Scalable Interconnect Strategies for Neuro-glia Networks using Networks-on-Chip

George Stephen Martin (BEng)

Faculty of Computing, Engineering and the Built Environment

A thesis submitted for the degree of Doctor of Philosophy (PhD)

August 2018

"I confirm that the word count of thesis is less than 100,000 words"

Abstract

Hardware has become more prone to faults, due to wear-out and faults caused during the manufacturing process. The reliability of hardware is becoming more dependent on the ability to continually adapt to faults and current fault tolerant approaches are susceptible to faults. A computational model of biological self-repair in the brain, derived from observing the distributed role of astrocytes (a glial cell found in the mammalian brain), has captured self-repair within neural networks; these are known as neuro-glia networks.

Astrocytes have been shown to facilitate biological self-repair in silent or near silent neurons in the brain by increasing the Probability of Release (PR) in healthy synapses. Astrocytes modulate synaptic activity, which leads to increased or decreased PR. To date, this has been proven with computational modelling and therefore, the next step is to replicate this self-repair process in hardware to provide self-repairing electronic information processing systems. A key challenge for hardware neuro-glia networks implementation is the facilitation of scalable communication between interacting neurons and astrocyte cells. There are large volumes of neurons/ astrocytes with different communication patterns and this network is viewed as a two-tiered network:

1. High speed temporal spike event (neural network)
2. Low speed numerical inositol trisphosphate information exchange (astrocyte network).

This thesis addresses the key challenge of providing scalable communication for a neuro-glia network with low-level Networks-on-Chip (NoC) topologies. This network supports astrocyte to neuron/synapse communication at a local level and astrocyte communication at a global level i.e. the astrocyte network. The astrocyte process is inherently slow, thus a ring topology exploits this slow change and sacrifices high throughput for a low area overhead, this is analogous to the astrocyte process. This astrocyte was applied in hardware and results demonstrate that novel ring topology provides a trade-off between low area/interconnect wiring overhead whilst supporting realistic communication speeds for both the slow-changing data between astrocytes and the higher throughput neuron networks.

NOTE ON ACCESS TO CONTENTS

I hereby declare that with effect from the date on which the thesis is deposited in Research Office of the Ulster University, I permit

1. The Librarian of the University to allow the thesis to be copied in whole or in part without reference to me on the understanding that such authority applies to the provision of single copies made for study purposes or for inclusion within the stock of another library.

2. The thesis to be made available through the Ulster Institutional Repository and/or EThOS under the terms of the Ulster eTheses Deposit Agreement which I have signed.

IT IS A CONDITION OF USE OF THIS THESIS THAT ANYONE WHO CONSULTS IT MUST RECOGNISE THAT THE COPYRIGHT RESTS WITH THE AUTHOR AND THAT NO QUOTATION FROM THE THESIS AND NO INFORMATION DERIVED FROM IT MAY BE PUBLISHED UNLESS THE SOURCE IS PROPERLY ACKNOWLEDGED.

(George Martin)

Table of Contents

List of Figures

List of Tables

Abbreviations

2-AG	*Glutamate*
ANN	*Artificial Neural Network*
AP	*Action Potential*
ASIC	*Application Specific Integrated Chip*
CA^{2+}	*Calcium*
CPU	*Central Processor Unit*
CRC	*Cyclic Redundancy Check*
DARPA	*Defence Advanced Research Projects Agency*
DSE	*Depolarization-induced Suppression of Excitation*
EMBRACE	*Emulating Biologically-Inspired Architecture in Hardware*
e-SP	*Endocannabinoid-mediated Synaptic Potentiation*
FIFO	*First In First Out*
FPGA	*Field Programmable Gate Array*
FSM	*Finite State Machine*
GPU	*Graphic Processor Unit*
HANA	*Hierarchical astrocyte network architecture*
HICANN	*High Input Count Analog Neural Network*
H-NoC	*Hierarchical Networks on Chip*
HPC	*High-Performance Computer*
IF	*Integrate and Fire*
IP_3	*Inositol trisphosphate*
LIF	*Leaky Integrate and Fire*
LUT	*Look Up Table*
MLP	*Multilayer perceptron*
MPSoC	*Multi-Processor System-on-Chip*
NoC	*Networks-on-Chip*
PE	*Processing Element*
PISO	*Parallel In Serial Out*
PR	*Probability of Release*

PWM	*Pulse Width Modulator*
SANN	*Spiking Astrocyte-Neuron Network*
SLP	*Single Layer Perceptron*
SNN	*Spiking Neural Network*
SoC	*System-on-Chip*
SRM	*Spike Response Model*
SyNAPSE	*Systems of Neuromorphic Adaptive Plastic Scalable Electronics*
TMR	*Triple Mode Redundancy*
VC	*Virtual Channels*
VHDL	*Very High-Speed Integrated Circuit Hardware Description Language*

Acknowledgements

It’s been a long time coming and I’m sure those close to me will know how happy it makes me to finally be here, concluding a chapter of my life. It has been an enormous effort, like that of Sisyphus, but less dramatic.

First of all, I would like to pay homage and thank those directly involved with my PhD. Dr. Jim Harkin, Professor Liam McDaid, Dr. John Wade and Dr. Junxiu Liu, without their work, wisdom and support I would never have made it this far and for that, I am indebted to them. To my friends and colleagues within the ISRC, thank you sincerely for the support, for the friendship, for the coffee breaks and the football, which were sometimes the means to an end.

Dedication

To my friends and family, for every day I had doubts and every day that I doubted myself, I dedicate this to you, you kept me going. To my father, George Martin and my godfather Stephen Martin, your support kept my feet on the ground and made sure I never lost sight of what was important. To my late grandfather George Martin and grandmother Anne Martin, I owe a lot to you, you showed me how love and hard work can overcome humble means. To my mother Catherine Cosgrove, without you none of this would have been possible, and finally my sister Corrina and her children Lucas & Khloe, and my brothers Christopher, Carlos and Thomas.

"It is not the critic who counts; not the man who points out how the strong man stumbles, or where the doer of deeds could have done them better. The credit belongs to the man who is actually in the arena, whose face is marred by dust and sweat and blood; who strives valiantly; who errs, who comes short again and again, because there is no effort without error and shortcoming; but who does actually strive to do the deeds; who knows great enthusiasms, the great devotions; who spends himself in a worthy cause; who at the best knows in the end the triumph of high achievement, and who at the worst, if he fails, at least fails while daring greatly, so that his place shall never be with those cold and timid souls who neither know victory nor defeat."

- *Theodore Roosevelt*

Chapter 1: Introduction

1.1 Background

Computing has advanced every aspect of our species from exploring the universe, the curiosity rover on Mars [1] and Juno on Jupiter [2], to the human brain exploring how the brains processes information in a fast and efficient manner. Scientists and engineers look to biology for inspiration. Over recent years humans have pushed the boundaries of computing and the technology used for computation. Computational performance using central processor units (CPUs) allows computers to perform a large number of calculations sequentially, therefore, computers can be used to solve complex problems. The hardware is based on sequential computing using single core processors and it is therefore limited; it is power hungry and inefficient because it performs calculations one after another. Using multiple CPUs can allow computers to operate with some parallelism but it is limited. The brain uses its innate parallel infrastructure to process information and this is much faster and much more efficient for complex problem solving. More recently, research has been carried out to mimic how the brain processes information using the timing and frequency of spikes to encode information [3]. Neurons process information in highly parallel manner, harnessing huge amounts of processing power efficiently. Models have been derived to replicate and emulate neurons and how they process information, and although these models are evidently based on biology, (i.e. neurons and human brain performance) there is still a lot to be desired i.e. NN applications cannot compare to the brain in regards to pattern recognition challenges such as visual and speech recognition [4], [5] as they don't have the same performance, are inefficient in terms of both time and computational power. This is because neural network applications use supervised learning for classification, this requires training, and due to the vast amount of data it is mostly done offline. Training requires the user to input data and "train the network" by using labelled outputs, another downfall is that software approaches lack efficiency and use up computational resources. Neural networks typically have one application specified by the user.

SNN applications overcome the computational and performance bottlenecks of traditional NNs and are based on how the brain processes data. SNNs differ from traditional NNs as information is encoded in spikes, this is more efficient and analogous to how the brain processes information but again the network is trained offline before it can be deployed. There is also the issue that simulating SNNs on hardware is computationally intensive but Neuromorphic hardware such as TrueNorth aims to replicate inherent parallelism by using hardware which deviates from traditional Von Neumann architecture as each neuron has memory, computational power and a communication aspect. There are however promising real-world applications of SNNs in audio processing [6] and image processing [7].

Self-repair is highly sought after in electronic systems due to technology becoming much smaller in size (geometric scaling), resulting in electronic systems becoming more prone to faults [8]. The ability to repair and tolerate faults leads to more reliable and robust systems. To maintain functionality and increase operational lifetime of an electronic system, a method of fault tolerance or self-repair is required [9]. Current fault tolerant mechanisms are based on coarse grained redundancy and employ the use of a central repair-decision agent to either find faults or correct them, such as Triple Mode Redundancy (TMR), which is typically used in mission critical systems [10]. TMR is the process of replicating critical components and using a comparator to detect discrepancies. This vastly increases area overhead and relies heavily on spare or redundant computing resources. The key weaknesses of existing approaches is limited granularity and the lack of a distributed repair-decision mechanism.

SNNs typically run on specialised hardware, this hardware is typically deployed in harsh environments and therefore, the hardware must use a form of fault tolerance. Rather than look at the electronic hardware, an SNN has neurons connected to a number of inputs; the network is trained to classify this data based on the inputs and information is passed via a huge number of synapses. Although the SNN may be susceptible to electronic failures and faults e.g. a sensor failing; rather than becoming completely useless, it is possible to employ what can be considered as a graceful degradation.

Recent research has shown that biological traits such as fine grained repair and distributed repair-decision making are performed in the brain via networks of glia cells (astrocytes) [11], [12]. It has been said there are 100 billion neurons and one trillion glial cells, with a glia to neuron ratio of 10:1. Even though such estimations are debated [13], it is the inherent parallelism and features such as fault tolerance and low-power consumption that have us look to the brain for inspiration. It has been discovered that astrocytes within the brain mediate synaptic plasticity [14] and this allows astrocytes to actively increase or decrease the probability of release (PR) of a synapse. Self-repair can be observed across faulty synapses when the PR on healthy neuron-synapses is increased, i.e. firing activity can be repaired to pre-fault levels, even with a substantial amount of faults (80%) [12].

In particular, computational models of such repair have been successfully captured and applied to spiking neural networks (SNNs) [15]. An SNN is a neural network which uses the timing and frequency of spikes to propagate information; a spike train is used to encode information. The research in [15] demonstrates how low neuron firing activity can be repaired at fine-grained levels, i.e. the synapses. The healthy synapses can be strengthened which enables the neuron to regain functionality (spiking). This can result in self-repair across the spiking neural network. The cell responsible for self-repair has been identified as the astrocyte. Astrocytes are highly distributed within networks of neurons [15]. This neuro–glia network paradigm addresses the key self-repair requirements of fine granularity and distributed decision making and this sets the focus of this PhD research.

There are a number of difficult challenges to overcome in order to realise a neuro-glia network in hardware:

- The realisation of both local and global communications within a neuro-glia network. Using astrocytes and providing self-repair requires a hierarchy of communication.
- Applying self-repair within a neuro-glia network is two separate networks working in unison; an SNN and an astrocyte network. There are additional interconnect

requirements including additional complexity and communication. This is a neural network with the functionality of an SNN for classification and astrocytes for self-repair, therefore, this application of self-repair is applied within an existing SNN framework.

- The scalability requirements, as the network scales the communication infrastructure must support the additional processing and communication requirements. The timing of communication between neuron and astrocyte networks diverges; a neural network emphasises throughput. The astrocyte is a lot slower process which focuses on self-repair and supporting the network and synapses. The network must be able to accommodate both communication processes and keep these working in unity.

One of the key challenges in progressing neuro–glia networks to hardware is connecting a network of astrocytes with an SNN. Astrocytes connect to groups of synapses and also form large networks of their own. Emulating a neuro–glia network in hardware introduces additional challenges such as increased complexity of the network, increased wiring interconnect overheads to maintain communication within a network and dealing with different time scales between neurons and astrocyte. There, we seek a scalable method of interconnecting astrocytes to astrocytes, and astrocytes to neurons, i.e. realise a low area and power strategy.

Networks-on-Chip (NoC) has emerged as a scalable approach for interconnecting many cores on a single chip, and is based on network engineering approaches over physical wires, buses and crossbars [16]–[18]. Therefore, we can look to exploit this mechanism in a manner which facilitates scalable data exchanges between neurons, synapses and astrocytes. This thesis explores the following:

Objectives:

1. Can HW models of networks of glia cells be successfully scaled using network on chip architecture?”

2. Is a ring-topology suitable for astrocyte to neuron and astrocyte to astrocyte communication at the low level of a network on chip hierarchy?

3. Is this topology capable of enabling both fault tolerance capability in synaptic connections and overall network scalability?

1.2 Thesis Contributions

This thesis covers a substantial body of research into self-repair. This self-repair is an innate feature of the human brain, provided by astrocytes. This thesis proposes using bio-inspired hardware within an existing neural network framework using on-chip interconnect strategies. That is, research into how the astrocyte process works and communicates with neurons and combining the features of astrocytes with neural networks, more specifically SNNs and H-NoC. The resulting contributions are the initial steps into realising large scale neuroglia networks using existing features of SNNs combined with a digital astrocyte in hardware thus exploring neuro-glia networks using a scalable field-programmable gate array (FPGA) interconnect.

The contributions presented within this thesis are:

1. A novel NoC interconnect based on using a ring topology to communicate e –SP for local communication in a neuro-glia network between neurons and astrocytes. This trades off area for communication speed (Chapter 5).
2. Detailed analysis of area and scalability, in regard to keeping low hardware overheads using a ring-based topology using NoC (Chapter 5).
3. Applying this interconnect for local commutation between astrocytes and neurons within an existing framework: Hierarchical Network on Chip (H-NoC) (Chapter 5).
4. A novel NoC router for global communication in a neuro-glia network in astrocyte networks. The astrocyte router is based on using a ring topology and uses low level logic to average and communicate IP_3 to all associated astrocytes (Chapter 6).

5. An analysis of area, power and scalability using an astrocyte tile router for communication between astrocytes. A trade-off between the biological time scale and low overheads in hardware (Chapter 6).
6. Validating the NoC interconnect in FPGA hardware using an example mobile robotic car (Chapter 7).

1.3 Thesis Outline

The outline of this thesis is as follows:

Chapter 2 provides a review of self-repair, astrocytes and neurons, and their role in computational software and hardware. From the biochemical reactions to computational neural networks, i.e. from biology to hardware. This provides an insight into the biological and computational motivation behind self-repair.

Chapter 3 investigates the application of the NoC paradigm as a hardware interconnect capable of providing a large-scale neuro-glia network with desired scalability. The chapter initially reviews NoC features, from topologies to existing NoC solutions in hardware. A review of current methods of fault tolerance and self-repair as well as a review of mechanisms already in place to facilitate self-repair is also provided. There is also a review of biological self-repair and the aim of implementing this self-repair process in a Neuro-glia network. The latter part of the chapter focuses on the challenges of implementing a neuro-glia network using a NoC interconnect.

Chapter 4 highlights why self-repair is desirable and why to date, current self-repair or fault tolerant approaches come at large overheads e.g. TMR. In particular, biologically inspired self-repair has been recently shown to provide fine grained and distributed self-repair using computational models, but due to limited scalability and performance constraints, require more creative, high performance and robust means to explore further.

Chapter 5 outlines the interactions and communication exchange between neurons and astrocytes, this being a local communication exchange. This chapter contributes a novel low-level NoC ring topology which facilitates astrocyte to neuron communication. The main focus is on the steps taken to implementing a ring topology and communication protocol. The results show that the e-SP ring provides a low area scalable interconnect solution to communicating e-SP to associated neurons within a node facility.

Chapter 6 addresses global interactions of a neuro-glia network and astrocyte to astrocyte interactions in hardware. It provides an overview of the interactions and communication exchange between astrocytes as a network, this being a global communication exchange. The main focus is on the steps taken to implementing a novel astrocyte router. The astrocyte router connects eight astrocytes in a ring topology and uses serial communication, trading off speed for area. The router uses a novel dynamic scheduler and token system to manage the synchronous aspects of astrocytes within the astrocyte network balancing urgency using timing and token requests as thresholds. This is to provide communication protocol for future use in neuro-glia networks, thereby providing a scalable solution to this interconnect challenge. FPGA results demonstrate the astrocyte router provides a good trade-off between low area and low power to interconnect overhead and relatively low communication speed.

Chapter 7 presents results on hardware validation of the interconnect strategies and demonstrate the self-repair functionality in FPGA hardware using an example: a mobile robot. The chapter reviews this mobile robot which uses a spiking astrocyte-neuron network. The ring topology (Chapter 5) is applied to the existing hardware to communicate signalling data to provide the neural network with a self-repair capability. This is a real-world application of a SNN with self-repair and results are presented that show that if the synapses within the neural network are faulty, the neuro-glia network has the ability to adapt and repair and maintain the direction and speed. Results have shown that with up to 80% of faulty synapses the network can repair and restore pre-fault functionality.

Chapter 8 draws a conclusion to the thesis and discusses the contributions. There is also a section indicative of future work which may be carried out to extend the research presented in this thesis.

1.4 Publications

This section presents the papers and publications which have been peer-reviewed publications in conferences and journals during this thesis:

1.4.1 Conference papers

G. Martin, J. Harkin, L. J. McDaid, J. J. Wade, J. Liu and F. Morgan, "Astrocyte to spiking neuron communication using Networks-on-Chip ring topology," IEEE Symposium Series on Computational Intelligence (SSCI), Athens, 2016, pp. 1-8. (Contributes to Chapter 5).

J. Liu, J. Harkin, L. McDaid, and G. Martin, "Hierarchical networks-on-chip interconnect for astrocyte-neuron network hardware," ICANN 2016: Artificial Neural Networks and Machine Learning – ICANN 2016 pp 382-390, Lecture Notes in Computer Science

1.4.2 Journal Papers

G. Martin, J. Harkin, L. J. McDaid, J. J. Wade and J. Liu, "On-chip communication for neuro-glia networks," IET Computer Digital Techniques: Special Issue Bio-inspired Hardware and Evolvable Systems, vol. 12, no. 4, pp. 130-138, 2018. (Contributes to Chapter 6).

J. Liu, J. Harkin, L. P. Maguire, L. J. McDaid, J. J. Wade and G. Martin, "Scalable Networks-on-Chip Interconnected Architecture for Astrocyte-Neuron Networks," in IEEE Transactions on Circuits and Systems I: Regular Papers, vol. 63, no. 12, pp. 2290-2303, Dec. 2016.

1.5 Summary of contributions

Local Communication: This contribution is a low-level NoC interconnect based on a ring topology. This focused on the excitatory signal from an astrocyte, i.e. e-SP. This signal is a product of the astrocyte, which is connected to neurons within a neural network. The e-SP is produced to provide an equilibrium across the synapse. It reinforces the PR on healthy synapses, which restores pre-fault firing activity in neurons. This work included an overview of H-NoC which was the existing SNN hardware used to stimulate the astrocyte by cloning packets and forwarding these packets to the astrocyte. The astrocyte then responded to the stimulus and produced 2-AG and subsequently e-SP. This e-SP is the excitatory signal which increases PR in a healthy synapse, this signal was communicated back to the astrocytes within the Node Facilities of H-NoC. The H-NoC paradigm is a scalable SNN, and using this existing framework it was possible to provide a scalable interconnect to communicate between an astrocyte and the SNN. The ring topology provided a scalable self-repair interconnect solution within a Neuro-glia network.

The low-level communication paradigm between a network of astrocyte cells and an SNN is realised in this contribution using a ring-topology. This supports parallelism as it requires normal SNN activity whilst supporting low-level interactions between spiking neurons. This low-level communication from the astrocyte occurs between each astrocyte and the SNN network. Neurons communicate with a sequential and time-based method, whereas, astrocytes communicate continuously at a much slower rate. Astrocyte data is numerical in value and to retain the precision of computational models, the bit resolution is 64-bit. As the signal is large and the network is vast, this provides an interconnect problem and it is necessary to adapt the interconnect thereby making it suitable for applying to neuro-glia networks in hardware. This e-SP ring topology allows an astrocyte to communicate e-SP within an existing SNN and this interconnect provides a scalable solution for low level communication between an astrocyte and neurons within a neuro-glia network. Within the astrocyte there is a lot of data exchanged at a less demanding throughput compared to that of spike events. In engineering terms, this slow continuous exchange of information can be exploited to save on area. This contribution also provided a solution for communication exchange between neurons and astrocytes at a local level

(inter-astrocyte) interchange using a NoC ring topology to exploit the slow changing communication in order to provide a low area interconnect capable of scaling for large scale networks.

Global Communication: As well as communicating with neurons, astrocytes also communicate within networks of astrocytes via a global communication protocol. This communication protocol within the neuro-glia network can be considered a multi-level communication which supports both local and global communication exchanges. The global astrocyte-to-astrocyte interchange communicates IP_3 data. Within an astrocyte, IP_3 oscillates and this causes a trigger to allow Ca^{2+} to be released.

This process balances glia-transmitters across all associated and neighbouring astrocytes. The network detects changes in IP_3 and this is communicated across all connected astrocytes. As astrocytes influence and affect each other, it removes the need for a central controller. It is this distributed and complex communication which allows self-repair on a global scale.

The astrocyte receives stimulus from the SNN. This event data is communicated from H-NoC to the astrocyte via an additional output port within the node router of H-NoC. The astrocyte model receives spikes from neurons which stimulate the release of 2-AG and the astrocyte produces the IP_3, DSE and e-SP signals. As IP_3 is a global communication signal it is shared across its neighbouring astrocytes. Each astrocyte has a level of IP_3 and any changes in this IP_3 indicates either increased or decreased levels. Astrocytes function by balancing and sharing their levels of IP_3 to ensure that there is enough IP_3 to facilitate repair and maintain normal functionality. This contribution is a multi-level solution for communicating signals (both local and global).

The global signal from each astrocyte is connected to an astrocyte tile router. The astrocyte router has two main roles: (1) receive IP_3 level data from up to eight astrocytes and, (2) calculate the average IP_3 level for all eight astrocytes and communicate this back to all eight. The rate at which IP_3 changes is much slower than spike events; typically 2-

3 orders of magnitude slower. Therefore, this contribution provided a hardware interconnect focused on balancing the physical area per astrocyte tile router facility while also meeting real-time requirements of the IP_3 exchange and update process. The astrocyte cluster facility is also an important component of the overall architecture of the astrocyte router. The ring topology in NoCs has previously shown benefits in area-speed trade-offs for area for both SNN and neuro-glia hardware. By exploiting the slower communication speeds of the biological IP_3 signal a time-multiplexed approach using ring structures can reduce area and power overheads. This contribution provides a scalable solution for global communication exchanges in neuro-glia networks.

Hardware Validation: The e-SP ring topology from contribution 1 is applied to a SANN. The astrocyte releases e-SP to strengthen PR and facilitates self-repair. When the e-SP strengthens the remaining healthy synapses the firing activity of the neuron is restored. The SANN is developed on an FPGA and consists of two neurons (Neuron #1 and #2), associated synapses and the astrocyte process. The neurons generate 2-AG which splits into two signals (e-SP and DSE). DSE goes to the astrocyte and reduces the PR on all synapses. The astrocyte then generates e-SP which increases PR on all synapses. The output spike from each neuron, is converted to an output frequency by a PWM and Neuron #2 controls the speed of the robot from this output frequency. Within this model there are no faulty synapses associated with Neuron 1, the faulty synapses are associated with Neuron 2. These are injected with faults during normal operation. The first to eighth synapses associated with Neuron 2 can have faults indicating the percentage of faulty synapses. When faults have been detected, the PR in associated synapses falls, and the e-SP rises. As the healthy synapses increase PR, the frequency of neuron #2 begins to recover to a pre-fault frequency.

The SANN can be broken down into two neuron facilities, two synapse facilities and an astrocyte facility. While 64-bit precision has a large overhead, it is used so that the system output can be compared against the computer simulated models. The first step in realising the e-SP with large scale astrocyte-neuron networks is to work within a biological timescale and so, the ‘e-SP comms’ module was integrated into the SANN. The e-SP

data comes from the astrocyte and is used as input data to the synapse facilities. The e-SP ring takes the e-SP output from the astrocyte and sends it to the associated synapses at each neuron. Results show that within the SANN with the e-SP ring in place, there is no deviation of results when compared to the SANN model hardware without the ring when additionally using the ring in hardware. The output frequency and the e-SP incurred overhead show quantifiable results directly affected by the e-SP ring within the SANN. These results show that although the ring focuses on a low-level serial communication protocol it is able to maintain accuracy compared to software models and similar accuracy when applied in hardware.

Chapter 2: From biology to hardware

2.1 Introduction

If a system is deployed to space and is unable to complete its mission due to a simple fault, it is a failed mission. In recent years, fault tolerance and self-repair have become a key research field especially in mission critical systems such as avionics and space applications, due to harsh environments [19]. Self-repair is a desired characteristic in such systems. This is because the device may repair its own faults, repairing or overcoming the fault or component, and returning to functional operation, or to some degree of acceptable performance.

Recent research has shown that biological traits such as fine grained repair and distributed repair-decision making are performed in the brain via astrocyte networks [11], [12]. In particular, computational models of such repair have been successfully captured and applied to SNNs [15] where neuron activity can be repaired to near pre-fault operation via the re-strengthening of the neuron's healthy synaptic connections. Since the repair occurs at the individual neuron synapses, this fault detection and repair is at a fine-grained level. The mechanism which makes the repair-decision within the brain has been identified as the astrocyte process, which is a type of glial cell highly distributed within networks of neurons [15]. This neuro–glia network paradigm addresses the key self-repair requirements of fine granularity and distributed decision making and sets the focus of this thesis. One of the key challenges in progressing neuro–glia networks to hardware is connecting a network of astrocytes with an SNN. Astrocytes connect to groups of synapses and also form large networks of their own. Therefore, emulating a neuro–glia network in hardware introduces additional interconnect challenges which also requires maintaining a scalable, low power/area overhead implementation. This will support a more complete understanding of the human brain, and the information exchanges, as well as providing biologically inspired self-repair for electronic systems.

2.2 Neural information processing

The brain is mainly made up of neuron and glial cells where early studies suggested that the neurons were considered to the main functioning cell whilst glial cells solely provided structural integrity. More recent research however, has shed light on the inner mechanisms and complexity of the brain. It was perceived, and accepted, that an adult had approximately 100 billion neurons and 10 times as many glial cells, but, this figure is closer to 86 billion neurons [20]. Interestingly, a study identified on average, the number of cortical neurons in new-borns was similar to that of an adults however, the number of glial cells was found to be approximately one-fifth to one-sixth to that of an adults total number of glial cells [21]. With more and more research, we unlock secrets within the brain, yet there is so much to understand. There are between 6-10 neurons to one astrocyte according to recent research [13]. This is discussed in more detail in Chapter 6. Section 6.2.1 Neurons and glia cells.

The Human brain is highly efficient and has the ability to process information efficiently using around 12 Watts of power [22], [23]. 10^8 times faster than traditional computers [24] and highly adept at problem solving, pattern recognition and performing cognitive functions [25]. Traditional computers, although fast when carrying out sequential operations, struggle with pattern recognition and other complex problems (such as data processing and classification). They are inefficient in terms of computational power and power consumption. One such model, the blue brain project, using the IBM Blue Gene/L supercomputer has 8,192 CPUs running at a clock frequency of 700MHz and at peak performance, should provide 360 teraflops. This was published in 2006 [26], since then Blue Brain has been updated: Blue Brain 4 was released in 2014 and more recently Blue Brain 5 was released in 2018 (see section 2.4.1). Within the brain, it is the parallel infrastructure and the interconnect between neurons and how they communicate, which provides efficient and powerful processing.

2.2.1 Neurons

Beginning in the early 1900s, the structure of neurons within the brain began to be explored, Fig.2.1. Shows Santiago Ramón y Cajal drawing of very early depiction of a neuron. The neuron is the main cell responsible for communicating information [27] and is made up of several components: the soma, axon, dendrite and synapse. From a simplistic viewpoint, the dendrites are inputs and the axon is the neuron's output (a neuron may have many inputs but only one output which branches out to many other neurons). Fig.2.2. Shows a synaptic cleft connecting an axon and dendrite, this is a chemical exchange between neurons. Fig.2.3. Depicts a more recent illustration of a neuron; labelling the neurons various components [28].

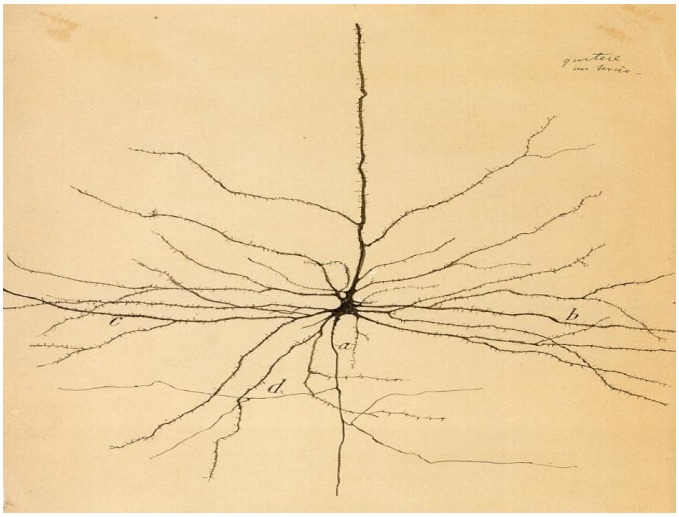

Fig.2.1 A neuron by Ramon y Cajal [29].

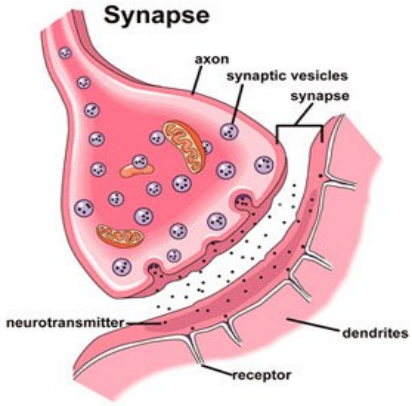

Fig.2.2 Synaptic cleft [30].

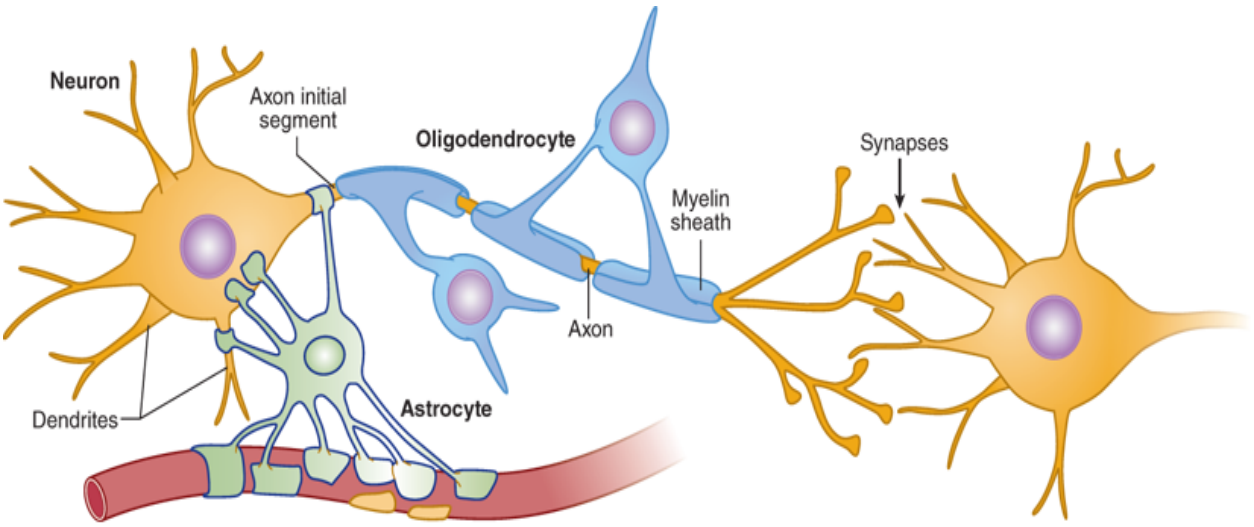

Fig.2.3 A closer look at the neuron. Here are the individual components of a neuron and connected to an astrocyte cell.

To understand how a neural network processes information we must first look at these individual components and their operation during a spike event i.e. the communication protocol between neurons [27]. The neuron body is referred to as the soma. This soma is made up of positively and negatively charged ions, but sodium (Na^+) and potassium (K^+) are the most important in neural computation and communication. These channels were first identified in the nerves of a giant squid when Hodgkin and Huxley dissected a giant squids nerves and applied electrical current to observe and explore the nature of neurons and their communication channels [31]. Information from neighbouring neurons stimulate the soma. This information arrives from the dendrites and cause a chemical reaction to occur which then causes an electrical reaction.

This reaction is caused as sodium channels open and the positively charged sodium ions flow through the membrane and into the soma, this is known as depolarization. The membrane potential has a voltage threshold (typically between -50 and -55 mV), as the neuron receives stimulus the membrane potential increases until it breaches the said threshold. This causes a spike or action potential (AP) to output through the axon to the synaptic cleft and into another neuron, i.e. a surge of glutamate across the synaptic cleft

[23], [32]. The axon is considered the output of the neuron and is connected to a dendrite via the synapse or synaptic cleft. Repolarization follows when the potassium channels open to allow potassium ions to leave the soma and restore membrane potential, this is the refractory period where the neuron is unable to fire again regardless of stimulus. Fig.2.4 is an action potential showing depolarisation and repolarisation.

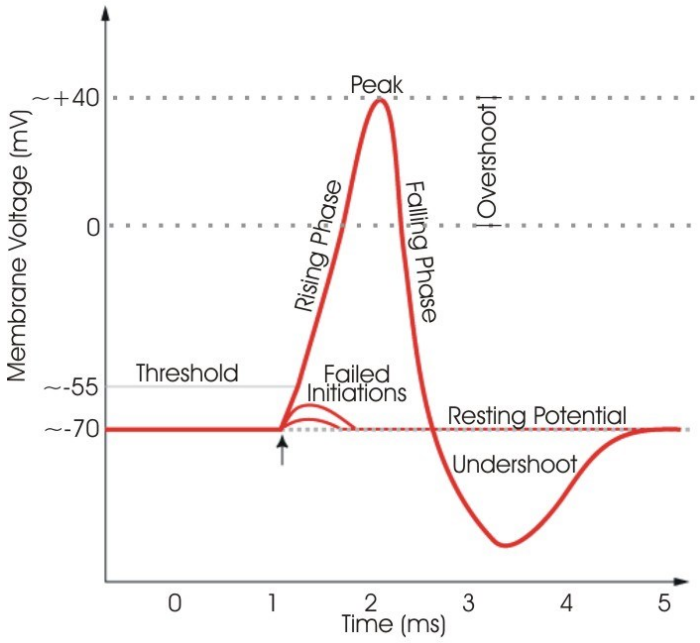

Fig.2.4 An AP/spike [33].

2.3 Neural networks

A typical neural network consists of artificial neural cells interconnected in a predefined topology with input and output layers and these layers consist of multiple inputs and outputs. The main aim of a neural network is to emulate an artificial system capable of complex problem solving and pattern recognition analogous to the brain, in a highly efficient parallel system. A neural network can be broken down into 3 main components: the I/O (input and output) layers, the interconnect or topology connecting neurons and the ability to learn. Recreating the architecture of the brain in a neural network with the capability to learn, brings us closer to solving more complex problems. Neighbouring neurons communicate via spikes, as the number of input spikes accumulate, the stimulus

surpasses a predefined threshold causing the neural cell to fire i.e. releases a spike. This emulates the neurological process and therefore is analogous to the biological neurons and their communication protocols [27], [28]. Fig.2.5. shows a simplified overview of an ANN. In this case the input (X_n) is multiplied by the weight (W_n) of the synapse and the sum of the inputs is compared to an internal threshold (activation threshold). If the sum of the inputs passes the threshold value, the output is 1, if not it remains 0.

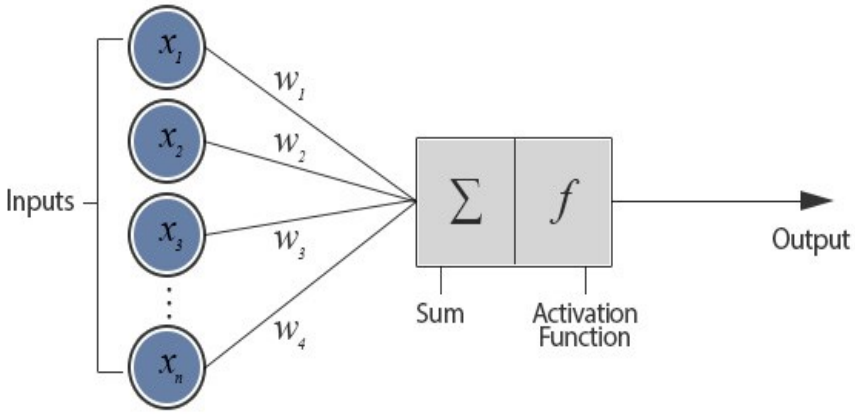

Fig.2.5 Simplified overview of an ANN [34].

Neural networks can be regarded as having three generations, the first- and second-generation models use rate coding, and these models were based on using the number of spikes to communicate. The McCulloch and Pitts model is considered the first generation of artificial neural network (ANN). The threshold for the firing rule (activation function) of the neuron is based on a step function. Neurons exchange information with neighbouring neurons and each input is multiplied by a weighted sum. If the sum of the vector surpasses the threshold, the neuron then fires a binary 1 or 0 [35].

Donald Hebb (Hebbian theory), proposed an explanation for the adaptation of neurons in the brain during learning, "Cells that fire together wire together". This is unsupervised learning. The causal link between neurons is strengthened based on the firing activity i.e. the weights associated with each synapse were updated based on the firing activity of the neuron. Reinforcing the relationships between connected neurons, the network would adapt and learn based on activity [36].The second generation of ANNs, furthered the work

on neural networks by way of learning. In 1958, a Single Layer Perceptron (SLP) was introduced. This used supervised learning to update the weights associated with neurons. Supervised learning uses training examples to associate an input to an output based on a training set. The user supplies a training set, and an expected output, to train the network, and then supplies a completely new set. The supervised learning should allow the network to classify new data correctly. By modifying the synaptic weights and threshold function, it is possible to perform pattern recognition [37].

Although the simplicity of using a SLP was a principal feature of such networks, they were unable to solve the basic XOR problem [38]. It was hypothesised, that introducing a hidden layer and devising a Multilayer perceptron (MLP) based on error back propagation, would solve the XOR problem. The trade-off of this approach was an increase in the complexity of the model, diminishing the simplicity of the SLP. A neural network may have a number of layers between the inputs and outputs, generally referred to as hidden layers. The number of layers depends on the classification or purpose of the network. An interesting aspect of the first and second generations of neural networks was that they focused on using the rate of spikes. However, this ignored the timing of the spikes completely. This leads us to the third generation of neural networks, a Spiking Neural Network (SNN). SNNs use the timing and frequency of their spikes to encode information. That is, they encode information based on the timing of spike events and/or rate coding. They communicate as such to neighbouring neurons, this is similar with how the brain processes information or data as spike events propagate from neuron to neuron [39], [40].

2.3.1 Spiking neural networks

As mentioned previously, SNNs are based on sending information between neighbouring neurons using the timing and frequency of their spikes (temporal and rate coding) rather than using the spikes shape or size [41]–[43]. SNNs can be used to carry out more complex computations and they are a more accurate representation of how neurons communicate based on biological evidence. However, it has become increasingly difficult to simulate large scale SNNs because of limited scalability when realizing the complex

interconnect [44] and as such there is trade-off between complexity and biological plausibility [45]. There are several computational models based on using the timing of the spikes to encode information. Hodgkin and Huxley, based on the nerve cell of a giant squid, found that the nerves consisted of a membrane sheath consisting of sodium and potassium currents as well as a leak current, as seen in Fig.2.6 and thus produced an accurate mathematical model of a neuron. This is the Hodgkin–Huxley model of the characteristics of cell membranes [31].

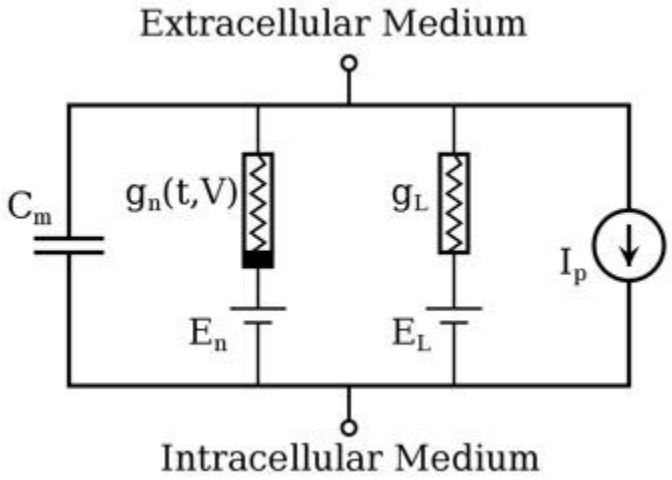

Fig.2.6 Hodgkin and Huxley model. This model uses voltage-gated (g_n and g_L) and leak channels to model the dynamics within a neuron [46].

The Hodgkin and Huxley model has four differential equations and as a result, the mathematical model is computationally intensive due to the number of variables [23]. There have been attempts to recreate a similar model without the computational complexity but without reducing the accuracy. The Integrate and fire (IF) model based on the most basic concept of a biological neuron consists of a single capacitor which represents the charging capacity of a neuron. When current is injected, the capacitor will charge until it reaches a voltage threshold and then discharge. This can be viewed as analogous to a spike [42]. The Leaky integrate and fire (LIF) model, goes a step further and provides the model with a resistor which represents the leak current. Each time a spike activates current, the capacitor charges, the resistor then causes the capacitor to lose this charge if there is no further stimulation. This simple model is similar to that of a neuron, with no spikes or voltage input into the neuron model, the charge returns to

resting potential [23]. Although very simple approaches, they maintain the most important aspects of a neuron model, that is the charging and leaking aspect. Other examples of computational models based on SNNs include: Spike Response Model (SRM) [23], and Izhikevich [39], amongst others. Interestingly, the Izhikevich model is a biologically plausible mathematical model and is capable of reproducing all different firing patterns exhibited by cortical neurons. It consists of two differential equations and this model can give a realistic insight into cortical neurons interacting with one another [39].

Due to the number of computational SNN models available, it is difficult to distinguish an overall “best” model. There are a number of key differences and ultimately it becomes, biological plausibility vs computational complexity as seen in Fig.2.7. [45]. The choice and selection of a neuron model depends on the purpose of the model and the amount of computational resources available. If the purpose of a model is to identify neural dynamics or the behaviour of neurons and how they interact and communicate, a Hodgkin and Huxley (HH) [31] model or Izhikevich model [39] may be used but will result in a smaller scale network i.e. having smaller numbers of neurons but with greater complexity. If, however, the purpose is to view the interconnect strategies between neurons on a large scale, the biophysical complexity of the model is sacrificed in order to increase the number of neurons and have a larger network. Therefore, the computational resources available can be applied to other aspects of the network e.g. the number of neurons or identifying interconnect topologies.

