accurate implementation, not simply for simulation purposes. Cases that require detailed neurological representations (such as applications interfacing directly to the nervous system itself) may require biophysically accurate details, allowing for the precise and complex emulation of neural systems. In contrast, neuromorphic systems that are designed for a single computational task may use a more abstract approach, resulting in greater computational efficiency at the cost of biophysical accuracy. A number of available models are shown in Figure 1, with the trade-off between biophysical accuracy and computational efficiency illustrated. One of the most famous biophysically accurate models was constructed by Alan Hodgkin and Andrew Huxley, who were exploring the relationship between chemical and electrical signalling within a squid (*Loligo forbesi*) giant axon [3]. This model has had a large impact in the field of neuromorphic systems [4], however the use of this model is limited because it is both computational expensive and does not scale well for simulations of many neurons at a time. A significantly simpler model, the binary model, was first developed in 1943 by McCulloch and Pitts [5]. In this model, each neuron has one or more inputs and a single output. The inputs are summed and

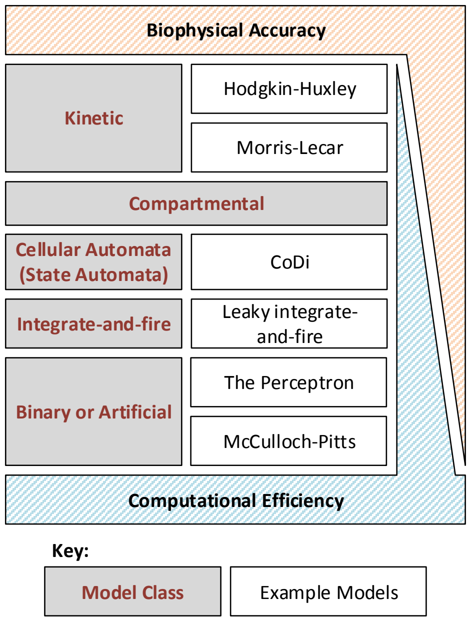


Fig. 1: The trade-off between biophysical accuracy and computational efficiency for a number of model classes.

compared with an internal threshold that, when exceeded, activates the output. This economical approach to neural dynamics is highly efficient, but less biologically relevant. There are some models that appear to exist outside of the scale shown in Figure 1, such as the Izhikevich model, which uses bifurcation theory to greatly simplify the Hodgkin-Huxley dynamics [6]. The resultant model is highly efficient while providing a good approximation of the bursting dynamics seen within nature, however, the accuracy of the temporal morphology of individual action potentials is forfeit. Different models will require different communications hardware support. Bio-physically accurate models may require attenuation or dendrite emulation as an inherent feature of their connection implementation. Spiking neural networks will produce sparse pulse trains which must be transmitted with known timings and amplitudes, while binary systems will only require the communication of a single bit. The model selection for any neuromorphic system must therefore be considered before the interconnect technology is designed. The trade-off for the modelling of neurons depends partly on the scale of the network to be analysed. For example if the number of neurons is low, the scope for using detailed biophysical models may be justified, however as the number of neurons moves into the 100s (and beyond) the simulation time required may be difficult to justify. An effective understanding of biological behaviour in neural networks can be achieved by using a simplified approach to modelling neurons either in simulations or on hardware emulators [7]. The method of choice for implementing neural networks can therefore be made in a pragmatic fashion based on a trade-off between accuracy and speed, depending largely on the scale and size of the network in question.

III. Implementing Neural Architectures

The implementation of connectivity in neuromorphic hardware is device specific. For example, the SpiNNaker platform has the ability to emulate massively parallel neural networks. It uses a packet switched asynchronous network, based on a fabric of ARM9 cores on dedicated multi-core processors [8], [9] and [10]. The SpiNNaker architecture is highly flexible, however it suffers from excessively high power consumption when compared to other approaches. SpiNNakers key strength is in its ability to handle arbitrary local and global connections, in other words it supports full arbitrary connectivity between any neurons, and it has been used in a range of applications including tasks such as vision systems [11]. Other example neuromorphic systems are TrueNorth and Neurogrid. TrueNorth is a platform whereby (local) short-distance communications are handled by an on-chip 64K crossbar array and asynchronous routers, while (global) long-distance communications are handled by protocols for intra-chip connections [12]. In contrast, Neurogrid is a mixed analogue-digital implementation that operates in the deep sub-threshold region of the semiconductor device. Synaptic inputs between spatially neighbouring neurons are distributed using a custom local arbor system, or a multi-cast router [13]. A common denominator of neuromorphic platforms is the flexibility to handle arbitrary levels of connectivity between any and all neurons. Whilst this is a clear advantage in general, for specific examples this can lead to less efficiency. A key approach to manage this issue is the implementation of a neural fabric that can be optimally configured for different scales of connectivity, whilst retaining the reconfigurability of the large scale neuromorphic implementations.

