An Overview of Event-Driven Architecture of SpiNNaker System

Komal Sinha[1], Dr. K S Jasmine[2]

PG Student, Department of MCA, RV College of Engineering, Bangalore, India[1]

Associate Professor, Department of MCA, RV College of Engineering, Bangalore, India[2]

ABSTRACT: The enhancement in AI and machine learning has led to build the system that mimics the biological brain with power efficiency, parallel processing, and increased computation powers that has ability to learn on its own. Moving forward towards SNN from ANN, the computing model are built that are totally event-driven and are in awake state only once any interrupt takes place. With the use of ARM968 in SpiNNaker System, scalability is assured to the system to run parallelly on multiple cores of the system. In order to resemble biological behavior of human, NEF has been proved to be most efficient for building neural model in the neuromorphic system. This paper gives an overview of the SpiNNaker System discussing the sub-routing packets structure, Neural Engineering Framework, and the SpiNNaker Software.

KEYWORDS: SpiNNaker System, Multicast packet, AER, NEF, SC&MP, SARK

I. Introduction

The first-generation Artificial Intelligence being rules-based, focused on emulation of classical logics for leading towards the conclusion which remained within a narrow and specific problem domain. Moving forward to the second generation which is the current generation, concerns are made for sensing and perception which uses the concepts of deep-learning neural networks. Extension to the current generation AI mechanisms, there is a need to step towards the automation of ordinary human activities that mostly corresponds to human cognition including unsupervised learning mechanisms for autonomous adaptation and interpretation. By mimicking the nervous system of humans, the integration of the memory, computation, and also communication is achieved at higher speed, increased complexity and a better energy efficiency.

In Spiking Neural Network, in order to produce a dynamic ANN model, the time factor has been included which generates the output, i.e. spike by the neurons when the pre-information collected becomes enough to uphold the threshold. Following this advanced concept of SNN, SpiNNaker has been built which is a multi-core neuromorphic platform that are built on the general-purpose ARM microprocessor that supports the scalability of approximately, 2^{16} x 18 cores. The SpiNNaker models 1% of the total human brain neurons present (10^{11} neurons) amounting to a billion of neurons out of which each neuron interconnects with thousands of other neurons. With the peak rate of 100s of Hz, the mean firing rate of neurons remains below 10 Hz. With this rate, for 10^9 neurons, 10^6 ARMs are required where each ARM corresponds to 10^3 neurons and each node consists of the memory of 64MB for 16ARM968 that comprises of 1.6 x 10^4 neurons/node. For 10^9 neurons, 6 x 10^5 nodes are required assuming each neuron gets 1,000 inputs. To maintain the commonality for all the inputs in order to set the neurons model on processors, each input event is made to interconnect to more than hundreds of neurons that are modeled by a processor.

II. Literature Survey

The authors[1] have presented the overview of SpiNNaker System discussing about high-level project goals of SpiNNaker System which is to mimic mammalian nervous system. The authors have discussed about the various other approaches towards brain modelling like Blue Brain, and SyNAPSE from IBM. In this paper, the architecture of the system towards quantitative characteristics that are involved in biological neural system are shown by comparing the system with biological brain working. The system components like ARM968, DMA control, VIC, Ethernet, Routing subsystem are also discussed in detail.

The authors[2] have presented an overview of the hardware implementation of the SpiNNaker System. The event-driven software model of SpiNNaker System is represented along with API for its support and how to run the sample program to run on it. The software SC&MP is described with the Ethernet interface and interconnect links among the core. The host software is also introduced as the workstation to run the system on the host side. Visualizer application has been used for visualizing the output of the SpiNNaker System on the host side in graphical format.

NEF acts as neural compiler, incorporates realistic local error-driven learning rules, models visuality, etc… The computation with NEF is shown by comparing the biological brain. NEF makes sharp distinction between the activity of a population of neuron and the values that are being represented. NEF claims that the input current to a neuron is a linear function of the value to be represented [3]. The authors [4] have proposed a new design rule for the development of brain inspired computing system and fabricated a neuromorphic chip, which is named as 'Tianji' chip. The demonstration of multi-chip architecture-based PCB board is represented by the authors along with the hardware and software implementation details.