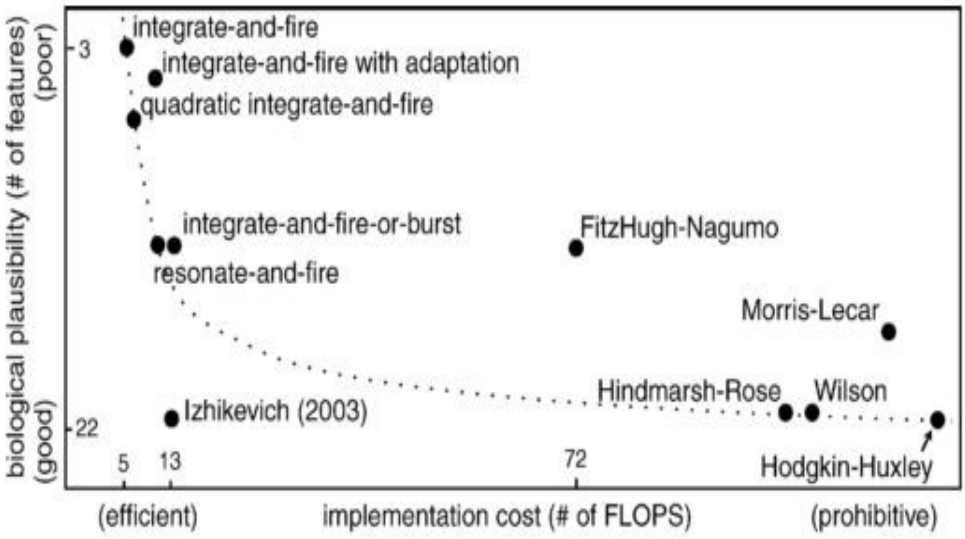

Fig.2.7 Biological plausibility vs computational complexity [45].

2.4 Neural networks: computational implementations

There are many reasons to model and investigate how the brain processes information. It may give a much better understanding of how a parallel dense interconnect populated with neurons can lead to huge data processing abilities or how to harness such processing power of such a simple model with an efficient power consumption [23], [42]. Alternatively it could allow the exploration of degenerative diseases that plague the brain such as epilepsy, Alzheimer's and Parkinson's [47]. Using models for more efficient processing systems capable of big data, speech recognition and pattern recognition, will lead to better systems in the future in terms of research within medicine [48]. It could also give us insight into other cells at work within the brain, as it has done with astrocytes [49].

2.4.1 Neural networks in software

Software applications offer researchers the flexibility of choosing complex or simplified models based on the nature of the research. It allows customisation throughout the network from start to finish, meaning the network can be specifically applied for certain problem solving or classifying purposes. Software models are also very useful for

investigating more complex neural networks and how neurons communicate in different parts of the brain e.g. the neocortex [45], and neuron models help explore the behaviour of neurons more closely. The biggest drawback of using software models is the processing power. The power required to explore and develop these models, is not always available off-the-shelf. In terms of software, modelling NNs suffers due to the hardware bottleneck used to create and run applications. This is due to sequential processors unable to execute parallel networks efficiently, this leads to an inefficient and power hungry processing [50]. There are two main platforms for realising neural networks in software using traditional processing units, general purpose Graphic Processor Units (GPU) and High-Performance Computers (HPC) [51]. A hybrid approach may also be considered using CPUs with specific NN hardware in order to accelerate the neural network applications [52]. In recent years, processing power and performance has increased significantly. The off-the-shelf Intel i9-7980XE 18-core, 36-thread CPU, clock speeds vary from 4.2GHz to 3.9GHz up to 12 cores, is due to be released Fall 2018 [53]. It would seem that the resources for neural networks on traditional CPU architectures may not be far away however, GPUs consist of more processing cores and dedicated processors to that of general-purpose CPUs. Graphics processing units, typically containing a lot more cores which makes them more suitable for realising the parallelism of neural networks than CPUs, and examples of such software simulations have been used in the past [54], [55]. General-purpose computing components aren't sufficient and are very limited in regard to simulating neural networks. These vast neural networks consist of only hundreds of thousands of neurons and synaptic connections, which would allow a very small-scale realisation. NeMo used a general purpose GPU to simulate one million neurons, with 1000 synapses each, firing at 10 Hz [54]. A GPU out-performs a CPU, but they are still inefficient and power hungry as well as slow [50]. It is the parallelism that researchers struggle to re-create. NCS6 was able to simulate one million cells and 100 million synapses by distributing data across eight machines with each having two video cards [56]. A CPU cluster e.g. Beowulf cluster which is a network of computers [57], similarly, GPUs can exploit their cores for parallelism, which can be used in realising NN models [57].

HPCs allow the realisations of large-scale neural networks with flexibility in terms of programming and creating networks, in [57] a computational model of the *thalamocortical* system was simulated on a Beowulf cluster. This simulation comprised of one million neurons and half a billion synapses. In computational terms, there were sixty 3 GHz processers, each with 1.5 GB of RAM yet, could not match real-time performance of biological systems. They aim to allow the user to modify the number of neurons, the number of hidden layers and the parallel interconnect.

HPCs can be used for deep learning, and are made up of combining GPUs, however, they have limitations such as scalability. In [58] a HPC capable of training one billion parameter networks was described but encountered scaling problems when trying to exploit parallelism across multiple GPUs. In [59] a Titan supercomputer was used in order to optimize the performance of deep learning algorithms. As the size of the network grows, the hyper-parameter space grows increasingly larger. Other such HPC approaches look to accelerate training using HPCs e.g. Caffe-HPC aims to train large models [60] using a HPC approach. So although HPCs can be applied to reduce certain aspects such as training times or accelerate models, they are limited.

With increasing numbers of neurons and additional complexity, the software simulations become more difficult to simulate and realise. As a result, this uses more computational resources available [26]. HPCs contain huge reservoirs of computational resources, allowing increasingly complex neural networks, but they are slow and do not scale efficiently when realising large scale SNNs [61]. Computational models have proven very effective at pattern recognition (speech and facial recognition), classifying biological information using neural networks [48]. SNNs have been emulated in software with the aim of creating a highly parallel implementation for acceleration purposes [8], [62], [63] and also as exploratory platforms [64], [65]. The BlueBrain project [26] is an example of a software implementation using a HPC.

The Blue Brain 4 and Blue Brain 5 are installed at the Swiss National Supercomputer Centre (CSCS). Blue Brain 4 aimed to simulate a rodent's brain (200 million neurons).The

Blue Brain 4 comprised of a four-rack IBM BlueGene/Q system 65,536 cores for computing, providing a peak performance of 839 Tera Flops [66]. The Blue Brain 5 core system is an HPE SGI 8600 system comprised of 372 compute nodes, providing 1.06 petaflops of peak performance. [67].

Software approaches are flexible due to the software control but exhibits significant physical space, cost and power overheads. In regards to software, the key challenge for implementing SNN software models on typical processing hardware is scalability. This stems from the need for efficient interconnect wiring, low area/power synapse and neuron designs, efficient weight storage, programmability of SNN topology and weights. It is the inability to do this without huge amounts of computational resources that make it very difficult to make an efficient neural network in software which can perform simulations in close to biological real time.

2.4.2 Neural networks in firmware

It is natural to progress to a hardware platform as a solution to the software approach's short-comings; researchers considered using Field Programmable Gate Arrays (FPGAs) [68], [69].

FPGAs provide a solution to the inherent problems of using software i.e. the hardware bottlenecks. Simulating large scale neural networks on a hardware platform, removes the limited CPU and GPU overheads as FPGAs are more efficient in terms of both hardware overhead and power consumption. FPGAs were initially used to realize large hardware integrated circuit (IC) designs. They offer a more flexible approach similar to that of software, but with the efficiency and performance of a full hardware design, with low overheads. FPGAs are made up of two-dimensional arrays of configurable digital logic blocks and registers. The FPGA interconnect, which connects blocks and registers is also reconfigurable between these blocks. Very High-Speed Integrated Circuit Hardware Description Language (VHDL) and Verilog are hardware description languages and are used to describe and program FPGAs. This allows the re-configuration of the FPGA (flexibility). Designs can be implemented on FPGAs quickly as there is no physical

process. Because of the parallel nature of FPGAs and how they operate they have become a popular choice when realizing neural networks [8], [50], [70], [71]. The reason FPGA platforms are preferred to software approaches is because they offer a low power and parallel infrastructure which is suitable to support large scale neural networks. FPGA's will be discussed in more depth in the next chapter.

2.4.3 Neural networks in ICs

A fully customized hardware design e.g. Application specific Integrated Chip (ASIC) provides an ideal scenario for neural networks. It provides a low area and power efficient interconnect with a large throughput. However, as modelling the size of the network scales, the traditional network interconnect struggles to deal with increasing numbers of neurons within the network. This is due to using bus systems for interconnecting PEs. Each neuron is considered to be an individual PE and thus, as the number of neurons increases, the number of PEs connected to the interconnect increases. This also increases latency and slows the throughput. Therefore, an IC design is not fully scalable using traditional digital interconnects because the bottleneck is the hardware interconnect. The topology of connected neurons is especially important in terms of how neurons are connected within a network, increasing complexity as well as increasing latency. This will affect the speed and performance of the full custom IC design. Neuromorphic ASICs have been designed to replicate the low power and area of the brain onto a chip. The Defence Advanced Research Projects Agency (DARPA)-funded program Systems of Neuromorphic Adaptive Plastic Scalable Electronics (SyNAPSE), aims to develop a full-custom hardware design for neural network implementations that scale to biological levels [72].

The SyNAPSE program develops TrueNorth neuromorphic chips and attempts to emulate the mammalian brain in electronic hardware. The overall aim is to build a microprocessor system that emulates the mammalian brain in terms of both function and power consumption (10 billion neurons, 100 trillion synapses consuming just one kilowatt [72]). As of 2014 there were 4,096 neurosynaptic chips capable of 1 million programmable neurons and 256 million programmable synapses, consuming just 4 kW of power. This

neurosynaptic chip is considered low power as it differentiates from traditional von Neumann architecture. It operates without a clock and has a low power consumption by optimizing event-driven operations, and therefore, it operates only when it needs to. A number of simulations and applications have been carried out by IBM and Darpa to recreate the efficiency and processing power of the mammalian brain. IBM used a super computer, the Dawn Blue Gene/P with 147,456 CPUs, to create a network similar in size to that of a cats neocortex [73]. Further research was released creating Compass [74] a large scale simulator for cognitive computing a follow up showed the ability to simulate 2.084 billion neurosynaptic cores containing 53 × 1010 neurons and 1.37 × 1014 synapses [75], as well as other applications which demonstrate the use of the neurosynaptic core [76].

TrueNorth [77] was created by IBM and is used on the DARPA SyNAPSE board, there are 16 TrueNorth chips. Each of the chip's 4,096 neurosynaptic cores includes the entire computing package: memory, computation, and communication. Fig.2.8 shows an infographic on TrueNorth released by IBM.

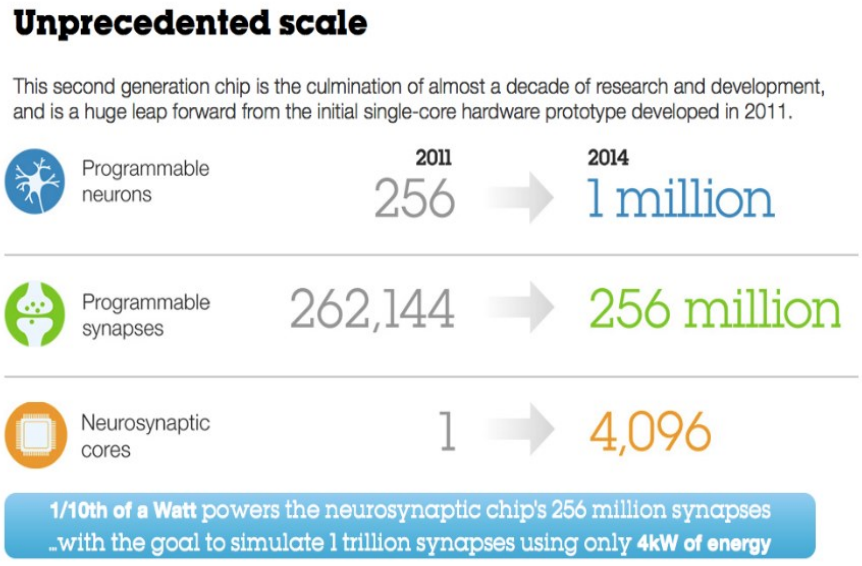

Fig.2.8 IBM infographic on TrueNorth [72].

2.4.4 Neural network hardware using on-chip interconnect

Significant progress has been made in synapse and neuron designs, programmability etc. Interconnect scalability remains a challenge which is still not fully addressed. Current approaches to this problem have explored networking concepts such as NoC. NoC technology applies network engineering and techniques for on-chip communication. This has notable improvements over conventional bus and crossbar interconnections. NoC improves the scalability, and the power efficiency of System on Chip (SoC) technologies. The next chapter will provide more insight into such technologies. Using dedicated hardware and digital interconnects, the speed and rate at which neural networks work has been increased. In terms of neuromorphic hardware the key research includes the following:

- **Spiking Neural Network Architecture (SpiNNaker)** [64] uses 18 ARM9 processors where 16 processors are used for simulating 1,000 neurons each. It uses a NoC packet switched interconnect and two NoC networks, and aims to simulate biological real time, large scale SNN consisting of a billion neurons and trillion synapses. The aim of SpiNNaker is to simulate up to a billion neurons in biological real time.
- **Neurogrid** [21] is a neuromorphic system (using analogue sensors and a digital interconnect) which simulates a neural system in biological real time. Neurogrid uses 16 neurocores (each neurocore simulates 65,536 analogue neurons) and aims to emulate 1 million neurons in real time with a low power overhead (3 watt) and connects neurocores using a NoC interconnect. Since it only uses 16 neurocores, it can simulate a million neurons and billions of synaptic connections in real time, with a low power consumption. Neurogrid uses software for interactive visualisation and hardware for real time simulation. Neurogrid, however, lacks software flexibility shown by High-Performance Computing (HPC) approaches and due to the limited number of neurons per layer, it is unable to offer biological real time [78].

- **FACETS** [62] is designed to use an uncut silicon wafer for communication between cores. The wafer consists of many High Input Count Analog Neural Network (HICANN) chips and rather than cut the wafer, a multi-bus NoC interconnect is integrated onto the wafer for communication between chips on the wafer. An FPGA provides wafer to wafer communication, although this limits performance. A single wafer can contain 384 HICANN chips which equates to 196,608 Neurons. FACETS offers hardware acceleration although the trade-off is a high consumption of power.

- **Emulating Biologically-Inspired Architecture in Hardware (EMBRACE)** [8], [63] was developed at Ulster University and uses analogue neurons with a NoC interconnect. EMBRACE offers a Field Programmable SNN solution which is reconfigurable, and due to the NoC architecture, it is also scalable with low power/area consumption. The Hierarchical Networks on Chip (H-NOC) Architecture, is an extension of EMBRACE, and focuses on the structure of the neurons and how the neurons communicate. It uses three communication layers with three separate routers on a single Cluster facility for communication. Fig.2.9 shows an overview of the H-NoC and how neurons are connected with a hierarchical router structure, this enables 400 neurons to interconnect whilst allowing communication on a global scale (many Cluster facilities). H-NoC also offers a spike compression technique [79], which reduces traffic congestion while maintaining biological real time. In addition H-NoC offers hardware acceleration where its NoC throughput performance outperforms that of SpiNNaker, Neurogrid and FACETs [79]. Fig.2.10. shows the H-NoC architecture interacting with an astrocyte network.

- **SyNAPSE and TrueNorth** [77]. TrueNorth is a neuromorphic CMOS integrated circuit chip. Memory, computation, and communication are handled in each of the 4,096 cores. In essence this is a highly parallel architecture intended to mimic neurons in the brain, however, they are updated serially. TrueNorth consumes 70 milliwatts (0.001% compared to traditional microprocessors) this is because the SyNAPSE chip only draws power for the computation of any calculations.

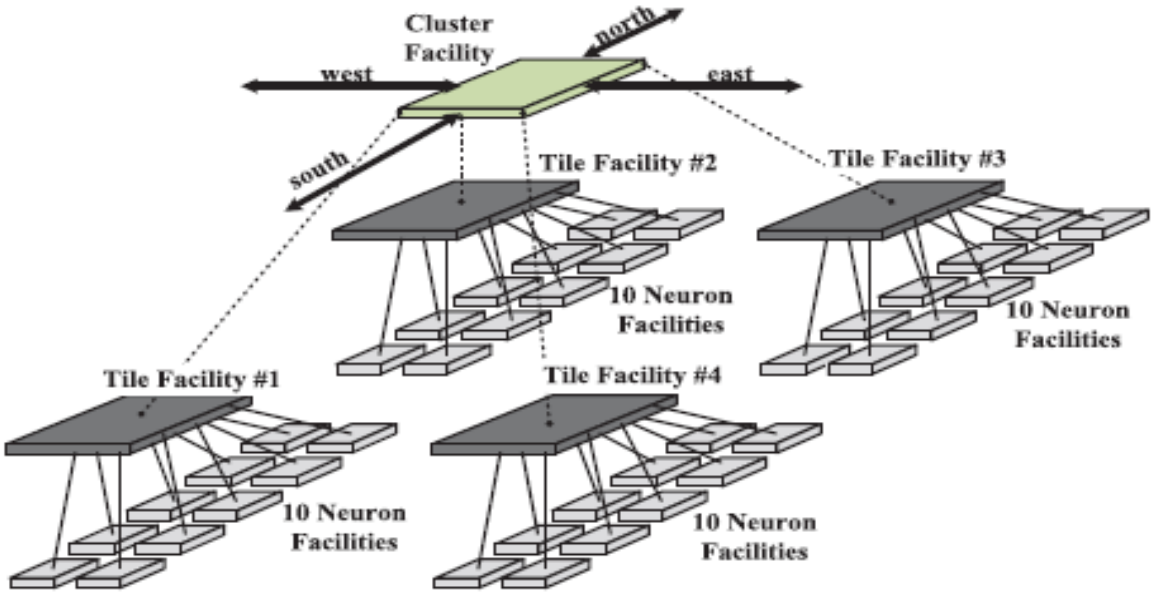

Fig.2.9 H-NoC Architecture: This Figure shows the hierarchical approach of H-NoC. The outcome is high throughput or increased spike communication across significant numbers of synapse and neurons [79].

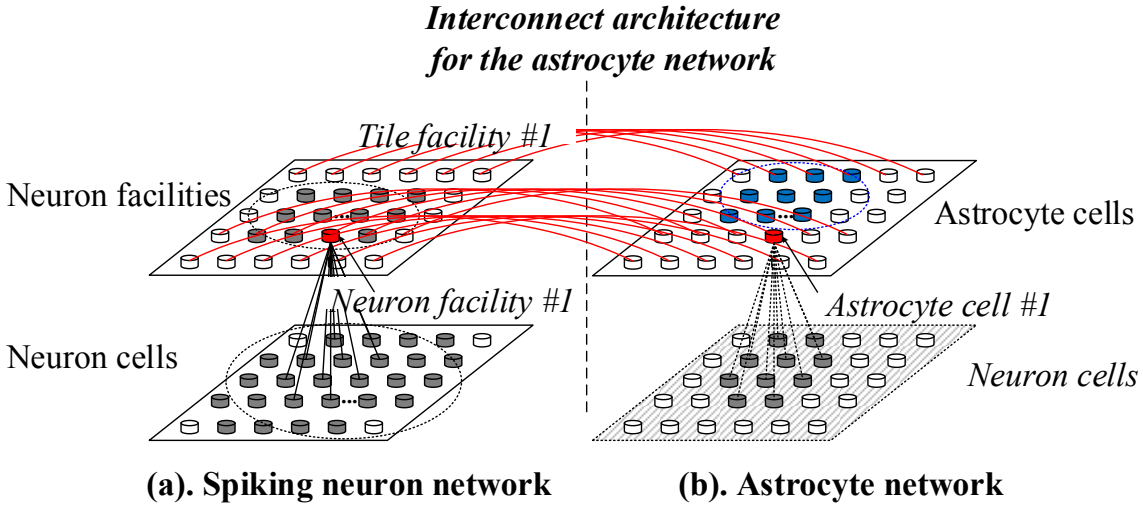

Fig.2.10 H-NoC/Astrocyte Architecture. This figure shows H-NoC connected to an astrocyte network: This Figure shows the hierarchical approach of H-NoC connecting neurons to astrocytes via the tile router.

H-NoC and FACETS are able to offer acceleration beyond that of biological real time. This shows the promising features of using hardware for a SNN and is therefore a natural progression from software. Fig.2.9. shows the H-NoC architecture. Fig.2.10. shows the H-NoC architecture interacting with an astrocyte network. A 3-D structure of interconnecting neurons and synapses captured in a flat 2-D NoC. This enables higher levels of flexibility for routing packets of data (spike events) to synapses and neurons.

2.5 Astrocytes and self-repair

A novel self-repairing strategy using astrocytes is based on recent biological evidence [12], [15], [80]. It was previously believed that astrocytes were not involved in neuron activity and were used solely for structure, however recent research shows that astrocytes are involved in regulating synaptic plasticity [14] and the self-repair process of the brain [11]. Astrocytes don't communicate in the same manner as neurons (spike events). Astrocytes communicate with neurons and other astrocytes using different chemical signalling pathways. Astrocytes communicate bi-directionally with neurons and other astrocytes by the uptake and release of transmitters, and because of this they modulate synaptic transmission [80]. Astrocytes contain receptors which are activated when a spike event or AP occurs. This triggers the release of glutamate from the presynaptic axon into the cleft and into the postsynaptic dendrite. When the postsynaptic neuron has been sufficiently depolarized, causing the neuron to emit a spike. 2-arachidonyl glycerol (2-AG) is released from the postsynaptic neuron. This is taken up by the astrocyte, and causes oscillations of calcium (Ca^{2+}) within the astrocyte. This in turn causes the release of glutamate or gliotransmitters; this is an indirect feedback mechanism from the astrocyte to the neuron(s) and is the mechanism which allows the astrocyte to communicate with the neuron. There are two feedback signalling pathways, the indirect feedback via the astrocyte and the direct feedback via the neuron. This feedback mechanism is referred to as Endocannabinoid-mediated Synaptic Potentiation (e-SP) which strengthens PR. The second feedback mechanism is the direct feedback referred to as Depolarization-induced Suppression of Excitation (DSE). DSE subsequently decreases the PR of the synapse. Moreover, astrocytes are connected via intracellular signalling routes via gap junctions. This allows Inositol trisphosphate (IP_3), which is an astrocyte secondary

messenger, to pass through thereby allowing astrocytes to communicate with and share information. This signalling behaviour has been modelled in previous work [81] and is the mechanism by which repair decisions are communicated at network level.

2.5.1 Example computational models of repair

The implementation of astrocytes within an SNN has recently been explored, and a software model [15] developed by Ulster University indicated that the use of astrocytes facilitates self-repair. Five astrocytes were interconnected in a ring fashion and each astrocyte connected to two neurons. Up to 80% of the neural synapses were made faulty during the simulation and results indicate that even with severe damage, the astrocyte model could return firing rates of damaged neurons to near pre-fault level [15]. This network level repair was based on the strengthening of PR on healthy synapses which caused the neurons to regain frequency output. Fig.2.11, illustrates two neurons firing, when one neuron stops, the excitatory signal provided by the astrocyte (e-SP) is maintained by the healthy neuron.

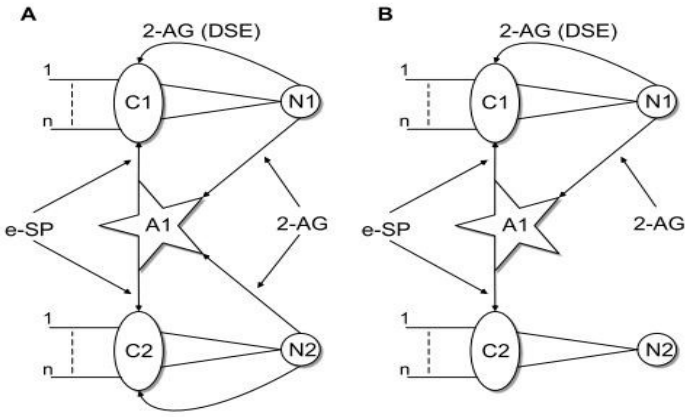

Fig.2.11 Astrocyte feedback: N1 and N2 depict neurons, A1 an astrocyte and C1 and C2 contain approximately ten synapses each. The signals e-SP and DSE are excitatory and suppressive feedback signals. In (A) both neurons are firing however in (B) N2 has stopped firing. Although DSE from N2 has stopped, the astrocyte e-SP feedback is still active due to N1 still remaining active. This leads to an increase in PR and weights within the remaining healthy synapses (C2) of N2 and a restoration of N2 firing activity [15].

Computational models of such repair have been successfully captured and applied to SNNs to demonstrate repair of neuron firing activity [15]. For example, when an active neuron suddenly stops firing it is deemed faulty, this is due to a low PR at its associated synaptic sites. These faulty neurons are referred to as silent or near silent neurons. Work has shown that astrocytes can detect faulty synapses (fine-grained level) associated with silent neurons, and by subsequently increasing the PR on surrounding healthy synapses they can restore the neuron to its original functionality i.e. the potentiation of PR on healthy synapses will restore pre-fault function.

By increasing the PR in the remaining healthy synapses, the neuron functionality is restored to its pre-fault level of activity. The increased complexity of the signalling between the astrocyte, synapses and neurons provides the capability to sense and repair synaptic connections (fine grained). The astrocyte regulates the degree of repair. It should be noted that astrocytes communicate globally with other astrocytes thereby providing a distributed repair-decision making capability. At an abstract level, one can view astrocytes as a network and the synapses and neuron as a separate network, with interactions between both networks occurring via the direct and indirect signalling pathways. Progress has been made in modelling the astrocyte process [81] and its interactions with SNNs [15] successfully in software models. The software is limited by the computational resources and the length of time it takes to compute simulations. SNNs have also been replicated using hardware and the level of parallelism exhibited by hardware has shown to improve performance over software models with a lower power and area overhead at a much lower cost. It is therefore, timely to explore hardware emulation as hardware models are now more readily available, thus it is possible to employ self-repair on a SNN.

2.6 Astrocytes in hardware

Hardware models of astrocytes have been developed in attempts to replicate the astrocyte functionality. Nazari et al. 2014 [82] emulates astrocyte signalling for communication using an analogue implementation of astrocytes. Introducing oscillators to simulate Ca^{2+} oscillations, they simulate communication between astrocytes, but there was no exploration of self-repair. Solemani et al. 2015 [83] introduces a digital

implementation of astrocytes, based on FPGAs. The neuron and astrocyte communication is recognized and this is seen as a platform for developing a feedback loop which is similar to that in the brain. When a neuron fires, the astrocyte releases waves of calcium this is used to modulate synaptic strength. Using a ratio of 1:1 the neurons are directly interfaced with a digital astrocyte feedback loop, which can modulate synaptic strength. This is the first step in digitally implemented astrocytes, however the approach is not scalable and has no repairing function. Using astrocytes for self-repair is a promising field of research and the work done on software modelling has been successful [15] and forms the basis of this PhD research. The intertwining of two very different networks (SNN and astrocyte networks) presents an interconnect challenge and using NoC strategies is a potential solution to this particular problem. Fig.2.12 shows how astrocytes interact with a neural network. N1 to N8 are neurons. There are two astrocytes which may have hundreds or thousands of connections to neurons.

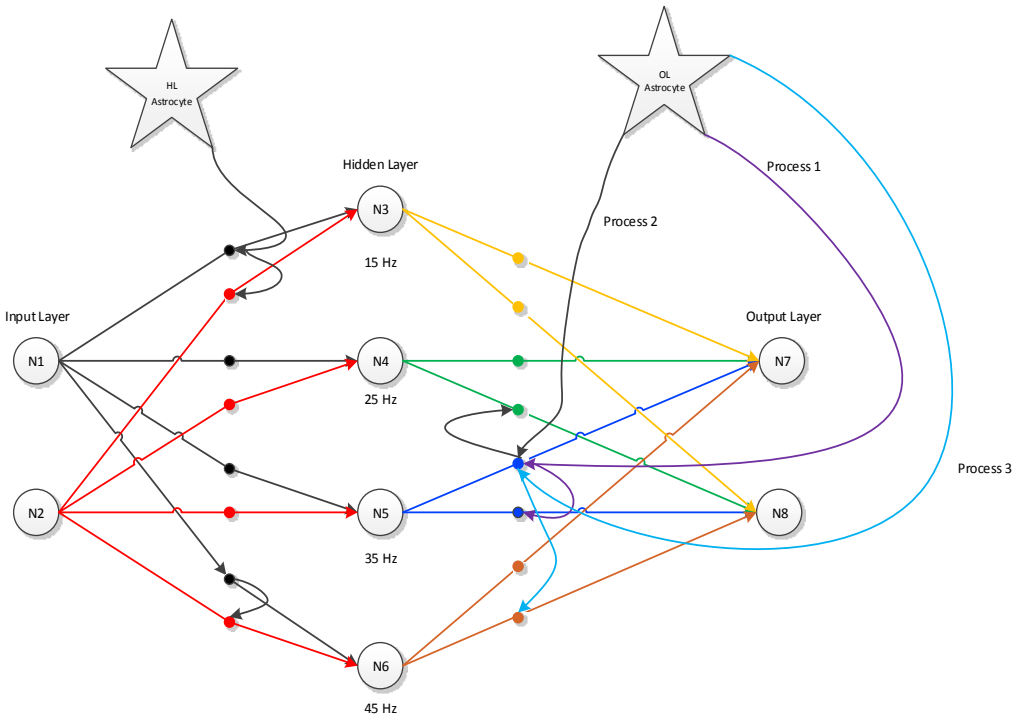

Fig.2.12 SNN with Neurons and Astrocytes [84].

The Self-repairing spiking neural network (SPANNER) architecture, developed at Ulster University, presents a self-repairing mechanism using astrocyte to neuron communication in hardware. Two neurons are connected and the synapses are connected to an

astrocyte. The two neurons have direct and indirect feedback signals, i.e. e-SP and 2-AG (DSE) and acts as an equilibrium stabilizing the PR at each synapse connected to the neuron and enable a stable firing rate. The self-repair capability in this process is monitored by the astrocyte process. When a fault occurs at the synapse associated with a neuron, both the e-SP and 2-AG (DSE) for that one synapse stop. The e-SP is a global signal which is associated with all synapses, therefore, when both signals cease, this creates an imbalance in PR at the healthy synapses. The e-Sp in the healthy synapses increases and thus the PR increases to restore the firing rate of the neuron. This self-repair therefore allows detection of faults in synapses and the repair of the neuron functionality occurs due to the healthy synapses increasing their PR. The SPANNER hardware architecture emulates the self-repairing mechanism of the brain. The results from SPANNER demonstrated a system with self-detection and repair capabilities and this has been applied in hardware.

Fig.2.13 and Fig.2.14 show a network with no faults and a network with faults respectively, in both, neuron 1 is indicated in blue and neuron 2 is indicated in red. This is to show that under normal operations both neurons are functional and have similar firing rates, e-SP and DSE. In Fig.2.14, 80% of faults have been injected into the network in neuron 2's synapses, at 200ms the faults are injected and the neuron firing frequency rate drops. The PR at the first synapse drops and at the tenth synapse the astrocyte increases the PR i.e. the healthy synapses increase the PR and even after a catastrophic (80%) fault, the frequency at which the neuron fires is close to pre-fault levels [85].

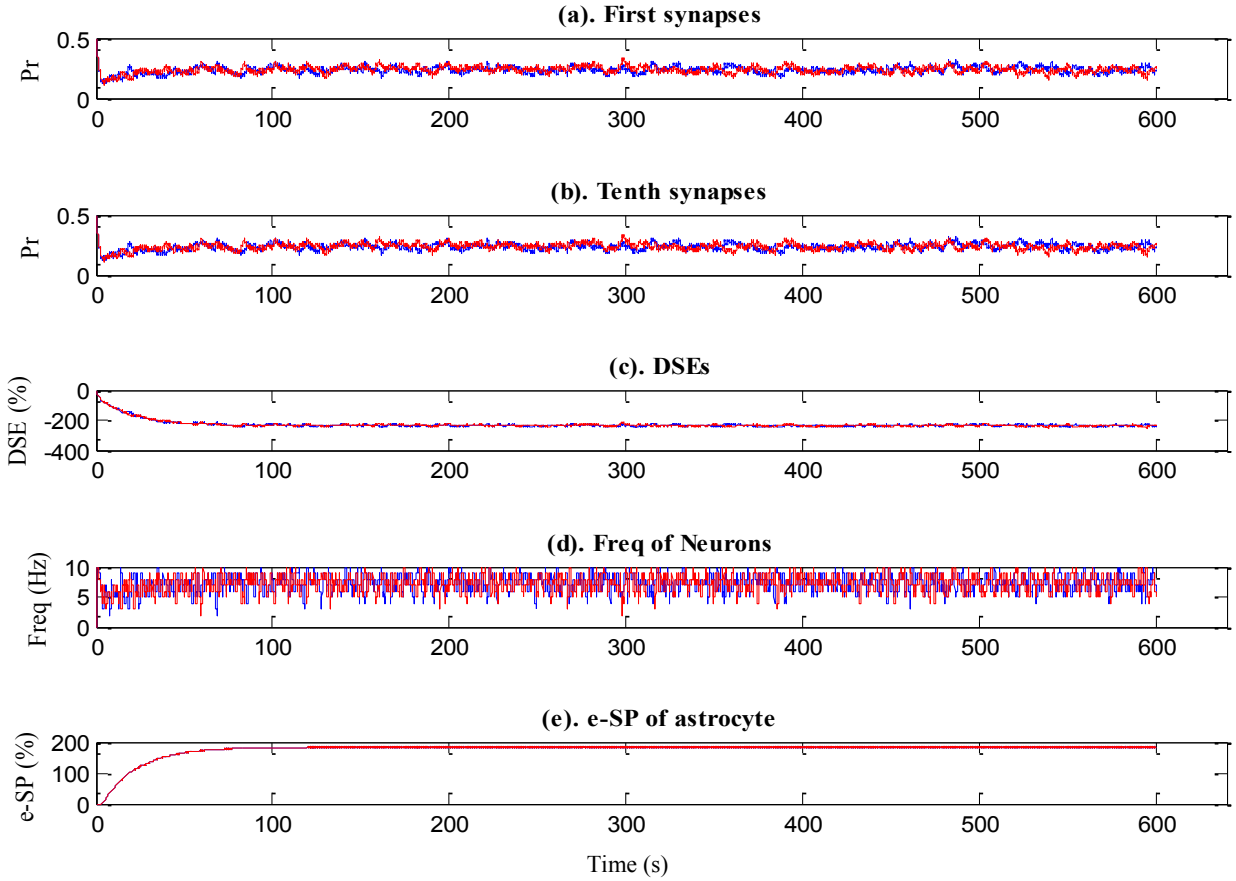

Fig.2.13 SPANNER repair mechanism no fault [85]. Neuron 1 is indicated in blue and neuron 2 is indicated in red, respectively. The figure shows the excitatory (e-SP) and inhibitory (DSE) signals under normal circumstances with no fault. This shows how the astrocyte signals can balance the PR and the frequency of the neurons.

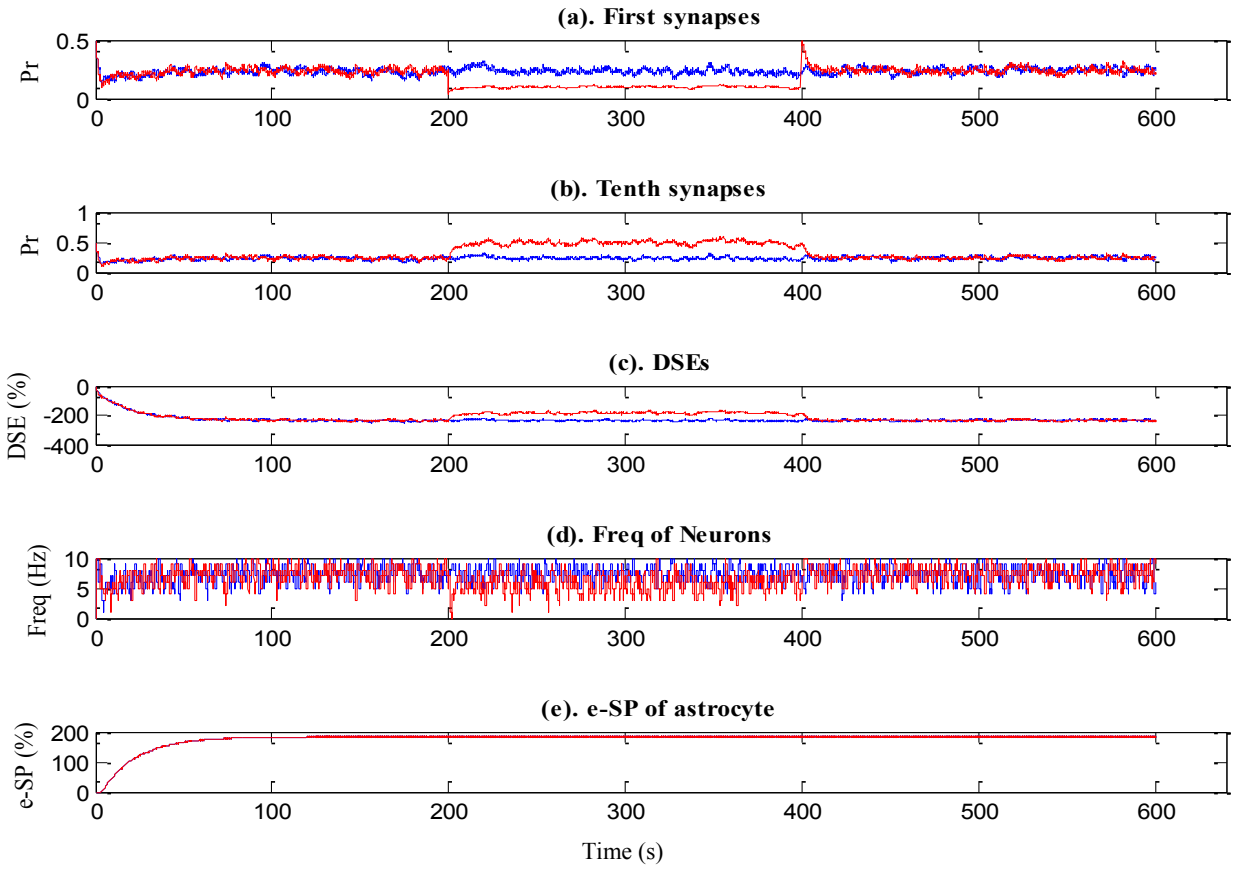

Fig.2.14 SPANNER repair mechanism with faults [85]. Neuron 1 is indicated in blue and neuron 2 is indicated in red, respectively. The figure shows the excitatory (e-SP) and inhibitory (DSE) signals under fault induced circumstances with 80% faulty synapses. The fault is injected at 200 seconds as this allows the signals to naturally balance before faults are induced.

Using this self-repair mechanism Ulster University applied the principles to a mobile robotic car using astrocyte-neuron networks [86]. This applied neural networks and self-repair capabilities to real world systems. The robot car consists of three main components: a spiking astrocyte-neuron network (SANN) implemented on an FPGA and a robotic wheel module. This is the controlling module for the robot car. A mobile robot car wheel hardware module and an FPGA hardware module which reads signal data and presents to monitoring software on a PC, the three modules are labelled (a), (b) and (c) respectively, these module and robot are shown in Fig.2.15.

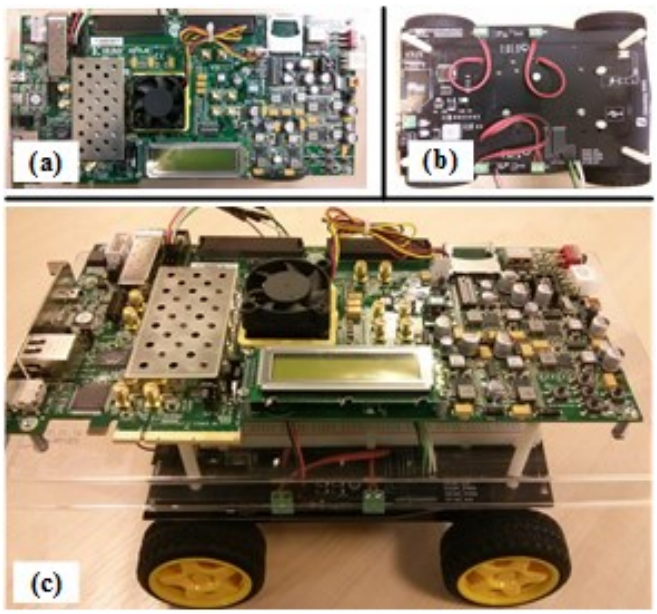

Fig.2.15 Mobile Car controlled by an FPGA based SANN [86].

The SANN is implemented on an FPGA hardware platform In previous work [85], the network consists of two neurons and one astrocyte and each neuron has several synapses associated. The neuron facility, which is neuron A and neuron B, are based on the LIF neuron model and is used to drive the car motors from two inputs, derived from neurons A and B respectively. As the neurons fire at a constant frequency rate the car moves forward, faults are then simulated within this neural network by setting the PR value at each synapse to a value of 0.1. Even with a catastrophic failure rate, 80%, the astrocyte facilitates self-repair. This is based on the global e-Sp signal increasing the PR of the healthy synapses and therefore the SANN can restore functionality [86].

SANNs have demonstrated the capability of self-repair and is representative of how the brain repairs. The vast number of neurons and astrocytes within this network introduces multiple computational paradigm problems. Neurons communicate with neurons, astrocytes to neurons, and astrocytes to astrocytes. Each using different communication protocols and patterns. It is difficult to realize this network in hardware using traditional interconnect mechanisms. Hierarchical astrocyte network architecture (HANA) [87] is an interconnect based on a hierarchical NoC interconnect to support the information exchanges between astrocyte cells within a neuro-glia network of neuron and astrocyte

cells. HANA is a two-layer interconnection structure which supports astrocytes and astrocyte tile facilities. This hierarchical interconnect consists of astrocyte tile facilities to allow the exchange of IP_3 (the signal used to communicate between astrocytes) between astrocyte cells. HANA focuses on astrocyte to astrocyte communication with no astrocyte to neuron communication. The main objective was balancing local and global traffic on the Astrocyte NoC using a two-tiered network and separate local and global communications. Analysis shows that the area overhead could be reduced however, as it focused on the hierarchy and the communication protocols the area efficiency wasn't the priority.

The communication protocol employs a ring topology, a token technique and a packet priority scheduling mechanism, this balances the local and global astrocyte network traffic within the network. The astrocytes connect directly to neurons and each astrocyte is connected to an astrocyte hub in a ring. The astrocyte hub is connected to a second level router, this is the astrocyte tile router. Therefore, the two levels of communication local and global are maintained within the two dimensional hierarchy of routers. Each astrocyte communicates within the ring for local astrocyte connectivity, and the astrocyte tile facilities connect in a mesh fashion to support global astrocyte communication. This also maintains the ability to scale the network with increasing numbers of neurons and astrocytes. It is important to note, each astrocyte tile facility consists of ten astrocyte cells, and each astrocyte cell connects directly to ten neurons, thus one astrocyte tile facility can accommodate ten astrocyte cells and one hundred neuron cells.