A. *Modelling Connectivity*

In order to understand the connectivity in hardware neural networks we can apply 'Rent's Rule' to relate the internal (local) and external (global) connections in the network. Rent's Rule was described in [14] however the basis was first explained earlier in [15]. More specifically, Rent's rule is the relationship between the number of terminals (T) and internal components, such as logic gates, (g) with the equation given in Eq. 1.

$$T = tg^p \tag{1}$$

Where t and p are variables that define the connectivity of the network in question and p <1. The relation can be used to indicate the level of connectivity, for example in neural networks, in which the terminals are synapses and the gates are neurons. Understanding the value of p and how it correlates to a network's connectivity can be used to optimize the resulting hardware implementation. An important aspect of Rent's rule is that if the equation is re-framed in terms of logarithms [16], the relationship between T and g becomes linear when both axes are expressed in logarithmic terms as shown in Eq. 2.

$$log(T) = log(t) + p * log(g) \tag{2}$$

This results in an offset value *log(t)* and the slope (in the log domain) determined by the Rent coefficient p. One of the interesting observations in nature is that this coefficient tends to be the same for all neural structures. Typically it is about p = 0.75 for both lower order animals such as the *C. elegans*, and human brains. A random circuit, however has a rent value of p = 1 [16].

B. Hardware Comparison

A comparison of SpiNNaker, Neurogrid and TrueNorth's relative power per neuron may be seen in Table I. SpiNNaker is largely simulation focused, using conventional processors to emulate its neural models. Both Neurogrid and TrueNorth, however, are hardware implementations and have recognised a significant power efficiency as a result. This supports the theory that such dedicated hardware may provide low-power neuromorphic processing solutions. All three of these systems make use of packet-based communications resulting in fully connected systems (i.e $T = 1$) that support the emulation of any network topology. From a practical perspective, the use of spiking neuron networks is an efficient mechanism for modelling neural networks, and is particularly appropriate for the hardware implementation on digital platforms such as FPGAs. Given the locally connected nature of biological networks it may be possible to improve the capabilities of the packet-based systems by exploring the effect of local vs global connectivity.

TABLE I: Power usage per neuron in three of the main hardware neural network systems.

System	Quoted Power (W)	Quoted Neuron Count	Power Per Neuron (μW)
SpiNNaker	0.75	17,000	44.12
Neurogrid	3.10	1,000,000	3.10
TrueNorth	0.65	1,000,000	0.65

IV. Existing Programmable Neural Array Architecture

In this section we demonstrate the limitation with an existing programmable neural array that exploits local connectivity, and we define our improved design that overcomes such limitations.

A. Locally Connected Architectures

An existing architecture for implementing locally connected neural arrays is that of Bailey [7]. In this approach the forming of networks of neurons may be implemented by connecting them all to a shared bus. If each neuron is assigned an address it is possible to ensure that only one neuron drives the bus at at a time using a global controller [7][17]. However, this implementation has a fundamental drawback that the address space must be accessed at every simulated time-step. The implication of such an approach is that two clocks must be implemented, the internal system clock, which drives the address unit, and the simulation clock, which marks off a single simulated time-step. Since every address (1 to n) must be accessed in each system step, the simulation clock frequency, F_c, is limited by the internal system clock frequency, F_i, as shown in Equation 3. This means that the simulation clock frequency is directly connected to the network size, making this design unscaleable.

$$F_c = \frac{F_i}{n} \tag{3}$$

An alternative solution is to consider the basic structure of multilayer feed-forward neural networks, which are widely used in a number of applications [18], [19], [20], [21]. In this type of system the synapses and neurons are brought together in what is referred to as a 'node'. These nodes can be arranged in layers, with each layer connected to the previous and next layer by two separate buses. Input/Output (IO) units are situated on the extremities of each layer and a full layer of IO units is added as the first and last layers of the system. Each row is assigned a unique address and a global address controller is used to dictate which row of nodes are driving the buses at any given time. An internal register in each node controls which addresses the node receives input from, allowing a node to connect to any of its neighbours in the previous layer. Each node can operate in a number of modes, such as forward, reverse or pass-through, depending on the requirements of the network. This system allows a designer to add as many layers as they desire with only the internal system clock propagation delay acting to limit the simulation clock rate. The simulation clock frequency is limited by the number of nodes in a single layer. The internal clock frequency is limited by the node propagation delay, which may differ between network implementations if pass-through nodes are utilized.

B. 2-Dimensional Hardware Architecture

We propose a 2-Dimensional architecture that overcomes the issues with scalability in locally connected networks that is present in the work of Bailey [7]. This Programmable Neural Array Architecture (PNAA), shown in Figure 2, expands the idea of locally connected networks into two-dimensions. The system is made up of two distinct elements, the node, which is an arbitrary neuron implementation and the IO block, operating as a connection to external actuators and sensors. Nodes are connected in loops, as shown in Figure 2. All nodes on a loop can communicate with one another and are assigned an address within the loop, with the largest loop dictating the