In SpiNNaker System, interconnection of all the chips are in the form of a mesh with toroidal topology that scales up to one million core processors. The information exchange among these core processors are done via the SDP protocol and communication is done asynchronously. Following event-driven programming scheme, SpiNNaker applications run without an operating system. Instead, it is linked to SARK (SpiNNaker Application Runtime Kernel) [5]. Multicast infrastructure for inter-processor communication uses the packets that are source-routed and can carry only information about the packet issuer. Multicast Router are able to replicate packets in order to implement the multicast function that are associated with sending the same packet to various other destinations [6].

The authors[7] have presented the comparison of the performance of Time-Step-Driven and Event-Driven Neural State Update Approaches in the SpiNNaker System by using 4 SpiNNaker chips (SPIN-3) and 48 SpiNNaker chips (SPIN-5). Theses chips run on the API on top of a specific event-based kernel. The conclusion of the experiment has shown that for time-step-driven method needs more processing time per spike than the Event-Driven Approach.

III. EVENT-DRIVEN COMPUTING MODEL OF SPINNAKER SYSTEM

SpiNNaker is not meant for service-provision system which uses convolutional operating system that runs on the core. Instead, SpiNNaker is entirely based on interrupt-driven in which each processor node on the chip consists of Vectored Interrupt Controller (VIC) that enables or disables the interrupts and awakes the processor mode from the sleep mode whenever any event occurs from one of the multiple sources. These sources may be among the following: Counter/timers, DMA (complete/error/timeout), Packet-error from the router, Off-Chip signals, System fault, Interrupt from another processor within the chip, Ethernet Controller, Software Interrupt, or 32KHz slow system clock. Each node consists of a DMA controller whose primary application is to manually control paging, and secondly, it incorporates a bridge between ARM968 and System NoC devices including SDRAM providing direct read and write access to the system. Depending on the activity of DMA transfers, ARM968 uses the bridge. Paging is used in the interconnects of neural connections for data transfer from SDRAM in the form of large blocks that corresponds to the response to an input driven processor and, for returning the updated information of the state change. The DMA controller handles the requests from ARM968 and allows it to pass through the processor providing CRC Error check control in the transferred blocks via dual buffers that supports simultaneous direct and DMA transfers. The DMA controller fulfills the request very transparently by letting the host processor retain full control of the transfer. The DMA controller provide DMA completion notification interrupts or error interrupt in case of any error occurrence. Figure 1 shows the Event-Driven SpiNNaker System.