2.7 Challenges

Implementation of a neuro-glia network has several difficult challenges to overcome. It is a challenge to provide scalable interconnect, between the neural network and astrocyte network. Astrocytes connect to other astrocytes and neurons, they also communicate using different protocols and time scales at which information is passed between cells (both neuron and astrocyte cells).
There are 3 main challenges:

1. A Vast number of connections: Neural networks, have a huge number of connections. If we can assume that in a neural network, the number of neurons to synapses is N^2, this is the number of neurons have N^2 connections. With the addition of *m* astrocytes the number of connections will become N^2 x *m*. As the network scales the number of connections grows at an exponential rate.
2. How information is exchanged: Binary events vs continuous numerical exchanges of information. Spike events are quick and basic, a 1 or a 0. Within astrocyte networks, there is a continuous exchange of calcium, this happens over a longer period of time as calcium waves oscillate. This is completely different and as of now, there have been few attempts to recreate this information exchange.
3. Timescale: Due to how astrocytes exchange information, the timescale of communication in each network is different. Spikes can be exchanged using clock speeds 1 Hz to 100kHz. This is easy to attach to spike rates on boards, this means that the rate of spikes can be accelerated. On a biological timescale, astrocytes communicate in the scale of seconds, this is very slow in comparison.

There are small positives to be gained from these dynamics. The biological timescale allows a slower timescale reducing the need for huge throughput. This allows room for a trade-off between throughput and area. Nevertheless, there are large and difficult challenges when implementing a neuro-glia network and providing a communication infrastructure between astrocytes and neurons allows a trade-off between parallel and serial communication.

2.8 Summary

This chapter has provided a review of current methods of fault tolerance and self-repair as well as a review of self-repair in biology. This chapter reviews current SNN hardware and how NoCs have provided a solution for the current network constraints of SNNS, this is they are inherently vast, as there are a huge number of connections between neurons within a network. NoCs have been implemented successfully in current SNNs to allow connectivity and high throughput which the SNN needs and now the digital interconnect

is a standard solution to connecting many PEs on a chip it is also a standardised solution for PEs or neurons, in a SNN.

Current fault tolerance methods come at the expense of a large area overhead and require reserves of spare parts. They rely solely on redundancy and spare cells to facilitate self-repair, this also increases complexity, in terms of reconfiguring the spare cells. As an alternative solution, SNNs have turned to the brain. Within the brain astrocyte cells facilitate self-repair, the aim of this biological self-repair is to of implement this self-repair process in a Neuro-glia network, that is, a network made up of both neurons and astrocyte cells, which aims to emulate a closer realisation of how the brain works. The aim of this is to emulate self-repair within hardware, building on an existing SNN application, to increase the lifespan of current SNNs, creating a more biologically accurate and plausible hardware SNN, similar to that of the brain, without downfalls current approaches such as TMR would incur such as a huge area overhead or imposing on the SNN application. The need for such a self-repair approach would allow an astrocyte network to work in parallel with a SNN such as H-NoC, creating a neuro-glia network and providing this SNN with a method of fault tolerance without huge power/area constraints.

However, there is a significant interconnect problem; as the network scales, the number of PEs increases and thus, the number of connections increases exponentially, this is therefore, not scalable. Using NoCs to provide a scalable interconnect, may be the solution to realizing this vast number of connections. As computational models of repair have been successful it is therefore timely, and the next step in developing and realizing these networks in hardware. It is necessary to explore and develop a SNN with self-repair, using non-traditional methods (buses and wires) to create an interconnect with low overheads. Such an approach is discussed in Chapter 3.

Chapter 3: Networks-on-Chip: an innovative solution

3.1 Introduction

In terms of System on Chip (SoC) and Multi-Processor System on Chip (MPSoC) architectures, typical interconnect strategies (wires and buses) are unable to support high performance i.e. the high throughput and high connection demands due to complexity and [17]. Therefore, the bottleneck of said traditional interconnect strategies are a network paradigm where performance and throughput is important, e.g. a typical system bus is good for commercial CPUs but does not scale very well. Fig.3.1 is a typical bus interconnect. With the increasing numbers of processing elements (PE's) in MPSoC architectures and the vast number of connections between these PE's, latency grows exponentially with each new PE as does complexity. The Network-on-Chip paradigm has become a revolutionary step in terms of developing interconnects based on computing networking protocols and is a promising solution for large dense parallel structures i.e. with the aim to create neural networks where networks consist of 10^{11} neurons and 10^{14} synapses are connected in parallel. Traditional bus systems are not fast enough and too complex. Recent examples of NNs using an NoC interconnect have been mentioned in the previous chapter, SpiNNaker [64], Neurogrid [65] and H-NoC [79] and again used in the current SANN at Ulster [85], [87].

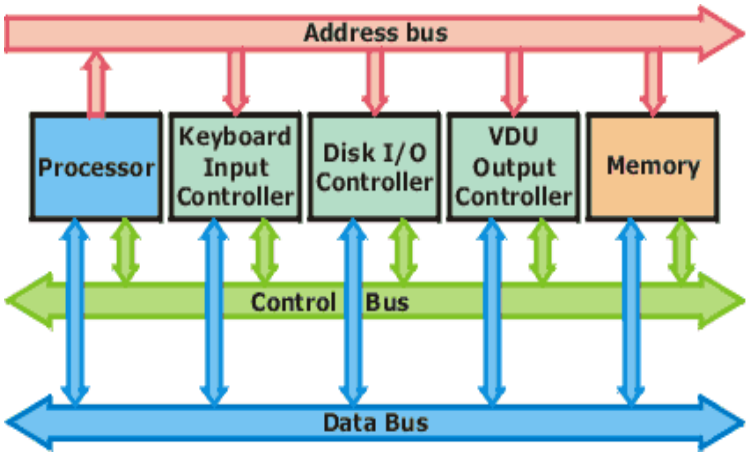

Fig.3.1 A typical system bus [72].

Networks-on-Chip (NoCs) has emerged as a standard approach in order to connect many cores on a single chip, over traditional approaches of using wires, buses and crossbars [16]–[18]. The NoC interconnect is inspired by computer networking, where a large network of computers connect together using routers and packets of information used to communicate data successfully in vast networks of PEs. Based on the same principles, connecting PEs via routers and packets, provides a scalable hardware connection mechanism with a low area/power overhead. Thus, it is timely to explore NoC technology as a solution to connect a large quantity of elements within a neuro-glia network. The NoC interconnect strategy provides a communication mechanism based on the packetization of information and the use of routers to provide a scalable, low power/ area solution. There are a number of aspects regarding the development of NoC interconnect, including the topology which refers to the physical layout of PEs within the interconnect, adaptive routing scheme and router microarchitecture. Firstly, NoC network consists of PEs or cores, these are then connected to network adapters, routers and links to route and send information throughout the network. Each PE or core is attached to a network adapter which is the physical link between PE and router, routers are connected in a predefined topology, selected based on performance, i.e. throughput or path diversity, and the router sends this information through physical links in the network. Fig.3.2 shows a typical NoC topology, connected in a mesh fashion, and shows how the routers connected to PE's are connected within the NoC.

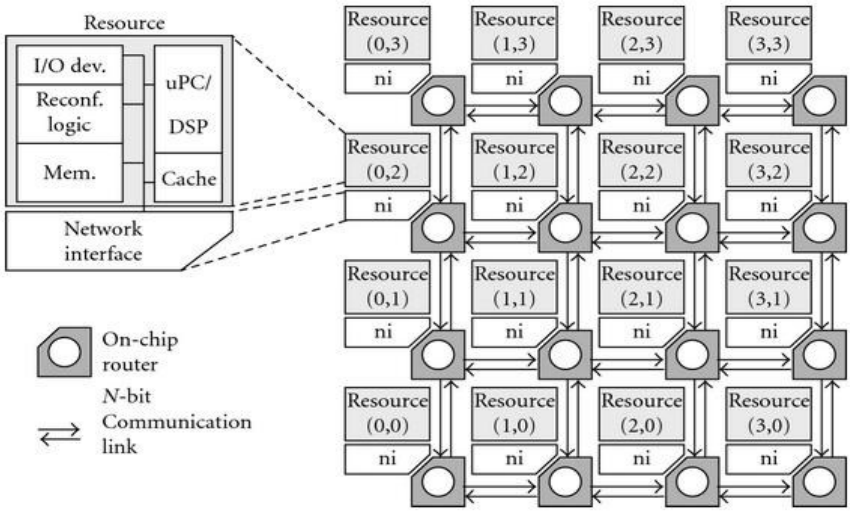

Fig.3.2 A typical mesh NoC infrastructure.

NoC is considered as a suitable solution to the interconnect problem within SoCs [16]–[18]. It is the manner in which the neurons are connected within the network which provides the low power and area solution, which is scalable and allows vast number of neurons to be connected within said network. An SNN can be viewed as analogous to a NoC interconnect, the neurons can be viewed as PEs, the links as synapses and the topology as the dense interconnect within the mammalian brain. NoCs have provided highly scalable, low power/area interconnects for SNNs [71]. A neuro glia network consists of more than one single type of neural cell, as it consists of many glial cells including astrocyte cells. Astrocytes, as previously mentioned in chapter 2, mediate information exchanges between neurons within the brain. They also exchange information with other astrocytes, creating a network within a network. The astrocytes provide an additional interconnect problem, as there are now two networks, consisting of very different communication protocols, working in parallel. H-NoC and NoC technology provides the foundation on which to build a low power/area and scalable interconnect capable of connecting the many astrocytes within a network of neurons. This will then provide a platform on which self-repair may be realized within SNN in hardware.

This chapter provides the following:

1. The advantages of using NoC as an interconnect for a neuro glia network.

2. The basic components required within a NoC interconnect e.g. router microarchitecture, topologies and routing algorithms.

3. A literature review detailing current large scale NoC implementations and those used in current brain-inspired paradigms.

4. The advantages of using NoC technology to provide a suitable interconnect for neuro-glia networks.

3.2 NoC interconnect advantages

NoC technology provides a suitable interconnect solution to networks. NoCs consist of vast numbers of connections and can provide a platform for performance-based networks requiring a large throughput or a fault tolerance strategy. They have already been used for neuro-computing paradigms providing an interconnect for current SNN applications [64], [79]. Providing a scalable and low area/power overhead with a high communication throughput between neurons. In terms of a neuro glia network, NoC provides a scalable and low power/area overhead solution capable of providing communication protocols for two separate networks, which is of vital importance. The advantages of NoC technology are summarized below.

Scalability: In terms of traditional interconnect protocols in hardware, previous approaches used a point to point or bus interconnect. These approaches work well with low numbers of PEs, however as the network scales, these approaches are unable to cope with the increasing load which affects the overall performance of the network. For example, in terms of using a bus. As the number of elements increases the bus is unable to provide adequate throughput as well as the network latency increasing with each additional PE. Fig.3.3. shows a 3x3 mesh network scales to a 5x5 mesh with low overhead and not a lot of added complexity. The bandwidth of a bus is proportional to the number of PEs which share the bus [17].

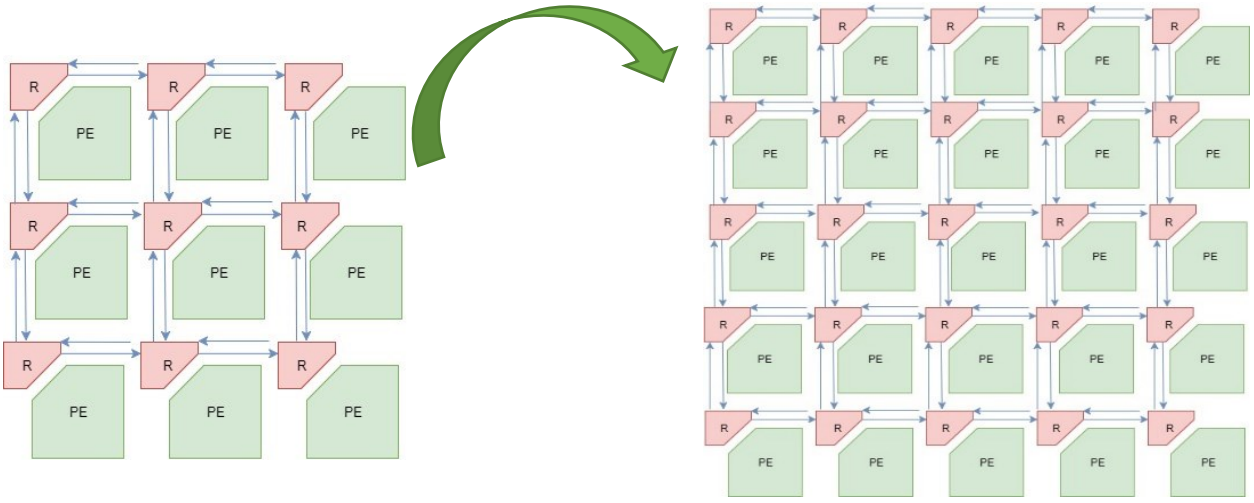

Fig.3.3 Scaling NoCs, comparing a 3x3 array to a 5x5 array.

Within NoC technology, the network scales up by exploiting the short physical links between routers. This allows the NoC to handle more traffic and higher latency a lot better than the bus, of course this is dependent on the topology. It should be therefore noted the NoC has better tools and resources to cope with additional strain of additional PEs i.e. as the network scales with increasing numbers of PEs, the NoC scales with the number of routers connected to each PE. SNNs implemented using a hierarchical approach allows the network to scale efficiently with the increasing numbers of neurons within a network. H-NoC [79] allowed 400 neurons to connect within a single cluster and this opened the possibility of therefore connecting clusters together, this will be discussed in more depth in the latter parts of the chapter. Therefore, using the same approach in terms of astrocyte communication, is a promising solution to the interconnect paradigm presented by a neuro-glia network.

Modularity: The NoC allows PEs to be connected within a network. A more interesting aspect is the modularity a NoC provides, allowing different PEs with different protocols. Each network adapter and router is connected to an independent PE. This allows the router to process the data and send it throughout the network. It is then possible to use one router for various types of communication e.g. the router receives 8-bit information from the network adapter. The network adapter identifies what the packet is and where it is going by reading the header within the packet and then sends it to the corresponding router. It may also accept a packet with 16 bits and again identify the packet and send it to its corresponding router. This allows the flexibility of using different network protocols within one network [88]. This is extremely appealing when understanding the mechanics and communication protocols within a neuro glia network. In terms of NoC engineering it is possible to connect two completely different PEs and connect those using similar routing schemes. For example, a GPU is made up of many processing elements, these are considered as dedicated hardware elements with different purposes such as graphics processing, within the GPU they are connected using the same routing interconnect as the router is separate to the PE, the router sends and receives the packets and sends them where they need to go. This can be applied to processing cells (neurons and astrocytes).

The topology can be considered a key component of a NoC. The topology defines how routers communicate. Another theoretical implementation of neuro-glia routers could be the use of different routers that use similar protocols for different communication types; an identifier within the packet is used to distinguish the type of communication. As well as where the information is required within the network, it could be used to exploit the locality of neurons and astrocytes i.e. connecting a neuron router to an astrocyte router. Fig.3.4 shows a NoC protocol which connects different routers with similar protocols and shows a router connecting a neuron router to an astrocyte router.

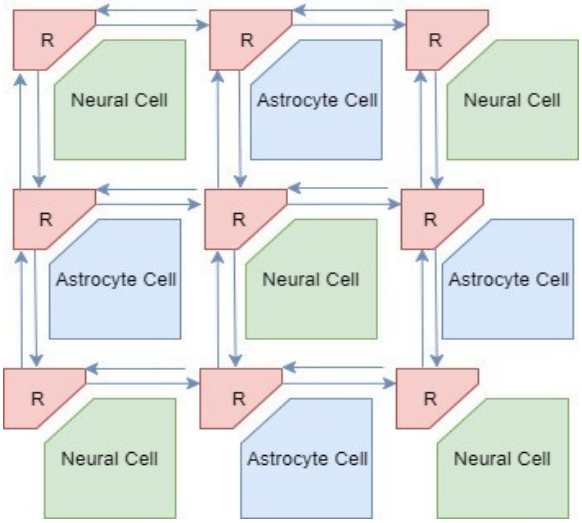

Fig.3.4 A theoretical overview of astrocyte and neuron routers communicating.

It can also be viewed that as a network scales the modularity and topology are promising aspects of using a NoC interconnect. The options provided by the NoC are therefore exciting and endless. The idea is that a designer creating a neuro-glia network can exploit the scalability and modularity of NoC. This creates a communication protocol which exploits the reusability of routers. Doing so by reusing the neural and astrocyte cells. This allows the network the ability to focus solely on the number of astrocytes, neurons and how they are interconnected without having to recreate or redesign new communication protocols every time.

Communication Parallelism: NoC interconnects allow data to be processed in a parallel manner. Of course, when using independent PEs, the PEs process information in parallel. However, when compared to traditional buses, the bus suffers from increasing latency. This is because of an increasing number of PEs as the performance is reliant on the interconnect. This is particularly important where performance is based on throughput. Bus shared schemes use a shared point of arbitration where a bus controller manages and prioritize communication requests, causing delays and therefore increases latency [16]. Within a neural network, neurons are firing constantly at different rates. A bus topology isn't ideal to deal with a network with this inherent parallelism. The NoC allows packets to traverse the network at will. When a spike occurs it can be directly sourced to its destination address. This parallelism can be exploited to take advantage of the interconnect and increase the throughput as neurons fire more rapidly. H-NoC [79] used a spike compression technique. After a neuron fired there was a biological time step of 10ms, any neurons that fired in the same node would consequently be recorded within this time period and traversed through the network. This increased throughput and reduced latency within the network. A neuro-glia network would benefit in terms of parallelism within the NoC as it uses different communication protocols between two networks. This allows two different neural cells the ability to communicate and work independently, as it works in biological terms.

Fault tolerance: A promising method of using NoC interconnects, is for the purpose of fault-tolerance in research applications. NoCs can be used in a fault tolerant manner as there are many routers and physical links between routers. The NoC can be utilized to reroute packets of information based on faulty links within the network. This is dependent on both the topology and routing algorithm e.g. the NoC can use adaptive routing and a mesh topology for fault tolerance. Fig.3.5. shows a number of faults occurring in a mesh NoC. Packets may be re-routed to avoid faulty links, using adaptive routing schemes a packet can be routed to avoid the faulty links. However, as this thesis focuses on self-repair, there is no need for fault tolerant methods.

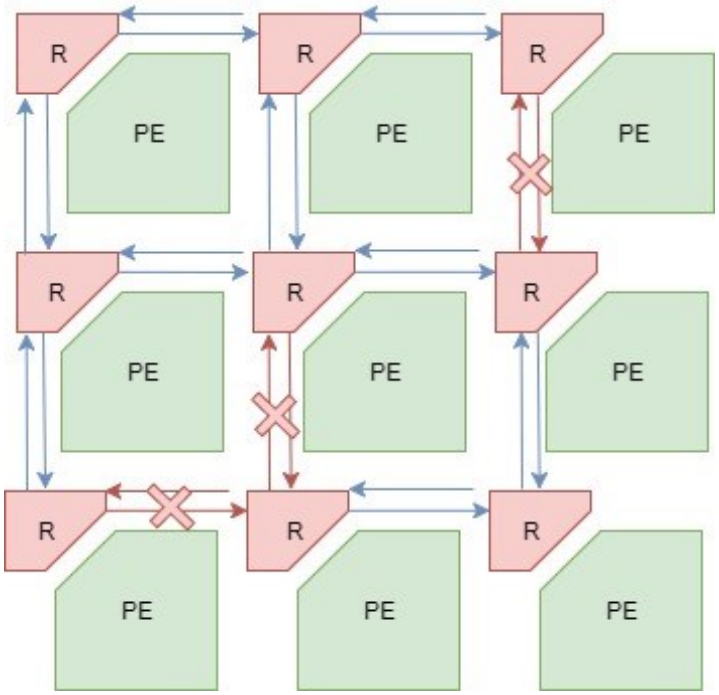

Fig.3.5 Faults occurring in an NoC interconnect. The PEs and routers are separate elements of the NoC, the blue arrows indicate a healthy line of communication between routers, both to and from the router, the red arrows and Xs indicate a faulty communication between routers.

Granularity: There are advantages of implementing NoC technology which provide large scale SNNs with a hardware platform of which, the same benefits can be transferred into a neuro-glia network. The NoC interconnect consists of many components and requires a lot of planning and iterations to be effectively employed. The following components will be discussed in more detail, with each having a significant impact; the topology of the interconnect, routing algorithms and router microarchitecture. Together these components make up an effective NoC interconnect which can be used to create a low area and power interconnect capable of handling both neurons and astrocytes within a neuro-glia network. The following subsections will discuss current NoC technology.

3.3 NoC components

A NoC can be broken into several components, which come together to define a network, and each component having its own characteristics.

3.3.1 NoC topologies

A topology can have two very diverse meanings within the field of neurobiology and NoCs. For the former it may explain how neurons are interconnected and communicate within the brain. Within the NoC it describes the layout of routers and PEs. However, in the same vain it is an overview of how PEs and routers are connected and communicate within a given interconnect. The similarity can therefore be exploited to emulate a biological infrastructure using engineered technology. The NoC topology is measured using specific metrics and can be reduced to trading off performance and area overhead or fault tolerance. The role of the topology has a direct impact on the performance and capabilities of the NoC interconnect. Each PE is attached to a router and the routers are attached in a specific manner to suitably support on board communication within any given network. The topologies are generally defined and compared using the following performance metrics [89], [90]:

Node Degree - This is the number of links at each node or router, including physical paths and communication channels to the closest neighbours, one or many. The number of input/output ports and size of each port can determine the complexity of the router (indicating approximate area and power overheads per router).

Network Diameter - This is the longest path between a source router and destination router. The diameter of a network is calculated based on the minimum number of hop counts across a network.

Hop Count – This is the number of hops a packet of information needs to traverse the network from the source/start router to a destination/end router. The average hop count

is calculated by determining the hop count between every source destination pair within the topology.

Bandwidth – Simply put, the maximum bps (bits per second), which can be injected into the network in a predefined window of time, before saturation occurs. The bandwidth is the traffic the network can support; therefore, more bandwidth indicates the networks ability to handle traffic without inducing latency.

Path Diversity - This is the flexibility in terms of routing channels within the topology. Each router may have a number of communication links, therefore the diversity of a path, refers to the number of paths a message can navigate between a start and end node. Usually this can be used in two ways, 1. To avoid traffic and this is determined by the routing algorithm and 2. The topologies fault tolerance capabilities.

Although, not usually considered a performance metric, the method of communication within a network must be considered as this may affect the topology. This is the manner of nodes communicating e.g. by sending packets from point to point: unicast is from one core directly to another core, multicast is from one to many and broadcast is one to all [91]. This may impact latency and bandwidth within the interconnect.

Variations of these metrics will impact the overall performance of the NoC. Performance metrics being throughput, latency and fault tolerance. It is difficult to directly compare network topologies unless the same number of nodes are present in each topology. Even so these structures can be implemented in hybrid or hierarchical topologies to improve overall performance. In terms of a flat 2D or direct topology there can be some comparisons made, to determine performance metric of given topologies with the same number of nodes.

3.3.2 Traditional topologies

The following are the most common direct topologies [89]:

Ring - This topology provides a network with a simple interconnect solution. It provides each node with a direct communication path to two nodes. The simplicity results in a high latency low area interconnect. Unfortunately, it does not support fault tolerance and is not scalable. In terms of throughput and latency it performs poorly, whilst also providing little fault tolerance, however it can be used to utilise a low area overhead in applications where throughput isn't of great importance Fig.3.6. shows a typical ring topology.

Mesh - This interconnect provides a simple interconnect capable of supporting fault tolerance and scalability. The mesh balances throughput and network latency. It is a basic topology and can be implemented in a number of ways due to its flexibility. Depending on how it is to be used the mesh may be used to support and focus on performance or fault tolerance. The node degree is either 2 or 4 depending on where the source node is situated, this is not good for traffic load balance, Fig.3.7. shows a typical mesh topology.

Torus - This is an improvised mesh which provides the outer nodes of the topology with a direct communication path with nodes on the opposite ends. This increases the node degree to 4 for each node. This allows the NoC to balance traffic appropriately, the average hop count also improves and again, as with mesh, it supports fault tolerance. Fig.3.8. shows a torus topology.

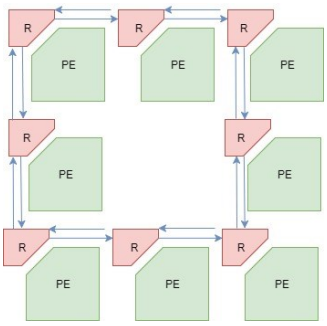

Fig.3.6 Ring Topology

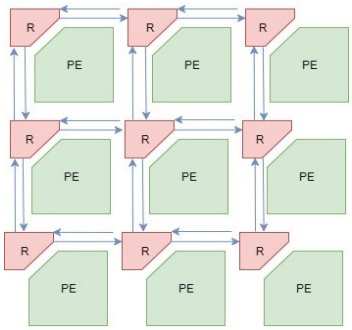

Fig.3.7 Mesh topology

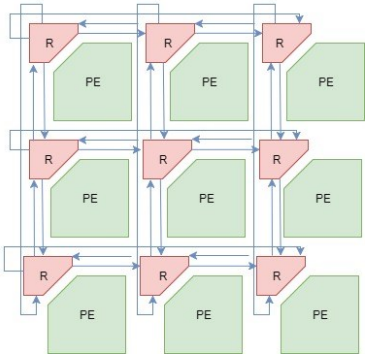

Fig.3.8 Torus topology

Star or Direct – Star or point to point, provides each node with a direct link to a central node. This provides a one-layer communication structure. The direct link results in low latency and due to the structure works very well with unicast communication or multicast

communication as the central node may replicate packets and direct them as appropriate to the leaves of the star. Star topology however does not support fault tolerance; there is only one path for each node and in cases of a faulty path the packets have no way around this link [90]. Fig.3.9 is a star topology.

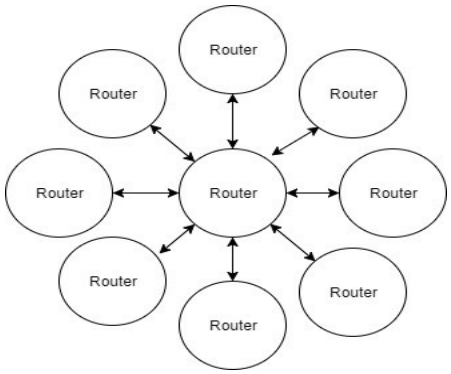

Fig.3.9 Star/Direct topology.

Tree – A Tree topology, this approach offers a hierarchical structure. This provides a high level of communication between nodes as messages are routed up from a node to a common ancestor and then routed down to the appropriate node. The advantages of using a multi-level approach are high levels of communication and the support of one-to-many messages. The main disadvantage is fault tolerance, as it doesn't provide sufficient path diversity [91].

Honeycomb - The Honeycomb structure can be compared to Mesh or Torus topologies where the advantages are low complexity in terms of hardware, fault tolerance and there is lower hop count between nodes. Honeycomb meshes have 25% smaller degree and 18.5% smaller diameter than Mesh with the same number of nodes [92]. Fig.3.10. below shows the concept of the honeycomb topology.

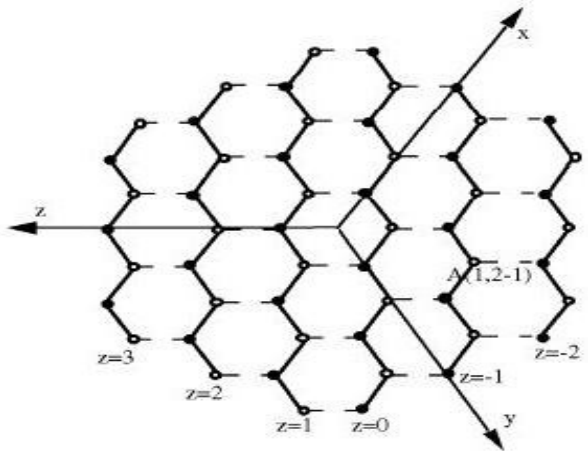

Fig.3.10 Honeycomb topology [92].

Table 3.1 compares 2D topologies using traditional performance metrics. Each topology has pros and cons depending on the main objective which will dictate the selection of a NoC topology. N is the number of nodes in a network, hop count is 2N – 2 in mesh and 2N/2 in torus.

Table 3.1 Comparing traditional network topologies

Network Size	4x4			16x16		
Topology	**Node Degree**	**Max. Hop Count**	**Path Diversity**	**Node Degree**	**Max. Hop Count**	**Path Diversity**
Ring	2	4	1	2	16	1
Mesh	2, 4	6	6	2, 4	254	6
Torus	4	4	N/A	4	128	N/A
Honeycomb	3	N/A	N/A	3	N/A	N/A
Star	1	1	0	1	1	0

There are also indirect topologies [89] including Butterflies, Clos Network and Binary Fat tree. These topologies can increase overall throughput. However, they come at increased complexity and power/area overhead.

This table based on performance metrics, may not give an overall definite conclusion to the best topology. For example, Ring shows poor performance and poor fault tolerance overall however, it is simple and supports slower communication protocols. Mesh supports fault tolerance with improved performance but must incur an increase to both area and power overheads as well as increasing complexity. Torus is similar to the mesh as it may balance traffic a lot more effectively but the cost comes with increased overhead. Star has the best performance as it has point-to-point connection between nodes and a router. The hop count is 1 although it has no room for any fault tolerant strategies. Honeycomb is similar to a mesh and torus with reduced area overhead and a reduced hop count [92]. When comparing topologies solely on performance metrics, it is difficult to draw conclusions, therefore the application for the NoC must be considered.

3.3 Advanced hierarchical topology

A Hierarchical topology approach aims to exploit the strengths of diverse topologies by combining them in virtual 3D or hierarchical regions [93], [94]. This has been shown to increase performance and reduce area and power overhead. This is due to the grouping of elements into clusters and focusing on short communication paths between elements with fewer longer paths. One very important aspect of hierarchical topologies is exploiting the communication locality within clusters of PEs as this can improve throughput and reduce traffic [95]. Examples of using a hierarchical topology include implementing a star ring topology for real time object recognition focusing on 1-N (1 to many) communication, reducing time and energy consumption. This is a multi-level topology where the routers are distributed between cores and connected by using both star and ring topologies. There are eight low-level routers (local) and four high-level routers (global), using two separate topologies which keeps local nodes connected with global nodes Fig.3.11 shows the star-ring topology [96].

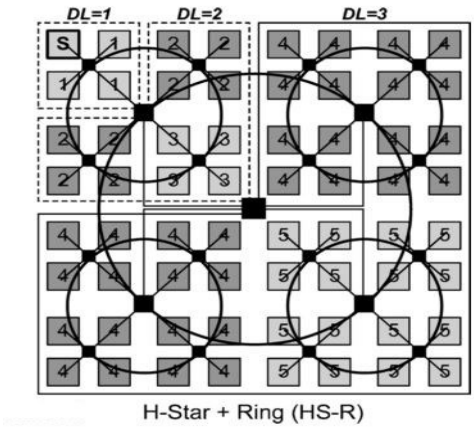

Fig.3.11 Star-Ring Hierarchical Topology [96]. This hierarchy is made up of a two-level star and a two-level ring. The dashed lines show a 4x4 mesh consisting of three different PEs labelled 1, 2 and 3, respectively. Overall, there are five PEs and the network is 8x8. This hierarchy is optimised to connect all the PEs within this network.

The focus of combining topologies aims to take the advantages of individual topologies and combine them, for example combining star and mesh, this will reduce hop count whilst maintaining a good path diversity for fault tolerance [90].

Due to natural progression, 3D topologies have been introduced and implemented [97], [98]. 3D interconnects propose adding a vertical axis into the interconnect by stacking 2D interconnects. This has been shown to increase the throughput and reduce power consumption and area overhead [99], however, it is more complex to realize in hardware due to the fabrication process.

3.4 NoC communication

A NoC uses packets of data to send information from router to router. The PE sends information to the network interface and this interface packetizes the information and thus, will send the data to the appropriate router to communicate. After a topology is defined a routing algorithm is implemented to send the packet of data through the network from source to destination [89]. Packets are messages containing information made up of a header and address information as well as the information which needs to be communicated to other PEs. Fig.3.12 outlines a typical NoC packet.

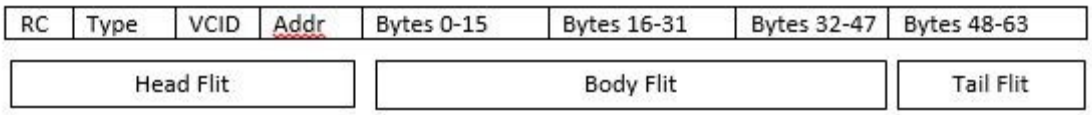

Fig.3.12 A Typical NoC packet: Consisting of a Head, Body and Tail flit, the Head flit generally contains the address header [89].

Routers can send packets as one to one (unicast), one to many (multicast) and one to all (broadcast). This will have an impact on which routing algorithm is to be used. The aim of the routing algorithm is twofold, balancing the traffic load across a network and also getting messages to the correct correspondent. Before choosing an algorithm, it is important to take these variables into consideration:

Number of destinations/ Type of transmission [89] – Based on the number of destinations and type of message transmission. The routing algorithm must be based on the needs of the network. Unicast, multicast and broadcast, have very specific requirements to be adhered to, unicast is useful for point to point communication between PEs and multicast and broadcast may be used to mimic biological signals. The neurons can send a signal to multiple or all targets in a neural network, this can be considered one to many or one to all communication. Due to neuro glia networks consisting of neurons and astrocytes. The variation in communication strategies, and is a great advantage of NoC due to astrocytes communicating in a one to all fashion when communication with other astrocytes. This whilst communicating with neurons directly in a point to point or direct fashion.

Routing decision [89] – A routing decision is dependent on how the network will distribute packets and communicate between nodes; centralised routing uses one controller router to makes decisions between routers. This controls routers communicating and sending packets. Source routing uses the source node i.e. the router communicating with the node sending information makes the decision or distributed routing where all routers involved in the communication to create a path i.e. wormhole routing. This is a difficult approach to identify and set in stone. As a result, the router

connected to the source node makes an initial decision e.g. the destination of a packet. The routers in between aim to complete this routing path. If there is traffic, the router is adaptive and may change the predefined path based on traffic within the network.

3.4.1 Routing algorithm

The three main approaches to routing algorithms are Deterministic Routing, Oblivious Routing and Adaptive Routing.
DOR/Deterministic Routing - This algorithm insists on one known pathway for a packet to travel between source and destination in a network. It is very basic and limited in regard to traffic congestion and/or faults within the network. Concerning congestion, if there is a blocked pathway stopping a packet from reaching its destination the router will still send packets regardless. If all pathways are predetermined to avoid congestion i.e. not allowing many nodes to access the same pathway, then this may improve throughput however due to faults being unpredictable. If a path is damaged the router cannot take this into account it will attempt to keep using the faulty pathway, which results in lost data. [89].

Oblivious Routing - This allows a random path to be chosen by the router between source and destination. Thus, packets will not always travel the same path. This routing algorithm doesn't take traffic congestion or faults into account and has no effect on balancing the traffic load causing congestion [89]. The advantage this algorithm has over the DOR is that because of its random nature, traffic load will be spread more evenly throughout the network without implementing a predetermined route between each node. It is also better at dealing with faults as only some packets will be lost during transmission.

Adaptive Routing - This is the most advanced routing strategy as it allows a router to select a path for a packet to travel between nodes. Factoring in traffic congestion and/or faults in the network, faults or hotspots can be avoided within the network by using a different path. This routing approach offers a level of fault tolerance as a faulty path may be avoided [100]. This is also used for balancing the traffic load within a network by avoiding congested paths. This leads to increasing throughput and reducing congestion

[101]. It will also result in increased complexity and area overhead within the router. Adaptive routing has also been applied to large scale SNNs [102].

Based on the methodology of the network and the optimal topology and routing decision and algorithm, a network can be implemented to communicate data within the network and therefore, in the context of a neuro-glia network, the above variables will have a large impact on the network and thus the implementation of this network. There are two main ways of implementing routing algorithms, finite state machines (FSMs) or look-up tables (LUTs). A LUT is a simple and quick technique, it uses a table within the router to predefine keys or paths for the router, when a packet comes in the router looks at the table and selects the path [89]. The downside with a LUT implementation, is the area overhead. As the table size increases, the more flip-flops are needed to fit the constraints which therefore uses more resources and increases area overhead per router. A FSM on the other hand, can be used in a very efficient and intelligent way, each state is designed to handle different parts of the routing process. A packet comes in to a router, the router looks at the header address and destination address within the first state, and then based on the information, it may go into the second or third state and thus state machine makes a decision on where the packet will go. FSMs are very flexible but more complex than using LUTs, they take a longer time to make a decision, which may increase latency within the network itself. Therefore, the implementation of the routing algorithm is as important as the algorithm itself.

3.4.2 Router micro-architecture

When a topology and algorithm have been consequently chosen, a router micro-architecture is developed, the topology and routing algorithm heavily influence the micro-architecture of the router. The router has several functions as it is responsible for receiving and sending packets from source to destination within a network, this information is contained within the header and address information contained on packets. The router consists of various elements: Crossbar switch, Link controller, Virtual Channel controller, Routing and arbitration unit, Buffers and Processor interface [91]. Fig.3.13 shows a basic NoC router design.

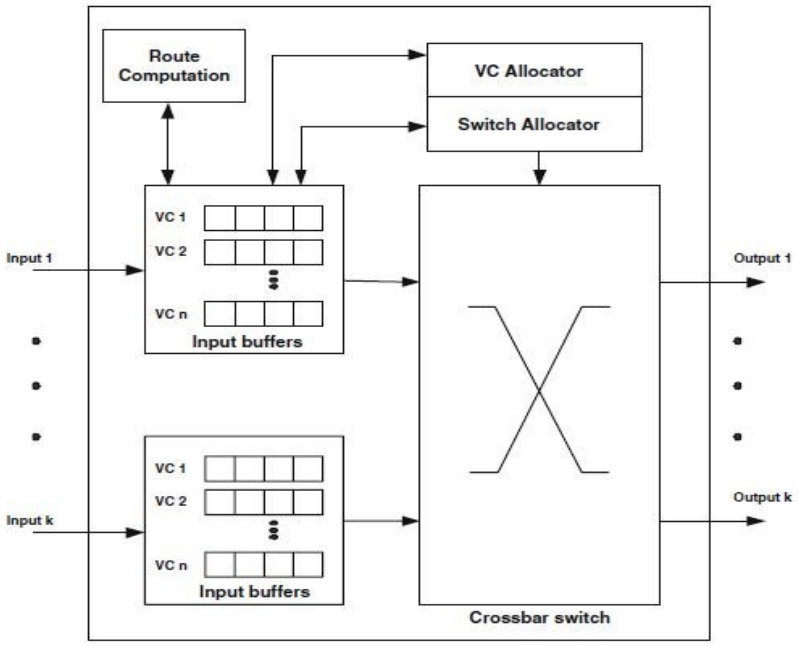

Fig.3.13 NoC Router Microarchitecture [89].

The router micro-architecture has an overall effect on the power consumption and area overhead as well as throughput [89] therefore the router design is critical.

A router is made up of the following components [89]:

Input/Outputs ports - these are physical input and output channels within the router, they connect the routers within the network.

Virtual Channels (VCs)/Buffers - A virtual channel is a buffer on the input, several VCs are placed on the one physical wire. When a packet is sent, it is split into flits, these flits are sent in a specific order. The next flit will not be sent until the previous flit has been sent, this causes queues in the inputs. A virtual channel allows a separate message to traverse the same physical input channel even though a separate VC may be blocked.

VC Controller - This controls the multiplexed nature of the VCs and ensures several VCs can use the same physical link.

Crossbar - This connects the inputs to the outputs. A crossbar allows full connectivity for all the physical channels.

Routing and Arbitration Unit - This is the component responsible for choosing the path of each packet, the packet will arrive and depending on the address header an algorithm will be used to communicate the packet.

The complexity of the router will mould and define the router, in terms of the network, the router has the greatest impact on power and area overhead. The router micro-architecture is dependent on several variables and there is no one way of creating a router. There are a number of components which should be standard for all routers. These are configured for each individual network and its constraints i.e. increasing the number of input/output ports etc. A typical flow of information from the inputs to the output ports, depends on the complexity of the router and is of utmost importance when evaluating the performance of a network in terms of throughput, latency, power and area overheads, and all must be considered when developing routers for a neuro-glia network.

3.5 Challenges of a neuro-glia NoC interconnect

Astrocytes connect to groups of neurons in clusters, and also form their own networks. Emulating a neuro–glia network in hardware introduces additional interconnect challenges beyond current SNN hardware. For example, it is not just the additional connections, which in itself is challenging, but the network must support two very diverse communication protocols. The two main factors are 1. The type of information communicated between astrocytes and neurons, and 2. The different timescales involved. The key challenge is to design a new interconnect mechanism in hardware which can accommodate (1) the increased number of interconnections of neuro–glia networks, (2) the different types of data being communicated and (3) the varied time-scales between signal exchanges.

This challenge is compounded by the need to achieve this with the following challenges in mind:

Power consumption – This is directly proportional to the number of routers within a network. The biggest source of power consumption is the routers within the NoC

interconnect [103]. The router architecture within a large-scale neuro-glia network along with the number of routers between astrocytes may have a huge impact on overall power consumption. These must be kept within tight constraints; the network topology is critical to optimise the number of routers used. Exploiting the parallelism of a neuro-glia network using NoC technology and a hierarchical topology could minimise power overheads and exploit the local and global communications whilst not reducing the neuro-glia capabilities.

Area overhead – The router complexity determines the area overhead. The complexity is dependent on the performance requirements and the number of routers required within a network. A simple router, without VCs e.g. as shown by [89], may be implemented when throughput isn't critical but there must be a balance between area and throughput. Hierarchical topologies can exploit the communication locality between nodes and maintain a scalable topology whilst maintaining a low overhead. A hierarchical topology can exploit the advantages of several topologies and with a slow communication throughput. The astrocyte can use a ring topology to exploit the simple topology and local communication between astrocytes and use a point to point for global communication.

Throughput – The router and application determine the throughput constraints. Along with an adaptive routing scheme, high throughput requires increased micro-architecture complexity. The number of VCs and buffers will affect the throughput, which will increase power and area overhead [89]. However, a neuro-glia network, has a very slow communication protocol as calcium waves oscillate and gradually change. This can be exploited as the network does not have to support throughput and can be designed to exploit the rate of change within astrocyte networks.

Traffic congestion – With a SNN, traffic is inherent due to the rate at which neurons spike. H-NoC [79] used a spike compression technique to increase throughput when neurons began firing. Astrocytes are slower in terms of speed of transmission and have a bigger payload. This may cause latency within the network. Using a similar technique traffic congestion may be reduced within the network. Neuro-glia networks change gradually and when one astrocyte wants to communicate a time step may be introduced

to listen to the other astrocytes before making a decision regarding communication to other astrocyte cells.

Complex topology – The complex interconnect of the brain is very difficult to mimic in hardware. The dense infrastructure of neurons and astrocytes, the communication, the connections and separate network protocols is difficult to capture. Using a hierarchical approach allows PEs to be connected in such a way to allow parallel networks to work independently. This includes, in theory, a SNN and a network of astrocytes. H-NoC [79] presented a topology that exploited separate advantages. Combining a point to point topology with mesh allowed the high throughput of the star topology and also maintained a scalable factor when the network is scaled. By extracting the constraints of a biological system, topologies can be explored and optimised to allow biological systems to be emulated. A hierarchical topology increases the overall complexity of a network. This approach would support using separate routers and protocols and allow communication between networks. It can maintain a low area and power overhead constraint and would remain scalable, i.e. the number of astrocytes would require scalability as the network of neurons scaled.