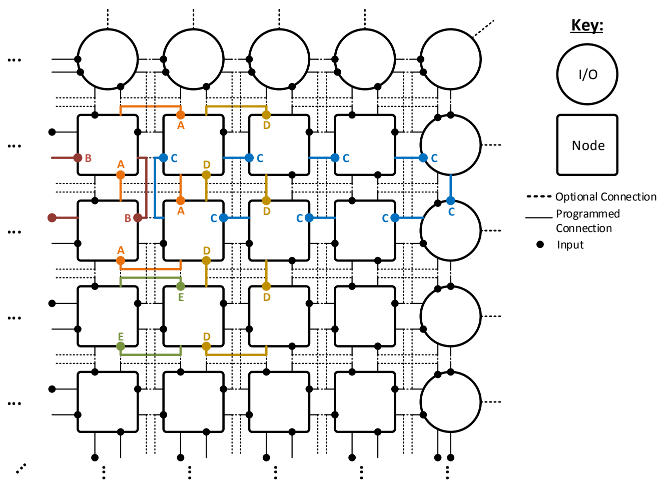


Fig. 2: Infrastructure of the PNAA, showing a number of 'loop' configurations labelled A to E. The circular nodes are the IO blocks that surround all edges of the architecture, the square nodes represent the processing, or neuron, blocks. Potential connectivity is indicated by the dashed lines and in the example loops given actual connectivity is indicated by a solid line.

largest address required. The system clock speed is dependent on the network and may be loosely associated to the largest number of connections that the most connected neuron requires to or from itself. In Figure 2 loop 'C' is seen to be the largest connected loop, with 8 units contained within its domain. The system clock must therefore run at least 7 times slower than the internal clock, regardless of the number of neurons used overall. The loops are implemented as a ring of shift registers, with a register located at each node input, meaning that the propagation delay remains fixed regardless of loop size. This architecture can mimic a fixed-width layered network by simply assigning loops down the full length of each column. This architecture supports two-dimensional scaling which is independent of the system clock timings. A pseudo-global connection can be achieved, with any node located a maximum of one node away from any other node, and a single neuron can send and receive in four directions at the same time allowing for rapid transmission through a network. This architecture can support locally connected networks with a small number of global connections.

C. Intermediate Representation

In order to produce a scalable and practical PNAA it is important to consider how a particular network may be synthesised onto hardware. There are a number of different methods for high-level representation of a network. An intermediate translation must be defined to allow a user to move from such high-level descriptions to a physical mapped solution and this often reflects the physical architecture through its defined structure. A number of different connection options are shown in Figure 3 as an example of the available connection routing. Each node can support four connections, with one starting on each face of the node. These optional connections are used to form loops of nodes, allowing data-flow between neurons.

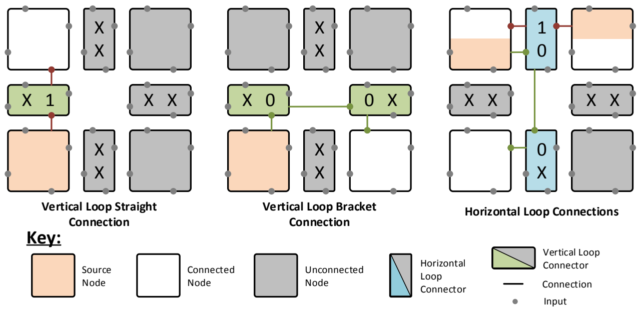


Fig. 3: Examples of connection options for a node in the PNAA, the interconnect blocks may be configured for X = not used, 1 = straight and 0 = clockwise routing patterns. This enables a programmable connection path between a source node and a connected node.

The implications of the interconnect options offered by this architecture must also be considered when mapping a high-level design, and inherent to this is the concept of locality between neurons. For two neurons to be connected they must exist within the one-another's neighbourhoods and it is therefore up to the synthesis engine to find an arrangement of nodes which results in all nodes sitting within the neighbourhoods of their connected nodes. This task becomes a fitting problem, similar to that performed by FPGA synthesis engines. Neurons sitting on the same row or column will share in the row or column neighbourhood respectively. This can be used to create layers of neurons, allowing a simple implementation of feed-forward neural networks.

D. Configuration

Each node is connected to a set of interconnect switches that can be configured to route connections to the neighbouring nodes in a North, East, South, West fashion. The PNAA has a rectangular structure and so the configuration process may be split into columns, where each column starts and stops with an IO block. Each neuron has three standard configurable parameters, the input weights, the input bias and the threshold. The bias and threshold values are singular for each neuron block, whereas the weights are a vector with an entry for each possible connected input.

V. Results from a *C. elegans* Locomotive Model

Section IV-B has provided an overview for a new type of PNAA, demonstrating the fundamental principles by which locally connected neural networks may be implemented in an efficient fashion. In this section we demonstrate this architecture using a working model of a biological system.