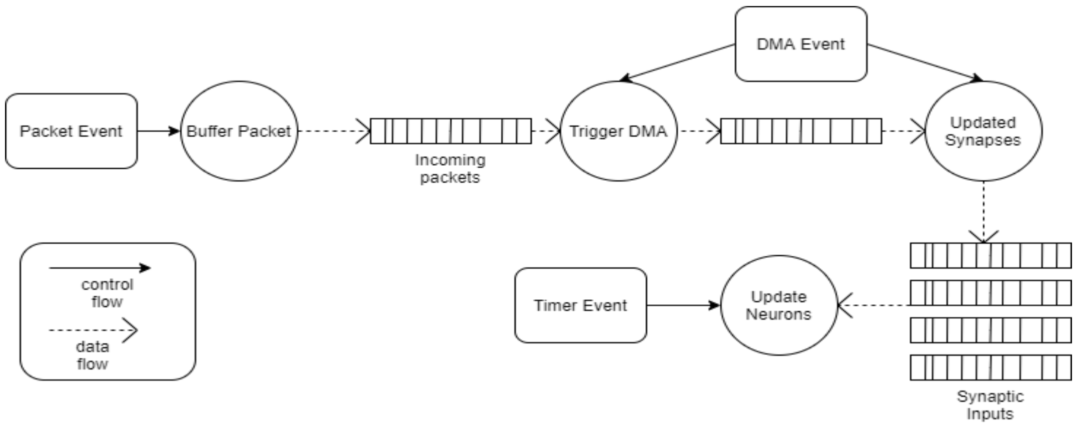

Fig 1. Event Driven SpiNNaker System

IV. Sub-Routing Packets

Inter-node communication in SpiNNaker System happens via packets which are launched by cores and transmitted over hardware to reach the local node router. Depending upon the target core which can be present on the same node as source or can be on another node, these packets are redirected. Each packet being 40 or 72 bits of data are broken up into control byte of 8 bits and one or two data words of 32 bits (1x 0r 2x 32 bits) which is optional. The two types of packets are used to set up the system where one concerns with data exfiltration and the reaming is called Multicast (MC) packet which are used during execution for conveying simulation information around the system. Figure 2 depicts the structure of 40 bits MC Packet.

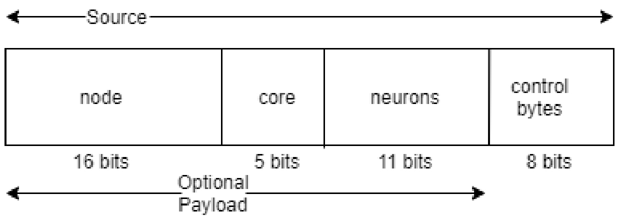

Fig 2: Multicast Packet Structure

Out of 32-bit address space, 16-bits are reserved for node address (node ID), 5-bits for the core (Core ID), and the rest 11-bits are reserved for each device that are hosted by a single core for uniquely address the devices per core. In order to route all the incoming packets to various outputs, routers are responsible for routing multicast neural event packets, point-to-point packets, nearest neighbour packets, fixed-route packet routing, default routing in case multicast packet is not matched with any of the entry in the multicast, and lastly emergency routing. Routers also handle various error conditions like packet parity errors, failure of output link or time-out condition. The passage of each types of SpiNNaker packets which are Nearest Neighbour (NN), Multicast (MC), Point-to-Point (P2P), and Fixed Route (FR), is arranged by hardware that comes through the router subsystem with the latency of 0.1microseconds irrespective of its type, source or destination.

V. Address Event Representation

Address Event Representation (AER) has become the basis of the emerging neural data serialization standard, which uses the packets after the source of the spike is encoded as an address in order to efficiently serialise and multiplex multiple neural signals to form the same series of lines. The knowledge about the target is included the routing tables in which each route table holds the incoming and the source address of the packets arriving. AER is implemented by using packet communication and multicast routing in SpiNNaker System. The address of a neuron spiking is broadcasted at a time to all the neurons to which the spiking neuron connects.

VI. Neural Engineering Framework

Neural Engineering Framework (NEF) allows to build neural models that resembles biological behavior of human brain in a large scale. The framework acts like a neural compiler where the properties of the neurons are specified along with the values to be represented and functions that are to be computed, and the framework solves the connection weights between the components performing the desired functions by converting the functions into realistic spiking neural networks. Using the distributed representations, NEF makes a clear distinction between the activity of a group of neurons and the values that are being represented which are usually considered as a vector of ‘x’. The input vector ‘x’ is passed to the set of encoders by the neural activity *a*, to each neuron *i*, resulting to the encoding vector e_i and a set of decoders are required in order to decode the spike train for the estimated value of the vector ‘x’. Neural activity can be computed by using the gain parameter, bias current for the neuron, and non-linear function:

$$a_i(x) = G(\alpha_i e_i x + b_i)$$

where G is a non-linear function representing the link between the current that is injected to a neuron and its firing rate, α_i represents the gain factor, e_i represents the threshold spiking rate, and b representing the background current that flows into the neuron. For the transformation of a vector into spikes and vice-versa encoders and decoders are used. For the desired connection of two neuron population, transferring of the value among the population is possible by specifying the interconnections weight adequately in order to perform operations. For each unit that are preferably linear, the output is obtained by reproducing the input values which are decoded form the spikes that are received from the first population and the output of these units are further encoded for the second population.