Granularity – The number of neurons and astrocytes within a neuro-glia network is vast. For every ten neurons there is one astrocyte, this will require a complex interconnect to allow communication between neuron to neuron, neuron to astrocyte and astrocyte to astrocyte communications. The NoC will require a hierarchy to support all processing elements and all subsequent communications within the NoC. This is two parallel networks running simultaneously. A small-scale network of 1,000 neurons would require 100 astrocytes.

3.6 Summary

The challenges of implementing a neuro-glia network is the connectivity of the two networks. Implementing a NoC interconnect mechanism to facilitate the complex connectivity of this structure is a promising solution. This chapter also reviews current SNN hardware and how the NoC has provided a solution for the high levels of

connectivity. Therefore, NoCs have been reviewed as a viable solution for a neuro-glia network connecting a SNN with astrocytes.

As NoCs are highly effective at supporting high throughput and dense communication infrastructures, such as SNNs, they provide a perfect platform for exploring neuro-glia networks and can support large numbers of PEs i.e. astrocytes and neurons. It discusses more recent work and research exploring astrocytes within neural networks, for the purposes of self-repair. A neuro-glia network of vast communication protocols presents an interconnect problem in terms of both power and area. These protocols are required for the number of connections for both neurons and astrocytes. Current spiking astrocyte-neuron networks are exploring large networks with many neurons and astrocytes. As the number of PEs increase the communication and interactions between PEs increases. A promising solution is to use network on chip protocols, as a way of realizing large networks and dense communication procedures. The following chapter introduces Networks on Chip (NoC) and the challenges and constraints of this communication protocol, and how it may be used to realize a neural-glia network in hardware.

Exploring a NoC approach for neuro-glia networks, is a promising prospect. NoCs are flexible, support many PEs, and are intrinsically parallel while maintaining low overhead and high performance. It is the development of the network to exploit these traits which remains difficult. The plausibility of supporting a large-scale neuro-glia network in hardware is plausible using network on chip technology. The neuro-glia network has the advantage of having a slow communication speed, thus allows some room to reduce the area overhead within the network itself and balancing the throughput of the network with the overhead it must incur. Using a hierarchical topology, allows vast numbers of PEs to connect and allows room for development of individual protocols for the astrocyte network. The network will be developed with the overall goal of self-repair in mind, using a hierarchical topology, adaptive NoC routers and efficient communication protocols.

Chapter 4: The Fault Model

4.1 Background

Electronic systems and devices are only as reliable as their components. As devices become smaller, components in turn, become smaller. The drawback is, they are less reliable. As such, the intention to increase longevity and functional life time of electronic systems is considered important. In consumer devices, repair is manageable, the device is sent back to the manufacturer and the faulty component replaced. When a component cannot be replaced, or a device cannot be physically repaired, it becomes useless. However, this is not an option for mission critical systems. The loss of funding and time researching due to a faulty device or system is unthinkable.

Systems are decreasing in size and this results in components becoming smaller and more prone to faults. Moore's law, a trend observed by Gordon Moore in 1965, predicted that the number of transistors on an integrated circuit approximately doubles every two years [20]. This scaling is referred to as geometric scaling; a decrease in the size of systems and components with identical physical capabilities [21]. However, this results in a system becoming more prone to faults where systems may be affected directly. By either faulty components or interconnect (this can be caused during the manufacturing process) [8]. Fault tolerance leads to more reliable and robust systems.

Faults can be characterized as either soft or hard; the latter being permanent and unrecoverable. Soft faults are temporary and common. They are caused by radiation and power fluctuations. They can be repaired or corrected by resetting or reconfiguring the device, usually a system reset restores functionality. Hard faults are caused by physical defects in either a component or the silicon interconnect; by either wear-out or defect during the device manufacturing process [22], [23]. To maintain functionality and increase operational lifetime of an electronic system a method of fault tolerance or self-repair is required. This is important for mission critical systems, when a system is deployed it is unable to be repaired using traditional physical means, usually in space or avionic applications [24]–[26]. Current fault tolerant mechanisms are based on coarse grained

redundancy and employ the use of a central repair-decision agent to either find faults or correct them e.g. Triple Mode Redundancy (TMR). TMR is used in mission critical systems and other areas of hardware development [10], [27]– [30]. TMR is the process of replicating critical components three-fold and using a voting mechanism (comparator) to compare the three outputs and detect discrepancies [31]. This process vastly increases the area overhead [32], and although it may ensure the system can endure faults or the loss of critical components, it relies heavily on spare parts and the use of the comparator (voter) [33]. Other methods include online detection/correction and autonomous self-repair, the former allows testing during operational lifetime. This however, can be invasive to the normal operations, interrupting or shutting down system resources, which is not ideal in system operations [34]. Low level granularity (gates and components) characterises the key weakness of existing approaches of repair, as repairs implemented at a low level incur a large area overhead. This low-level approach does not use TMR, this is due to the work focusing on self-repair within SNNs and replicating all synapses three-fold would increase the area significantly. It is possible to use both TMR and astrocytes however, this is unnecessary as astrocytes perform self-repair at a low level (the synapses) and the astrocytes perform repair by increasing the PR on healthy synapses. This design ensures a graceful degradation, as the network degrades over time the astrocytes perform repair, this ensures reliability over time and is an alternative to TMR. For example, the overhead incurred at the lowest level, a low-level router which is one wire for communication between one astrocyte to ten neurons. The overhead to ten neurons connected in a star topology to an astrocyte is ten wires, 10x, including TMR this is 30x. This chapter also provides a review of existing fault models and fault-tolerant techniques.

4.2 Fault tolerance and self-repair strategies

As technology advances, most applications have become reliant on using general purpose CPUs. As we push the boundaries of traditional engineering for huge tasks, general CPUs aren’t enough. In remote areas such as space, it is common to use radiation hardened microprocessors, as the harsh environment causes bit flips and errors [104], [105]. Radiation hardened FPGAs are more common in space applications [1] and

as NASA release projects like the Mars rover Curiosity [1] and Juno [2] which was sent to Jupiter, these missions require huge amounts of engineering resources. The technology must also include fault tolerance and is becoming a major cost factor when developing and manufacturing electronic applications [106]. This is due to the number of things that can actually go wrong, in other words, there are more components so there is an increased potential for faults or errors. That being said, applications are also prone to faults due to geometric scaling, which affects electronic chips and reduces the functionality and reliability. The smaller the component becomes, the more sensitive it is to both manufacture variations and tolerance accumulations. Faults are developed due to normal degradation and wear out, or design/manufacturing defects. Faults may cause degradation in the performance of a system before it completely fails.

Faults can be classified into two main groups, referred to as soft errors or hard faults. Soft errors are temporary and more common, they are caused by radiation and power fluctuations, and they can be repaired or corrected by resetting or reconfiguring the device [107]. Hard faults are caused by physical defects in either a component or the silicon interconnect; by wear-out or during the device manufacturing process, rendering the afflicted areas useless [108], [109]. There are numerous methods employed to increase robustness and reliability within electronics which are discussed in the following sections.

Current fault tolerant mechanisms are based on coarse grained redundancy and employ the use of a central repair-decision agent to either find faults or correct them e.g. TMR is generally employed in mission critical systems which cannot be repaired after deployment [10]. TMR is the process of replicating critical components three-fold and using a voting mechanism (comparator) to compare the three outputs and detect discrepancies. This process vastly increases the area overhead. Although it may ensure the system can endure faults or the loss of critical components, it relies heavily on spare parts and the use of the comparator (voter). Other methods include online detection/correction and autonomous self-repair, the former allows testing during operational lifetime, this can be intrusive and interrupt or completely stop normal functioning which is not ideal for most applications [110]. The key weaknesses of existing approaches is limited granularity at

which repairs can be implemented (gate, component level) and in particular, the lack of a distributed repair-decision mechanism. It is important to note that the area overhead will increase when replicating components and if the comparator fails, the system also fails to detect faults. There are several methods currently used to detect faults.

4.2.1 Online testing

Online testing refers to detecting errors throughout a systems operational lifetime and is typically applied in mission critical systems when operation cannot be interrupted. Online testing typically utilizes a central unit or additional hardware for the detection of errors. Errors are generally pre-defined and a detection scheme for these errors is predetermined e.g. the testing mechanism may focus on detecting faults at switches, where errors may cause packets to get stuck in the wrong output port [111]. Error detection schemes must be used in conjunction with other methods or detection schemes to be effective as a fault tolerant system [112]. Within NoCs fault detection is distributed and focuses on the three main components: PEs, routers and the interconnect [113]. The PEs are cores or CPUs. The routers are required to transport packets of data within a network and the interconnect refers to physical wires which connect cores, this will be discussed later in the chapter.

There is a trade-off for online testing as it must not impact typical behaviour and the frequency at which testing occurs must be balanced. This is necessary to detect errors early but also not consume excessive access time of the NoC. The aim is to test certain aspects of the system with minimal intrusion and detect errors before they cause further damage. In isolation, detection is limited, and it should be applied in conjunction with some sort of error correction or self-repair procedure, e.g. adaptive routing in NoC systems. The main disadvantage of online test methods is increased area overhead and is generally used for the detection of specific faults or errors that may occur. Although an online system provides testing for faults, it doesn't include any form of self-repair. For further examples of online detection please refer to [110], [114].

4.2.2 Hardware redundancy

Hardware redundancy approaches such as TMR [10] are common. A single voter mechanism resides amongst the hardware and detects irregularities by comparing all outputs against each other. In a working system all outputs should be identical. This can then be used in conjunction with a Cyclic Redundancy Check (CRC) in case more than one instance of a component is damaged [115]. In particular, the key weakness occurs with the single voter unit for detecting a fault as it's a single central component and prone itself to faults. In addition, when one of the backups fail, there is no scope to accommodate further faults. Modern neural based processing systems like SpiNNaker also utilize redundancy in cases where a processor fails [64] as they provide spare processors on the chip. This is done at a processor level (coarse-grained) and is not efficient in terms of area/power (scalability) as often a fault may only affect a very small element of a complete processor and when this occurs the complete processor is non-operational.

4.2.3 Autonomous self-repair

The process of self-repair goes beyond simple redundancy models and can be defined as a systems ability to overcome faults by adapting, with minimal degradation of performance. In biology this phenomenon has been developed through evolution; however, looking to biology for self-repair mechanisms is difficult as an electronic system cannot grow components in silicon. Self-repair without the constraint of a central fault/detect repair unit has been explored [109], [112], [115] [116]–[118]. These self-repair mechanisms focus on distributed methods of self-repair not relying on one central controller or one method of fault tolerance. One such mechanism [109] indicates the use of reconfigurable hardware as a method of self-repair on FPGAs, thereby taking advantage of the reconfigurable aspect of the hardware and implementing spare cells which can be used for self-repair. The authors claim that this strategy does not require spares as they can build all repair techniques within the system.

Research from [115] provide two methods of biologically inspired technology, a three layer "POEtic chip" using their "ontogenetic model" for self-repair. This chip facilitates self-

repair by allowing partial reconfiguration of cells. The model can replicate the functional part of an existing cell to that of a spare cell and creates, or reconfigures, data paths between resources. A second method introduced in the same paper is based on the same principles, inspired by cell division. Fault detection is facilitated through the use of redundancy (TMR) and the use of CRC. When a fault is detected, routing takes place to enable reconfiguration in a spare cell. The spare cell is then reconfigured from the voter mechanism and when it is finished it destroys the input signal as to not be affected from further reconfiguration. While this does offer a detailed and adaptive self-repair model, although it isn't often reported, the area overhead incurred by using TMR for fault detection will increase these components three fold [115]. The spare cells used for reconfiguration will also incur a large area overhead. There is no mention of how disruptive the cell duplication may be or how this may affect performance or typical operation. These current self-repair mechanisms are still fundamentally based on redundancy and replicated instances of hardware e.g. TMR and Dual-FPGA architecture [109]. The use of spare cells is a double-edged sword, on one hand it allows reconfigurable hardware for self-repair however the system is complex and is then limited by the number of spare cells available which also vastly increases the area overhead. Biology and evolution progress to date, sets the motivation to continue exploring how we can exploit them in advancing electronic system design. Implementing astrocytes to facilitate self-repair is a radically new model of repair and could provide a low power/area solution which facilitates repair.

4.3 Fault models

An SNN can be deployed as an application using an FPGA or ASIC (specialised hardware), this is because SNNs are efficient, reliant and robust. The hardware is installed in an area which may be considered a harsh environment. For example, an SNN was deployed in Spain where it was trained to classify sensor data in order to predict forest fires [119]. It is very possible that SNNs will be used to classify sensor information within these harsh environments such as space, the ocean or in radiated environments, such as deploying robots for exploration or predicting natural hazards in order to react faster.

The hardware used must be robust, as there are a number of things that could go wrong, fault tolerance using TMR, is not a complete and reliable solution as the system when compromised, is unreliable. A more complete paradigm in the form of astrocytes and reliable and self-repairing electronic hardware provides a more promising solution. For example, an SNN has input neurons connected to sensors; these sensors are used to classify damage or predict natural hazards. The SNN is trained offline and is used to classify sensor data based on the sensor inputs. The SNN may be susceptible to electronic failures and faults e.g. a sensor failing; but rather than becoming completely useless when it is first compromised by a fault e.g. a sensor failing, it is possible to employ what can be considered as a graceful degradation i.e. if a sensor fails or neuron fails the SNN will continue to work. As long as the network can classify data, even though a neuron has failed within the network, the SNN is still reliable, the accuracy may be reduced, but the longevity and increased operational lifetime have huge benefits.

It is possible to have faults at all levels and these fault models have been outlined below. These have been derived as possible faults based on an SNN implemented on hardware, as a result, there are potential faults in more than one aspect. There are three main aspects:

1. Biological model

The SNN/Astro-network has a number of components where failure can occur:

- Neuron – If a neuron fails it is seen as being a silent neuron, this doesn't affect the model as the network will disregard the silent neuron and increase the PR of healthy neurons. If the neuron is an input neuron the network will be able to classify with the remaining data up to a point, but this at this moment currently unknown.

- Astrocyte – The astrocyte has the potential of being faulty, as it is a complex process. This could be similar to losing a central repair component in a small network however, the astrocyte network is distributed, and it is this distributed manner that makes the neuro-glia network robust. If an astrocyte has a fault, the neighbouring astrocytes within the network will exchange signals in order to restore

the network, this complex network can be seen in Fig.4.1. which shows a large-scale neuro-glia network. This large-scale network is yet to be understood fully.

- Synapse – this is just one part of the network, and typically it may be viewed as the least significant, however, it is the interconnectedness of the network which makes it robust. The large number of synapses connected to each neuron ensures that if there is a fault i.e. a neuron has stopped firing, the PR in healthy synapses increases and restores functionality to the SNN.

This fault model is based on software models where up to 80% of synapses can be damaged. Damage to neurons and synapses is expected, this biological model offers an alternative and biologically inspired model of self-repair for SNNs.

2. Low-level electronics

The fault model is built up from low-level logic and electronic circuits. There are again several aspects; “biological” faults and electronic faults. For the former, if there is a fault in a neuron or astrocyte it can be considered a biological fault, that is the loss of a biological component, this is because even if the hardware is destroyed the self-repair capabilities can overcome the faults and the network will degrade, but it will do so over a longer period of time, this is graceful degradation as electronic components eventually falter and fail.

Other fault scenarios in the electronics are expected; such as stuck-at-one/zero or transient glitches. Overall, such faults would not affect the operation of synapses or neurons. These are viewed as soft-faults, typically a system reset is required, however, the SNN is a reliable and robust network typically restoring activity when affected by hard-faults. If a neuron or synapse is not operational they are viewed as hard faults, the neuro-glia network can reliably overcome such faults.

Note: These faults might affect the astrocytes but are yet to be tested, a more complete neuro-glia network would have to be realised in order to explore how these faults would affect the operational lifetime.

3. FPGA/NoC interconnect

Faults within an NoC interconnect can be very common. There are several methods of fault tolerance applied when choosing an NoC topology (e.g. mesh); using adaptive routing it is possible to circumvent faults in the NoC interconnect by allowing the routers to choose the path of the communication. In terms of the SNN hardware this can again be viewed as a biological fault within the network, the loss of neurons or synapses within a large scale network will affect the overall functionality of the SNN, but the aim of self-repair is to tolerate these faults. For astrocyte communication, dedicated routers are used in Chapter 5 and 6 in order to facilitate self-repair within the SNN.

The SNN will not be affected by soft- or hard- faults as the neuro-glia network will restore functionality which will lead to a robust SNN within a neuro-glia network. The focus of self-repair is employed to allow a network to degrade over time and prevent loss of accuracy and precision. It is possible to have faults in the astrocyte hardware, but a large-scale neuro-glia network, will have a number of astrocytes, the operation of these astrocytes is not the objective of this thesis, rather the aim and objective are to build an interconnect capable of supporting SNN operations using an astrocyte network.

4.4 Drawbacks of existing approaches

Current fault-tolerant methods are limited as they are application specific and are mostly based on redundancy. For example, TMR. This process vastly increases the area overhead [32], and although it may ensure the system can endure faults or the loss of critical components, it relies heavily on spare parts and the use of the comparator (voter) [33]. TMR will fail when a critical component has failed or is compromised whereas, the neuro-glia network can be developed to keep on functioning, providing a graceful solution to this particular problem. Regardless of the fault; hard or soft, biological or electronic, the

neuro-glia network may continue to operate in harsh environments, without the expensive area overhead induced by redundancy.

There are other methods such as online detection/correction and autonomous self-repair, the former allows testing during operational lifetime. This however, can be invasive to the normal operations, interrupting or shutting down system resources, which is not ideal in system operations [34]. Low level granularity (gates and components) characterises the key weakness of existing approaches of repair, as repairs implemented at a low level incur a large area overhead. This low-level approach does not use TMR, this is due to the work focusing on self-repair within SNNs and replicating all synapses three-fold would increase the area significantly. It is possible to use both TMR and astrocytes however, this is unnecessary as astrocytes perform self-repair at a low level (the synapses) and the astrocytes perform repair by increasing the PR on healthy synapses. This design ensures a graceful degradation, as the network degrades over time the astrocytes perform repair, this ensures reliability over time and is an alternative to TMR. For example, the overhead to ten neurons connected in a star topology to an astrocyte is ten wires, using a solution such as TMR this is 30x the wires alone.

4.5 The Objective

A neuro-glia network is a large-scale network capable of self-repair. The network is made up of a neural network and an astrocyte network which work in parallel. The SNN is used for classification and supported by the astrocyte network. The astrocyte can appropriately deal with faults within the SNN, typically silent neurons or faulty synapses, this area of research is at an early stage however, the aim is to deploy SNN hardware in harsh environments, wherein typical applications may fail, to provide a long-term solution and a reliable and robust hardware application. Figure 4.1 shows a completed neuro-glia network realisation and provides an overview of a large-scale application. This figure shows five astrocytes interconnected and each astrocyte supporting a large-scale SNN application.

The objective is to deploy neuro-glia networks on specialised hardware in harsh environments. This hardware contains an SNN, which has neurons connected to inputs, and the network is trained to classify the data based on the inputs and information is passed via synapses to outputs. The hardware will also consist of an astrocyte network which facilitates self-repair.Although the SNN may be susceptible to electronic faults and failures after deployment rather than being compromised, the network will repair itself when possible. This is referred to as a graceful degradation. This model can accommodate all fault models including longer-term drift, as the objective is to address all possible faults (electronic and biological) using this model of self-repair, at least to begin with. If a sensor or motor fails or if an input neuron fails, the SNN will functionally operate, it will lose accuracy and it will become less reliable, self-repair allows graceful degradation and this is a new paradigm within SNNs. This model can support hard faults such as neurons failing within the network. As long as the SNN is reliable, the longevity and increased operational lifetime of this self-repair may have huge benefits.

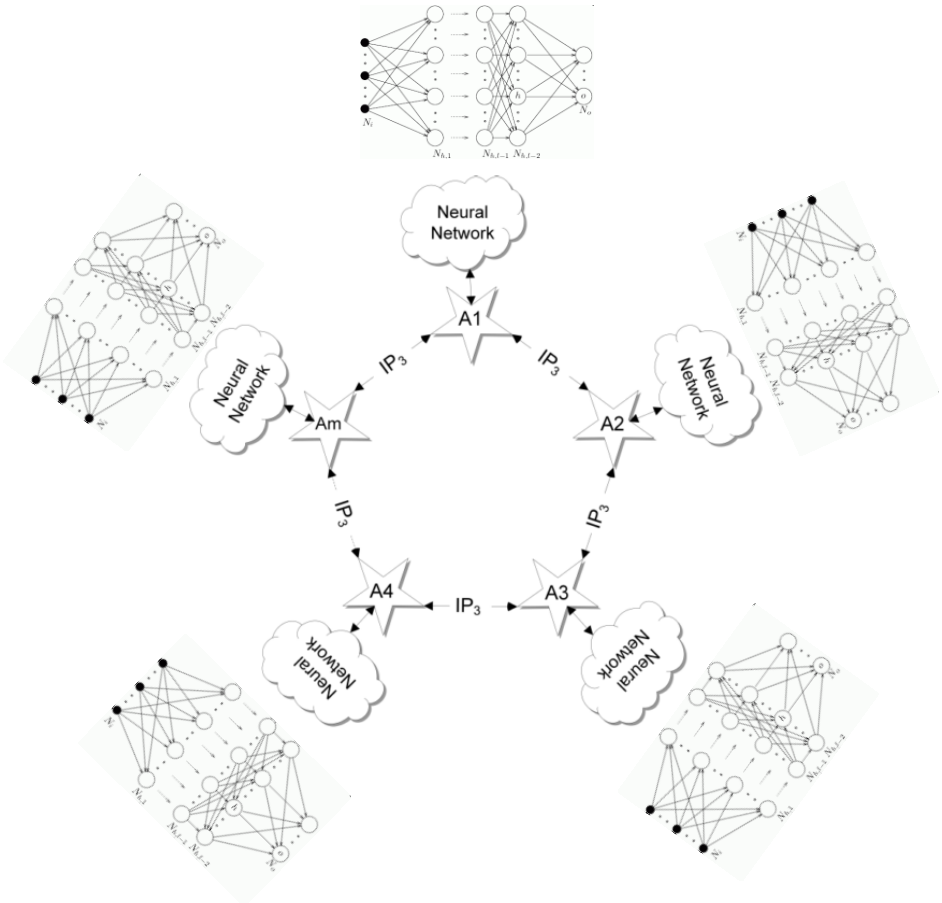

Fig.4.1 A large scale neuro-glia network.

4.6 Summary

Fault tolerance and current models of self-repair are based on redundancy. A biological model of self-repair was implemented in software with promising results. However, when emulating this model on hardware there are additional hardware constraints that must be considered. A neuro-glia network is two networks running in parallel. Implementing a neuro-glia network (i.e. an SNN with neurons and synapses having self-repair capability) on hardware is challenging. This chapter reviews current fault models, current fault tolerance and other methods of repair in hardware and how NoCs have provided a solution for the high levels of connectivity. Additionally, it reviews possible faults and discusses how a neuro-glia network, a network connecting an SNN with astrocytes is a viable method of self-repair.

Exploring an NoC approach for neuro-glia networks, is a promising prospect. As NoCs are flexible, support many PEs, and are intrinsically parallel while maintaining low overhead and high performance. It is the development of the network to exploit these traits which remains difficult. The plausibility of supporting a large-scale neuro-glia network in hardware is plausible using network on chip technology. The neuro-glia network has the advantage of having a slow communication speed, thus allows some room to reduce the area overhead within the network itself and balancing the throughput of the network with the overhead it must incur. Using a hierarchical topology, allows vast numbers of PEs to connect and allows room for development of individual protocols for the astrocyte network. The network will be developed with the overall goal of self-repair in mind, using a hierarchical topology, adaptive NoC routers and efficient communication protocols.

Chapter 5: Local communication: Astrocyte to neuron communication using a ring NoC

5.1 Introduction

This chapter introduces a novel interconnect topology for providing the local communication between astrocytes and neurons in a neuro-glia network. Using a low-level NoC hardware communication architecture, information can be exchanged between astrocytes and neurons. This work extends the H-NoC SNN paradigm which was not developed to support scalable Neuro-glia network hardware. While H-NoC currently provides scalable communication for the SNN, the contribution in this chapter is facilitating scalable communication between astrocyte cells and the SNN. This novel NoC interconnection scheme for communicating e-SP enables a significant number of astrocytes to communicate with neurons within a minimal area constraint.

Implementing a neuro-glia network in hardware is a difficult challenge. Astrocytes implemented within the network can provide a fine-grained distributed repair process within a neural network. The communication infrastructure consists of a vast network that must take into account the number of connections between two separate networks with different communication protocols. The local level communication between the astrocyte and synapses has two individual aspects of communication:

- Normal SNN activity
- Interactions between large numbers of spiking neurons and astrocytes via synaptic connections.

In effect, the aim is to facilitate efficient and scalable communication between an astrocyte and a SNN. This is to allow self-repair capabilities in the SNN as the astrocyte modulates synaptic activity. When neurons stop firing due to faulty synapses, the astrocyte increases the PR across healthy synapses to restore functionality to clusters of neurons. This can be applied to existing SNN approaches by implementing an astrocyte in hardware and communicating between the astrocyte and neurons.

This chapter aims to provide an overview of this biological self-repair process where subsequently this self-repairing capability is investigated in hardware. This chapter also

provides an overview of the interactions and communication exchange between neurons and astrocytes, this being a local communication exchange. An outline of this novel topology and communication protocol for astrocyte to neuron communication in hardware is also provided which enables scalable neuro-glia networks to be implemented in hardware.

A ring topology was implemented to facilitate communication between a single astrocyte and 10 neurons: hereafter referred to as a H-NoC neuron facility. The ring topology reduces area overheads in communicating with the ten neurons and associated synapses and this comes at the cost of increased latency. The ring topology affords a smaller area overhead as the parallel bus is reduced to a serial signal. This reduces the number of wires required to transmit the signal from the astrocyte to the synapses. The signal takes longer to transmit however, this does not impact the astrocyte because the astrocyte to neuron speed of communication is a slow changing signal. Therefore, reducing area overheads exploits the slow transfer rate.

FPGA results demonstrate that the new ring topology provides a good trade-off between low area/interconnect wiring overhead and communication speed for the relatively slow-changing data between astrocyte and neurons.

This chapter provides the following:

1. The challenges including the vast number of connections regarding local communications in a neuro glia network;
2. How the astrocyte facilitates repair at a low level and how a ring NoC can be used to exploit the inherently slow astrocyte communication;
3. The advantages of using a ring NoC as an interconnect for local communications in neuro glia network;
4. The overheads of using NoC technology to provide a scalable interconnect for neuro-glia networks.

5.2 Network design challenges and constraints

The connectivity requirements in a typical SNN causes networking issues because of the complexity of connecting all the neurons. In a simple neural network where all neurons are connected, this can be seen in Equation (5.1).

$$\text{Number of connections} = Ns \times Nn \qquad (5.1)$$

Where *Ns* = number of synapses and *Nn* = number of neurons.

This is a network of neurons and synapses where each neuron is connected. Coupling a SNN and an astrocyte requires a larger communication protocol between the astrocyte and each neuron as seen in Equation (5.2).

$$\text{Number of connections} = Ns \times Nn + Ns\left(\frac{Nn}{Na}\right) \times Na + Na \qquad (5.2)$$

Simplified as: $2(Ns \times Nn) + Na$

Where: *Ns – no.of synapses. Nn – no.of neurons. Na – no.of astrocytes*

The wiring complexity of this design imposes a large connectivity constraint on the design of a communication protocol and the number of wires required induces a huge area overhead. Therefore, reducing both the complexity and the area overhead (number of wires) using a ring topology in NoC, provides a simple and area saving solution.

The challenge is to provide a solution, capable of combining the two networks: a neural and astrocyte network. Thus, the communication protocol must support the network performance whilst reducing complexity and area overheads.

The main constraint is the number of astrocytes in a network that bi-directionally couple to synapses. As the number of astrocytes increase, the number of connections increases linearly. Coupled with communicating using different data types e.g. events and numerical values and varying communication speeds, the networks complexity increases. The challenges and constraints can be summarised as follows:

- Complexity of the network: the wiring complexity for a large-scale neuro-glia network requires a vast area overhead.
- Communication time scales: neurons communicate via spike events, these are quick exchanges typically in orders of KHz. Astrocytes communicate via numerical exchanges, continuous but a much slower exchange of bio-chemicals signals. The timescales vary as astrocytes communicate in order of seconds.

Therefore, the communication protocol must adhere to the constraints above. Due to the slow exchange of information within the astrocyte network, throughput can be reduced with the focus on reducing area overheads. Therefore, a topology that uses a serial communication protocol with less wiring is suitable, i.e. it has sufficient time or 'slack' to ensure all astrocyte to neuron signal exchanges can be communicated within the correct time.

5.3 Neuro-glia network repair

A faulty synapse is modelled by a rapid drop in PR at its associated synaptic sites which results in silent or near silent neurons. Research has suggested that astrocytes can detect faulty synapses associated with silent neurons [11]. This fine-grained repair can be achieved by successively increasing the PR on surrounding healthy synapses and this process restores the faulty neuron to its pre-fault firing activity: the astrocyte increases PR of the healthy synapses which in turn restarts the learning process resulting in the potentiation of PR. The potentiation of PR (repair) re-starts the firing activity of the neuron. Within this process glutamate is released from the presynaptic dendrite. 2AG is released by the postsynaptic dendrite and triggers the oscillation of Ca^{2+} in the astrocyte. Ca^{2+} and IP_3 signals provide astrocytes with communication channels. This process also activates the e-SP and DSE signals which mediates the PR of the synapse. Fig.5.1. from chapter 2, shows a high-level model of repair.

C1 and C2 contain tripartite synapses. In scenario (A) both neurons are firing with zero faults. In scenario (B) N2 has stopped firing due to faulty synapses. From here the

feedback signals e-SP and DSE can be observed. As mentioned, N2 has stopped firing. The DSE stops but the astrocyte e-SP feedback is still active due to N1 still remaining active. The e-SP begins to increase, and this leads to an increase in PR across all associated synapses. The weights of the remaining healthy synapses of C2 which leads to a restoration of the firing activity to N2. N2 can repair until near pre fault levels of firing activity [15]. Table 5.1 shows an overview of the astrocyte signalling values within a neuro-glia.

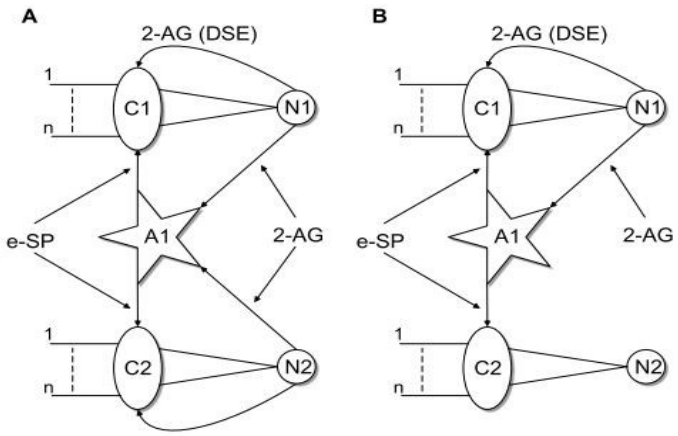

Fig.5.1 Computational astrocyte model. N1 and N2 depict neurons and A1 an astrocyte; e-SP and DSE are excitatory and suppressive feedback signals [15].

Table 5.1 Astrocyte signals and values (for information purposes not related to hardware).

Signal	Signalling Value
e-SP	0 to 200%
DSE	0 to -250%
Pr	0 to 1
Ca^{2+}	0 to ~ 1 microMolar per Synapse
IP_3	0 to ~ 2 microMolar per Synapse
2AG	0 to ~ 0.02 microMolar

Simply put, if there are two neurons firing, when one neuron stops, the excitatory signal provided by the astrocyte (e-SP) is maintained by the neighbouring neuron and this initiates the repair process [15]. Therefore, achieving a self-repair paradigm within a SNN can be realised by the interaction with a nearby astrocyte and their control over the PR of healthy synapses. The increased complexity of the signalling between the astrocyte, synapses and neurons provides the capability to sense and repair synaptic connections, where the astrocyte regulates the degree of repair. Astrocytes are also connected via intracellular signalling routes (gap junctions) which allow the secondary messenger IP_3 (inositol trisphosphate) to pass through, thereby allowing astrocytes to communicate with each other thus providing a distributed repair-decision making capability. At an abstract level, one can view astrocytes as a high-level network which exercises plasticity over neural networks, with interactions between both networks occurring via the direct and indirect signalling pathways.

Progress has been made in modelling the astrocyte process [81] and its interactions with SNNs [15]. These models are limited by the computational resources and the performance of simulations. Implementing astrocyte cells within existing hardware is difficult due to the vast size and complexity of the astrocyte mathematical models. There have been a various applications using astrocytes in both Neuromorphic circuitry [120], [121] and within digital hardware devices [83], [85], [122], [123]. However, progressing neuro-glia computational function requires the implementation of scalable hardware where neural and astrocyte networks must work together in parallel. Neurons spike and communicate to other neurons while astrocytes support this communication and the self-repair mechanism within a neuro-glia network.

SNNs have also been implemented using FPGA hardware. The level of parallelism exhibited by hardware improves the SNN performance over that of software models along with lower power and area overhead. Therefore, hardware emulation has made it a promising prospect to experiment with self-repair on a SNN using a bio-inspired model of astrocytes.

5.4 Low-level neuro-glia interconnect

This section describes a low-level NoC interconnect based on a ring topology, which focuses on the excitatory signal (e-SP) which is activated in the astrocyte when there are faulty synapses: this is hereafter referred to as the astrocyte-NoC. Within a neuro-glia network, e-SP increases the PR of healthy synapses which restores the neuron firing activity. Extending on H-NoC, this paradigm presents a scalable method to support large Neuro-glia network hardware implementations. H-NoC currently provides scalable communication for SNN activity, but not for astrocytes in a neuro-glia network.

Facilitating a low-level communication construct between a network of astrocyte cells and a SNN can be realised using a ring-topology as discussed in the introduction to this chapter, section 5.1 and this is discussed further in section 5.4.1. Implementation of a neuro-glia network in hardware, must consist of two parallel aspects: normal SNN activity and interactions between astrocytes and large numbers of spiking neurons via synaptic connections. The low-level communication between an astrocyte and neurons must happen in parallel to SNN activity and in a non-interfering manner. In effect, the aim is to facilitate communication within and between astrocytes and the SNN network, while forming a single unified neuro-glia network. Neurons communicate with a sequential and event-driven method whereas astrocytes communicate continuously at a much slower rate. Astrocyte data is numerical in value and for the purpose of precision the bit resolution in the present case is 64-bit [124].

To adhere to the constraints of the computational model of repair, it is necessary to adapt the hardware. A “close” hardware model implements the computational model using 64-bit precision. However, this provides a paradoxical problem, regarding insignificant data exchanged at a less demanding throughput compared to the rate of spike events. The biological solution is to do this at a slow continuous timeframe which could cause a lot of traffic and congestion within the network. Fig.5.2. outlines the major signals communicated in a neuro-glia network: the astrocyte cells (stars) and neurons (circles) and the astrocyte network is depicted by the dashed star in the centre. This is a small astrocyte network with 5 astrocytes each with two pre- and post- synaptic neurons. This

network underpins the interactions and communication exchange between neurons and astrocytes (A-N communication) and between astrocytes (A-A communication).

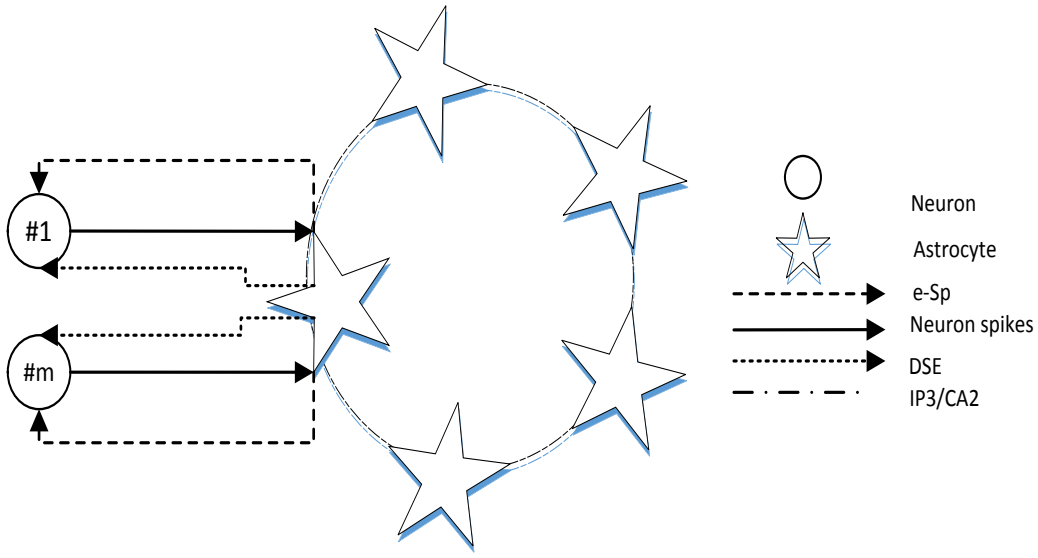

Fig.5.2 Neuro-glia network outlining the major signals within the network.

The inter-astrocyte and low-level astrocyte-neuron communication uses a separate communication infrastructure. H-NoC supports SNN activity, whilst the astrocyte-NoC supports astrocyte activity and communicates the essential data to H-NoC. These networks must function in parallel and exchange appropriate data as required. As previously mentioned, H-NoC is a 3-D hierarchical infrastructure consisting of a Node Facility, Tile Facility and Cluster Facility. At this lowest level, ten neurons are directly connected to a node router, and the topology used in the node is point to point (star). This topology allows PEs and neurons direct connection to the node router. From an abstract viewpoint, it allows higher throughput when the rate of information exchanges is critical. Each neuron spike is received by the node router as a packet [79]: the Node router therefore registers every spike. These packets are received in a non-intrusive manner from the node router. This is due to the inclusion of an extra output port within the node router. It is from this point that the astrocyte-NoC reads the packets (spikes) and communicates with the astrocyte process. Fig.5.3. shows an overview of the two communication processes. The SNN communicates using spikes which are events and the astrocyte is a continuous numerical value. The 2D boxes are the tile facilities and the

circles are the neurons. The blue loop is neuron communications, these are spikes which occur in the neurons and are sent to each node and tile router respectively via an up/downstream within H-NoC. The purple indicates the astrocyte process. Due to the nature of the e-SP signal, throughput is not a key requirement, the process is slow and gradually changes in time, therefore, the tile router in H-NoC sends data to the astrocyte tile router and this communicates in one way to the neurons. The hardware design must accommodate two very different communication protocols.

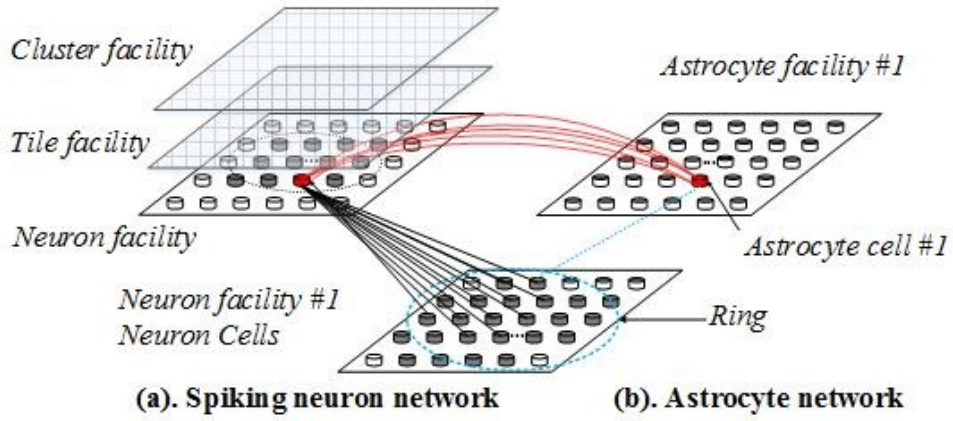

Fig.5.3 Neurons vs astrocyte communication. Two separate protocols within the same network. HNoC has three routers; at the lowest level neurons spike and go to the Neuron facility, blue indicates the ring within the astrocyte-NoC, this communicates back to the neurons.

Within H-NoC, the most efficient and least complex level to implement the astrocyte communication, is from the tile to the node router. The tile router within H-NoC duplicates the data packet (spike) it has received and sends it to the astrocyte-NoC which forwards this e-SP signal to the neurons. H-NoC spike events continue uninterrupted during this process and the astrocyte-NoC does not impact on traffic data load in the H-NoC. In the next section this will be talked about in greater detail.

5.4.1 The astrocyte process in hardware

The astrocyte process consists of two 2-AG generators and an astrocyte core. Fig.5.4. shows an overview of the astrocyte process which is made up of two 2-AG generators.

The spikes are received from the neurons and the 2-AG generators create DSE and 2-AG. The 2-AG is received in the astrocyte core which subsequently outputs the e-SP signal which controls PR within the synapses. The spikes from neurons 1 and 2, are de-packetized and re-packetized in an appropriate format so they can be directed into the 2-AG generator to produce DSE and 2-AG/e-SP.

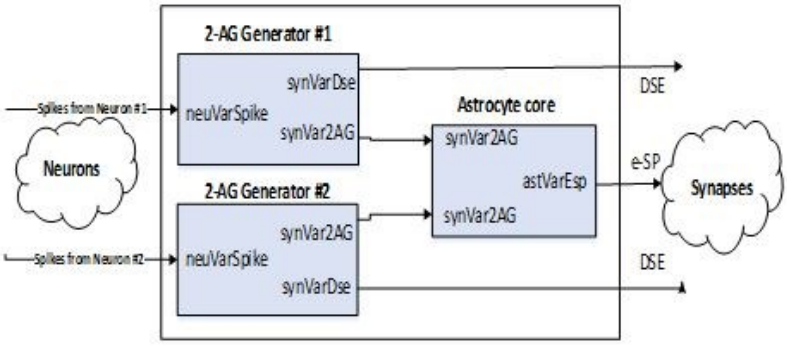

Fig.5.4 2-AG generator communicating to the Astrocyte producing e-SP. In hardware the astrocyte process consists of two separate 2-AG generators, these are connected to the neurons and take spikes as inputs. They produce the signals 2-AG (glutamate) and DSE. The 2-AG from all (in this case two) 2-AG generators is directed into an astrocyte core which produces e-SP. This is sent back to the neurons/synapses.

The e-SP signal is also broadcast to each synapse and therefore it may be viewed as a global signal within a node facility. This means that it may be the most challenging signal to scale due to its one-to-all connectivity. To address the scalability issue of the e-SP signal the ring topology is investigated as this permits broadcasting, but with an increased latency dependent on the number of packet hops. Given the slow dynamics of the e-SP signal (changes infrequently over seconds), one can trade-off higher latency for fewer routing connections (serial pathway). The ring topology enables this trade-off. A ring topology is used to address two key low-level issues within the neuro-glia implementation; (1) reduce the number of physical wires per node facility and (2), exploit the slower communication speeds of the biological e-SP signal (i.e. spike events are typically 2-3 orders of magnitude faster in exchange rates). The ring topology has previously shown benefits in area-speed trade-offs for SNN hardware [125]. Therefore, the e-SP communication is based on a ring or daisy chain topology for scalability and exploits the

one-to-many global communication between the neurons to traverse in a ring fashion around all 10 neurons in a node tile of the H-NoC, see Fig.5.3.