A. Overview of the C. elegans locomotory System Model

The locomotory system of the free living nematode *Caenorhabditis Elegans* (*C. elegans*) was used demonstrate the ability of this programmable neural array to replicate a real neural network that exhibits a high level of local connectivity. The *C. elegans* is a transparent free-living nematode with a generation time of about 3.5 days [7]. With only 302 neurons in its nervous system, the *C. elegans* is the only organism to have had its connectome (the neuronal 'wiring diagram') fully mapped [22]. The relative simplicity, and high level of local connectivity of the *C. elegans* locomotory nervous system makes it a good naturally occurring network against which artificial solutions may be verified. Subsets of neurons within the *C. elegans* can be identified by the incorporation of appropriate promoter sequences to drive expression of fluorescent proteins. A micrograph image is given in Figure 4 where the animals were transgenic for the expression of two fluorescent proteins DS-red and GFP (green fluorescent protein). The subset of neurones expressing the neuropeptide FLP-8 fluoresce green, and those containing the putative receptor fluoresce red. To demonstrate the operation of our model at a network level we present simulations of the locomotory system of *C. elegans* on the PNAA described in Section IV-B. The structure of the *C. elegans* locomotory system is regular, and there are very few variations from animal to animal within the species. In total there are 86 neurons and 180 synapses [22]. There are six different types of neuron and four different types of synapse. Further, the locomotory system of the *C. elegans* has already seen common usage in both the MBED Cellular Automata Neuron model by Claverol et al [23] and by Bailey [7]. The locomotion system that was modelled by Bailey used a direct implementation VHDL model [17], and was based upon work performed by Claverol [23]. This model showed that an artificial neural network was capable of generating the same motor neuron patterns as had been observed in living

C. elegans. The locomotive model of the *C. elegans* may be divided into small, repeated segments (as shown in Figure 5a). Each segment exhibits a high order of local connectivity, with relatively sparse connections from one segment to another. By utilising the local connectivity seen within this model, it may be implemented using the PNAA system described in IV-B (an example mapping is shown in Figure 5b). The largest loop within this implementation, which lies along the width of a single segment, contains only 10 neurons. This system therefore requires only 9 clock cycles per simulated system time-step regardless of the number of segments implemented. This represents a $9\times$ speed increase in the 10 segment system when compared to the direct implementation of Bailey [7]. The sensory AVA and AVB neurons are actually two single neurons that connect along the entire length of the *C. elegans*. These neurons have no incoming connections from within the network and they may therefore be divided into a number of identical neurons that are all driven by the same external stimulus, as shown in Figures 5a and 5b.

B. *Validation of the C. elegans Model using bi-phase stimulation*

In order to validate the mapping of the model onto the PNAA, an RTL simulation was performed for a hypothetical *C. elegans* with 10 segments. Forward and backwards locomotion was achieved using a bi-phasic control signal which produced the appropriate stimulation to the head and tail of the model, the resulting motor neuron activations are shown in Figure 6a. The top trace (Control) shows the control signal, resulting in motor neuron activation along both the ventral and dorsal sides in a sinusoidal fashion. The result of the bi-phasic control signal is an alternating forwards and backwards locomotion. The top bank of traces represent the Dorsal motor neurons and the lower bank the Ventral motor neurons, activation is shown as a solid block of colour due to the high frequency oscillations

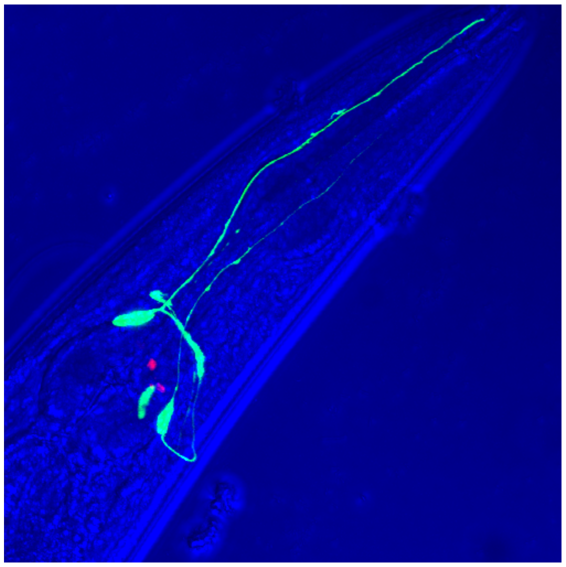
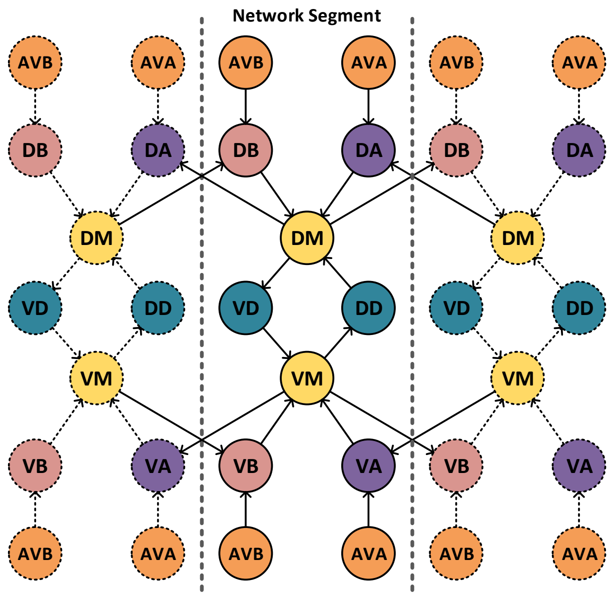
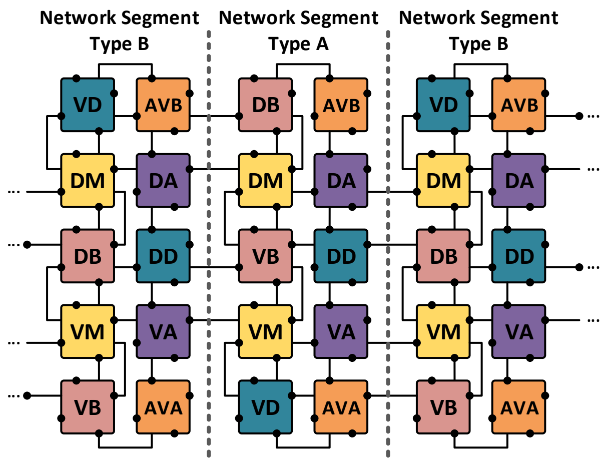