VII. SpiNNaker Software

The SpiNNaker Control and Monitor Program (SC&MP) is the control software that runs on the SpiNNaker System. Ethernet and IP-based protocols becomes the interface between the SpiNNaker Systems and the outside world. The bootstrap codes that are used for loading the codes via Ethernet interface are contained on the SpiNNaker chips which are used to load SC&MP to a single chip. After the propagation of SC&MP over the entire system through the interchip links, it is run continuously on the core over the selected monitor processor providing a range of services allowing the applications to be loaded on the remaining application cores on individual chip. SpiNNaker Datagram Protocol (SDP) is the simple packet protocol that is used within the SpiNNaker system for which SC&MP acts as a router that allows the packets to be sent to or from any core in the system including the Ethernet to the external endpoints using the shared memory interface. The packets are embedded in UDP/IP packets in order to carry SDP out of the system via the Ethernet to the external endpoints. A SpiNNaker application which is typically written in C is a program that runs on various application cores on a single SpiNNaker System by utilizing SDP or multicast packets for communication purpose. SpiNNaker Application Runtime Kernel (SARK) is the support library that is linked with each application providing the booting code for the application core in order to set up the runtime environment. SARK is also used for maintaining the communication interface with SC&MP that runs on the monitor processor allowing the application to communicate to the system and control SpiNNaker chips or external systems.

VIII. Conclusion

The development of the complex systems like SpiNNaker pushes the boundary towards neuromorphic computing making the power of hardware build and the software development. Development of such system is a step towards fulfilling the challenges that matches a human’s flexibility towards the ability to learn from unstructured stimuli by using the SNN model that emulates natural neural network existing in the biological brains. SpiNNaker System withholds the complexity of running million cores parallelly, building communication infrastructure, power management, and computation of state data. With the design of the software that can run on a non-deterministic machine with no internal debug has led to the beginning of the realization of the potential of the system not only in biology but also in other simulation areas.

References

1. S. B. Furber et al., "Overview of the SpiNNaker System Architecture," in IEEE Transactions on Computers, vol. 62, no. 12, pp. 2454-2467, Dec. 2013, doi: 10.1109/TC.2012.142.
2. S. B. Furber, F. Galluppi, S. Temple and L. A. Plana, "The SpiNNaker Project," in *Proceedings of the IEEE*, vol. 102, no. 5, pp. 652-665, May 2014. doi: 10.1109/JPROC.2014.2304638
3. Terrence C.Stewart (Oct 29, 2012). A technical Overview of the Neural Engineering Framework
4. L. Shi et al., "Development of a neuromorphic computing system," 2015 IEEE International Electron Devices Meeting (IEDM), Washington, DC, 2015, pp. 4.3.1-4.3.4, doi: 10.1109/IEDM.2015.7409624
5. F. Walter, M. Sandner, F. Rcöhrbein and A. Knoll, "Towards a neuromorphic implementation of hierarchical temporal memory on SpiNNaker," *2017 IEEE International Symposium on Circuits and Systems (ISCAS)*, Baltimore, MD, 2017, pp. 1-4, doi: 10.1109/ISCAS.2017.8050983
6. E. Painkras *et al.*, "SpiNNaker: A multi-core System-on-Chip for massively-parallel neural net simulation," *Proceedings of the IEEE 2012 Custom Integrated Circuits Conference*, San Jose, CA, 2012, pp. 1-4, doi: 10.1109/CICC.2012.6330636
7. E. Painkras *et al.*, "SpiNNaker: A multi-core System-on-Chip for massively-parallel neural net simulation," *Proceedings of the IEEE 2012 Custom Integrated Circuits Conference*, San Jose, CA, 2012, pp. 1-4, doi: 10.1109/CICC.2012.6330636