The synapses of the node are interfaced with the astrocyte core using an '*e-SP comms*' module. This module consists of an 'e-SP Tx' transmitter block and several 'e-SP Rx' receiver blocks (one connected to each neuron) which communicate via a serial link. The e-SP packet is serially propagated through each e-SP Rx module via the ring topology enabling all 10 neurons to receive and store the e-SP data value. The astrocyte-NoC is indirectly connected to ten neurons via a node router. It displays the connectivity between H-NoC neurons and the astrocyte core. The neurons (shown as #1 to #10) are connected via *ns* synapses, (where *ns* is the max number of synapses per neuron). The astrocyte NoC is implemented at the same level as the Tile Router in H-NoC. Here the spikes are directed to both the Tile router in H-NoC and formatted to be directed into the astrocyte core. The astrocyte core has one e-SP Tx module which is serially connected to the 10 e-SP Rx modules connected to each neuron. This is one data line, from the e-SP Tx to the first e-SP Rx module. The e-SP will be traversed back to the synapses in this manner. This shows the two separate networks (H-NoC and the astrocyte-NoC network) interacting at the lowest level and can operate independently. This means the spikes and astrocyte communications can operate concurrently. The node router is responsible for receiving the spikes and forwarding packets based on the address within the header. The node router packetizes the information and the routing engine directs them to the tile facility or to the neighbouring neurons within its Node facility.

In H-NoC, the e-SP packet is 64-bits in length which is significant in size and not practical for every neuron within each node router to have a dedicated 64-bit bus. For example, ten 64-bit buses would require around 640 wires to communicate e-SP to each neuron. This is highly inefficient. Therefore, a serial communication structure allows the use of one wire for communicating e-SP is adopted here. Reducing the path to a single serial link means the e-SP packet is serialized using a Parallel in Serial out (PISO). There is an 'e-SP Tx' block and this receives a 64-bit value from the astrocyte core and produces a 1-bit bus consisting of a 66-bit serial transmission. This information is forwarded through

a single wire to the ‘e-SP Rx’ block. The e-SP signal is communicated in a serial fashion on a single line thereby, reducing the number of buses and wires required.

5.4.2 Packetisation of the e-SP signal

The payload within the ‘e-SP comms’ packet size is vast (64-bit). This causes an increased number of wires or a larger bus used to communicate information as the size of the payload correlates with the packet size. The ring topology minimizes the overhead induced by the reducing the size of the bus. The e-SP packet is decoded and sent serially, using one wire or a 1-bit bus to communicate from the astrocyte back to the neurons/synapses. The serial message will be sent to all neurons and associated synapses. Although the trade-off is speed of communication with a higher latency, there is no traffic congestion caused by the ring approach due to the information eventually making its way to associated synapses at each of the neurons.

The e-SP is a 64-bit value the astrocyte core sends to the neurons. While this is significant it still maintains precision and accuracy when compared to computational models of repair. H-NoC uses a 48-bit packet which can be seen in Fig.5.5. with a 1-bit header, target address and source address. The source address includes both the node address and the “Neuron Position”. The neuron position indicates which neurons within the node facility have fired. It is from this packet that the information is directed to the astrocyte core.

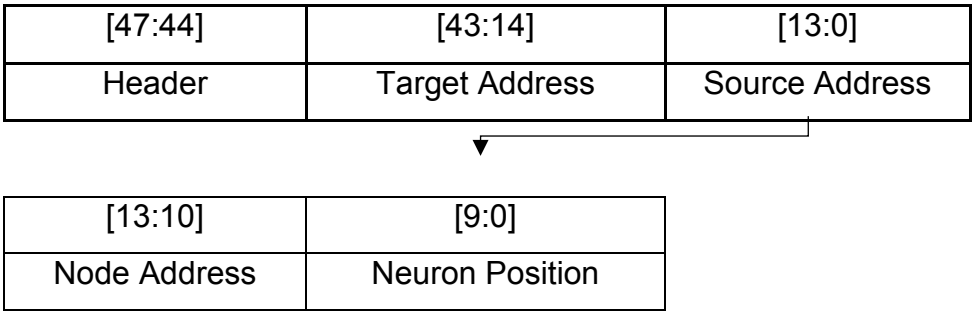

Fig.5.5 H-NoC packet description. This is a typical packet breakdown in H-NoC, the packet is 48-bits in size and has a header and target address which preludes the payload

or source address. This payload consists of a node address and the neuron/s that have fired within a node, the time of spike is included in the packet.

The e-SP signal is created by the astrocyte core from the signals received from the 2-AG generators. Each 2-AG generator receives spike events and produces signals, DSE and 2-AG. The 64-bit value which represents the e-SP is then packetised in the e-SP Tx module. To maintain simplicity, there is a start bit, the payload (which is the 64-bit value from the astrocyte core) and an end bit. The 'e-SP Tx' within the astrocyte NoC performs parallel to serial packetisation and the 'e-SP Rx' performs serial to parallel de-packetization and storage. The e-SP packet is shown in Fig.5.6.

[65]	[64:1]	[0]
Start Bit	e-SP Payload (64 bit)	End Bit

Fig.5.6 The e-SP packet in closer detail. This is a typical packet in the astrocyte-NoC. It is significantly larger than the H-NoC packet due to the size of the e-SP payload. It has 1 start bit and 1 end bit.

This packet format includes a start bit which indicates the start of the e-SP value communicated from the e-SP Tx module to the e-SP Rx modules. The e-SP value is held in the payload and the start bit indicates there is an e-SP signal to be communicated. The e-SP Tx module, a parallel to serial converter or PISO, sends the value in a series of bits. The series of bits then traverses through the flip-flop and into a shift register which stores the e-SP value. When the e-SP value is sent from the e-SP Tx, it is as a serial 66-bit packet. This can be seen in Fig.5.7. where the e-SP Tx module has a one- bit bus or a single wire output.

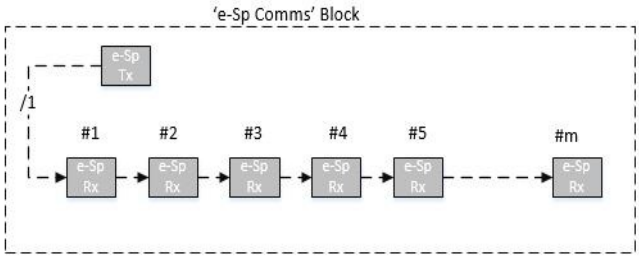

Fig.5.7 The e-SP Tx and Rx modules. This is the e-SP comms module as a whole. The block consists of one Tx module and multiple Rx modules. This creates the ring topology as the Tx transfers and the Rx modules receive information.

After the packet is sent serially, the e-SP Rx block uses a series of flip flops to control the input and output from the e-SP Tx block. The start bit indicates the beginning of the e-SP, which starts the process and the first bit starts the D flip-flop. This bit is set to '1' and so when both the start-bit, and the clock is a '1' the JK flip-flop receives a '1'. The next 64 bits is the e-SP payload which is sent to the 64-bit shift register where it can eventually be outputted to the neurons. The packet contains an end bit, to indicate the end of the signal and this bit is always set to '1' to ensure that when both the end bit and the clock is a '1' the JK flip-flop receives a '1' and the output toggles. The JK flip-flop has two inputs, the D flip-flop and the shift register output, where the output of the shift register feeds back to form one of the JK flip-flop inputs. When both the JK flip-flop and the clock are '0', the NAND gate is a '1' which indicates to the JK flip-flop that there are no more bits and the JK flip-flop stops any activity in the shift register. Therefore, this ring topology is only active when bringing communicating the current e-SP value.

The e-SP Rx module communicates the e-SP signal to every other e-SP Rx forming a ring topology. The 'e-SP Rx' requires a start bit to begin the transmission of the bits through the buffer and an end bit to indicate the end of data to be communicated and subsequently the 'e-SP Tx' will stop sending data. The area overhead of the serialised block is 65 LUTs and 132 Slice Registers and de-serialised block is 34 LUTs and 67 Slice Registers. Fig.5.8. shows the 'e-SP Rx comms' block in closer detail.

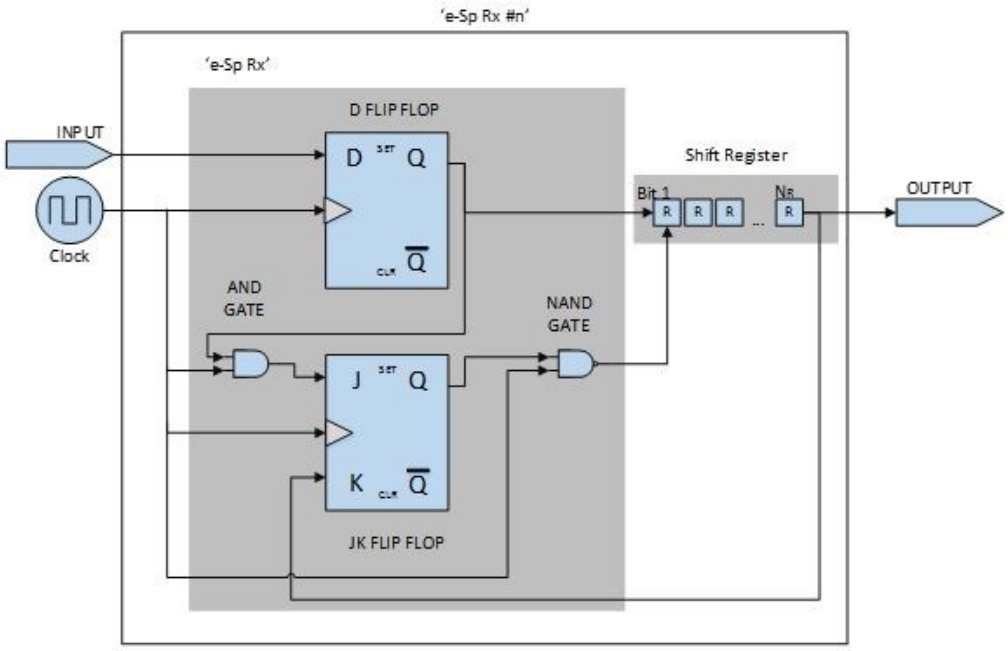

Fig.5.8 The e-SP Rx module. This is the Rx module, it consists of two flip flops, a D and JK respectively, an AND gate, a NAND gate and a shift register. This receives the 64-bit signal from the Tx module and passes the data serially to every neuron within the ring.

5.4.3 The e-SP signal

Fundamentally, the e-SP packet size depends on the bit-resolution of the astrocyte model. The e-SP signal value is currently 64-bits to maintain precision. The e-SP comms module is made up of one 'e-SP Tx' logic block used to interface the astrocyte core with the neurons within H-NoC. The e-SP data is converted into a series of bits using a parallel to serial conversion. This series of bits is then forwarded to the 'e-SP Rx' block, with a D flip-flop simply acting as a buffer and its output is then fed into a JK flip flop and the shift register. The serial data is passed through to a shift register (i.e. First in First out (FIFO)) consisting of a series of registers (each neuron from #1 to #m has a FIFO). The size of shift register depends on the size of the packet. N_R. N_R = 64 with a 64-bit packet. The JK flip-flop within the 'e-SP Rx' is utilised to automatically stop the shift register from receiving data bits when the FIFO has received all of its 64-bit data. The size of the packet has a direct impact on the size of the shift register. Due to a direct connection from the astrocyte to all the neurons in a single node facility there is no need for an address as the data

across all synapses is identical. There is a start bit and end bit, required to switch the transmission on or off.

5.5 Results

This section outlines the test bench and provides performance analysis of the ring-topology for the ‘e-SP comms’ block. The area overhead of the e-SP communication is compared with the H-NoC and an astrocyte core to demonstrate its compactness. It is also assessed for area utilization across various packet sizes.

A. Testbench and setup

The H-NoC neuron facility, astrocyte core and astrocyte NoC ring have been captured in VHDL and synthesised for a Xilinx Virtex-7 XC7VX485T-2FFG1761C FPGA evaluation board using Xilinx Vivado 2016.2. The NoC ring topology with astrocyte to neuron communication has been validated on the FPGA Xilinx ARTY 35T evaluation board.

The astrocyte accepts packets in the form of 8 bits (a spike would be represented as binary “00000001”)this is because the astrocyte is developed using C++ and converted to VDHL using Vivado synthesis software which uses buses for all signals (part of reducing the astrocyte), and produces a number of signals (2-AG, DSE, Ca^{2+} and IP_3) in this chapter the important signal is the global e-SP signal. As previously mentioned, the ‘e-SP comms’ block is interfaced with the neurons using a single wire in a ring topology this is shown in Fig.5.9 as the signal main_output_final_top. Fig.5.9. shows a high-level RTL schematic in Vivado indicating the Tx and Rx modules. The Tx receives the e-SP signal, performs parallel to serial conversion and sends it to the first Rx module.

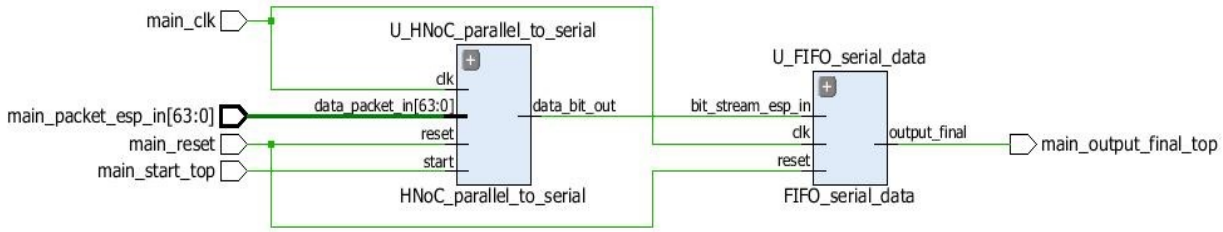

Fig.5.9 "e-SP comms" module. This is the e-SP Tx and Rx modules high-level RTL schematic in Vivado. The H-NoC parallel to serial is the Tx module and the FIFO serial data is the Rx module. The Tx module receives a 64-bit bus (main_packet_esp_in) and has a single wire output (data_bit_out). This is sent to the Rx which has a 1-bit input (bit_stream_esp_in) and one output (output_final) which traverses the ring.

B. Simulated operation of low-level e-SP comms module

Fig.5.10 is a simulation of the low-level e-SP comms module where the test bench was designed for an astrocyte to accept spikes from neuron 1 and 2. The astrocyte would then send the 64-bit e-SP value. The e-SP data packet is sent to the e-SP Tx module and outputted serially. This would then be received by the buffer store which accepts 66-bits including the start and end bit: this process extracts the e-SP value.

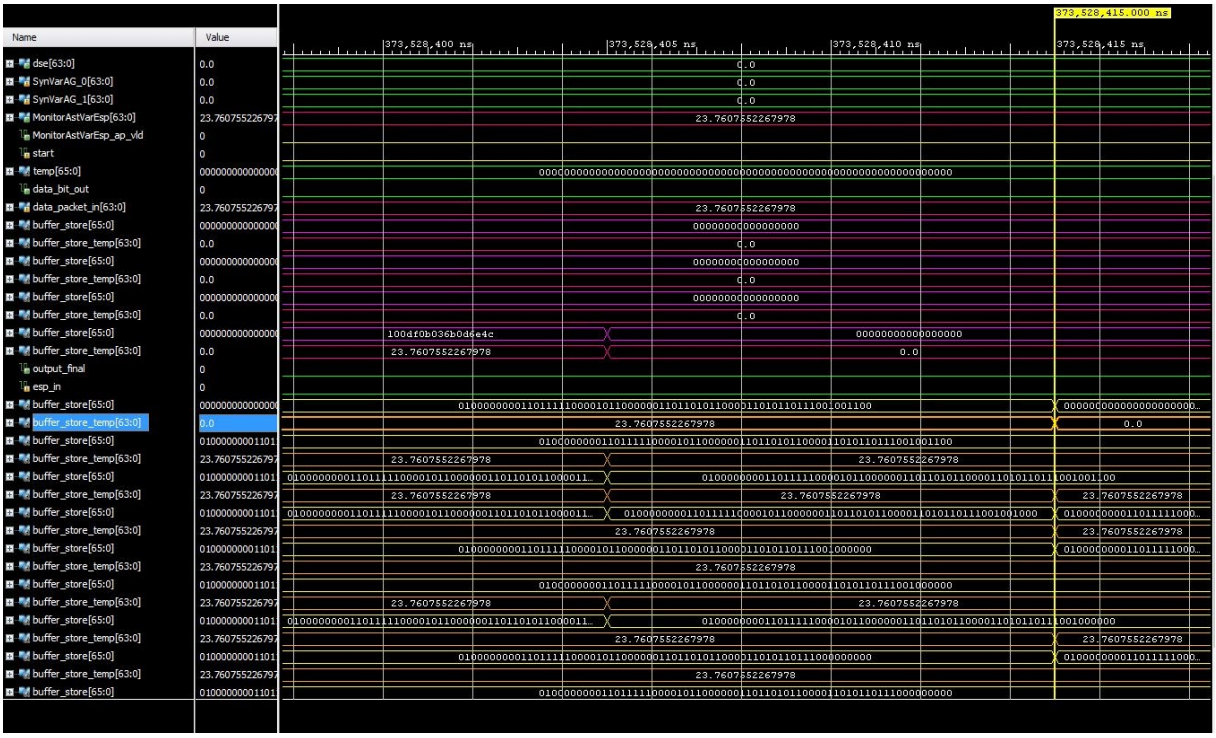

Fig.5.10 “e-SP comms” module showing the simulation. This is a simulation using Xilinx Vivado software. At the top of the figure, in green are the astrocyte signals, and in red the e-SP signal in 64-bits. The temp [65:0] is the PISO receiving a 64-bit packet (data_packet_in) and outputting a 66-bit string of bits, this is the data_bit_out, this is the Tx module. The buffer_store and the buffer_store_temp is the Rx module, which take the 66-bit packet and formats it to get the e-SP signal value. The yellow and gold lines are used to show how the value traverses the ring without a distortion of data, each buffer_store coupled with a buffer_store_temp indicates a new module in the ring. The simulation occurs over a long period of time to ensure there are no dropped bits or distortion of the signal, this is after 373,528,415,000 ns or 373.5 seconds.

C. Analysis of the e-SP comms module

The area overhead incurred by the e-SP comms block is defined in Table 5.2, compares area overhead associated with the astrocyte and H-NoC neuron facility and the area utilization is shown in terms of percentages due to the size of the astrocyte core.

Therefore, comparing against the astrocyte in terms of area overhead, the H-NoC neuron facility is around 3.34% in terms of LUTs and 4.51% in terms of slice registers. The e-SP is much lower at 0.61% in terms of LUTs and 1.2% in slice registers which supports spike communications**.** This correlates with the size of the e-SP packet data i.e. the size of the packet generated by the astrocyte, which at this point in time is a fixed value. In terms of scalability the number of look up tables (LUTs) and slice registers were used to determine how the 'e-SP comms' block scaled in comparison to both the area consumed by the H-NoC Neuron facility and the astrocyte core. This is to show that relative to the astrocyte the area incurred due to the inclusion of the e-SP comms is minimal.

Table 5.2 Area analysis "e-SP comms"

Component	LUTs	(%)	Slice Registers	(%)	Device Utilisation (%)
Astrocyte	16,182	-	16,305	-	4.33
H-NoC Neuron- Facility	540	3.34	735	4.51	0.28
e-SP comms	98	0.61	198	1.21	0.06

In terms of both LUTs and slice registers (area utilization) the 'e-SP comms' block is very small compared to the H-NoC neuron facility with which it communicates. Fig.5.11 compares the ring and star topologies using area overhead and latency metrics. The star has a greater overhead and lower latency and the ring has a higher latency but a lower area overhead.

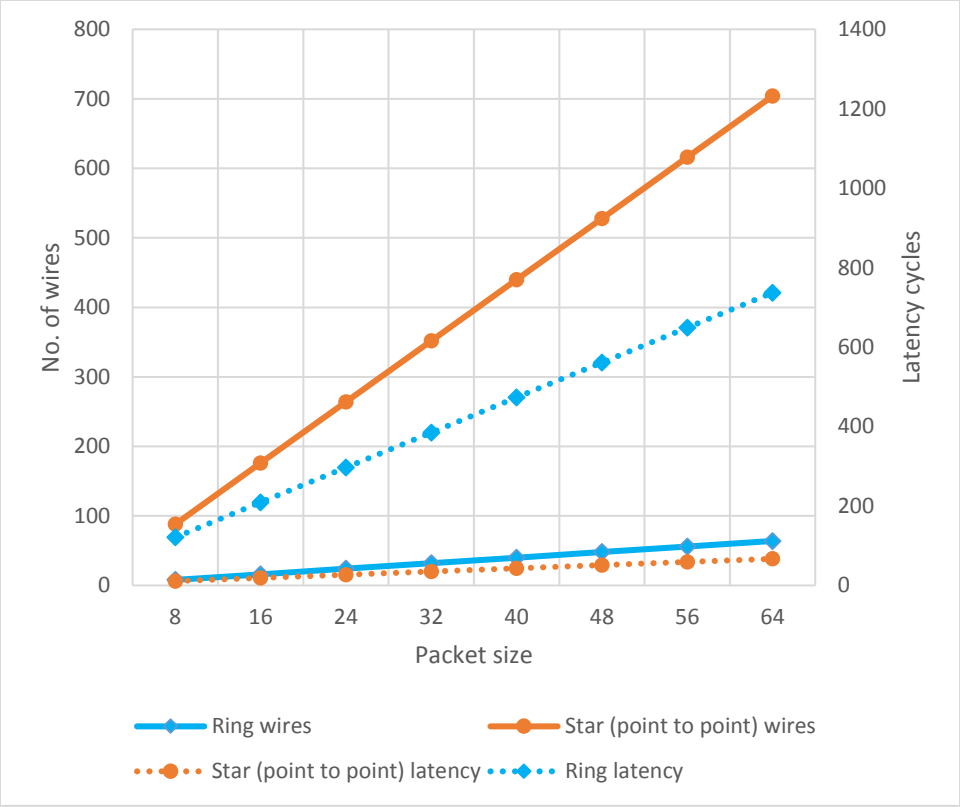

Fig.5.11 e-SP comms module scalability. This compares ring and star topologies using area overhead and latency metrics.

Fig.5.12. shows the LUT resource usage of the e-SP comms as the network scales, with reference to the H-NoC neuron facility. The e-SP comms block scales more linearly, this is because the e-SP comms has fewer resources and is a more simplistic design than the H-NoC element which demonstrates that the e-SP comms is not a limiting factor in scaling neuro-glia networks. Fig.5.12. shows the number of LUTs when using a 64-bit resolution. This is the absolute maximum bit resolution and could be considered the limiting factor.

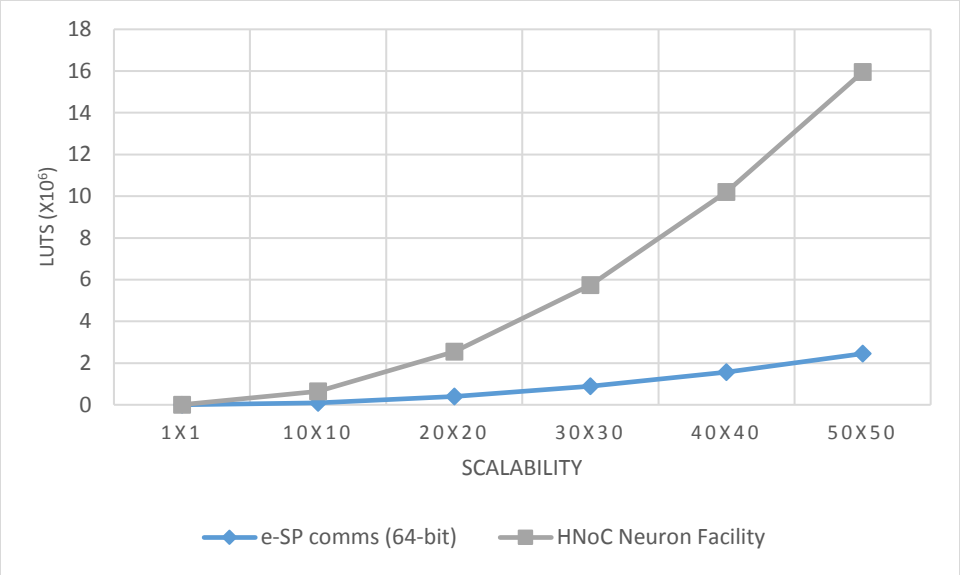

Fig.5.12 e-SP comms module vs H-NoC neuron facility (scalability). This graph shows area overhead using LUTs. The e-SP comms module is compared against the H-NoC neuron facility as they scale up over various array sizes.

Fig.5.13. shows the number of LUTs (area utilization) in the e-SP comms block, across different packet sizes (e.g. between 8 to 64 bit). The graph shows that the area overhead incurred when the e-SP comms and H-NoC neuron facility, as this represents, one e-SP module to one H-NoC neuron facility, are scaled up from an array size of 10x10 to 50x50. This shows the LUT (area) has a gradual increase and can be reduced by minimizing the e-SP packet size; i.e. optimize the astrocyte core. This indicates that the e-SP comms block incurs a conservative area overhead (0.61% compared to the astrocyte). Due to the worst-case scenario of 64 bits being used (as the astrocyte core uses 64-bit precision), it is important to realize that a reduced packet size would result in the area overhead decreasing.

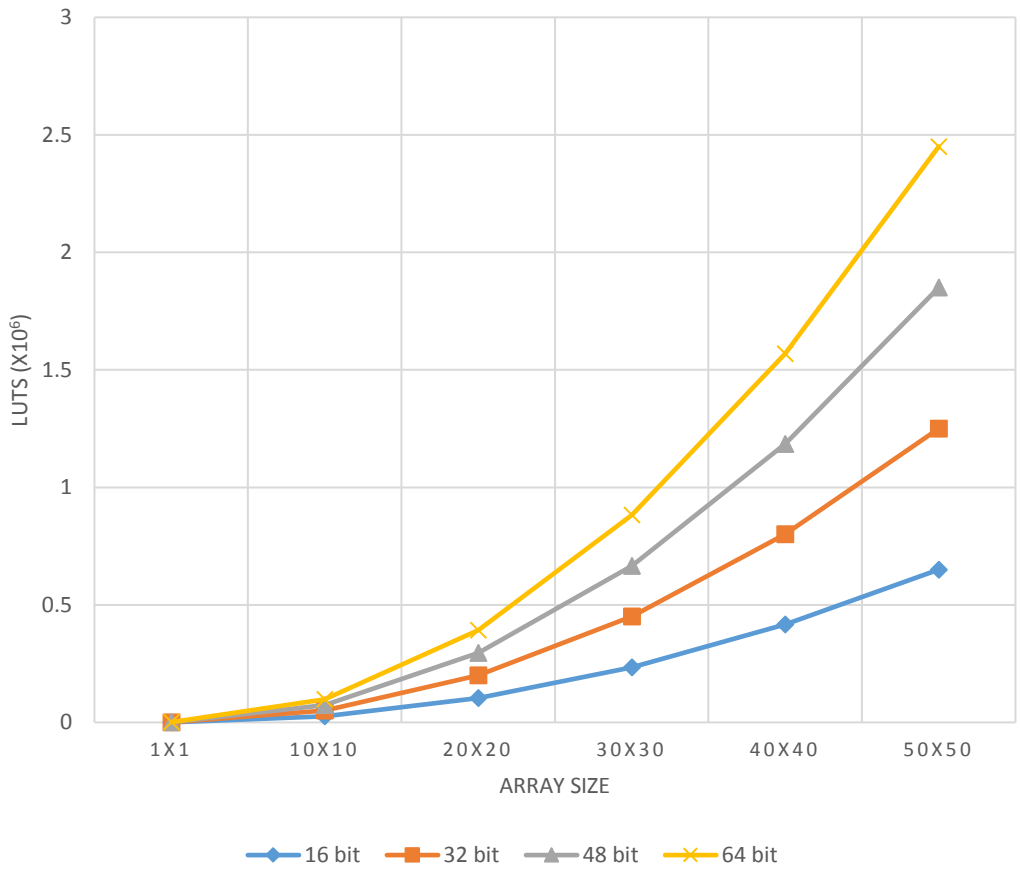

Fig.5.13 Area utilisation scaling the e-SP comms module. This graph shows the area overhead incurred by comparing the array size and then comparing packet size. The packet size is important as it can be noted here, that as the packet size increases in number of bits the area overhead required to support this packet size increases exponentially which has a huge impact on scaling the network.

The results show that the area overhead incurred by adding the 'e-SP comms' is not significant compared to H-NoC and therefore the proposed approach enables large neuro-glia networks to be realised. From the graphs above, scaling and packet size are huge influencers of area overhead when supporting a large-scale network. As the network scales it is more area efficient than the H-NoC neuron facility, however, reducing the 64-bit packet is the key to reducing the area footprint. As throughput is not the key factor within the neuro-glia network, a ring topology is used to reduce the area overhead, as it has lower area overhead trading off increased latency. As the network scales, the array

size dictates the area incurred and compared to a H-NoC neuron tile, where the ratio of e-SP comms to neuron facilities is 1:1, the e-SP comms scales better and supports a large-scale network. However, it can be observed that 64-bits is redundant and that reducing the size of the packet would have a huge impact in terms of area and throughput. Overall, a higher latency allows a ring topology to take advantage of the slow changing astrocyte signal and reduce the area overhead whilst maintaining 64-bit precision.

5.6 Summary

In this chapter a novel NoC ring topology to support e-SP signals between astrocytes and neurons has been proposed. This ring topology takes advantage of a slow changing astrocyte process used in self-repairing networks, and sacrifices throughput as it uses an area efficient design. Implementing the ring topology for e-SP communication within a neuro-glia network allows a low area design, which could be further reduced by reducing the packet size from the astrocyte core. Due to the vast size of both individual networks, the number of PEs (neurons and astrocytes) and number of communication signals increase exponentially. The interfacing of the two completely different networks is a complex challenge. Previous work on high level astrocyte to astrocyte communications [126] and this low level astrocyte to neuron/synapse communication indicates a ring topology NoC provides a scalable solution to the interconnect challenge. The use of the ring topology in the NoC provides a good trade-off between reducing area/wire overheads and relaxing the communication speed of data provided by the astrocyte to synapses/neurons.

This novel NoC interconnection scheme enables communication from the astrocyte to neuron/s within a single neuron facility in H-NoC. This allows the e-SP signal to be communicated and enables a significant number of astrocytes to communicate with neurons within a minimal area constraint. This process will enable self-repair emulation (as shown in chapter 6) with a distributed and fine-grained nature without a central controller. This is based on the biological and computational models of previous works. The results show that trading off throughput for area efficiency is ideal because of the slow changing astrocyte. Compared to the astrocyte for relative size, the e-SP ring incurs

a small overhead and scales efficiently compared to the H-NoC neuron facilities. The size of the packet is essential but implicates the area. As the network scales the resources to support the 64-bit signal from the astrocyte scales exponentially, however, reducing this packet size would significantly reduce the area further. This work has been explored further in chapter 6, implementing the e-SP ring in a SANN (with self-repair capabilities in a SNN with an astrocyte cell in hardware). This low-level interconnect in addition to the high-level astrocyte communications provide a platform for developing a neuro-glia interconnect for future inspired computing paradigms regarding self-repair strategies in hardware.

The e-SP signal is considered an astrocyte-neuron signal, as it and DSE are communication signals between the astrocyte and neurons. In the next chapter, inter-astrocyte signals (from astrocyte-to-astrocyte) signals are explored. This allows both high- and low-level communication signals to be realised in a neuro-glia network using NoC technology. This chapter focuses on the self-repair mechanics or fine-grained aspect of repair whereas, the next chapter focuses on the distributed aspect of repair, and how it affects neuro-glia networks, and how to efficiently communicate IP_3 efficiently using an astrocyte core.

Chapter 6: On-chip communication for neuro-glia networks: Global NoC

6.1 Introduction

Neuro-glia networks have a vast number of connections between glial and neural cells. As discussed in Chapter 3 and 4, NoCs can be implemented in order to address scalability issues when there are significant numbers of PEs requiring connectivity. In Chapter 5 a NoC ring-topology was used to interconnect astrocytes with neurons and this interconnect exploited the local slow changing astrocyte values communicated to neurons and reducing area implementations in hardware.

A key communication requirement in the repair process is the global exchange of data between astrocytes themselves, e.g. inter-astrocyte (astrocyte-to-astrocyte). Neuro-glia networks become more complex due to additional connectivity between multiple astrocytes which adds to the existing interconnect challenge.

IP_3 is one of two signals (the other being Ca^{2+}) communicated directly from astrocyte to astrocyte, see chapter 5, Fig.5.2. This is part of the global repair process; as IP_3 oscillates in the cell, it triggers Ca^{2+} releases between astrocyte cells. To address this global interconnect challenge, a novel NoC-based astrocyte tile router is applied to a cluster of astrocytes. This implementation provides a reduced area/power communication infrastructure in hardware where the cluster comprises of eight astrocyte cells.

This chapter explores using this Global NoC approach and consists of three novel aspects:

1. Astrocyte tile router and ring topology; the inherent slow changing astrocyte signal is exploited by serialising the data with a ring topology in order to reduce area/power overheads for scalable implementations.

2. An IP_3 accumulator; this is a low-level logic system consisting of logic gates, multiplexors and counters. All IP_3 signals in a given cluster are accumulated and a single adder is used to add 64-bit packets (from the astrocyte cluster), average

and communicate this information back to the astrocytes cells. This is the first time this type of low-level system has been applied to astrocyte communications.

3. An update manager implements a novel token-based control dynamic centred around the frequency of astrocyte communications. Using this novel token system, there are two variables which control the update rate (no. of tokens and time) within the tile router. This ensures the astrocytes remain in sync, the clusters update frequency can be changed in the future as required to suit variable communication rates. The dynamic scheduler manages token requests and will enable the start of the IP_3 process, after a certain number of requests or a time period has passed.

This contribution addresses the key challenge of providing a scalable communication interconnect for global requirements by contributing to a multi-level NoC ring topology. It extends the work in Chapter 5 as it provides a solution to a complete neuro-glia network and explores the crucial intra-astrocyte communication signals. Note: this contribution has been published by the author [127].

The Chapter is organised as follows: Section 6.2. discusses the scale of the neuro-glia network challenge and section 6.3 outlines the novel astrocyte tile router. Section 6.4. presents results and analysis while section 6.5. provides a conclusion.

This chapter provides the following:

1. The ratio of neurons to glial cells and why this is important;
2. The challenges of creating a scalable interconnect to support the vast number of connections regarding global communications in an astrocyte network;
3. How an astrocyte router is used at a global level and how it can be used to exploit the inherently slow astrocyte communication;
4. The advantages of using a dedicated astrocyte NoC router for interconnect for global communications in neuro glia network, including an update manager and dynamic scheduler;

5. The overheads of using NoC technology to provide a scalable interconnect for neuro-glia networks.

6.2 Neuro-glia networks: Biology

Astrocyte communication is both local and global and is analogous to a small-world network graph. A small-world network is a communication phenomenon found in networking where groups or clusters of nodes exchange information and can be found in real world environments such as social networks, navigation [128] and biology [129]. A recent example such as cryptocurrency, and peer-to-peer networks, has been shown to demonstrate small-world principles [128]. As the network scales the communication infrastructure scales efficiently. The features of a small-world network emphasise many local communication clusters and fewer global connections between clusters. A neuro-glia network can be considered a small-world network denoted by signal exchanges on a large and intricate scale between astrocyte and neuron networks. This section provides an overview on astrocyte to neuron ratios in the brain, how these cells communicate and self-repair within biological neuro-glia networks.

6.2.1 Neurons and glia cells

The ratios between neurons and glia cells in the brain largely differ. According to recent literature there is a 1:1 (glia to neuron ratio) [13]. This is the suggested overall ratio in the brain, however, this ratio of glia to neurons varies e.g. the cerebral cortex is around 1:3 and the cerebellum 1:4 glia to neurons. However, as 20% of glial cells are astrocytes within each region, the 1:1 works out at around 1:5 (of astrocyte to neurons) in the brain, 1:15 in the cerebral cortex and 1:20 in the cerebellum [13]. Table 6.1 shows approximate ratios of glial cells to neurons cells and astrocyte cells to neuron cells in different regions of the brain. Table 6.2 shows a comprehensive review on how the ratio has changed as our technology and understanding of the brain has evolved. However, this is an estimate and there is still a significant debate regarding the issue in the literature [13] with a large variance in the reported ratios, as shown in Fig.6.1. Undoubtedly, the glia to neuron ratio is diverse and difficult to define. However, in this chapter the ratio used is 1:10 astrocyte

to neurons, as was the case in Chapter 5. The work within this chapter uses eight astrocytes where each astrocyte is interfaced with 10 neurons and each astrocyte tile router accommodates 8 astrocytes which allows 80 neurons per astrocyte tile facility: also it is easier and more efficient to average this number and divide using a shift register. This will be discussed in more detail later in the chapter.

Table 6.1 Neurons to glia cells (based on region)

Brain region	**Glial cells to neurons (Ratio)**	**Astrocytes to neurons (approx. Ratio)**
Human brain	1:1	1:5
Cerebral cortex	1:3	1:15
Cerebellum	1:4	1:20

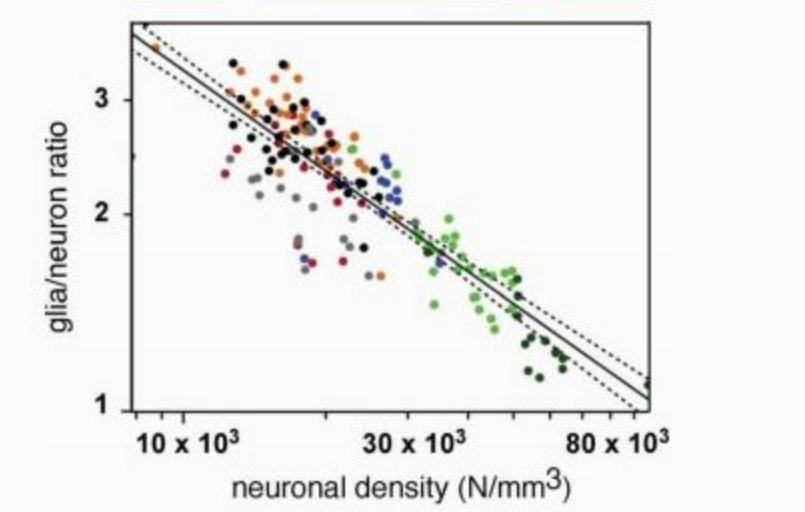

Fig.6.1 Glia/neuron ratio [13].

Table 6.2 Neuron to glia ratios [13]**.**

Reports of glia-neuron ratios (GNRs), numbers of total cells, glia, and neurons in human brain

GNR	Total cell number	Glia #	Neuron #	Method	Reference
			3 bn		Donaldson, 1895
10:1 ("perhaps")					Glees, 1958
10:1 ("perhaps")					Pope, 1958
10:1 ("around")					Hyden, 1960
10:1 ("perhaps")					Galambos, 1961
10:1 ("about")					Hyden, 1961
10:1	110 bn	100 bn	10 bn		Asimov, 1963
10:1					Maron, 1963
(glia "more abundant" than neurons)					Kuffler & Nicholls, 1966
		100–130 bn		Histology	Blinkov & Glezer, 1968
"glia …outnumber neurones by several fold"					Dobbing & Sands, 1970
5:1 – 10:1					Noback & Demarest, 1975
10:1 ("at least") >10 bn					Kuffler & Nicholls, 1976
~10:1					Ganong, 1977
"Glia … far outnumber(s) neurons"			20–200 bn		Wittrock, 1977
			50 bn		Edelman & Mountcastle 1978
~10:1					Ganong, 1979
			10bn – 1 trn		Hubel, 1979
			10 bn – 100 bn or 1 trn		Nauta & Feirtag, 1979
			100 bn		Stevens, 1979
5:1					Jensen, 1980
5:1 – 10:1					Snell, 1980
9:1	[10 trn]	[~9 trn]	~1 trn		Kandel & Schwartz, 1981
10:1 ("perhaps")		[~1 trn]	100 bn		Nolte, 1981
			30 bn ("roughly")		Szentagothai, 1983
10:1					Damask & Swenberg, 1984
10:1 – 50:1	[11–51 trn]	[10–50 trn]	1 trn ("best estimate")		Kandel & Schwartz, 1985
10:1					Nicholls et al., 1985 2nd ed

6.2.2 How the brain facilitates self-repair

As previously stated in Chapters 2 and 5, local communication pathways connect the astrocyte directly to neurons. In addition, a global information exchange occurs between astrocyte. This neuro-glia structure can be viewed as two separate networks, where pathways are in place to support the information exchanges of both neurons and astrocytes. Neurons exchange information with astrocytes and other neurons via spike events. Astrocytes communicate using separate signalling pathways (global pathway) and glio-transmitters (non-spike events). Spike events between pre- and postsynaptic neurons stimulate signalling exchanges between astrocytes. A neuron, in a very basic and simplified manner, is made up of dendrites (inputs) and an axon (output). When a

spike occurs, there is an intracellular chemical reaction which takes place and astrocytes can modulate or mediate these reactions [11]. This results in the increase or decrease of the PR in associated synapses. This self-repair behaviour is the signalling process which facilitates repair decisions at a low or local level. This however, increases the complexity and signalling processes within the network. Astrocytes are connected via gap junctions or intracellular routes which allow IP_3 exchange. The astrocyte network can be viewed as a high-level network, working in parallel with the neural network, and is responsible for regulating synaptic plasticity in the neural network.

6.2.3 NoC for neuro-glia network

The H-NoC hierarchical approach assists in identifying a communication network for astrocytes which communicate on a local and global scale. Using routers to distinguish between these local and global communications, they can be separated and viewed in a hierarchical manner. This not only supports parallelism, allowing data to be passed from neuron to neuron and astrocyte to astrocyte, but also the interactions between neurons and astrocytes. Neurons communicate through spike events whereas astrocytes communicate with each other through the exchange of IP_3. These are very different communication patterns and speeds which must be adhered to when realizing a neuro glia network. Inspiration from using different topologies, such as ring, establishes the motivation for exploring the trade-off between reduced communication bandwidth and area overhead from larger buses/wiring.

6.3 Astrocyte tile router overview

The H-NoC hierarchical approach assists in identifying a communication network for astrocytes which communicate with other astrocytes via a separate communication protocol to that of astrocyte-neuron communication. The global astrocyte-to-astrocyte channel communicates IP_3 data, whereby astrocytes function by balancing and sharing their levels of IP_3 to ensure that there is enough IP_3 to facilitate repair and maintain normal functionality. Within an astrocyte, IP_3 can be considered as a pool of water connected to a reservoir, this reservoir extends to neighbouring astrocytes and maintains similar levels

of IP_3. When the IP_3 level drops in one astrocyte, the other reservoirs exchange IP_3 to provide an equal balance across all pools. The main communication paths are outlined in Fig.6.2.

This process balances the glio-transmitters across all associated and neighbouring astrocytes and provides the network with the ability to detect changes in IP_3 and Ca^{2+}. Therefore, astrocytes influence and affect each other, removing the need for a central controller. It is this distributed and complex communication which allows self-repair on a global scale to occur.