Fig. 4: Fluorescent subset of neurones within *C. elegans* showing projections running alongside the pharynx. The overall structure of the head region can be seen in the blue, overlaid transmission. *Image courtesy of John Chad and Ilya Zheludev*



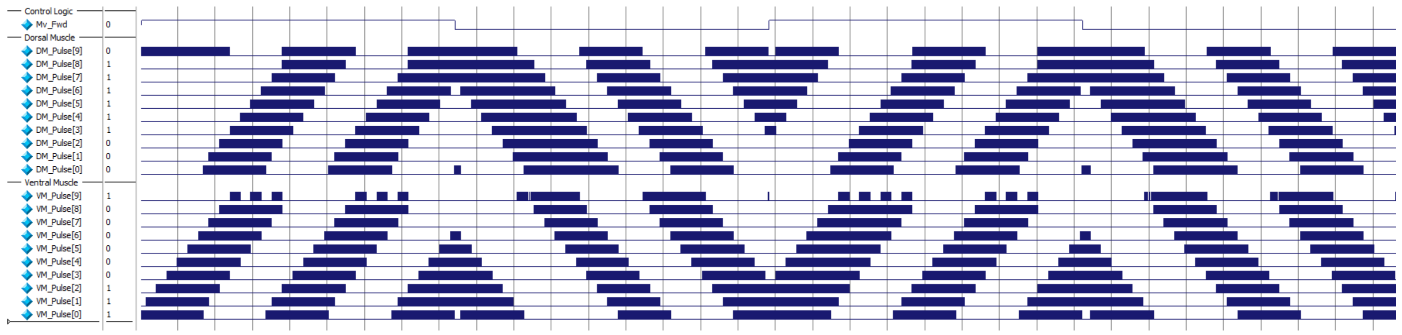
(a) The *C. elegans* locomotion model inspired by Enric Claverol [23], shown to be divisible into repetitive segments in each column. It is clear that there is a high level of local connectivity within each column, or section, with relatively sparse global connectivity between all neurons.



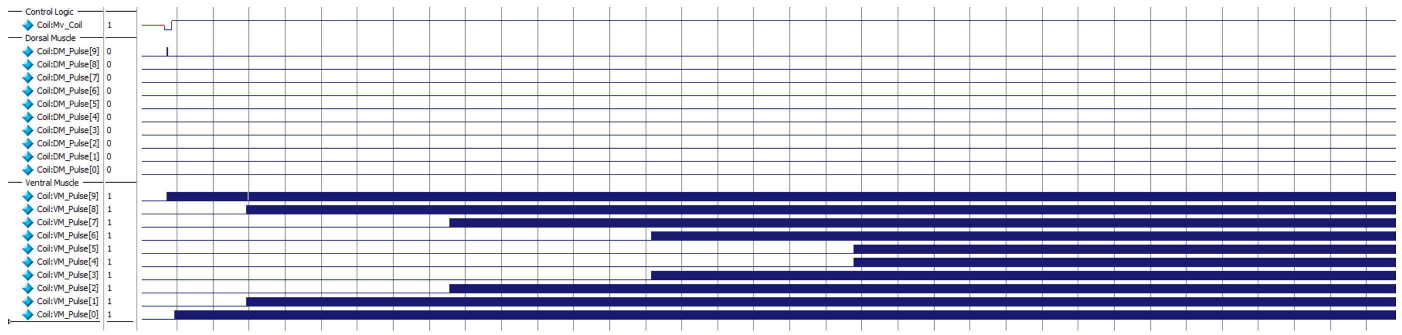
(b) The segment from Figure 5a implemented on the PNAA architecture with full scalability, extending the model to an arbitrary number of segments is a simple case of inserting more columns, no changes to the fundamental architecture are required.