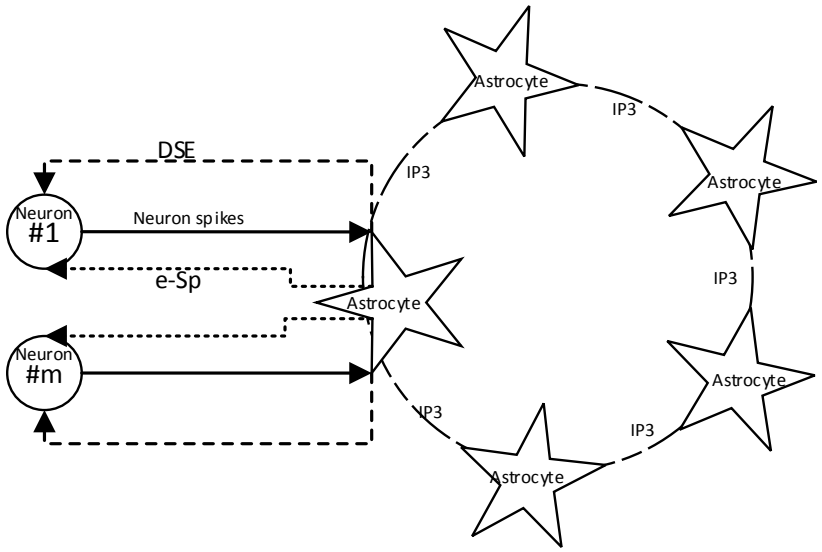

Fig.6.2 Communication signals within a neuro-glia network.

6.3.1 Global communication in a neuro-glia network

The astrocyte receives spike stimulus from the SNN, event data is communicated from H-NoC to the astrocyte via an additional output port within the node router of H-NoC. This is shown in the previous chapter and is used to send information to both the tile router of H-NoC and the astrocyte. The node router of H-NoC sends the data simultaneously to both the tile router within H-NoC and to the astrocyte core. The astrocyte model used, consists of two 2-AG generators and an astrocyte process; received spikes from neurons stimulate the release of 2-AG and the astrocyte produces the IP_3, DSE and e-SP signals,

see Fig.6.2. The IP_3 is a global communication signal within the astrocyte network, and is shared across neighbouring astrocytes where all astrocytes strive to balance out the levels of IP_3 across the cluster of astrocytes as a whole. As the IP_3 oscillates there is a flux of Ca^{2+} between astrocytes which ensures this balance is maintained. Therefore, at an abstract level, the spike events from H-NoC can trigger changes in an astrocyte's IP_3 level, and this value must be communicated to other astrocytes (global communication). Within Chapter 5 the local communication consists of communicating e-SP from the astrocyte, the e-SP Tx interfaces with the astrocyte core, and sends information down to each e-SP Rx through the ring topology. This chapter focuses on the global signal using the same astrocyte model and communicating the IP_3 data between all astrocytes within a cluster.

Each astrocyte is therefore connected to the astrocyte tile router through a channel which exchanges IP_3 data, this is a requirement for global communication. The astrocyte router has two main modules, an IP_3 accumulator and an update manager. The router as a whole will:

(1) Receive IP_3 level data from up to eight astrocytes.
(2) Calculate the average IP_3 level for all eight astrocytes (carried out by the IP_3 accumulator) and communicate this back to all astrocytes.

The astrocyte core represents IP_3 as a 64-bit packet [85]. The rate at which IP_3 changes is much slower than spike events; typically, 2-3 orders of magnitude slower. The key objective for hardware is to balance the physical area per astrocyte tile router facility while also meeting real-time requirements of the IP_3 exchange and update process. The astrocyte cluster facility is also an important component of the overall architecture of the astrocyte router. The ring topology in NoCs has previously shown benefits in area-speed trade-offs for both SNN and neuro-glia hardware [87], [125], [130].

This ring topology allows eight astrocytes within a cluster to communicate IP_3 within an astrocyte network. In biology, 4-8 astrocytes are generally connected [131], however, eight astrocytes allow a 3-bit shift register to divide and average the IP_3 from all astrocytes. As there are 4-8 astrocytes, in biology, eight astrocytes were chosen, one,

because this corresponds with numbers seen in biology and two, a 3-bit shift register is needed to divide by eight which is part of the design consideration. This router consists of a router and novel ring topology, a novel IP_3 accumulator, a novel update manager consisting of a token system and dynamic scheduler. This design exploits the slower communication speeds of astrocytes. Using the biological timescale, the astrocyte router exchanges IP_3 data using a time-multiplexed approach based on a ring topology. Each aspect will be described further in the next sections.

6.3.2 Astrocyte tile: Inter-router module and topology

The 64-bit precision constraint on the IP_3 data size is significant and becomes area inefficient if implemented using traditional channels of parallel-lines, i.e. a 64-bit channel. The astrocyte tile router is comprised of three main components, an adder, a ring interface and an update manager. Each astrocyte is attached to an inter router which manages the parallel to serial conversion using a Parallel-In Serial-Out (PISO) module and a ring transmission interface. When an astrocyte has an updated IP_3 value it releases an IP_3-vld signal to the inter-router, this is followed by its 64-bit IP_3 value. The 'inter router' accepts this data value and consequently requests a token from the update manager, this will be explained in the next section in more detail. Due to the potential for numerous IP_3 values (changing within a short timeframe) from a number of astrocytes, it is important to manage the token requests and the communication process, this will allow the astrocytes to remain in sync. The inter router will request a token from the update manager and the update manager accepts the request if the token is free and then grants the token to the requesting 'inter router'. This process starts the chain of events regarding the IP_3 pathway and is the main process regarding global communication in an astrocyte network. When an 'inter-router' token has been granted, the 'inter router' will serially transmit its IP_3 data to the IP_3 accumulator via a PISO contained within each 'inter router' unit. This serial link enables a reduction in the physical wiring because a traditional bus may consist of eight 64-bit buses/512 physical wires, the serial line reduces this to 8 single lines (one for each astrocyte) containing all the IP_3 data. The inter-router sends its value serially and simultaneously transmits a signal to inform the next 'inter router' to subsequently send its data to the IP_3 accumulator. Each 'inter-router' sends its data serially whilst sending a

start signal to the next ‘inter-router’. This data propagates through the IP_3 accumulator, adding each astrocytes IP_3 value and sending back an average value of IP_3 to the astrocytes. The resulting new IP_3 value is serially propagated back through each ‘inter router’ facility using the ring topology. The IP_3 data is sent on a 64-bit bus to the ‘inter router’, and then sent serially via a single data wire.

Overall, the astrocytes send data to the astrocyte tile router and the IP_3 accumulator manages the incoming data from all astrocytes via a multiplexer, this data is contained in a packet, for the packet layout see Fig.6.3. Each IP_3 value is in a 64-bit packet, astrocyte #0 contains the 64-bit packet and is appended with a single start bit. Each 64-bit packet is numbered from 0 to 63, as shown below in Fig.6.3.

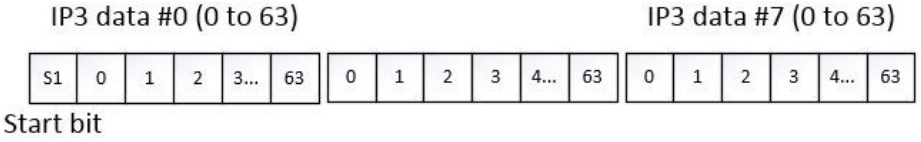

Fig.6.3 Packet layout

The astrocytes send data synchronously with a 1 clock cycle delay between each astrocyte, thus an 8-bit mux is used. This multiplexer switches between all eight astrocytes IP_3 data and is controlled by Counter #1. Counter #1 starts with the first input and will count to 8 before resetting to 0 and controls Mux # by switching from 000 (zero in binary) to 111 (7 in binary). Each input is directed, in a synchronised fashion to a full adder, adding all eight astrocytes IP_3 values. Initially Mux #2, a 2-input mux, defaults at 0. As the first bit comes in to the B port of the adder, A is 0, this guarantees the first bit is added correctly. This is also controlled by Counter #1. The adder accumulates all eight IP_3 values and forwards this to the ‘Divider’ where a shift-by-3 operation is performed to complete the IP_3 averaging process. A shift-by-3 is used to divide the IP_3 by eight, averaging and outputting the average IP_3 to all astrocytes. This is controlled by Counter #2 which will count the first 3 bits and discard them. Fig.6.4 shows the IP_3 accumulator in more detail as it manages and accumulates 8x64 bit serial values. The output of multiplexer is connected to a serial adder circuit.

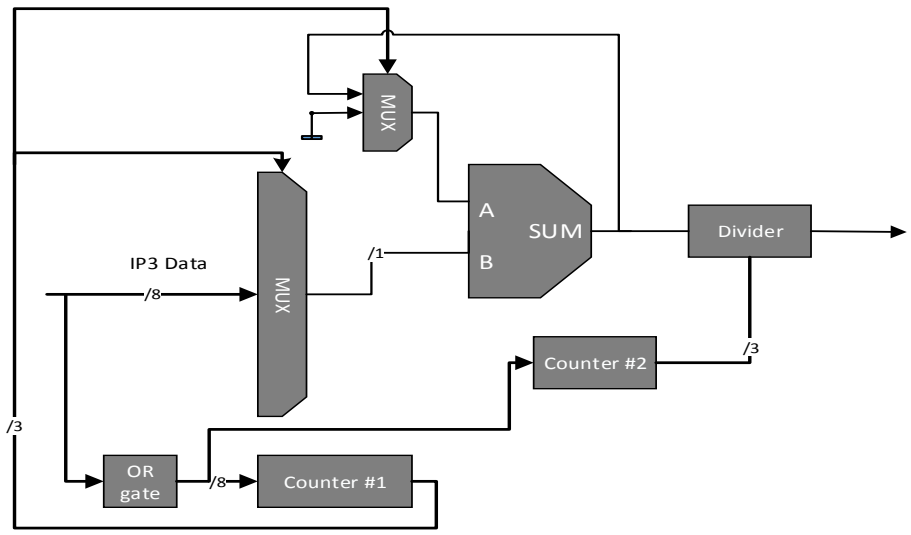

Fig.6.4 IP_3 accumulator

Using a ring topology minimizes wiring overhead which is dictated by the large packet size. However, using a serial data line allows the ring topology to maintain connectivity with all astrocytes. A single data line is used between each astrocyte and the 'inter-router' whereas the 'inter-router' uses four separate data lines to communicate back and forth with the astrocyte. The 'inter-router' also communicates to the update manager (requests tokens) and IP_3 accumulator (sends current IP_3 value). Each inter router has four wires to communicate back and forth with the astrocyte. It also has four additional wires, one to request a token, one to send IP_3 to the IP_3 accumulator and two to communicate IP_3 the new IP_3 value. Inter router #0 has an extra wire for receiving the token from the update manager to begin the update of IP_3 within the ring topology.

The 'Ring I/F' interface uses the ring of the inter-routers to communicate the average IP_3 value back to each astrocyte. Fig.6.5 highlights all the input and outputs in regard to each 'inter-router'. The 'inter-router' manages the values from the astrocytes and keeps the astrocyte tile router, this accumulates and averages IP_3 values from all astrocytes and communicates the resulting information back to the astrocytes within the network.

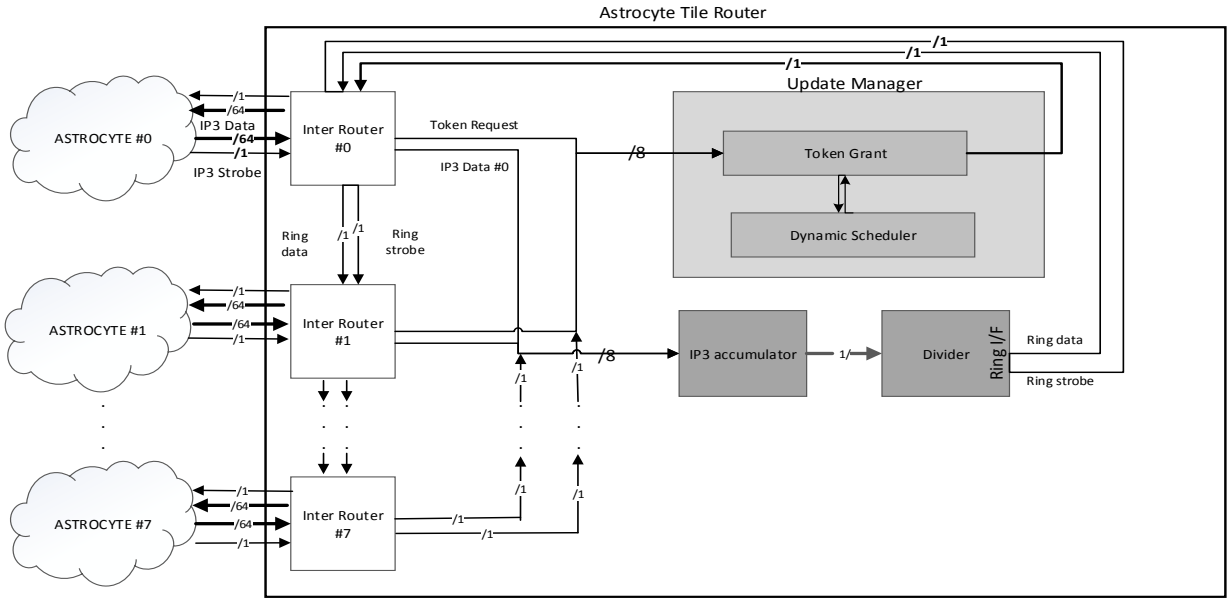

Fig.6.5 Astrocyte tile router communications. IP_3 from the astrocyte is sent to the inter router. The inter router requests a token. If the token is granted, each astrocyte sends its IP_3 value to the IP_3 accumulator. The updated IP_3 value is then sent back serially via the ring interface to each astrocyte.

6.3.3 Update manager and dynamic scheduler

The update manager is interconnected with eight astrocytes, there is additional flexibility to allow more astrocytes to communicate with less astrocyte tile routers if required; i.e. increase efficiency and reducing overhead. The importance of the update manager is twofold, managing the synchronisation of the input IP_3 data from the astrocytes and 'inter-routers' into the accumulator. Secondly, it enables the control of the rate at which the update or averaging process is done as IP_3 values change very slowly.

To perform a complete update with every astrocyte state would be inefficient. Therefore, rather than perform an average IP_3 calculation each time a single astrocyte requests the action, the dynamic scheduler provides a means by which to minimise unnecessary averaging calculations. This reduces power consumption as the token is released when either:

(1) A number of token requests(n_{toc}) from the ‘inter-routers’ has been requested
or
(2) A max time period (T_{DS}) has elapsed.

The dynamic scheduler (DS) is connected to an update manager, this manages the requests from ‘inter-routers’ connected to astrocytes which upon a change in IP_3 in an astrocyte, it requests a token. Only after one of these thresholds is met can a computation or update be performed. Fig.6.6 depicts a flow chart detailing the DS process.

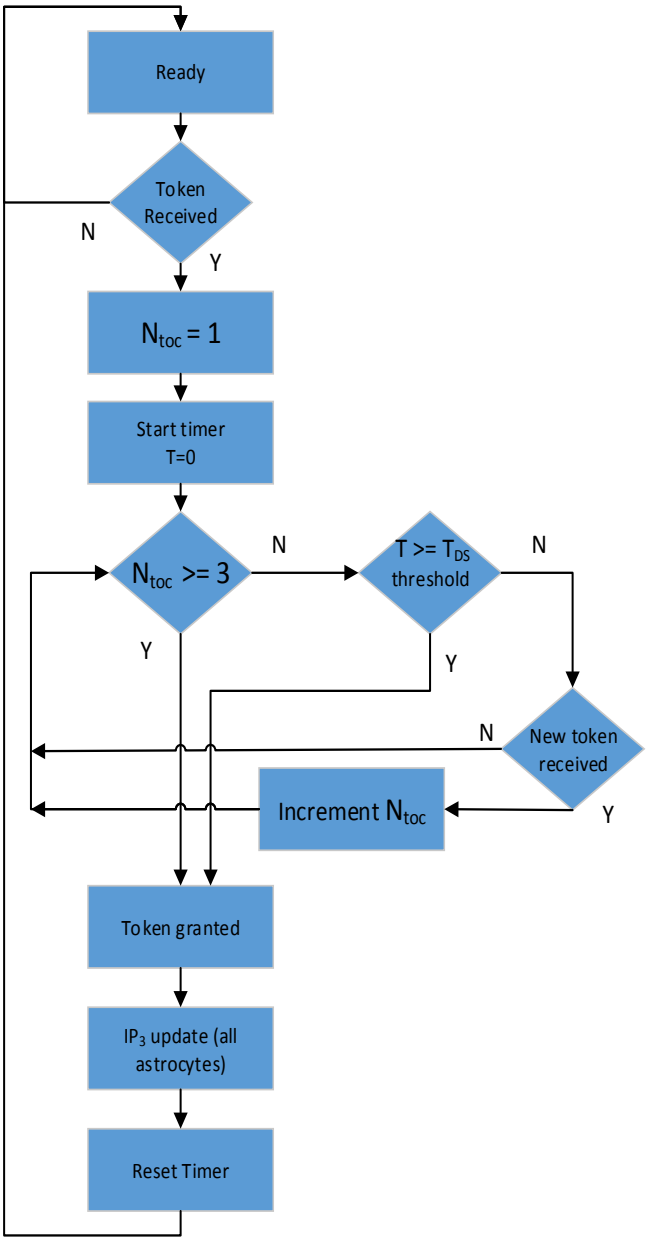

Fig.6.6 Update manager DS flow chart.

The update manager waits for an initial token request. When the first token request is received, the value of n_{toc} is incremented by 1 and a timer starts, T=0. This is the start of the update manager flow. The update manager will remain inactive as long as the current time (T) is under T_{DS}. If the update time window has expired, without receiving another token request the update will start. T_{DS} is a variable threshold and this threshold is set when the first token request has been sent regardless of the number of subsequent token

requests. This ensures that the system updates periodically with changes in IP_3. If there are three or more token requests within this time period (T_{DS}) the DS will enable an update of the IP_3 values by granting the token to 'inter-router #0'. Thus, the DS aims to reduce the frequency of the update based on the two conditions.

The update manager holds a single token and only when this token is released the update process start. The token system is used to manage updates because the values of the IP_3 will change frequently and result in more frequent IP_3 updates. This change of IP_3 is insignificant at times and does not warrant a complete update, to balance this, when three requests, the threshold has been met and it is deemed adequately urgent to perform the update this balances updates and keeps the astrocytes in sync. If there are three token requests (N_{toc}) within this time frame the IP_3 will update: three was chosen as an arbitrary starting value for testing purposes. It would then be comparable to using six and nine tokens, which yields more results and greater insight in less steps e.g. in 2's (2, 4, 6, 8, 10). If there is one token request the update will not occur until the update window expires. The update window T_{DS} is based on timescales of 10ms, 100ms and 1 sec. These values were chosen again for testing purposes, 1 sec is comparable to biological timescales and reducing this by magnitudes of 10, would give an ideal testing timeframe. These are the max update rates of IP_3, where the refresh/update rate is not immediate but rather dynamic within the time constraints of the update window.

This DS process allows scope for tweaking either for a more power efficient model or a model based on a higher throughput. It can also be altered to release the token using less or more requests or by changing the time threshold. These variables can be optimised to provide either a more power efficient or higher throughput strategy.

6.4 Results and analysis

This section outlines the test setup and provides area-power performance analysis of the astrocyte tile router. The performance of the router used in the ring topology of the astrocyte is compared with the astrocyte core itself (computation component) and the

spike-based H-NoC (interconnect) to demonstrate its compactness and hardware scalability.

The H-NoC neuron facility, astrocyte cell and proposed astrocyte router have been described in VHDL and synthesised for a Xilinx Virtex-7 XC7VX485T-2FFG1761C FPGA evaluation board using Xilinx Vivado 2016.4.

6.4.1 Performance: Astrocyte tile router

The area and power estimates are obtained using Vivado, which estimates area using LUTs and slice registers. Power is compared using both the static and dynamic power respectively. Table.6.3 outlines the area overhead of the astrocyte tile router and compares it with the astrocyte core, H-NoC neuron facility and “e-SP comms”.

In Table 6.3, the astrocyte core (i.e. computation component) is used as a benchmark to compare area-size of the various interconnect components. A single H-NoC neuron facility (Node router) consumes 3.2% in terms of LUTs and 4.5% in terms of slice registers of the FPGA device. In comparison, the “e-SP comms” interconnect is very compact with only 0.6% of LUTs and 1.2% in slice registers. The astrocyte tile router is 2.2% in terms of LUTS and 4% in terms of slice registers which is larger but still more compact than the area of the H-NoC Node router.

Table 6.3 ‘Astro-Router’ block evaluation

Component	**LUTs**	**(%)**	**Slice Register**	**(%)**
Astrocyte Core	16,305	-	16,182	-
Node Router (H-NoC)	527	**3.2**	735	**4.5**
AstrocyteTileRouter	365	**2.2**	651	**4**
e-SP comms	99	**0.6**	199	**1.2**

There is a small area overhead incurred by the astrocyte tile router and the 8 inter-routers. These inter-routers act as signal managers, directing the IP_3 from the astrocyte core to the astrocyte tile router, each tile consumes 70 LUTS and 64 slice registers, and the inter-router uses 27 LUTs and 55 slice registers.

6.4.2 Scalability analysis

In regard to scalability, Fig.6.7 shows the interconnect area overhead for various sizes of neuro-glia network implementations. There are two relatable aspects a single neuron facility.

1. A single tile router (astrocyte).
2. A single ‘e-SP comms’ block.

In one implementation of a neuro-glia network, one tile router (astrocyte) connects to eight neuron facility (ten neurons) where each neuron facility has one ‘e-SP comms’ block. Therefore, to emphasise how a neuro-glia network scales, each aspect is scaled and compared e.g. array sizes of 10x10 up to 50x50.

Fig.6.7 shows that the astrocyte tile router interconnect is a scalable interconnect solution. In each scaled array there is an exponential growth as the network scales, however this is expected as its slope is less than that of the neuron facility (H-NoC) and as there are 10 neuron facilities per 1 astrocyte tile router: the astrocyte network will have less area overhead. The X-axis is the number of components and the array size. The two Y-axis show the area used up in terms of LUT’s (using a solid line) and slice registers (using a dashed line), respectively. The number of LUTs and slice registers increases exponentially as the number of tiles increases, this is expected as the network scales. One neuron facility of H-NoC and the ‘e-SP comms’ block shows the minimal overhead incurred by the ‘e-SP comms’ block. There are ten neuron facilities per tile facility in H-NoC and each tile facility correlates with a 1:1 ratio of tile facilities (astrocyte). The neuron facility and the astrocyte tile router are comparable in size and scale similarly, however in one implementation it is one astrocyte tile router to ten neuron facilities.

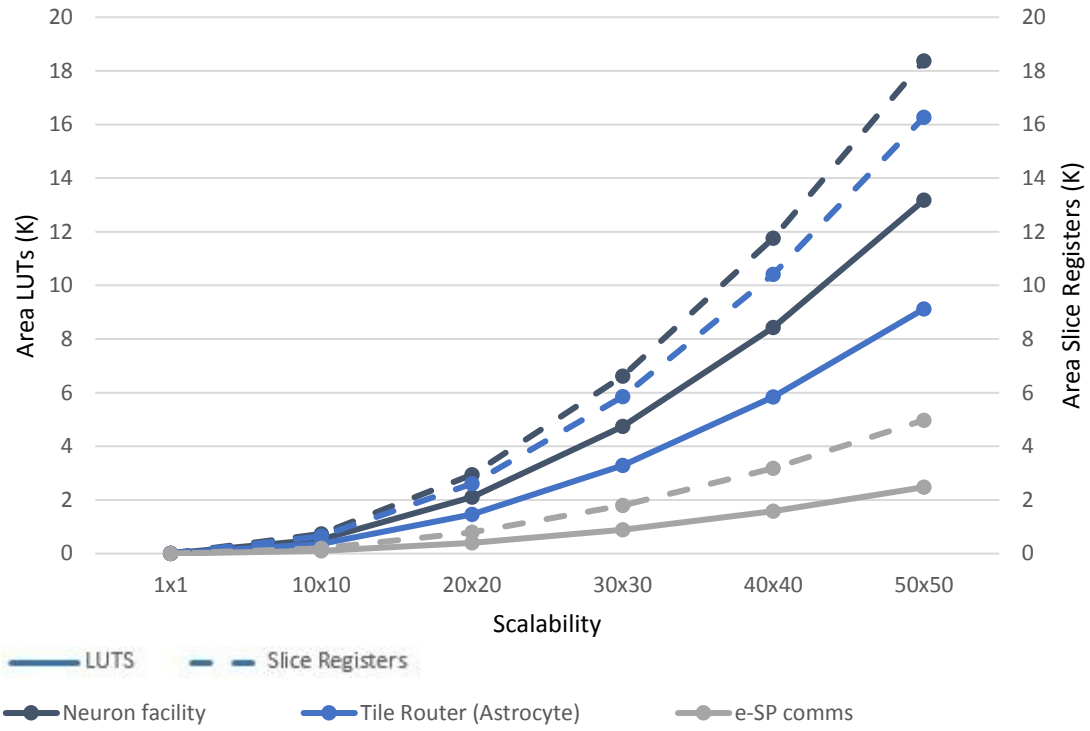

Fig.6.7 Scalability LUTs and slice registers.

These results indicate that there is a very small area overhead when communicating IP_3 between astrocytes and this is important for global self-repair as astrocytes communicate using IP_3. This helps facilitate a neuro-glia network communication between astrocyte cells. The computational model uses double point precision and the astrocyte requires this precision to remain faithful to the computational requirements. Therefore, the 64-bit data is communicated serially using a ring-topology to minimise area. The results show that the area overhead incurred by adding the astrocyte tile router and 'e-SP comms' interconnect block is small when compared to the size of the actual astrocyte computation core. There is one astrocyte tile router to every eight astrocytes and each astrocyte has an 'e-SP comms' block. This shows the relationship between the interconnection and computation, e.g. one astrocyte tile contains eight astrocytes with eight 'e-SP comms' blocks.

6.4.3 Power analysis with dynamic scheduler

A power analysis was carried out using time as a variable and then the number of iterations, doing so identified a balance between the time and number of iterations per each update. Therefore, these results reflect updates within a single astrocyte tile router. Firstly, t_{DS} is set to 10, 100 or 1000, that is, an update occurs once per t_{DS} within 1,000ms. Table 6.4, compares the power consumed over the course of 1,000 Ms (1 second). The quantity t_{DS} is the time interval of each update whilst updating the global IP_3 once within one tile facility astrocyte. Therefore, 1 update is 2.45 Watts and if we update 10 times during this time period the power consumed is 24.56 Watts. One entire update cycle consists of adding the IP_3 values from each astrocyte. This update process accumulates, averages the IP_3 and communicates this new value around each astrocyte. Fig.6.8. shows that as we reduce the number of updates or iterations for a certain time period, it significantly reduces the power consumed: a slower update against the power consumed is can be varied to find the best performance metric. Table 6.4 shows that as the update period, t_{DS}, increases the power consumed reduces. As the astrocyte is typically a slow changing process in the order of seconds, there is scope to reduce the update rate in hardware and therefore this will enable the reduction of power to support scalable implementations.

Table 6.4 Power analysis with varied t_{DS}

t_{DS} (ms)	**10**	**100**	**1,000**
Power (Watts)	245.6	24.56	2.456

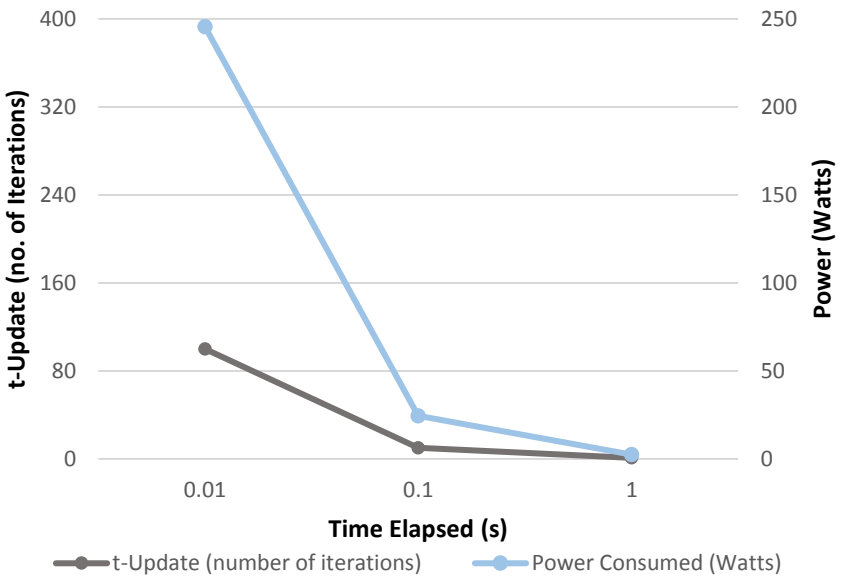

Fig.6.8 Dynamic scheduler evaluation.

Furthermore, it is important to explore the power consumed during each update iteration and this was carried out using xPower within Vivado 2016.4. Using a typical activity file, Table 6.5 outlines the power consumed by the astrocyte tile router compared with the astrocyte computation itself, and the 'e-Sp comms' block. This demonstrates that the interconnect for the astrocyte communication is scalable as it only exhibits ~12% (3.071/25.471) of the power used by an astrocyte computation.

Table 6.5 Power analysis

Component	(Watts)		
	Static	**Dynamic**	**Total**
Astrocyte cell	0.732	24.739	**25.471**
Astrocyte tile router	0.088	2.465	**2.543**
'e-SP comms' Block	0.074	0.454	**0.528**

Table 6.6 outlines the power consumed by each individual design; the astrocyte cell, the astrocyte tile router and 'e-SP comms'. This table compares the average for one iteration and then scaled for 1, 10, 20, 30, 40 and 50 iterations. This table shows that the low power design consumes on average around 2.44 watts per iteration and is broken down

further into 1.28w (signals) and 1.24w (data). This design shows a small overhead in terms of both power and area.

Table 6.6 Power Analysis of individual components within the astrocyte tile router.

Component	1	10	20	30	40	50
Tile Router (Astrocyte)	2.4	24	48	72	96	120
Tile Router component	1.26	12.6	25.2	37.8	50.4	63
Inter-router	0.11	1.1	2.2	3.3	4.4	5.5
TOTAL	3.77	37.7	75.4	113.1	150.8	188.5

Fig.6.9. compares the power consumed as the interconnect array size scales. As the size of the network scales the power consumption scales linearly, this is preferred as the network is expected to be reproduced on large scales. The components required for astrocyte communication from Chapters 5 and 6 respectively, are the “e-SP comms” and the astrocyte tile router. As the number of tile routers and “e-SP comms” blocks increase there is an increase in power. For example, if there are 40 astrocyte tile routers in hardware the power consumed will be approximately 100 watts. The “e-SP comms” (in grey) shows the power consumption of each individual “e-SP comms” block. However, for every astrocyte tile router there will be eight “e-SP comms” blocks, the black line represents the total which is one astrocyte tile router and 8 “e-SP comms” blocks. Table 6.7. shows the breakdown of power consumed by each component and compares the interconnect of both the astrocyte tile router and the e-SP comms block with the astrocyte. This also shows dynamic and static power consumption. It can be seen that compared to the astrocyte the total power consumed is 10x less. With eight astrocytes connecting to one astrocyte tile router consuming only $1/10^{th}$ of the power of a single astrocyte, it can be seen that the astrocyte tile router is a good trade-off between area and throughput yielding low area and power overheads in comparison to the astrocyte.

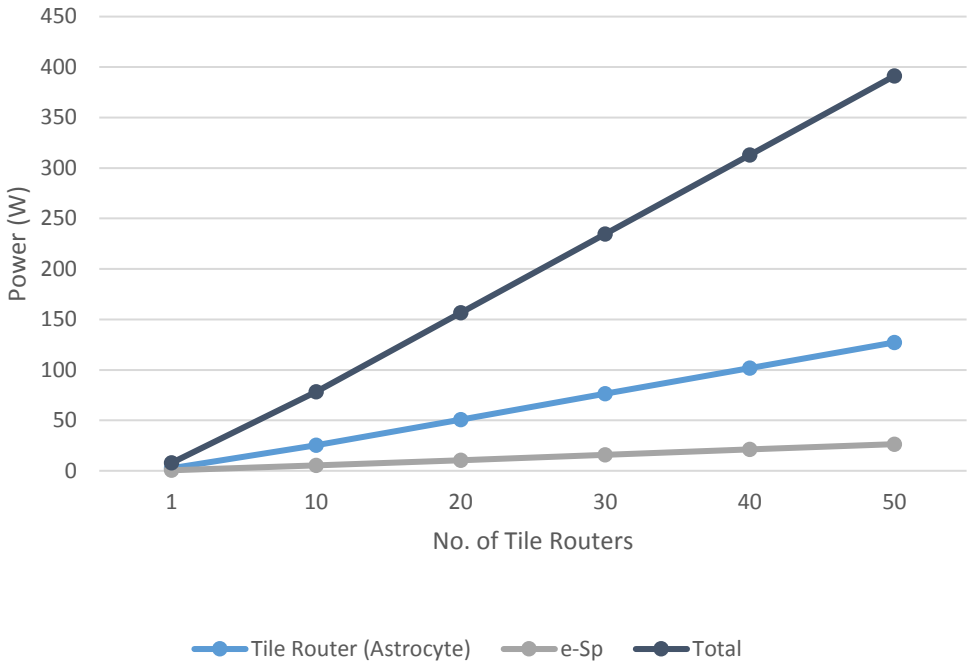

Fig.6.9 Scalability in terms of astrocyte tile routers.

Table 6.7 Power analysis per component.

Power analysis		**(watts)**		
Component	**Static**	**Dynamic**		
Breakdown		**Total Power**	**Signals**	**Data**
Astrocyte	0.732	24.739	13.989	12.229
Astrocyte_tile_router	0.088	2.465	1.296	1.253
e-SP comms Block	0.074	0.454	0.229	0.229

6.5 Conclusion

In this chapter a novel astrocyte tile router and ring topology supporting the exchange of IP_3 signals between eight astrocytes has been proposed. This ring topology is similar to that in Chapter 5 and proposes exploiting the slow changing astrocyte process used in

self-repairing networks. This allows the overheads such as power and area to be minimal as it uses an area and power efficient design that is scalable.

Astrocytes are computationally expensive and therefore demand a lot of resources on an FPGA platform. With both individual networks, neural network and astrocyte network the number of PEs (neurons and astrocytes) and communication signals is significant in size and interfacing the two networks is a difficult challenge. Chapter 5 addresses the vast number of communication signals: high level astrocyte to astrocyte communications and combining local and global communication protocols. This allows the astrocyte to use necessary resources whilst maintaining the low overheads in both area and power to provide a scalable solution to the interconnect challenge.

This novel NoC interconnection scheme for communicating enables a significant number of astrocytes to exchange data with other astrocytes with minimal area and low power constraints. As previously stated each astrocyte is interfaced with 10 neurons and each astrocyte tile router accommodates 8 astrocytes which allows 80 neurons per astrocyte tile facility. This will enable self-repair within SNN hardware and facilitates the exploration of self-repair in electronic systems using a biologically inspired approach. The proposed NoC interconnect provides a hardware building block for developing neuro-glia interconnect for self-repair strategies. IP_3 is an important communication signal from astrocyte to astrocyte. This novel interconnect allows eight astrocytes to communicate IP_3 within an astrocyte network.

This neuro-glia networks exploits the innate slow changing astrocyte signal in order to reduce area and power overheads. This chapter provides a scalable interconnect to address the above communication and interconnect challenges within neuro-glia networks. The chapter also explores this paradigm using a novel NoC astrocyte tile router. This interconnect consists of three major novel components within neuro-glia networks

1. Astrocyte tile router connected in a ring topology
2. A novel IP_3 accumulator
3. A novel update manager (token system and dynamic scheduler).

The next Chapter (Chapter 7), implements the e-SP ring, described in Chapter 5, in a SANN (with self-repair capabilities in a SNN with an astrocyte cell in hardware). This low-level interconnect in addition to the high-level astrocyte communications provide a platform for developing a neuro-glia interconnect for future inspired computing paradigms regarding self-repair strategies in hardware.

Chapter 7: Application of spiking astrocyte-neuron network using Networks-on-Chip

7.1 Introduction

This chapter presents an implementation of a robotic controller in FPGA hardware using the proposed NoC mechanism. The mobile robot uses a SANN to control motor speed. The firing activity of the output neurons are converted into a pulse width modulator (PWM) signal, which in turn, controls the speed of wheel motors. Previous results [86] show that if the synapses within the astrocyte-neuron neural network are faulty, the network has the ability to adapt and repair, thus maintaining the direction and speed. This experimental phase was carried out using fault densities of 0%, 20%, 40%, 60% and 80%. Results have shown that with up to 80% of faulty synapses, the network can repair and restore pre-fault functionality with only a slight degradation of output frequency. This chapter focuses on using an e-SP ring where area overhead is the key requirement and latency is not a critical requirement, due to the slow astrocyte process in biology. Results demonstrate that the output frequency had an average of 5.77 Hz down from 7.27 Hz when the network suffers catastrophic failure (80% of faults). The e-SP ring has a low area overhead relative to the astrocyte, 1.68% of the total number of slice registers and 3.62% of slice registers.

This chapter aims to provide an overview of the self-repairing process in hardware where the SANN and the communication exchange between neurons and astrocytes is discussed. The main focus is to implement the ring topology and communication protocol developed in Chapter 5, thus providing a scalable solution to the interconnect challenge within an astrocyte-neuron network. This may be applied to larger networks in the future. Results demonstrate that the ring topology provides a good trade-off between low area/interconnect overhead and communication speed for the relatively slow-changing data between astrocyte and neurons as well as providing a self-repairing capability in a real-time application.

7.2 The spiking astrocyte-neuron network

This section introduces the spiking neuron network (SNN), which when coupled with an astrocyte, is a first step in realising the challenges and constraints of an astrocyte-neuron network. In the present case, the SNN contains two neurons where each neuron has ten input synapses coupled to an astrocyte. The astrocyte mediates synaptic activity and PR across all twenty synapses. The hardware architecture includes three facilities – probabilistic tripartite synapse [124], LIF model [23], and the De Pitta et al. astrocyte model [132].

7.2.1 An overview of the robotic car architecture

The robotic car used in this chapter, is based on a SANN. The SANN accepts input neuron spikes as stimulus and outputs a spike frequency, which is converted via pulse width modulation (PWM) modules and then used to drive and control the speed of the robot. A software model of the SANN had been used previously to demonstrate fault detection and self-repair.

Fig.7.1 shows two neurons (N1 and N2) firing which results in two the feedback signals e-SP and DSE; e-SP is excitatory and DSE is suppressive. C1 and C2 are tripartite synapses associated with the neurons which are coupled to the astrocyte (A1). There are two scenarios as shown in Fig.7.1 where:

(A) No faults: all synapses are capable of passing information.

(B) Faults: a fraction of synapses are damaged and are not capable of passing information and N2 has stopped firing.

DSE (the suppressing signal) reduces as N2 becomes silent due to faulty synapses. The e-SP feedback signal to synapses associated with N2 increases as N1 remains active. This leads to an imbalance of e-SP and DSE as there is an increase of e-SP but DSE is reduced within the array of synapses in C2. As a result, the PR of the healthy synapses associated with N2 (C2) is increased. This results in the restoration of N2 to a pre-fault firing activity.

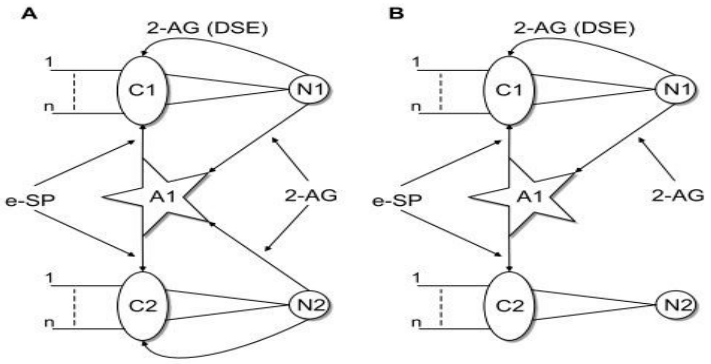

Fig.7.1 Astrocyte feedback. N1 and N2 are neurons and A1, an astrocyte. (A.) shows a no faults and (B.) shows N2 as faulty. This shows e-SP and DSE as excitatory and suppressive, feedback signals to associated synapses.

Fig.7.2 is an overview of the physical components used to build the car and includes: the FPGA hardware with the SANN controller, the motor units and the completed mobile robotic car.

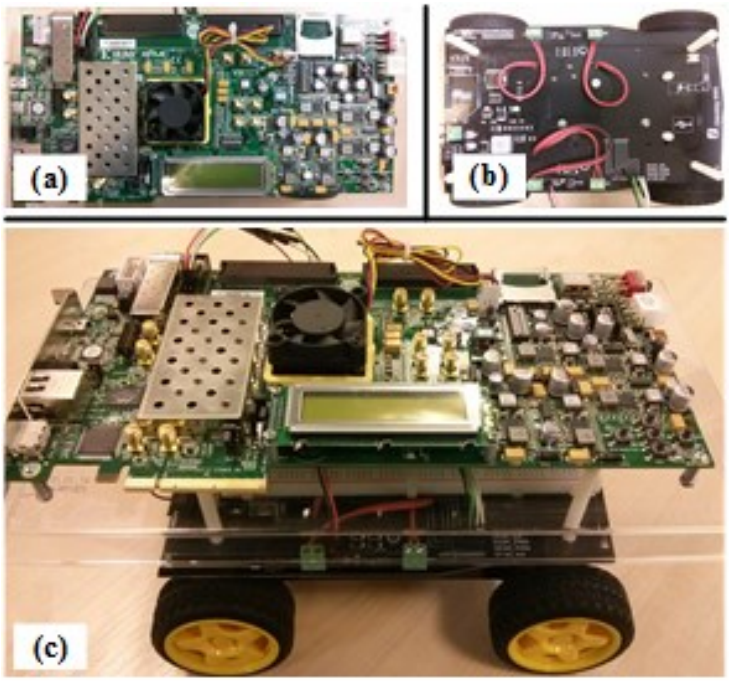

Fig.7.2 The FPGA based robotic car. (a). The FPGA device. (b). Motor circuits controlled by FPGA. (c). The final robotic car [86].

The FPGA hardware is programmed to execute the SANN where each neuron has 10 associated synapses. This small network controls the mobile robotic car. With this small-scale network self-repair can be demonstrated using a real hardware application. Fig.7.3 is the hardware architecture. The three main components described are: (a) the SANN (b) the FPGA connections to the mobile car and (c) the computer interface used to collect and monitor signals.

The SANN is developed on an FPGA and consists of two neurons (Neuron #1 and #2), associated synapses and the astrocyte process. The neurons generate 2-AG which splits into two signals, DSE which goes to the astrocyte and reduces the PR on all synapses. The astrocyte then generates e-SP which increases PR on all synapses. The neurons are modelled using a LIF model [86]. The output spike from each neuron, is converted to an output frequency by a PWM. Neuron #2 controls the speed of the robot from this output frequency.