Fig. 5: *C. elegans* locomotion model and equivalent architecture

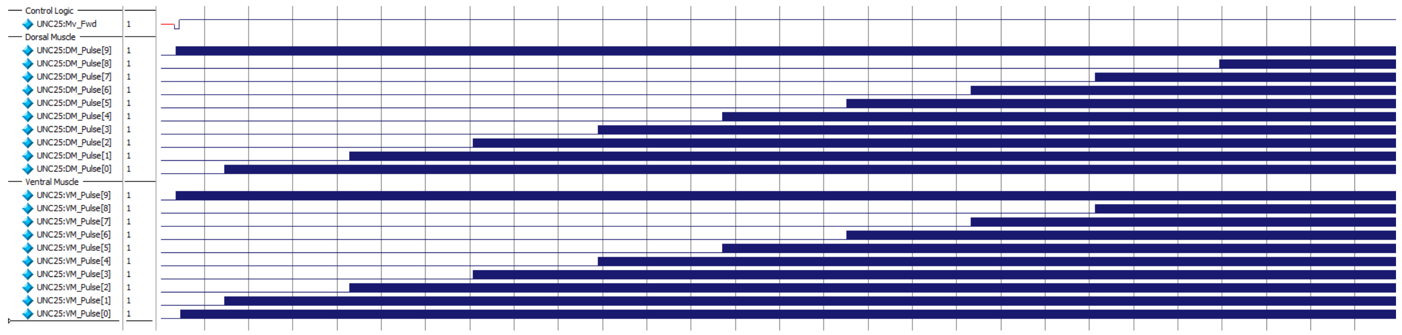
of the neurons outputs. Neurons numbered 0 are the head end, and 9 the tail end. As the stimulus pulse is alternated from head to tail and back again there is a propagating wave of muscle activation along the length of the model. This wave produces a sinusoidal locomotion action. Vertical lines are drawn every 500 milliseconds on the x-axis. Activity begins on the Ventral side at the head (VM0) and propagates steadily down through the Dorsal muscles reaching the tail end (VD9) in approximately 2900 milliseconds. Comparing this result to that of Bailey and Claverol confirms the same propagating



(a) Forward and backward locomotion behaviour of a 10 segment *C. elegans* model. The top trace (Control) shows a bi-phasic control signal, generating stimulus applied to both the head and tail of the model, this results in motor neuron activation along both the ventral and dorsal sides in a sinusoidal fashion. The result of the bi-phasic stimulus is an alternating forwards and backwards locomotion.



(b) Coiling behaviour of a 10 segment *C. elegans* model. A single control signal was used to trigger stimulus applied to both head and tail end on the ventral side of the model, this results in motor neuron activation along the ventral side only, producing a coiling action towards the centre.



(c) UNC25 knockout behaviour of a 10 segment *C. elegans* model. A constant forward stimulus is applied to the model, this results in motor neuron activation along the ventral and dorsal sides, producing a shrinking action towards the head end.

Fig. 6: *C. elegans* simulation results

waves are observed in all models. This evidence shows that the PNAA model is behaving correctly when compared against previous work.

C. Validation of the C. elegans Model Coiling Behaviour

A second behaviour explored in earlier models was *coiling*. Figure 6b shows the activation of the Dorsal and Ventral motor neurons in response to a stimulus applied to the Ventral side of both the head and tail simultaneously. Motor neuron activation begins at each end of the model and propagates towards the centre. This muscle activation results in a coiling motion *towards* the stimulus, which is consistent with the models of both Bailey and Claverol and further confirms the effectiveness of the PNAA model when compared to previous work.

D. Validation of the C. elegans Model UNC25 knockout Behaviour

A third behaviour explored in earlier models was when the UNC25 gene knockout mutation is applied to the model. In this circumstance the Dorsal and Ventral motor neurons response to a constant stimulus is modified. Motor neuron activation begins on both the Dorsal and Ventral sides of the model and propagates from head to tail, this muscle activation results in a shrinking behaviour as each of the motor neurons lock into a constant firing mode. This behaviour manifests itself in both the Dorsal and Ventral motor neurons firing in the same sequence leading to all the neurons firing simultaneously This result is consistent with the models of both Bailey and Claverol and further confirms the effectiveness of the PNAA model when compared to their previous work.

VI. Discussion of results

A. Introduction

Given the range of possible modelling approaches for systems such as the *C. elegans* locomotory system, there is always a question over the correct level of detail to accurately reflect the true biological behaviour in simulation. In many cases it comes down to a subjective choice based on software preferences, hardware availability and ease of use. However, it is possible to have some basis for comparison using four fundamental criteria; biological accuracy, scalability, performance (which could be in terms of speed, area or both) and flexibility. Therefore we will consider the three fundamental parameters of scalability, performance and flexibility in a qualitative manner, using quantifiable metrics wherever possible. In each case the proposed approach will be compared with the previous hardware implementations of Claverol [23] and Bailey [7].

B. Biological Accuracy

As has been demonstrated in section V, the basic locomotory behaviour, coiling and the effect of UNC-25 knockout has been validated in the model as implemented on the PNAA architecture. These results are consistent not only with the observed biological behaviour, but also with the hardware models developed by Claverol [23] and Bailey [7]. There are more mutation conditions that could be evaluated however these three fundamental tests are validation of the efficacy of the model as they show the independent neuron behaviour under locomotion, the asymmetric behaviour of coiling and the shrinking behaviour of the UNC-25 knockout. This clearly demonstrates the fundamental behaviour of the model is correct.