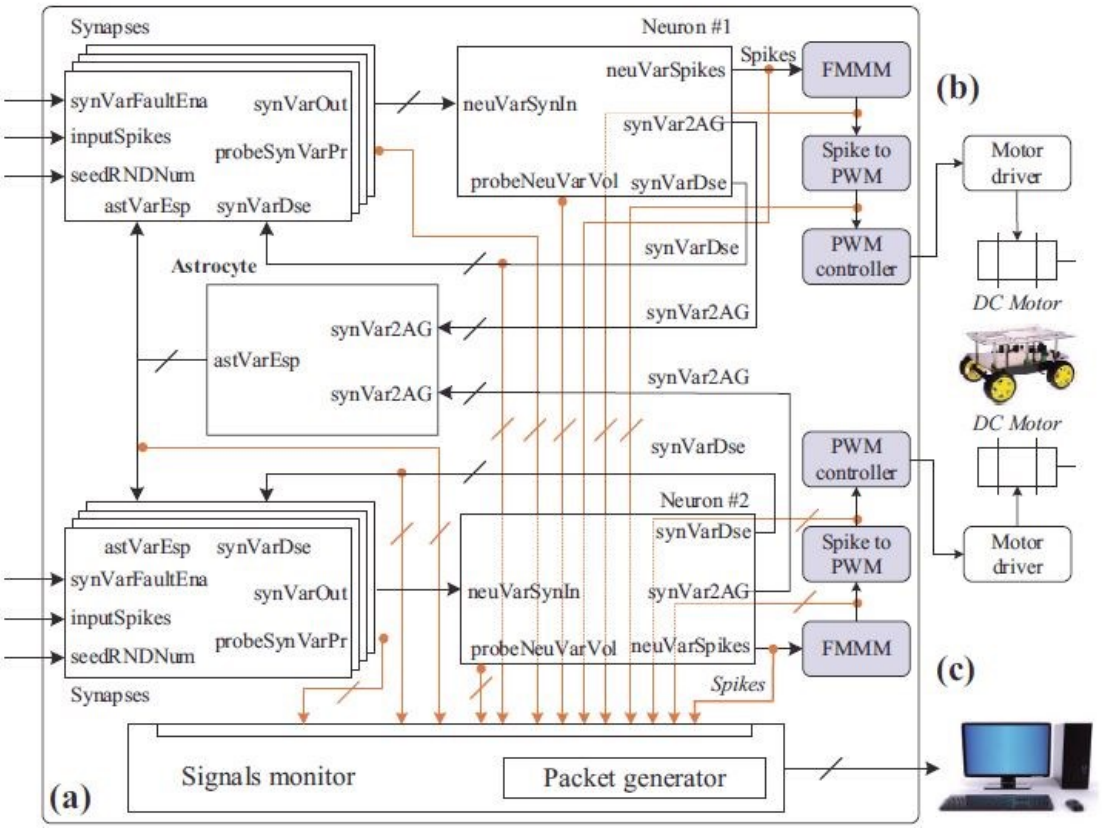

Fig.7.3 Overview of the hardware architecture. (a). SANN FPGA implementation (b). Mobile robotic car. (c). Software used to collect and monitor signals [86].

7.2.2 The astrocyte process

Within the SANN, there is a two-tier hierarchy of communication:

I. Low-level communication: when the neurons and synapses interact with the astrocyte.

II. High-level communication: Inter-astrocyte communication.

When a neuron stops firing it is considered silent or near silent which is caused by a decrease of PR in associated synapses i.e. faulty synapses. Astrocytes detect faulty synapses associated with silent neurons and this is referred to as fine-grain detection and repair of defective synapses. To facilitate repair the astrocyte increases the PR of healthy synapses thus restoring the neuron activity to its pre-fault level. From an abstract point of view, the astrocyte within the SANN receives 2-AG from the neuron facilities and this outputs the signal, e-SP, to the synapses which serves to modulate synaptic PR. The astrocyte model consists of two 2-AG generators and an astrocyte core; spikes from neurons produce DSE and 2-AG signals where the latter produces the e-SP signal, as seen in Chapter 7. Fig.4.4.

Fig.7.4. shows the hardware architecture with e-SP ring NoC. Within the SANN, there are two neurons. Each neuron facility is connected to a synapse facility and an astrocyte facility. The astrocyte facility outputs a signal (e-SP) to the synapse facilities and increases PR at the pre-synaptic terminal. The synapse facility is based on a probabilistic-based synapse model and has many input signals which influence PR. The inputs are based on input spikes, it receives a signal “signal for the random number generator”, and both the DSE and e-SP signals from neuron and astrocyte facilities, respectively. It has one output signal (synVarOut), which is connected to the neuron facility [79].

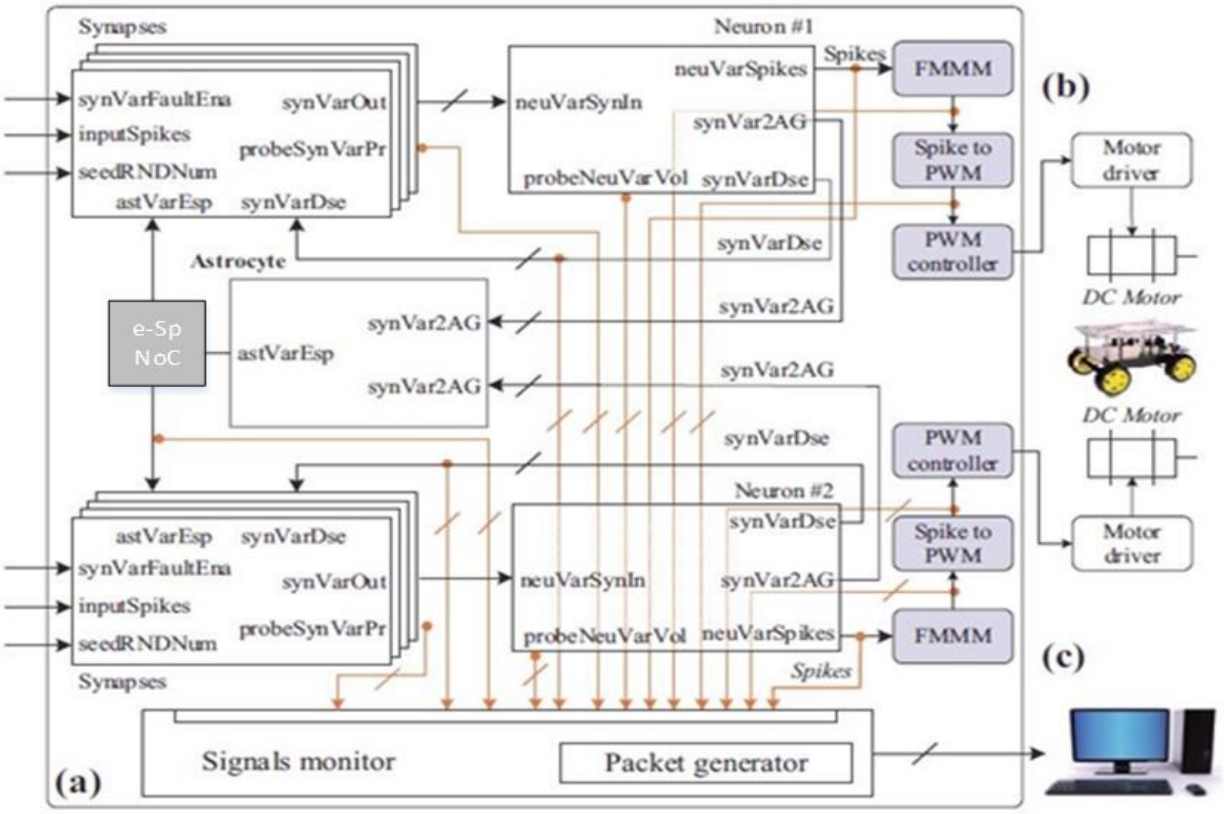

Fig.7.4 Overview of the hardware architecture with e-SP ring NoC.

Within the synapse facility, the PR is regulated by the DSE and e-SP signals. These are the signals within biological constraints which act to balance the PR of the synapses. In Fig.7.5 there are several graphs to monitor the performance and output of the mobile robotic car without faults. This is used to compare against computational simulations [15], [133]. The PR is affected solely by the DSE and e-SP and has an initial value of 0.5. An input fault enable signal (synVarFaultEna) is used to simulate faults and if this signal is high, the PR value is set to 0.1 which is adequate to simulate a fault.

Within Fig.7.6, there are again several graphs. The PR of the first synapse and the PR of the tenth synapse associated with Neuron 2. There is the excitatory e-SP and the depressing DSE signals and finally, the output frequency of the neuron, this is the Freq (Hz). Results show, as expected, the output spike frequency is maintained, the e-SP rises and the DSE falls. Over the time period PR fluctuates half way at approximately 0.25.

Neuron 2 and the output frequency controls the robots speed. These graphs enable visualisation of the results over a 600s period.

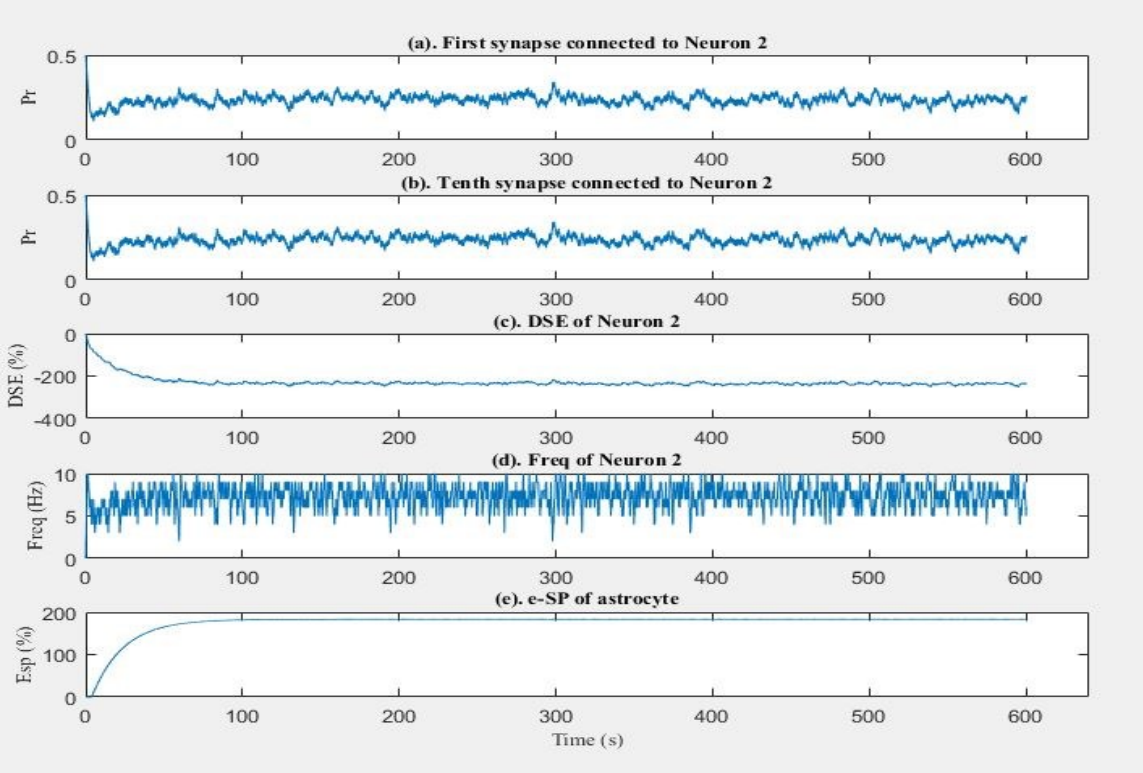

Fig.7.5 Neuron 2 regular activity with no faulty synapses. Various graphs to visualise the results and the output frequency over a 600s period. The DSE drops and the e-SP rises.

7.3 e-SP and the astrocyte neuron network

The astrocyte releases e-SP to strengthen PR and facilitate self-repair by strengthening weights associated with the remaining healthy synapses and the firing activity of the neuron is restored.

Fig.7.6 and Fig.7.7 show the first and tenth PR variation over time of the synapses associated with Neuron 2. During the first simulation (Fig.7.6) the e-SP does not go back to the synapses and therefore the only signal present at the synapse is DSE; this results in a significant depression of PR at all synapses. At 200s faults are injected. Fig.7.6 shows there is virtually no self-repair taking place. However, there are faulty synapses. The PR in the first synapse drops and the tenth synapses PR partially increases due to the fall in the decrease in the DSE signal due to the slower neural firing frequency.

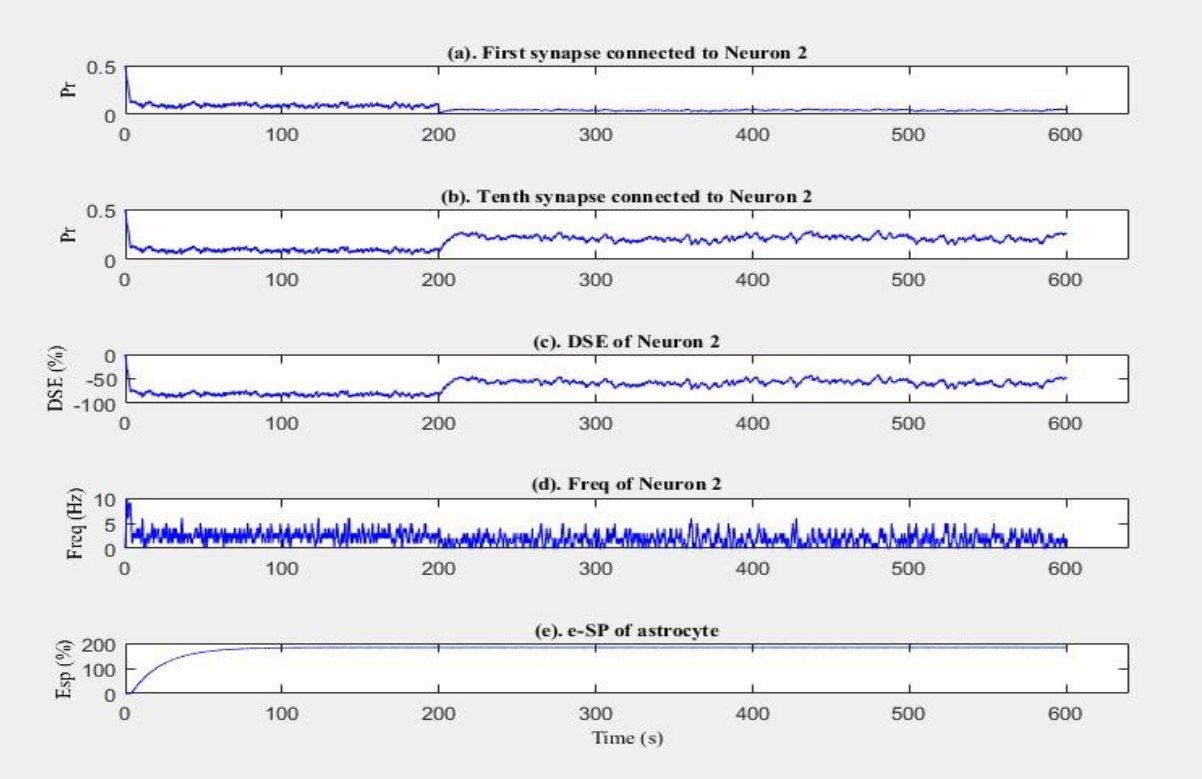

Fig.7.6 Neuron 2 faults with no e-SP.

The simulation in Fig.7.7 shows the same set-up as the previous simulation where faults are injected into the synapses at 200s. In this case, the e-SP is directed into the synapses. The first synapse associated with neuron #2 is faulty. The drop in PR indicates this is a faulty synapse and consequently, the output frequency of Neuron 2 falls. As the e-SP is directed to the healthy synapses, the healthy synapse PR increases and the frequency of neuron #2 begins to recover to a pre-fault frequency. This shows the importance of the e-SP signal in facilitating self-repair.

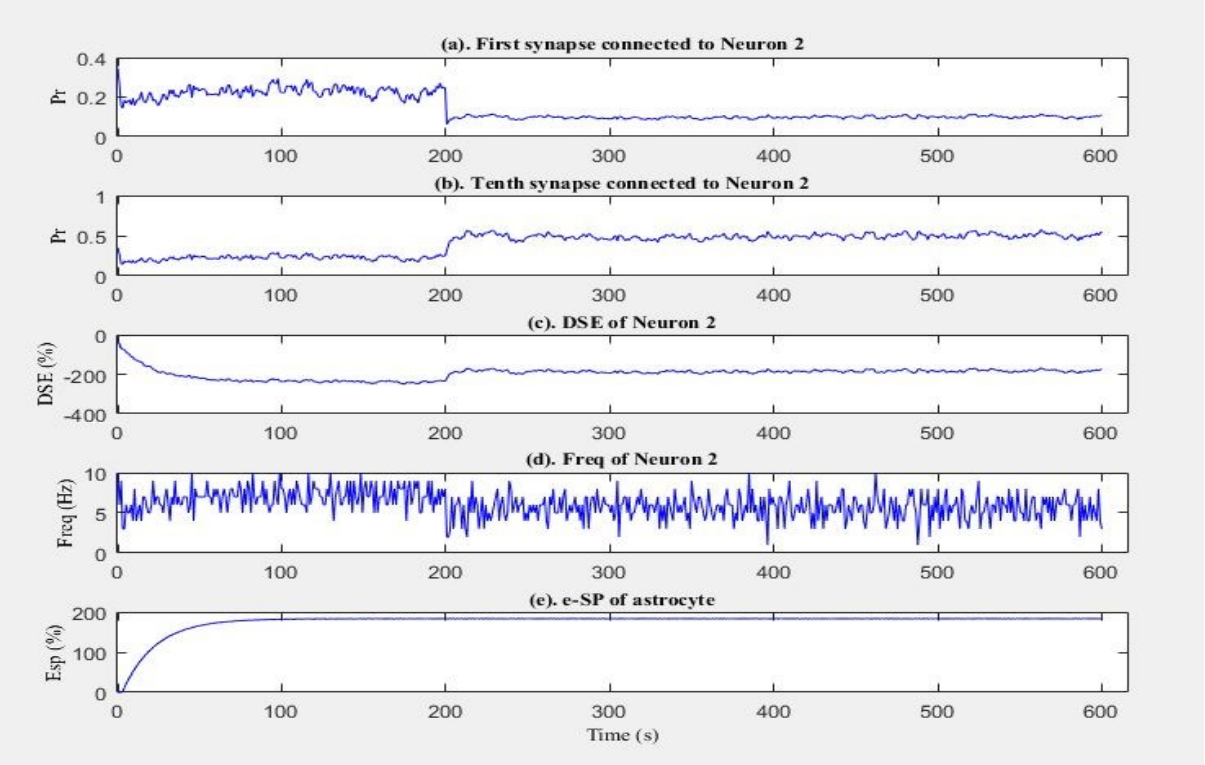

Fig.7.7 Neuron 2 faults with e-SP.

Communicating e-SP and DSE are essential within the SANN. The SNN and astrocyte network operate in parallel and exchange data continuously maintaining self-repair capability. However, the complexity of the astrocyte- neuron coupling takes up significant resources in terms of area on FPGAs. Below table 7.1 [86] shows the size of each component of the SANN.

Table 7.1 SANN area analysis

	Slice LUTs	Slice Registers	Slice	Block RAM Tile	DSPs
Astrocyte	11,394	11,666	3,552	5	42
Synapse & Neuron	9,865	10,383	3,120	0.5	45
DSE generator	4,353	4,688	1,394	0	14
FMMM	65	80	40	4	0
PWM Controller	21	20	6	0	0

The overhead within this SANN is a cause of concern as the network increases in size. To address this we look to use the low-level NoC hardware communication architecture, used in chapter 5 which extended the H-NoC.

7.4 Self-repair using “e-SP comms” ring

The SANN is developed using Vivado synthesis tools and due to the Modular nature of the SANN it can be broken down into two neuron facilities, two synapse facilities and an astrocyte facility. Within this FPGA modular architecture there are 64 bit buses, which is a requirement due to the binary precision used for the astrocyte data representation; i.e. the astrocyte requires 64 bit precision to remain accurate when compared with software computational models. While 64-bit precision does indeed impose a large overhead on the communications protocol, it is used so that the system output can be compared against the computer simulated models and optimising the area of the astrocyte core is not the focus of this thesis. The reduction of the 64-bit precision and therefore optimisation of communication overhead is a target for future embedded systems implementations. The first step in realising the e-SP with large scale SANNs is to work within a biological timescale and so, the ‘e-SP comms’ module was integrated into the SANN. The e-SP data comes from the astrocyte and is used as input data to the synapse facilities.

The e-SP ring takes the e-SP output from the astrocyte and sends it to the associated synapses at each neuron. Fig.7.8 and Fig.7.9 show the SANN with the e-SP ring in place, under a 0% fault condition and this shows that there is no deviation of results when using the ring in the hardware. Both figures show the PR at first and tenth synapses at the associated neurons respectively. The DSE is also shown in each. The output frequency and the e-SP are shown in red, these signals are directly affected by the e-SP ring. The results are plotted using MATLAB.

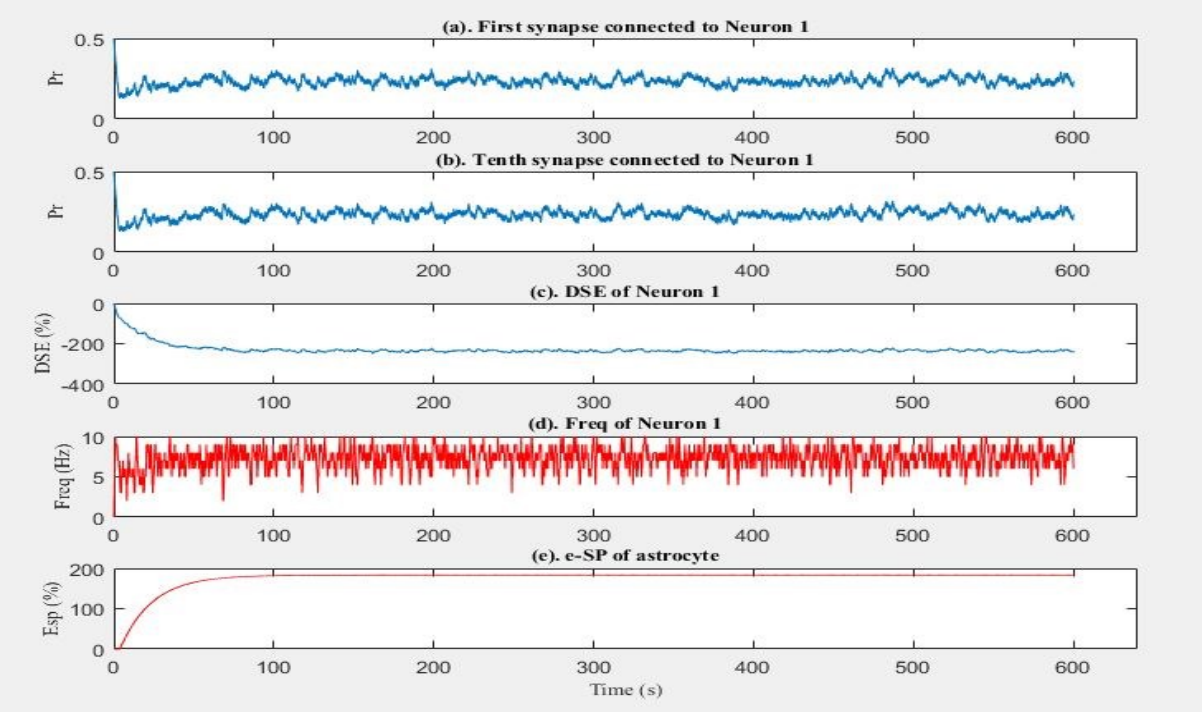

Fig.7.8 Neuron 1 with “e-SP comms” (no faults).

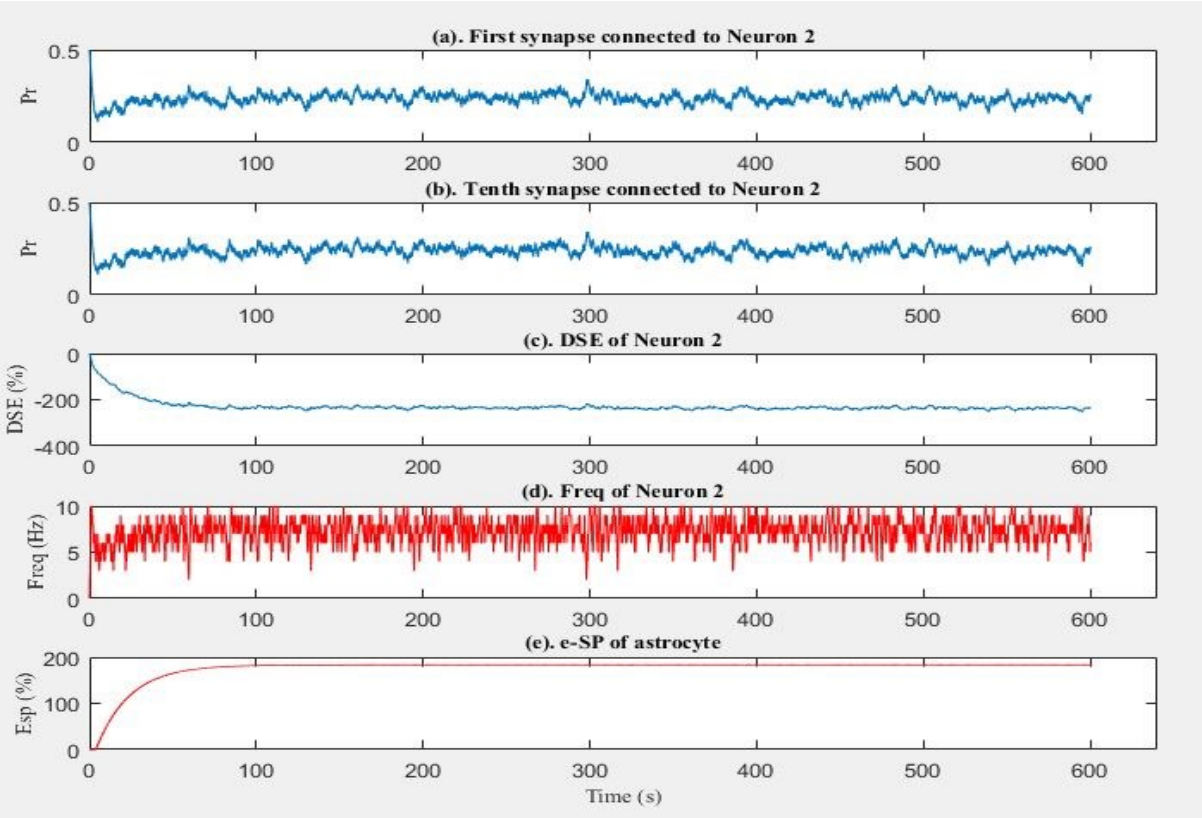

Fig.7.9 Neuron 2 with “e-SP comms” (no faults).

Fig.7.10 and Fig.7.11 show the same plots as before but with faults injected into the SANN during simulation. Fig.7.10 has 40% faults injected in synapses associated with Neuron 2 and Fig.7.11 has 80% of faults injected. This shows that the e-SP ring works within the NoC and provides self-repair with no deviation from the original design.

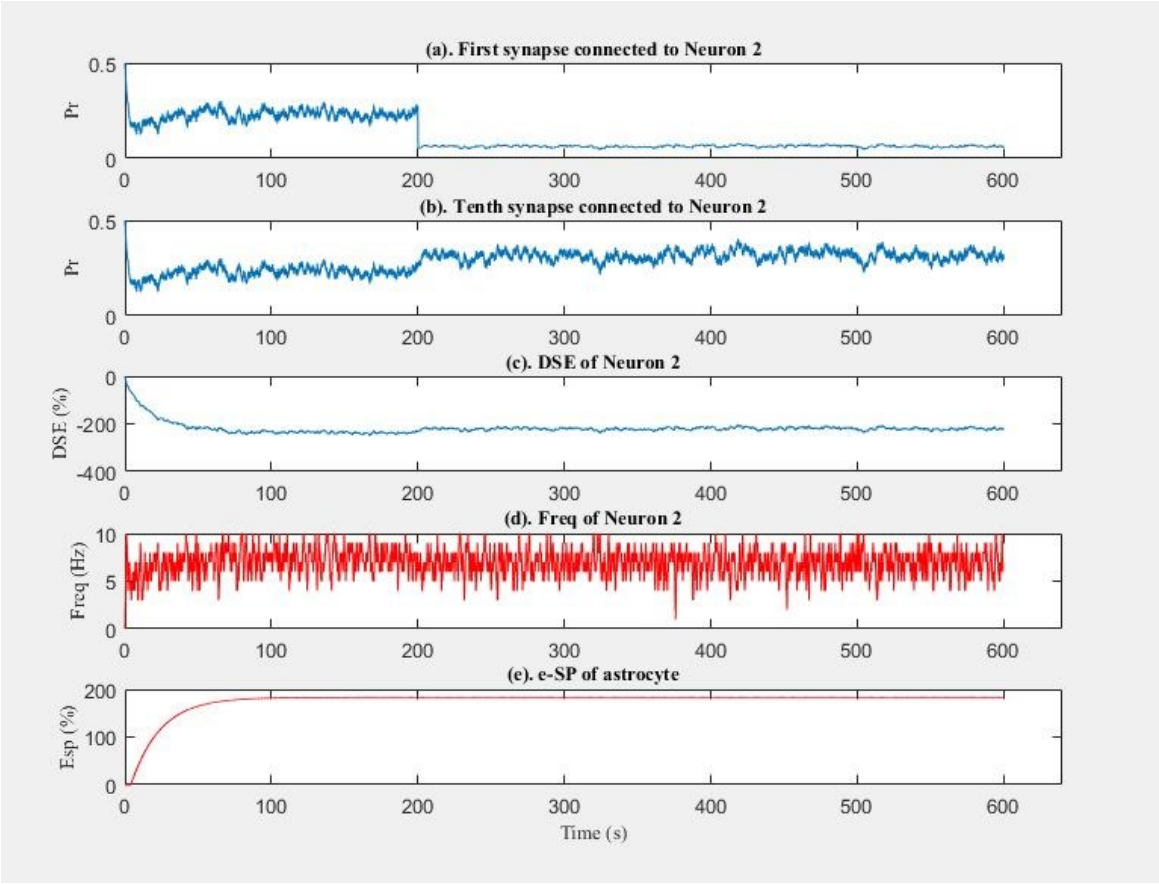

Fig.7.10 Neuron 2 with “e-SP comms” (40% faults).

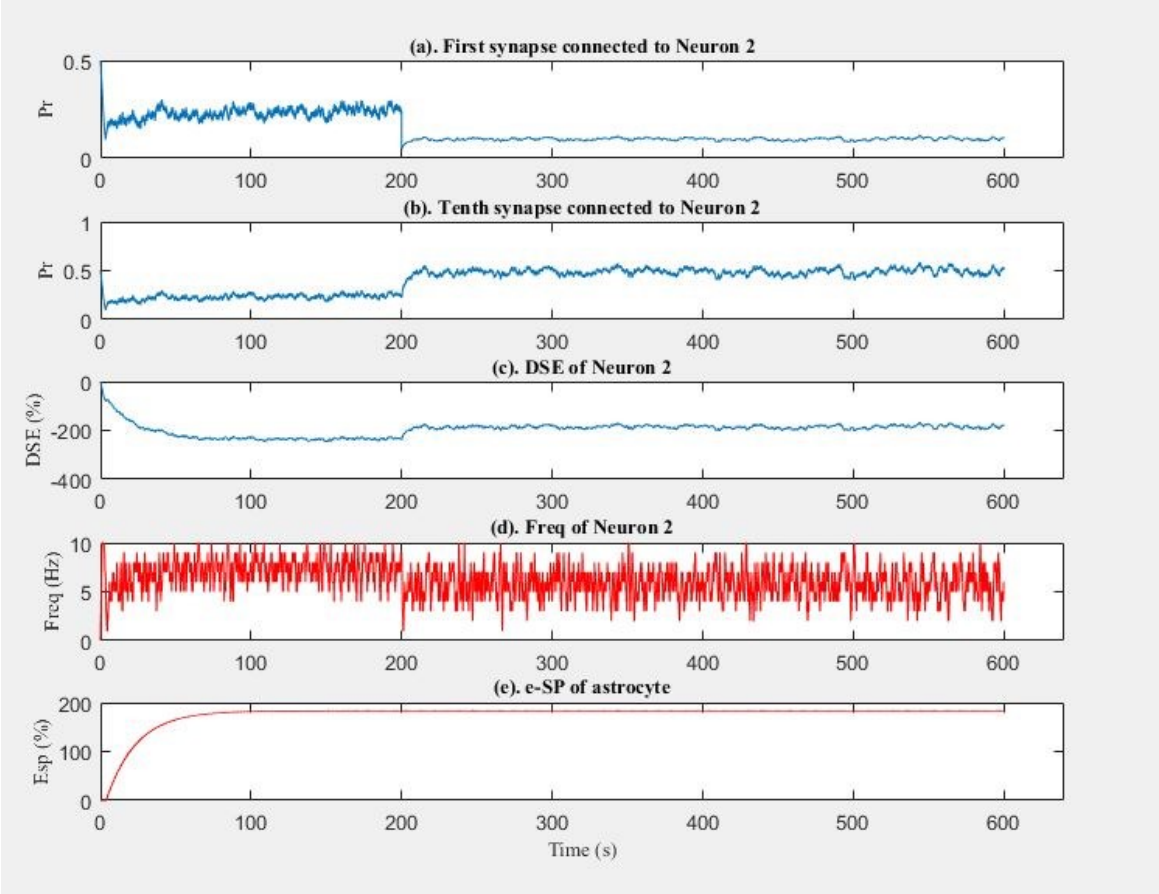

Fig.7.11 Neuron 2 with “e-SP comms” (80% faults).

Table 7.2 shows that the e-SP ring is 1.68% of the total number of slice registers and 3.62% of slice registers relative to the astrocyte.

Table 7.2 e-SP ring relative area analysis

	Slice LUTS	Slice Registers	Block RAM Tile	DSPs
Astrocyte	11,394	11,666	5	42
Synapse & Neuron	9,865	10,383	0.5	45
DSE generator	4,353	4,688	0	14
FMMM	65	80	4	0
PWM Controller	21	20	0	0
e-Sp ring	191	422	0	0

Table 7.3 and Table 7.4 compare the output frequency of neuron #1 and neuron #2 with and without the e-SP ring. Table 7.3 demonstrates that the average output frequency of the simulation vs the hardware, this was to show there was very little difference when applied to hardware. This was carried out for 0%, 40% and 80% of faulty synapses [124]. These simulations were recreated with the e-SP ring in place to show there was very little difference in the average output frequency. The e-SP ring converts a 64 bit packet in a serial transmission, this could cause latency if there is a delay in the e-SP getting back to the synapses. The average output frequency has very little deviation of results (+/- 0.2), which is caused by a slightly different averaging result. This shows that the e-SP ring does not incur latency in the original design, this is because of the slow changing astrocyte process. The e-SP changes very slowly and the e-SP ring successfully carries the e-SP to the synapses within this timescale.

Table 7.3 Average frequencies of different platforms under various fault densities (Hz)

		Average output frequency	
Fault Density	**Platform**	**Neuron 1**	**Neuron 2**
0%	Simulation	7.19	7.20
	Hardware	7.28	7.27
40%	Simulation	7.38	6.81
	Hardware	7.37	6.88
80%	Simulation	7.38	5.68
	Hardware	7.37	5.75

Table 7.4 Comparing average frequencies with and without the e-SP ring

		Average output frequency		Average output frequency with e-SP ring	
Fault Density	**Platform**	**Neuron 1**	**Neuron 2**	**Neuron 1**	**Neuron 2**
0%	Simulation	7.19	7.2	N/A	N/A
	Hardware	7.28	7.27	7.28	7.27
40%	Simulation	7.38	6.81	N/A	N/A
	Hardware	7.37	6.88	7.38	6.88
80%	Simulation	7.38	5.68	N/A	N/A
	Hardware	7.37	5.75	7.37	5.77

7.5 Fault Scenarios

There are a number of areas that a fault can occur which are outside of the current fault model:

- A fault affects the neuron, and not the synapse:

If there is a fault in the neuron, the network considers this a silent neuron, with all synapses connected to the neuron as faulty. This model used 2 neurons and 10 synapses, due to this small network, when the neuron fails the entire network fails. The astrocyte increases PR in healthy synapses but as there are no healthy synapses the model cannot recover and the robot stops completely. This is not a representative of a large network, this is because in the model there are simply two neurons.

- The astrocyte or astrocyte network routers fail:

This fault was addressed in section 7.3. Using a dedicated router within the neural network, has the potential to have faults. Within this network there are potential faults in the astrocyte or astrocyte router, this is part of the future work in realizing a large scale neuro-glia network. Due to the network having an e-SP signal in a single wire from the astrocyte to synapses, if the signal cannot be communicated, there is no self-repair in the network. However, repair is not only part of astrocyte to neuron interactions, but astrocytes to astrocyte communication would enable the network to detect changes. Using a small-scale network, provides a small scope, this is addressed in section 8.3, R*ealizing a large-scale neuro-glia network in hardware.* Future work focused on a larger scale would allow observation of faults within the astrocytes and the routers, as this work has focused on the astrocyte to neuron interaction.

- A “loud” synapse, i.e. one that constantly excites:

A “loud” synapse fault would be an increased initial PR within each faulty synapse, this means that the neuron would fire more quickly and the output frequency would be higher, and therefore the robot would move more quickly. This was tested as the PR was set high on many of the synapses, the expected output was to observe the network regulating and

adjusting the PR so these faulty synapses would reduce PR and the robot would stabilise. However, with the absence of DSE there was no way of reducing the PR. The e-SP signal works by increasing the PR, typically there is an equilibrium acting on the synapse. The e-SP increases PR and DSE decreases PR, they cancel each other out, when the neuron stops firing, the DSE to the synapse is reduced, the e-SP is increased and thus the PR increases. The hardware model does not use the DSE on the signal as a “loud” synapse wasn’t initially tested. Further work on the software model should be carried out and then the hardware can use the DSE signal to decrease the PR and repair.

- The sensors or motor drivers fail:

The motor was removed to simulate a fault. However, because the motor is controlled by the output frequency of the neuron, there was no feedback and the neuron continued to fire. This is expected, in this small scale set-up there are two neurons and two motors to control speed. Within this experiment, the damage to the synapses showed self-repair within the neural network but the motor is external, and controlled by the output of the network. An interesting aspect would be using a large scale network with sensor inputs. These sensor inputs would affect the output of the neural network. This is a long term aim of the neuro-glia network, it is employed in harsh environments and as faults occur to synapses, neurons or astrocytes or even further, faulty sensors the network will recover partially, and the overall aim is a graceful degradation of the SNN. Unfortunately this fault model cannot be simulated on such a small network, especially within this SNN as the motors have no influence on the network.

7.6 Summary

This chapter presented a robotic controller in FPGA hardware using the proposed NoC mechanism from chapter 5. A ring topology for e-SP was used within a SANN and results show that with up to 80% of faulty synapses the network can repair and restore pre-fault functionality. Results show that the area incurred by the e-SP ring is 1.68% of the total number of slice registers and 3.62% of slice registers relative to the astrocyte, this is a tiny overhead incurred in terms, of area. Table 7.4. Compares the output frequency of the neuron #1 and neuron #2 with and without the e-SP ring. This had shown there was little

to no difference in the output frequency caused by the e-SP ring. The results indicate that using a biological timeframe the e-SP ring can operate without inducing latency, as the average output frequency has very little change (+/- 0.2). This will be vital for future implementations of large-scale astrocyte networks. The fault models above were experimental and could be considered as part of future work, the objective of this work is to address faults at a local level within the SNN which is the focus of this thesis. It is however, interesting to view the model as a whole and consider various fault scenarios, upon realizing a large scale neuro-glia network.

Chapter 8: Conclusion and future work

8.1 Conclusion

The brain is a large complex system with an overwhelming number of tiny biological processes. Yet, it is the most complex and powerful computing paradigm, the most complex and powerful known to science. Whether it's processing huge amounts of information or solving complex problems incredibly quick, it is a highly adaptive system, constantly learning and powered by an area efficient and low power infrastructure. The brain is a dense parallel system consisting of a complex network of cells. The possibility of creating a complex system capable of the traits we observe in the human brain, is beyond the imagination, yet with each step we get a bit closer. Biological traits such as fine-grained repair and distributed repair-decision making are performed in the brain via astrocytes. Astrocytes are glial cells within the brain and have provided an insight into a more complete and complex paradigm. Astrocytes mediate synaptic plasticity; thus, astrocytes have the capability of dynamically increasing or decreasing the PR of a synapse, this facilitates self-repair. This neuro–glia network paradigm addresses the key self-repair requirements of fine granularity and distributed decision making which is a constraint of current self-repair techniques in hardware.

In particular, computational models of such repair have been successfully captured and applied to SNNs. As such, neuro-glia networks have been applied computationally and this paradigm has only begun to be explored in hardware. This PhD thesis provides insight into neuro-glia networks in hardware, providing a scalable interconnect to communicate local and global signals, exploring the astrocyte and its internal processes in hardware. The scope of the research which forms this PhD, focused on providing a scalable NoC interconnect for self-repair using SNNs on hardware. Chapter 5 implemented a low-level ring topology which communicated e-SP from astrocytes to neurons, which modulates the PR of synapses. The low-level ring allowed an astrocyte process to be implemented within a current SNN framework (H-NoC), which was never intended to have self-repair, this scalable design provides the roadworks for astrocyte to neuron communication. This allows astrocytes to facilitate repair in current SNN hardware. Chapter 6 focused on

providing a global astrocyte network with a scalable interconnect for global network communications. An astrocyte tile router was designed to exploit the inherent slow communication of astrocytes and allow clusters of up to eight astrocytes to communicate IP_3 in an efficient and scalable manner. This used low level hardware to average and distribute the global IP_3 when there were changes in one astrocyte. The router used a dynamic scheduler and token system to provide an efficient update process which balanced the number of requests and prioritised efficiency over throughput. These low-level scalable interconnect strategies were implemented to communicate important signals within a neuro-glia network, these are required for self-repair.

This thesis consisted of six chapters, each with a role to play in identifying the constraints of a neuro-glia network. Using NoC inspired interconnects to overcome the complex connectivity and scalable interconnect issues of a neuro-glia network. Each chapter is summarised below:

Chapter 2 provided a review of neural networks, astrocytes and self-repair. This explored neurons and astrocytes and their role in computational models and showed how biochemical reactions can be used within computational neural networks. As SNNs and their applications develop, a biologically inspired self-repair provides a more complete and deeper understanding of processes within the brain where astrocytes provide fine grained and distributed self-repair. When realising a neuro-glia network in hardware there are scalability issues, implementing a common networking paradigm NoC was identified as a scalable interconnect solution within the next chapter.

Chapter 3 reviewed and opened with the scalability issues within large scale neuro-glia networks and this chapter reviewed the application of the NoC paradigm as a hardware interconnect. NoCs have been used to provide large scale neural networks with scalability. These same techniques significantly reduce area and power overheads when compared to typical bus and SoC interconnects. NoCs are also highly effective at supporting high throughput and dense communication infrastructures such as SNNs. Due to the vast interconnectedness of a neuro-glia network, this presented an interconnect

problem (in terms of both power and area overheads). Therefore, using NoCs provided a solution to realize large networks and dense communication procedures. This approach has many benefits for neuro-glia networks:

- Provided a scalable interconnect
- Low area overhead implementations (compared to traditional topologies)
- Reduced complexity
- Flexibility in design and development of a network

A NoC is essentially parallel in nature, this allowed some room to reduce the overheads within the network itself. Using a hierarchical topology, vast numbers of PEs are connected within a network and therefore, this allowed space for development of an astrocyte network. NoCs are based on networking engineering principles, *"Route packets, not wires"*. The next chapter focused on how to implement a neuro-glia network using a NoC interconnect.

Chapter 4 Provided a review on fault methods and repair. Self-repair is a desirable trait for electronic devices and this chapter reviewed current methods of hardware redundancy and self-repair. To date, current self-repair or fault tolerant approaches come at large overheads. A neuro-glia network would allow a SNN to be used within a harsh environment, this would provide an application in mission critical areas which is capable of prolonging it's operational lifetime and will continue to operate even when it suffers faults such as losing sensors.

Chapter 5 presented how a NoC ring topology could be used to provide e-SP communication within a neuro-glia network. This interconnect was a low-level scalable ring topology which communicated e-SP from an astrocyte to neurons. This was achieved by using serial data to communicate the signal to all neurons within a single Node Facility of H-NoC without impeding the SNN and its typical spike communications. The chapter provided an overview of this self-repair process and applied this process in hardware. It provided an overview of the interactions and communication exchange between neurons and astrocytes at the local level of communication exchange. A neuro-glia network is vast

in size and had both neuron and astrocyte processes, with each of these, the number of neurons, astrocytes and communication signals grow. The complex nature of interfacing two completely different networks has complications. Using a ring topology provides a good trade-off between reducing area/wire overheads and a slower rate of communication provided by the astrocyte.

This novel NoC interconnect provided the astrocyte with a way to communicate with neurons and provided the signals necessary for self-repair. It is regarded as essential for a SNN to facilitate self-repair in parallel to the normal operation of a SNN as the SNN focuses on high throughput, as a result faults would be detected using a separate process which cloned the packet. Moreover, the developed low level interconnect supports local communication from a high level astrocyte to a low level neuron/group of neurons. NoC provided a scalable interconnect solution with minimal area overhead providing the communication which is capable of facilitating self-repair at a fine grained and distributed level. This contributes a low-level NoC ring topology for astrocyte to neuron communication.

Chapter 6 presented a NoC interconnect solution which addressed the global IP_3 communication signal in an astrocyte network. The IP_3 interaction, is a communication protocol within a neuro-glia network. This is an astrocyte to astrocyte interaction. The chapter explored the interactions and communication exchange between astrocytes as a network which included identifying the ratio of astrocytes to neurons in biology and replicating this in hardware. This is a high-level global communication exchange. The steps taken to implement an astrocyte router within an astrocyte network were outlined and the astrocyte router used a low-level communication protocol to average and distribute IP_3 within a neuro-glia network using groups of up to eight astrocytes. This work provided a scalable solution to the vast interconnect and communication paradigm. Results demonstrated that the astrocyte router provided a good trade-off between low area and power to overhead with a relatively low communication speed.

As the astrocyte network was developed in hardware, it required a low overhead scalable interconnect with a balance between speed and accuracy. Astrocytes are computationally expensive and therefore, demand a lot of resources on an FPGA platform. The use of a ring topology and the low overhead router in the NoC provided a reduced area/wire overhead and as astrocytes communicate at slow speeds in biological terms, the interconnect provided a slow communication speed which was biologically appropriate. This novel NoC interconnect, provided communication protocols which allows a significant number of astrocytes to exchange data with other astrocytes. Each astrocyte is interfaced with 10 neurons and each astrocyte tile router accommodates 8 astrocytes which allows 80 neurons per astrocyte tile facility. Moreover, the NoC interconnect provided a hardware building block for developing neuro-glia interconnect for self-repair strategies, aiming to provide a distributed and fine-grained self-repair using astrocytes in hardware.

Chapter 7 validated the local ring topology hardware using self-repair implemented for use with a mobile robot. The chapter reviewed the mobile robot and the SANN designed and developed with self-repair in mind. The ring topology from Chapter 5 was applied into the existing framework and this provided an e-SP interconnect to provide the neural network with self-repair. This is a real-world application of a SNN with self-repair. Results showed that if the synapses within the neural network are faulty, this SANN has the ability to repair, even with a potentially catastrophic rate of faults, that is, 80% of faulty synapses within the network.

8.2 Thesis Contributions

This thesis covered a substantial body of research into biological self-repair and realizing a computational model using NoC technology. Self-repair is a characteristic of the human brain and is provided by astrocyte cells. This thesis covered the communication protocols within an existing SNN framework, and combined the features of astrocytes with neural networks, more specifically SNNs and H-NoC. The resulting contributions are the first steps into realising large scale neuro-glia networks.

The contributions presented within this thesis are:

- A novel NoC interconnect for local communication in a neuro-glia network between neurons and astrocytes (Chapter 5).
- Detailed analysis of area footprint and scalability, in regard to keeping low overheads in hardware (Chapter 5).
- Applying this interconnect for local communication exchanges within an existing framework (H-NoC) (Chapter 5).
- A novel NoC router for global communication in a neuro-glia network between astrocytes (Chapter 6).
- Detailed analysis of area and power footprints and scalability, in regard to keeping low overheads in hardware (Chapter 6).
- Validating the NoC interconnect in hardware using an existing mobile robotic car (Chapter 7).

8.3 Future Work

This entire body of work has focused on providing a scalable interconnect for large scale neuro-glia networks. That is, providing a hardware interconnect to support hardware models of the astrocyte. Astrocyte networks consist of local and global communication signals which are essential for self-repair. However, it is still within the embryonic stages of development in hardware. Using NoC techniques and engineering, this thesis provides scalable solutions with low area and power overheads to help realize large scale networks.

This thesis has explored the astrocyte process, it has explored SNNs, mainly H-NoC, and how astrocytes may be applied into current SNN paradigms. The benefits of using NoCs has opened up the possibility of scalable communication protocols between astrocytes and neurons (inter-neuron) and astrocytes (inter-astrocyte) within a neuro-glia network. Extracting the most important signals and realizing the required protocols in hardware to exploit the slow innate astrocyte process, coupled with the faster SNN protocols, thereby

allows large scale neuro-glia networks to be explored further in the future. However, there is scope for future work within this body of work.

Understanding astrocytes: Within the fields of neurobiology, computational neuroscience and computational modelling, astrocytes are still within their embryonic stage of development, especially, in regard to biophysical models of astrocytes. Within neural networks, astrocytes open up the possibility of self-repair and more complex models of the brain, yet we still don't fully understand the relationships between neurons and astrocytes as the biological exploration of neural-glial interactions is still at a very early stage. However, each small progression in this understanding is another step in the right direction.

As our understanding of glial cells and astrocytes progresses, interconnecting astrocytes and neurons within hugely dense parallel networks reinforces the necessity for scalable hardware capable of supporting hugely complex systems. The astrocyte models produced will lead to more complex and simplistic models, however, unlike neurons where there have been accurate models from the 50's such as the Hodgkin and Huxley model [31], astrocyte models are still lacking and will need time to catch up. Eventually these new models will lead to a leap in computational models of repair.

Global astrocyte models need to be developed further. The interaction between astrocytes and neurons provides repair, however, global self-repair is less understood. These networks of astrocytes have complex properties and until these are completely understood, the models will be limited. The role of IP_3 and Ca^{2+} has been investigated on a small scale but realising this network in full is a challenge.

Applying self-repair within a working SNN: Chapter 5 uses aspects of H-NoC, using the Tile Facility, Node Facility and neurons to communicate typical SNN data whilst cloning the packets and using this as a stimulus for the astrocyte process. The biggest constraint was the absence of dynamic synapses, this is both excitatory and inhibitory and works independently, as the PR on the synapses was hard coded. This was based

on the original H-NoC architecture. The PR values and weights had been based on the values used in computational models. The synapse PR could be increased but would have no effect on the firing frequency of the neurons within H-NoC. This model of the synapses could not be changed and therefore, the PR could not be increased. Therefore, in order to change H-NoC, it would need a newer model. A SNN with variable synapses would need to be integrated with an astrocyte in order to facilitate self-repair.

Developing H-NoC with physical synapses and testing the ability of self-repair, is a long way off. The first step is to reduce the astrocyte and observe the role of interaction between H-NoCs neurons and the astrocyte. Progress has been shown in Chapter 5, however, this work requires implementing the ring topology and applying the mediation of PR in dynamic synapses, this would enable a neuro-glia network to adapt and repair to faults as this process occurs in the synapses.

Back off algorithm to balance traffic and throughput: The exponential back off algorithm is a networking paradigm and an IEEE 802.11 standard based on ethernet protocols [134]. This algorithm uses feedback to overload a system with data and then decrease the rate of this process, in order to find an acceptable rate. This approach generally finds the cross over point between throughput and latency. Chapter 6 describes an astrocyte router which router waits for tokens to begin the IP_3 averaging process. An ideal scenario is to use a full-scale model of astrocytes and begin overloading the astrocyte router with as much data as possible. The rate is then reduced, and the rate of which it operates with maximal throughput and minimal latency would be the ideal number of tokens, the system can handle. This work would require a lot more hardware to complete, and in the future, either increasing the hardware or reducing the astrocyte process can make this a viable stress testing method.

Realizing a large-scale neuro-glia network in hardware: Using NoC interconnects has provided a scalable and efficient solution to realizing neuro-glia networks in hardware. A neuro-glia network consists of two layers, an astrocyte network and a neural network. NoC interconnects can provide hardware to bring together the self-repair and SNN

paradigms of the respective networks in hardware, yet this is still a long way off. A large scale neural network is still the main focus of researchers developing computational models such as the Blue Brain Project [26]. There has been research within IBM which is "the culmination of almost a decade of research and development". From 2011 there has been an increase in programmable neurons (up to1 million) and programmable synapses (256 million). The ultimate goal, according to IBM, is to build a system with ten billion neurons and one hundred trillion synapses, all while consuming only one kilowatt of power [72]. A large scale neuro-glia network is limited by the scalability of the astrocyte. The astrocyte is a complex process that demands a large device utilisation, and this makes it difficult to simulate or realize multiple astrocytes on hardware. The number of astrocytes in a small neuro-glia network would be around eight, this would require 4-8 FPGA boards, and this means that the boards would also have to use physical wired connections to communicate. For instance, using one FPGA for two astrocytes, and thus, four FPGAs to realise eight astrocytes, this requires additional hardware as the astrocytes and their respective neural networks and digital interconnects would be spread amongst a number of boards which communicate using standardized on board communications to communicate between boards, reducing the analysis of the digital NoC interconnect.

Addressing drift: Drift is typical in neural networks where the training has been offline. Offline training is typically preferred as online training is computationally intensive and takes times, the network can be trained offline using powerful software/hardware and the network can be applied to hardware and it can classify in real time. This has problems as the training of the network can have a bias, that is, it has no further learning enabled and so, even though something might slightly change with the application, the network has been trained based on input and output relationships, and may classify incorrectly [135]. There is scope within a neuroglia network to address drift. Astrocytes are capable of adjusting the PR of a synapse, although this body of work focuses on repair, it could be possible to employ a method of online training, if the output of the classifier is incorrectly classified, the network may be able to affect the PR of synapses and adjust the output.

These neural network models are focused on building large scale neural networks, which focus on using neurons and harnessing the processing power of neurons by mimicking the parallel nature of neurons, yet these neural models ignore the majority of signalling processes within the brain. As the astrocyte has emerged, the complexity of neuro-glia networks has exponentially increased, in order to fully understand how the brain processes information and facilitates repair, neural models should focus on producing a large-scale neuro-glia network with both astrocyte and neuron cells, each network with its own individual complexity. Combining these networks is at the time of writing, seemingly impossible. This work would require millions of astrocytes, billions of neurons and trillions of synapses. In order to realize these networks, it would require considerable work on exploring the role of astrocytes in biology and reducing the astrocyte in hardware whilst maintaining precision. From there creating a small astrocyte network and implementing this within a small scale fully functioning neural network, which would identify how these networks interact in hardware and how these processes may be optimised. This would allow a small-scale neuro-glia network with both the capabilities of complex information processing from the neurons and the self-repair capabilities of the astrocyte network with all the interactions between. It would then be paramount to scale this network combining clusters of astrocytes and using more than one small scale neural network. This work would lead to realizing large scale neuro-glia networks and thus, having a more complete understanding of how the brain works, repairs, combines networks in a scalable and efficient manner and not solely focus on the abstraction of how the brain processes information, as this completely ignores the complexity of astrocytes and their role within the brain.

Bibliography

[1] D. Ratter, “FPGAs on Mars,” *Xcell J.*, 2004.

[2] NASA, “Juno Mission to Jupiter,” p. 2, 2015.

[3] F. Y. H. Ahmed, B. Yusob, and H. N. A. Hamed, “Computing with spiking neuron networks a review,” *Int. J. Adv. Soft Comput. its Appl.*, vol. 6, no. 1, pp. 1–21, 2014.

[4] R. P. Lippmann, “Speech recognition by machines and humans,” *Speech Commun.*, vol. 22, no. 1, pp. 1–15, 1997.

[5] P. Sinha, B. Balas, Y. Ostrovsky, and R. Russell, “Face recognition by humans: Nineteen results all computer vision researchers should know about,” *Proc. IEEE*, vol. 94, no. 11, pp. 1948–1961, 2006.

[6] A. Tavanaei and A. S. Maida, “A spiking network that learns to extract spike signatures from speech signals,” *Neurocomputing*, vol. 240, pp. 191–199, 2017.

[7] J. T. Jose, J. Amudha, and G. Sanjay, “A Survey on Spiking Neural Networks in Image Processing,” in *Advances in Intelligent Informatics*, 2015, pp. 107–115.

[8] J. Harkin, F. Morgan, L. McDaid, S. Hall, B. McGinley, and S. Cawley, “A Reconfigurable and Biologically Inspired Paradigm for Computation Using Network-On-Chip and Spiking Neural Networks,” *Int. J. Reconfigurable Comput.*, vol. 2009, pp. 1–13, 2009.

[9] R. Isserman, *Fault -Diagnosis Systems: An Introduction from Fault Detection to Fault Tolerance*. Springer Science & Business Media, 2006.

[10] K. Kyriakoulakos and D. Pnevmatikatos, “A novel SRAM-based FPGA architecture for efficient TMR fault tolerance support,” *FPL 09 19th Int. Conf. F. Program. Log. Appl.*, pp. 193–198, 2009.

[11] J. J. Wade, L. J. McDaid, J. Harkin, V. Crunelli, J. A. S. Kelso, and V. Beiu, "Exploring retrograde signaling via astrocytes as a mechanism for self repair," *Proc. Int. Jt. Conf. Neural Networks*, pp. 3149–3155, Jul. 2011.

[12] J. Wade, L. McDaid, J. Harkin, V. Crunelli, and S. Kelso, "Self-repair in a bidirectionally coupled astrocyte-neuron (AN) system based on retrograde signaling.," *Front. Comput. Neurosci.*, vol. 6, no. September, p. 76, Jan. 2012.

[13] C. S. von Bartheld, J. Bahney, and S. Herculano-Houzel, "The search for true numbers of neurons and glial cells in the human brain: A review of 150 years of cell counting," *J. Comp. Neurol.*, vol. 524, no. 18, pp. 3865–3895, Dec. 2016.

[14] M. De Pittà, N. Brunel, and A. Volterra, "Astrocytes: Orchestrating synaptic plasticity?," *Neuroscience*, vol. 323. pp. 43–61, 2016.

[15] M. Naeem, L. J. McDaid, J. Harkin, J. J. Wade, and J. Marsland, "On the Role of Astroglial Syncytia in Self-Repairing Spiking Neural Networks," *IEEE Trans. Neural Networks Learn. Syst.*, vol. 26, no. 10, pp. 2370–2380, 2015.

[16] L. Benini and G. De Micheli, "Networks on chips: A new SoC paradigm," *Computer (Long. Beach. Calif).*, vol. 35, no. 1, pp. 70–78, 2002.

[17] W. J. Dally and B. Towles, "Route packets, not wires: on-chip interconnection networks," *Proc. 38th Des. Autom. Conf.*, pp. 684–689, 2001.

[18] A. Hemani, A. Jantsch, S. Kumar, A. Postula, J. Öberg, M. Millberg, and D. Lindqvist, "Network on a Chip: An architecture for billion transistor era," *Proc. Norchip - 2000*, pp. 166–173, 2000.

[19] J. Downer, E. C. for A. of R. and Regulation, and L. S. of E. and P. Science, *When Failure is an Option: Redundancy, Reliability and Regulation in Complex Technical Systems*.

CARR, 2009.

[20] S. Herculano-Houzel, “The human brain in numbers: a linearly scaled-up primate brain,” *Front. Hum. Neurosci.*, vol. 3, p. 31, 2009.

[21] D. P. Pelvig, H. Pakkenberg, A. K. Stark, and B. Pakkenberg, “Neocortical glial cell numbers in human brains,” *Neurobiol. Aging*, vol. 29, no. 11, pp. 1754–1762, 2008.

[22] S. Achard and E. Bullmore, “Efficiency and Cost of Economical Brain Functional Networks,” *PLOS Comput. Biol.*, vol. 3, no. 2, pp. 1–10, 2007.

[23] W. Gerstner and W. M. Kistler, *Spiking Neuron Models: Single Neurons, Populations, Plasticity*. Cambridge University Press, 2002.

[24] J. J. Hopfield, “Neural networks and physical systems with emergent collective computational abilities.,” *Proc. Natl. Acad. Sci.*, vol. 79, no. 8, pp. 2554–2558, 1982.

[25] M. P. Mattson, “Superior pattern processing is the essence of the evolved human brain,” *Frontiers in Neuroscience*, vol. 8. 2014.

[26] H. Markram, “The blue brain project.,” *Nat. Rev. Neurosci.*, vol. 7, no. 2, pp. 153–60, 2006.

[27] F. Rieke, *Spikes: Exploring the Neural Code*. MIT Press, 1999.

[28] W. Maass, “Networks of spiking neurons: The third generation of neural network models,” *Neural Networks*, vol. 10, no. 9, pp. 1659–1671, 1997.

[29] T. T. & M. H. Report, “Santiago Ramon Y Cajal’s Neuron.” [Online]. Available: http://trauma.blog.yorku.ca/2013/02/santiago-ramon-y-cajals-neuron/neuron/. [Accessed: 08-Aug-2017].

[30] Memrise, “Synaptic Vesicles.” [Online]. Available: https://www.memrise.com/user/ziming-learning/?page=11. [Accessed: 08-Aug-2017].

[31] A. L. Hodgkin and A. F. Huxley, “Currents carried by sodium and potassium ions through the membrane of the giant axon of Loligo,” *J. Physiol.*, vol. 116, no. 4, pp. 449–472, 1952.

[32] R. F. Thompson, *The Brain: A Neuroscience Primer*. Worth Publishers, 2000.

[33] Wikipedia, “Action Potential.” [Online]. Available: https://en.wikipedia.org/wiki/Action_potential. [Accessed: 06-Aug-2018].

[34] T. project Spot, “Introduction to Artificial Neural Networks.” [Online]. Available: http://www.theprojectspot.com/tutorial-post/introduction-to-artificial-neural-networks-part-1/7. [Accessed: 01-Jan-2017].

[35] W. S. McCulloch and W. Pitts, “A logical calculus of the ideas immanent in nervous activity,” *Bull. Math. Biophys.*, vol. 5, no. 4, pp. 115–133, Dec. 1943.

[36] D. O. Hebb, *The Organization of Behavior: A Neuropsychological Theory*. Taylor & Francis, 2002.

[37] F. Rosenblatt, “The perceptron: A probabilistic model for information storage and organization in …,” *Psychol. Rev.*, vol. 65, no. 6, pp. 386–408, 1958.

[38] M. L. Minsky and S. A. Papert, *Perceptrons: Expanded Edition*. Cambridge, MA, USA: MIT Press, 1988.

[39] E. M. Izhikevich, “Simple model of spiking neurons,” *IEEE Trans. Neural Networks*, vol. 14, no. 6, pp. 1569–1572, 2003.

[40] S. Ghosh-Dastidar and H. Adeli, “Third Generation Neural Networks: Spiking Neural Networks,” in *Advances in Computational Intelligence*, W. Yu and E. N. Sanchez, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 167–178.

[41] D. C. Tam, “Decoding of firing intervals in a temporal-coded spike train using a

topographically mapped neural network," *Neural Networks, 1990., 1990 IJCNN Int. Jt. Conf.*, pp. 627–632, 2002.

[42] T. P. Trappenberg, *Fundamentals of Computational Neuroscience*. Oxfort U. Press, 2002.

[43] E. Kandel, *Principles of Neural Science, Fifth Edition*. McGraw-Hill Education, 2013.

[44] J. Šíma and P. Orponen, "General-Purpose Computation with Neural Networks: A Survey of Complexity Theoretic Results," *Neural Comput.*, vol. 15, no. 12, pp. 2727–2778, Dec. 2003.

[45] E. M. Izhikevich, "Which model to use for cortical spiking neurons?," *IEEE Trans. Neural Networks*, vol. 15, no. 5, pp. 1063–1070, 2004.

[46] Wikipedia, "Hodgkin-Huxley model." [Online]. Available: https://en.wikipedia.org/wiki/Hodgkin–Huxley_model. [Accessed: 01-Jan-2017].

[47] J. Gratwicke, M. Jahanshahi, and T. Foltynie, "Parkinson's disease dementia: a neural networks perspective," *Brain*, vol. 138, no. 6, pp. 1454–1476, 2015.

[48] A. Esteva, B. Kuprel, R. A. Novoa, J. Ko, S. M. Swetter, H. M. Blau, and S. Thrun, "Dermatologist-level classification of skin cancer with deep neural networks," *Nature*, vol. 542, no. 7639, pp. 115–118, Feb. 2017.

[49] J. J. Wade, L. J. McDaid, J. A. Santos, and H. M. Sayers, "SWAT: A spiking neural network training algorithm for classification problems," *IEEE Trans. Neural Networks*, vol. 21, no. 11, pp. 1817–1830, 2010.

[50] D. B. Thomas and W. Luk, "FPGA accelerated simulation of biologically plausible spiking neural networks," *Proc. - IEEE Symp. F. Program. Cust. Comput. Mach. FCCM 2009*, pp. 45–52, 2009.

[51] J. Misra and I. Saha, "Artificial neural networks in hardware: A survey of two decades of

progress," *Neurocomputing*, vol. 74, no. 1–3, pp. 239–255, Dec. 2010.

[52] F. M. Dias, A. Antunes, and A. M. Mota, "Artificial neural networks: A review of commercial hardware," *Eng. Appl. Artif. Intell.*, vol. 17, no. 8, pp. 945–952, 2004.

[53] Intel, "Intel® Core™ i9-7980Xe X-series Processor," 2018. [Online]. Available: https://www.intel.co.uk/content/www/uk/en/products/processors/core/x-series/i9-7980xe.html. [Accessed: 06-Aug-2018].

[54] A. K. Fidjeland, E. B. Roesch, M. P. Shanahan, and W. Luk, "NeMo: A platform for neural modelling of spiking neurons using GPUs," *Proc. Int. Conf. Appl. Syst. Archit. Process.*, pp. 137–144, 2009.

[55] D. Strigl, K. Kofler, and S. Podlipnig, "Performance and scalability of GPU-based convolutional neural networks," *Proc. 18th Euromicro Conf. Parallel, Distrib. Network-Based Process. PDP 2010*, pp. 317–324, 2010.

[56] R. V. Hoang, D. Tanna, L. C. Jayet Bray, S. M. Dascalu, and F. C. Harris, "A novel CPU/GPU simulation environment for large-scale biologically realistic neural modeling," *Front. Neuroinform.*, vol. 7, no. October, pp. 1–10, 2013.

[57] K. S. Oh and K. Jung, "GPU implementation of neural networks," *Pattern Recognit.*, vol. 37, no. 6, pp. 1311–1314, 2004.

[58] A. Coates, B. Huval, T. Wang, D. Wu, and A. Y. Ng, "Deep learning with COTS HPC systems," *Proc. 30th Int. Conf. Mach. Learn.*, pp. 1337–1345, 2013.

[59] S. R. Young, D. C. Rose, T. P. Karnowski, S.-H. Lim, and R. M. Patton, "Optimizing deep learning hyper-parameters through an evolutionary algorithm," *Proc. Work. Mach. Learn. High-Performance Comput. Environ. - MLHPC '15*, pp. 1–5, 2015.

[60] Y. Yu, J. Jiang, and X. Chi, "Using supercomputer to speed up neural network training,"

Proc. Int. Conf. Parallel Distrib. Syst. - ICPADS, pp. 942–947, 2017.

[61] H. de Garis, C. Shuo, B. Goertzel, and L. Ruiting, "A world survey of artificial brain projects, Part I: Large-scale brain simulations," *Neurocomputing*, vol. 74, no. 1–3, pp. 3–29, 2010.

[62] J. Schemmel, J. Fieres, and K. Meier, "Wafer-scale integration of analog neural networks," *Proc. Int. Jt. Conf. Neural Networks*, pp. 431–438, Jun. 2008.

[63] S. Cawley, F. Morgan, B. McGinley, S. Pande, L. McDaid, S. Carrillo, and J. Harkin, "Hardware spiking neural network prototyping and application," *Genet. Program. Evolvable Mach.*, vol. 12, no. 3, pp. 257–280, Apr. 2011.

[64] E. Painkras, L. A. Plana, J. Garside, S. Temple, F. Galluppi, C. Patterson, D. R. Lester, A. D. Brown, and S. B. Furber, "SpiNNaker: A 1-W 18-core system-on-chip for massively-parallel neural network simulation," *IEEE J. Solid-State Circuits*, vol. 48, no. 8, pp. 1943–1953, 2013.

[65] B. V. Benjamin, P. Gao, E. McQuinn, S. Choudhary, A. R. Chandrasekaran, J. M. 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," *Proc. IEEE*, vol. 102, no. 5, pp. 699–716, May 2014.

[66] CSCS, "Blue Brain 4." [Online]. Available: https://www.cscs.ch/computers/dismissed/blue-brain-4/. [Accessed: 06-Aug-2018].

[67] CSCS, "Blue Brain 5." .

[68] J. Zhu and P. Sutton, "FPGA Implementations of Neural Networks -- A Survey of a Decade of Progress," in *Field Programmable Logic and Application*, 2003, pp. 1062–1066.

[69] J. L. J. Liu and D. L. D. Liang, “A Survey of FPGA-Based Hardware Implementation of ANNs,” *2005 Int. Conf. Neural Networks Brain*, vol. 2, pp. 915–918, 2005.

[70] L. P. Maguire, T. M. McGinnity, B. Glackin, A. Ghani, A. Belatreche, and J. Harkin, “Challenges for large-scale implementations of spiking neural networks on FPGAs,” *Neurocomputing*, vol. 71, no. 1–3, pp. 13–29, 2007.

[71] F. Morgan, S. Cawley, B. Mc Ginley, S. Pande, L. J. Mc Daid, B. Glackin, J. Maher, and J. Harkin, “Exploring the evolution of NoC-based spiking neural networks on FPGAs,” *Proc. 2009 Int. Conf. Field-Programmable Technol. FPT’09*, pp. 300–303, Dec. 2009.

[72] IBM, “Brain Power.” [Online]. Available: http://www.research.ibm.com/cognitive-computing/brainpower/.

[73] R. Ananthanarayanan, S. K. Esser, H. D. Simon, and D. S. Modha, “The cat is out of the bag,” *Proc. Conf. High Perform. Comput. Networking, Storage Anal. - SC ’09*, no. c, p. 1, 2009.

[74] R. Preissl, T. M. Wong, P. Datta, M. Flickner, R. Singh, S. K. Esser, W. P. Risk, H. D. Simon, and D. S. Modha, “Compass: A scalable simulator for an architecture for cognitive computing,” *Int. Conf. High Perform. Comput. Networking, Storage Anal. SC*, 2012.

[75] T. M. Wong, R. Preissl, P. Datta, M. D. Flickner, R. Singh, S. K. Esser, E. McQuinn, R. Appuswamy, W. P. Risk, H. D. Simon, and D. S. Modha, “IBM Research Report 10^14,” *IBM Res. Rep. 10^14*, vol. 10502, pp. 1–3, 2012.

[76] N. Imam, T. Cleland, R. Manohar, P. Merolla, J. Arthur, F. Akopyan, and D. Modha, “Implementation of Olfactory Bulb Glomerular-Layer Computations in a Digital Neurosynaptic Core,” *Front. Neurosci.*, vol. 6, p. 83, 2012.

[77] P. A. Merolla, J. V. Arthur, R. Alvarez-Icaza, A. S. Cassidy, J. Sawada, F. Akopyan, B. L.

Jackson, N. Imam, C. Guo, Y. Nakamura, B. Brezzo, I. Vo, S. K. Esser, R. Appuswamy, B. Taba, A. Amir, M. D. Flickner, W. P. Risk, R. Manohar, and D. S. Modha, "A million spiking-neuron integrated circuit with a scalable communication network and interface," *Science (80-.).*, vol. 345, no. 6197, 2014.

[78] S. Choudhary, S. Sloan, S. Fok, A. Neckar, E. Trautmann, P. Gao, T. Stewart, C. Eliasmith, and K. Boahen, "Silicon neurons that compute," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 7552 LNCS, no. PART 1, pp. 121–128, 2012.

[79] S. Carrillo, J. Harkin, L. J. McDaid, F. Morgan, S. Pande, S. Cawley, and B. McGinley, "Scalable hierarchical network-on-chip architecture for spiking neural network hardware implementations," *IEEE Trans. Parallel Distrib. Syst.*, vol. 24, no. 12, pp. 2451–2461, 2013.

[80] M. Navarrete and A. Araque, "Endocannabinoids potentiate synaptic transmission through stimulation of astrocytes," *Neuron*, vol. 68, no. 1, pp. 113–126, Oct. 2010.

[81] J. J. Wade, L. J. McDaid, J. Harkin, V. Crunelli, and J. A. S. Kelso, "Bidirectional coupling between astrocytes and neurons mediates learning and dynamic coordination in the brain: A multiple modeling approach," *PLoS One*, vol. 6, no. 12, p. e29445, Jan. 2011.

[82] S. Nazari, K. Faez, M. Amiri, and E. Karami, "A novel digital implementation of neuron???astrocyte interactions," *J. Comput. Electron.*, Nov. 2014.

[83] H. Soleimani, M. Bavandpour, A. Ahmadi, and D. Abbott, "Digital implementation of a biological astrocyte model and its application," *IEEE Trans. Neural Networks Learn. Syst.*, vol. 26, no. 1, pp. 127–139, Jan. 2015.

[84] J. Liu, L. J. Mcdaid, J. Harkin, S. Karim, A. P. Johnson, A. G. Millard, J. Hilder, D. M.

Halliday, A. M. Tyrrell, S. Member, J. Timmis, and S. Member, “Exploring Self-Repair in a Coupled Spiking Astrocyte Neural Network,” *IEEE Trans. Neural Networks Learn. Syst.*, vol. PP, pp. 1–11, 2018.

[85] J. Liu, J. Harkin, L. Maguire, L. McDaid, J. Wade, and M. McElholm, “Self-repairing hardware with astrocyte-neuron networks,” *Proc. - IEEE Int. Symp. Circuits Syst.*, vol. 2016–July, pp. 1350–1353, 2016.

[86] J. Liu, J. Harkin, L. McDaid, D. M. Halliday, A. M. Tyrrell, and J. Timmis, “Self-repairing mobile robotic car using astrocyte-neuron networks,” *2016 Int. Jt. Conf. Neural Networks*, pp. 1379–1386, 2016.

[87] J. Liu, J. Harkin, L. McDaid, and G. Martin, “Hierarchical networks-on-chip interconnect for astrocyte-neuron network hardware,” in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2016, vol. 9886 LNCS, pp. 382–390.

[88] J. Flich and D. Bertozzi, *Designing Network On-Chip Architectures in the Nanoscale Era*. Chapman & Hall/CRC, 2010.

[89] N. E. Jerger and L.-S. Peh, *On-Chip Networks*, vol. 4, no. 1. Morgan & Claypool, 2009.

[90] L. Carrillo Snaider, “Scalable Hierarchical Networks-on-Chip Architecture for Brain-Inspired Computing,” 2013.

[91] L. N. J. Duato, S. Yalamanchili, *Interconnection Networks: An Engineering Approach*, no. Morgan Kaufmann. 2002.

[92] I. Stojmenovic, “Honeycomb networks: Topological properties and communication algorithms,” *IEEE Trans. Parallel Distrib. Syst.*, vol. 8, no. 10, pp. 1036–1042, 1997.

[93] U. K. Lee, K. I. Kang, and G. H. Kim, “Mass concrete curing management based on

ubiquitous computing," *Comput. Civ. Infrastruct. Eng.*, vol. 21, no. 2, pp. 148–155, 2006.

[94] M. Abd-El-Barr and T. F. Al-Somani, "Topological properties of hierarchical interconnection networks: A review and comparison," *J. Electr. Comput. Eng.*, vol. 2011, 2011.

[95] R. Das, S. Eachempati, A. K. Mishra, V. Narayanan, and C. R. Das, "Design and evaluation of a hierarchical on-chip interconnect for next-generation CMPs," *Proc. - Int. Symp. High-Performance Comput. Archit.*, pp. 175–186, 2009.

[96] J. Y. Kim, J. Park, S. Lee, M. Kim, J. Oh, and H. J. Yoo, "A 118.4 GB/s multi-casting network-on-chip with hierarchical star-ring combined topology for real-time object recognition," *IEEE J. Solid-State Circuits*, vol. 45, no. 7, pp. 1399–1409, 2010.

[97] Z. Lu and A. Jantsch, "Trends of terascale computing chips in the next ten years," *ASICON 2009 - Proc. 2009 8th IEEE Int. Conf. ASIC*, pp. 62–66, 2009.

[98] V. F. Pavlidis and E. G. Friedman, "3-D topologies for Networks-on-Chip," *2006 IEEE Int. Syst. Conf. SOC*, pp. 285–288, 2007.

[99] B. S. Feero and P. P. Pande, "Networks-on-chip in a three-dimensional environment: A performance evaluation," *IEEE Trans. Comput.*, vol. 58, no. 1, pp. 32–45, 2009.

[100] M. Ebrahimi, M. Daneshtalab, and J. Plosila, "High performance fault-tolerant routing algorithm for NoC-based many-core systems," *Proc. 2013 21st Euromicro Int. Conf. Parallel, Distrib. Network-Based Process. PDP 2013*, pp. 462–469, 2013.

[101] E. J. Chang, H. K. Hsin, S. Y. Lin, and A. Y. Wu, "Path-congestion-aware adaptive routing with a contention prediction scheme for network-on-chip systems," *IEEE Trans. Comput. Des. Integr. Circuits Syst.*, vol. 33, no. 1, pp. 113–126, 2014.

[102] S. Carrillo, J. Harkin, L. McDaid, S. Pande, S. Cawley, and F. Morgan, "Adaptive routing

strategies for large scale spiking neural network hardware implementations," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 6791 LNCS, no. PART 1, pp. 77–84, 2011.

[103] K. C. Hale, B. Grot, and S. W. Keckler, "Segment gating for static energy reduction in networks-on-chip," *2009 2nd Int. Work. Netw. Chip Archit.*, pp. 57–62, 2009.

[104] F. G. De Lima, Cota, L. Carro, M. Lubaszewski, R. Reis, R. Velazco, and S. Rezgui, "Designing a radiation hardened 8051-like micro-controller," *Proc. - 13th Symp. Integr. Circuits Syst. Des.*, pp. 255–260, 2000.

[105] S. Zhang and H. Liu, "Synthetical analysis on space radiation tolerance techniques in ASICs and FPGAs," *2011 Int. Conf. Syst. Sci. Eng. Des. Manuf. Informatiz. ICSEM 2011*, vol. 2, pp. 305–310, 2011.

[106] P. Kabisatpathy, A. Barua, and S. Sinha, *Fault diagnosis of analog integrated circuits*, vol. 30. 2005.

[107] K. N. Dang, M. Meyer, Y. Okuyama, and A. Ben Abdallah, "A low-overhead soft--hard fault-tolerant architecture, design and management scheme for reliable high-performance many-core 3D-NoC systems," *J. Supercomput.*, vol. 73, no. 6, pp. 2705–2729, Jun. 2017.

[108] S. Murali, T. Theocharides, N. Vijaykrishnan, M. J. Irwin, L. Benini, and G. De Micheli, "Analysis of error recovery schemes for networks on chips," *IEEE Des. Test Comput.*, vol. 22, no. 5, pp. 434–442, 2005.

[109] S. Mitra, W.-J. Huang, N. R. Saxena, S.-Y. Yu, and E. J. Mccluskey, "Reconfigurable Architecture for Autonomous Self-Repair," *IEEE Des. Test Comput.*, vol. 21, no. 3, pp. 228–240, 2004.

[110] J. Liu, J. Harkin, Y. Li, and L. Maguire, "Online fault detection for Networks-on-Chip

interconnect," *Proc. 2014 NASA/ESA Conf. Adapt. Hardw. Syst. AHS 2014*, pp. 31–38, 2014.

[111] C. Grecu, A. Ivanov, R. Saleh, E. S. Sogomonyan, and P. P. Pande, "On-line Fault Detection and Location for NoC Interconnects," *Int. On-Line Test. Symp.*, pp. 145–150, 2006.

[112] K. Reick, P. N. Sanda, S. Swaney, J. W. Kellington, M. Mack, M. Floyd, and D. Henderson, "Fault-tolerant design of the IBM Power6 microprocessor," *IEEE Micro*, vol. 28, no. 2, pp. 30–38, 2008.

[113] M. Hervé, P. Almeida, F. L. Kastensmidt, É. Cota, and M. Lubaszewski, "Concurrent test of network-on-chip interconnects and routers," *LATW2010 - 11th Latin-American Test Work.*, 2010.

[114] a. Alaghi, N. Karimi, M. Sedghi, and Z. Navabi, "Online NoC Switch Fault Detection and Diagnosis Using a High Level Fault Model," *22nd IEEE Int. Symp. Defect Fault-Tolerance VLSI Syst. (DFT 2007)*, pp. 21–29, 2007.

[115] W. Barker, D. M. Halliday, Y. Thoma, E. Sanchez, G. Tempesti, and A. M. Tyrrell, "Fault tolerance using dynamic reconfiguration on the POEtic tissue," *IEEE Trans. Evol. Comput.*, vol. 11, no. 5, pp. 666–684, 2007.

[116] M. E. Eapen, C. Pradeep, A. A. Varghese, and J. M. Nair, "Built-in Self Repair Approach by Module Relocation for FPGA Based Reconfigurable Systems," *Procedia Technol.*, vol. 24, pp. 1587–1594, 2016.

[117] S. Karim, J. Harkin, L. McDaid, B. Gardiner, J. Liu, D. M. Halliday, A. M. Tyrrell, J. Timmis, A. Millard, and A. Johnson, "Assessing Self-Repair on FPGAs with Biologically Realistic Astrocyte-Neuron Networks," *Proc. IEEE Comput. Soc. Annu. Symp. VLSI,*

ISVLSI, vol. 2017–July, pp. 421–426, 2017.

[118] A. P. Johnson, J. Liu, A. G. Millard, S. Karim, A. M. Tyrrell, J. Harkin, J. Timmis, L. J. McDaid, and D. M. Halliday, "Homeostatic fault tolerance in spiking neural networks: A dynamic hardware perspective," *IEEE Trans. Circuits Syst. I Regul. Pap.*, vol. 65, no. 2, pp. 687–699, 2018.

[119] G. Biot-Marí, J. Liu, Q. Fu, L. McDaid, J. Martínez-Corral, J. Harkin, Q. Lu, and Y. Luo, "Forest fire detection using spiking neural networks," pp. 371–375, 2018.

[120] Y. Irizarry-Valle and A. C. Parker, "Astrocyte on neuronal phase synchrony in CMOS," *Proc. - IEEE Int. Symp. Circuits Syst.*, pp. 261–264, 2014.

[121] Y. Irizarry-Valle and A. C. Parker, "An astrocyte neuromorphic circuit that influences neuronal phase synchrony," *IEEE Trans. Biomed. Circuits Syst.*, vol. 9, no. 2, pp. 175–187, 2015.

[122] B. A. Abed, A. Ismail, and N. A. Aziz, "Real time astrocyte in spiking neural network," *SAI Intell. Syst. Conf. 2015*, pp. 565–570, 2015.

[123] M. Hayati, M. Nouri, S. Haghiri, and D. Abbott, "A Digital Realization of Astrocyte and Neural Glial Interactions," *IEEE Trans. Biomed. Circuits Syst.*, vol. 10, no. 2, pp. 518–529, 2016.

[124] J. Liu, J. Harkin, L. P. Maguire, L. J. Mcdaid, and J. J. Wade, "SPANNER : A Self - Repairing Spiking Neural Network Hardware Architecture," vol. 29, no. 4, pp. 1287–1300, 2018.

[125] S. Pande, F. Morgan, G. Smit, T. Bruintjes, J. Rutgers, B. McGinley, S. Cawley, J. Harkin, and L. McDaid, "Fixed latency on-chip interconnect for hardware spiking neural network architectures," *Parallel Comput.*, vol. 39, no. 9, pp. 357–371, 2013.

[126] J. Liu, J. Harkin, L. P. Maguire, L. J. McDaid, J. J. Wade, and G. Martin, "Scalable Networks-on-Chip Interconnected Architecture for Astrocyte-Neuron Networks," *IEEE Trans. Circuits Syst. I Regul. Pap.*, vol. 63, no. 12, pp. 2290–2303, 2016.

[127] G. Martin, J. Harkin, L. J. McDaid, J. J. Wade, and J. Liu, "On-chip communication for neuro-glia networks," *IET Comput. Digit. Tech.*, vol. 12, no. 4, pp. 130–138, 2018.

[128] O. Sandberg, "Searching in a Small World," *Div. Math. Stat. Dep. Math. Sci.*, vol. Licentiate, p. 78, 2005.

[129] O. Sporns, D. R. Chialvo, M. Kaiser, and C. C. Hilgetag, "Organization, development and function of complex brain networks," *Trends Cogn. Sci.*, vol. 8, no. 9, pp. 418–425, 2004.

[130] G. Martin, J. Harkin, L. J. McDaid, J. J. Wade, J. Liu, and F. Morgan, "Astrocyte to spiking neuron communication using Networks-on-Chip ring topology," in *2016 IEEE Symposium Series on Computational Intelligence, SSCI 2016*, 2016.

[131] M. M. Halassa, T. Fellin, H. Takano, J.-H. Dong, and P. G. Haydon, "Synaptic Islands Defined by the Territory of a Single Astrocyte," *J. Neurosci.*, vol. 27, no. 24, p. 6473 LP-6477, Jun. 2007.

[132] M. De Pittà, V. Volman, H. Levine, and E. Ben-Jacob, "Multimodal encoding in a simplified model of intracellular calcium signaling," *Cogn. Process.*, vol. 10, no. 1, p. 55, Nov. 2008.

[133] J. Wade, L. McDaid, J. Harkin, V. Crunelli, and S. Kelso, "Self-repair in a bidirectionally coupled astrocyte-neuron (AN) system based on retrograde signaling.," *Front. Comput. Neurosci.*, vol. 6, no. September, p. 76, 2012.

[134] IEEE, "IEEE Standard for Information Technology- Telecommunications and Information Exchange Between Systems-Local and Metropolitan Area Networks-Specific

Requirements-Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications," *IEEE Std 802.11-1997*. p. i-445, 1997.

[135] J.Kohlmorgen, K.R.Müller, and K.Pawelzik, "Analysis of Drifting Dynamics with Neural Network Hidden Markov Models," *Adv. Neural Inf. Process. Syst.*, vol. 10, pp. 735–741, 1998.