C. Scalability

The key advantage of our PNAA approach is its 2-dimensional nature and the ability to scale the model in both axes arbitrarily. Using the software interface, the design is abstracted to a network level making it very easy to generate and programme a network accordingly. The connectivity is assumed to be highly local, and while this will not provide the full connectivity for completely arbitrary networks, global connections can be defined in addition to the core local networks. This is analogous to the approach used in the well known GALS systems. Importantly, this feature permits the rapid scaling of models such as *C. elegans* to an arbitrary number of segments without any change in network architecture.

D. Performance (speed/resources)

From a resource perspective the PNAA is an efficient method, by using a level of granularity that maps from the neuron network level directly to hardware, with sufficient local connectivity to ensure that the model utilizes the connectivity available. The intrinsic efficiency of the clocking approach leads directly to a speed improvement of $9\times$ in the 10 segment system when compared to the direct implementation of Bailey [7]. This is primarily due to the efficient connectivity in this implementation that limits the loop size, therefore for locally connected networks, with relatively low numbers of connections, a performance improvement can be dramatic as in this case. The largest loop within this implementation, which lies along the width of a single segment, contains only 10 neurons. This system therefore requires only 9 clock cycles per simulated system time-step regardless of the number of segments implemented.

E. Flexibility

The abstraction of the model to a hardware architecture makes the whole approach extremely flexible from a network definition and programming perspective. Unlike the fully coded approaches of Claverol and Bailey. The library based approach of Bailey meant that a standard synthesis software tool would be required to configure the network, making the ultimate design not necessarily representative of the locally connected nature of the nervous system being analysed. Using the software interface to define the connectivity and the weights makes the process straightforward, and most importantly the definition of an underlying hardware fabric gives a known structure and fundamental behaviour that is consistent and simple to extend or modify.

F. Future Work

Using the PNAA as a test platform, the impact of hardware enforced locality on neural systems may be easily assessed. Classic artificial neural models, such as convolutional neural networks, demonstrate high measures of locality and further mapping of these systems onto the PNAA will provide meaningful insight into how a networks underlying structure and it's connection requirements relate to one another. A digital model of the *C. elegans* is also under construction, using physical constraints and a soft-body IK model to support simulations of the nematodes locomotive response to neural stimulus.

VII. Conclusion

With Moore's Law reaching its inevitable end, a new processing paradigm must be developed. A number of neurological hardware implementations have been identified and compared, leading to the observation that most hardware implementations are large systems with support for full global connectivity. However, many naturally occurring neurological systems are locally-connected specialised systems. A constraint focused approach has been outlined, resulting in a new programmable neural array architecture or PNAA. Sections IV outlines our novel approach to connectivity within neural hardware. Focusing on predominantly local resources while allowing a number of neurons to connect to any other neuron within the system, this architecture can recognise speed improvements over other address based systems. A form of synthesis or intermediate representation is discussed, showing how a design may be implemented on such a reconfigurable architecture. Further, an implementation of the *C. elegans* locomotive model is developed, validating our architecture and resulting in a speed increase of around $9\times$ that of previous solutions when modelling a 10 segment system. Our approach can be applied to the modelling of many different neural networks, with a potential for greater performance, easier configuration and most importantly a high level of scalability. Such systems can provide a novel solution to the problems associated with conventional processing methods leading to new and efficient processing technologies.

Acknowledgements

The authors gratefully acknowledge the permission to reproduce the image of the *C. elegans* nervous system taken by Dr Ilya Zheludev working with Professor John Chad at the University of Southampton.

References

[1] D. Monroe, "Neuromorphic computing gets ready for the (really) big time," *Communications of the ACM*, vol. 57, no. 6, pp. 13–15, 2014.

[2] C. Mead, "Neuromorphic electronic systems," *Proceedings of the IEEE*, vol. 78, no. 10, pp. 1629–1636, 1990.

[3] A. L. Hodgkin and A. F. Huxley, "A quantitative description of membrane current and its application to conduction and excitation in nerve," *Journal of Physiology*, vol. 117, pp. 500–544, Jan. 1952.

[4] W. A. Catterall, I. M. Raman, H. P. C. Robinson, T. J. Sejnowski, and O. Paulsen, "The Hodgkin-Huxley Heritage: From Channels to Circuits," *Journal of Neuroscience*, vol. 32, no. 41, pp. 14 064–14 073, Oct. 2012.

[5] W. S. McCulloch and W. Pitts, "A logical calculus of the ideas immanent in nervous activity," *The Bulletin of Mathematical Biophysics*, vol. 5, no. 4, pp. 115–133, Dec. 1943.

[6] E. Izhikevich, "Simple model of spiking neurons," *IEEE Transactions on Neural Networks*, vol. 14, no. 6, pp. 1569–1572, Nov. 2003.

[7] J. A. Bailey, "Towards the neurocomputer: An investigation of VHDL Neuron Models," PhD Thesis, University of Southampton, 2010.

[8] S. B. Furber, D. R. Lester, L. A. Plana, J. D. Garside, E. Painkras, S. Temple, and A. D. Brown, "Overview of the SpiNNaker System Architecture," *IEEE Transactions on Computers*, vol. 62, no. 12, pp. 2454–2467, Dec. 2013.

[9] S. B. Furber, F. Galluppi, S. Temple, and L. A. Plana, "The SpiNNaker Project," *Proceedings of the IEEE*, vol. 102, no. 5, pp. 652–665, May 2014.

[10] E. Painkras, L. A. Plana, J. Garside, S. Temple, S. Davidson, J. Pepper, D. Clark, C. Patterson, and S. Furber, "SpiNNaker: A multi-core System-on-Chip for massively-parallel neural net simulation," in *Proceedings of the IEEE 2012 Custom Integrated Circuits Conference*, vol. 4. IEEE, Sep. 2012, pp. 1–4.

[11] T. Kawasetsu, R. Ishida, T. Sanada, and H. Okuno, "Live demonstration: A hardware system for emulating the early vision utilizing a silicon retina and SpiNNaker chips," in *2014 IEEE Biomedical Circuits and Systems Conference (BioCAS) Proceedings*, Oct. 2014, p. 188.

[12] F. Akopyan, J. Sawada, A. Cassidy, R. Alvarez-Icaza, J. Arthur, P. Merolla, N. Imam, Y. Nakamura, P. Datta, G.-j. Nam, B. Taba, M. Beakes, B. Brezzo, J. B. Kuang, R. Manohar, W. P. Risk, B. Jackson, and D. S. Modha, "TrueNorth: Design and Tool Flow of a 65 mW 1 Million Neuron Programmable Neurosynaptic Chip," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 34, no. 10, pp. 1537–1557, Oct. 2015.

[13] B. V. Benjamin, Peiran Gao, E. McQuinn, S. Choudhary, A. R. Chandrasekaran, J. Bussat, R. Alvarez-Icaza, J. V. Arthur, P. A. Merolla, and K. Boahen, "Neurogrid: A Mixed-Analog-Digital Multichip System for Large-Scale Neural Simulations," *Proceedings of the IEEE*, vol. 102, no. 5, pp. 699–716, May 2014.

[14] M. Y. Lanzerotti, G. Fiorenza, and R. A. Rand, "Microminiature packaging and integrated circuitry: The work of E. F. Rent, with an application to on-chip interconnection requirements," *IBM Journal of Research and Development*, vol. 49, no. 4.5, pp. 777–803, Jul. 2005.

[15] B. Landman and R. Russo, "On a Pin Versus Block Relationship For Partitions of Logic Graphs," *IEEE Transactions on Computers*, vol. C-20, no. 12, pp. 1469–1479, Dec. 1971.

[16] D. S. Bassett, D. L. Greenfield, A. Meyer-Lindenberg, D. R. Weinberger, S. W. Moore, and E. T. Bullmore, "Efficient Physical Embedding of Topologically Complex Information Processing Networks in Brains and Computer Circuits," *PLOS Computational Biology*, vol. 6, no. 4, pp. e1 000 748+, Apr. 2010. [Online]. Available: http://dx.doi.org/10.1371/journal.pcbi.1000748

[17] J. A. Bailey, R. Wilcock, P. R. Wilson, and J. E. Chad, "Behavioral simulation and synthesis of biological neuron systems using synthesizable vhdl," *Neurocomputing*, vol. 74, no. 14-15, pp. 2392–2406, Jul. 2011.

[18] M. S. Z. Azalan, M. P. Paulraj, and S. bin Yaacob, "Classification of hand movement imagery tasks for brain machine interface using feed-forward network," in *2014 2nd International Conference on Electronic Design (ICED)*, vol. 3. IEEE, aug 2014, pp. 431–436.

[19] B. Deng, M. Zhang, F. Su, J. Wang, X. Wei, and B. Shan, "The implementation of feedforward network on field programmable gate array," in *2014 7th International Conference on Biomedical Engineering and Informatics*, no. Bmei. IEEE, oct 2014, pp. 483–487.

[20] D. Upadhyay, N. Tarun, and T. Nayak, "ANN based intelligent controller for inverted pendulum system," in *2013 Students Conference on Engineering and Systems (SCES)*. IEEE, apr 2013, pp. 1–6.

[21] P. Arce and P. Arce, "Forecasting high frequency financial time series using parallel FFN with CUDA and ZeroMQ," *Information and Telecommunication Technologies (APSITT), 2012 9th Asia-Pacific Symposium on*, no. Nov, 2012.

[22] J. G. White, E. Southgate, J. N. Thomson, and S. Brenner, "The Structure of the Nervous System of the Nematode Caenorhabditis elegans," *Philosophical Transactions of the Royal Society B: Biological Sciences*, vol. 314, no. 1165, pp. 1–340, Nov. 1986.

[23] E. T. Claverol, "An event-driven approach to biologically realistic simulation of neural aggregates," Ph.D. dissertation, University of Southampton, 2000.