Artif Intell Rev (2014) 41:491–534
DOI 10.1007/s10462-012-9321-7

A new classification approach for neural networks hardware: from standards chips to embedded systems on chip

N. Izeboudjen · C. Larbes · A. Farah

Published online: 17 March 2012

Abstract The aim of this paper is to propose a new classification approach of artificial neural networks hardware. Our motivation behind this work is justified by the following two arguments: first, during the last two decades a lot of approaches have been proposed for classification of neural networks hardware. However, at present there is not a clear consensus on classification criteria and performances. Second, with the evolution of the microelectronic technology and the design tools and techniques, new artificial neural networks (ANNs) implementations have been proposed, but they are not taken into consideration in the existing classification approaches of ANN hardware. In this paper, we propose a new approach for classification of neural networks hardware. The paper is organized in three parts: in the first part we review most of existing approaches proposed in the literature during the period 1990–2010 and show the advantages and disadvantages of each one. In the second part, we propose a new classification approach that takes into account most of consensual elements in one hand and in the other hand it takes into consideration the evolution of the design technology of integrated circuits and the design techniques. In the third part, we review examples of neural hardware achievements from industrial, academic and research institutions. According to our classification approach, these achievements range from standard chips to VLSI ASICs, FPGA and embedded systems on chip. Finally, we enumerate design issues that are still posed. This could help to give new directions for future research work.

Keywords ANN hardware · Classification · Neurocomputers · VLSI · ASICs · FPGA · Embedded systems on chip

N. Izeboudjen (✉)
Centre de Développement des Technologies Avancées (CDTA), lotissement 20 Aout, 1956 Baba Hassen Algiers, Algeria
e-mail: nizeboudjen@cdta.dz

N. Izeboudjen · C. Larbes · A. Farah
Ecole Nationale Polytechnique, 10, Avenue Hassen Badi, BP 182,16200 El-Harrach Alger, Algeria

1 Introduction

Artificial neural networks (ANNs) are systems based on special mathematical algorithms, which are derived from the field of neuroscience and are characterized by intensive arithmetic operations. These networks display interesting features such as parallelism, classification, optimization, adaptation, generalization and associative memories. Since the McCulloch and Pitts pioneering work (McCulloch and Pitts 1943), much research has been done to implement these algorithms using software tools supported by conventional computers which are based on the model of the Von Newman machine and in which a single instruction is executed at a time. Software tools such as the MathWorks MATLAB neural network toolbox (Demuth and Beale 1992), the Stuttgart Neural Network Simulator "SNNS" (Zell et al. 1990) and the General Neural Simulation System "GENESIS" (Bower and Beeman 1998), provide the user with a flexible environment and libraries that suit different applications. However, their major drawback is that the parallelism inherent to neural networks is not exploited. To overcome this problem, since the early of 1980s, researchers supported the idea that implementation of neural networks into a special hardware and in which multiple processing elements are connected in parallel, represents a promising future for the improvement and exploitation of the neural networks capabilities. This position is well described in a paper by Lippmann (1987): "The great potential of neural networks remains in the high speed processing that could be provided through massively parallel implementation". In another paper (Trealeven 1989) reported that "the important design issues of ANNs are parallelism, performance, flexibility and their relationship to silicon area". However, despite these well justified arguments, and from 1980 to the beginning of 1990s, development and commercialization of neural hardware has been slow and had a modest success; this is due to the fact that, at that time, there was not a clear consensus on how to exploit the capabilities of the VLSI technologies for hardware implementation of massively parallel neural networks. Another raison that has hampered the construction of the neural hardware regarding software over this period is explained by the fact that the number and variety of neural models and algorithms has grown rapidly: For many models, performance is almost not known and not established. Paradoxically, at that period researches in the hardware implementation of ANNS have coincided with the tremendous growth of computing power of conventional processors of the Von Newman machine such as the Intel Pentium processor, which allowed the software neural tools to achieve a great success in a wide range of applications. This has led some researchers to ask the question: Why hardware implementation of neural networks? Answers to this question were mainly justified by the speed that could be achieved for large neural networks. Indeed, even the fastest processor cannot achieve learning in real time response for large neural networks (Denby 1993). Other factors such as: miniaturization, portability, weight, confidentiality and power dissipation are used to justify dedicated hardware implementation.

From there, a new interest was given for the hardware implementation of neural networks, and since the early 1990s, the number of achievements that have had great success, has increased significantly. A lot of approaches have been proposed ranging from general purpose computers, to DSP circuits, standards chips, special purpose VLSI or ASIC chips, FPGA circuits, and recently systems on chips and embedded systems.

Classification of this hardware is essential to understand the basic properties of new implemented versions, to compare between them and draw a roadmap for hardware neural networks. Certainly, a lot of approaches have been proposed in the literature to classify neural hardware architectures. However, at present there is not a clear consensus on classifications criteria's and performances. Also, with the evolution of the microelectronic technology and the design tools and techniques, new ANNs implementations have been proposed in the

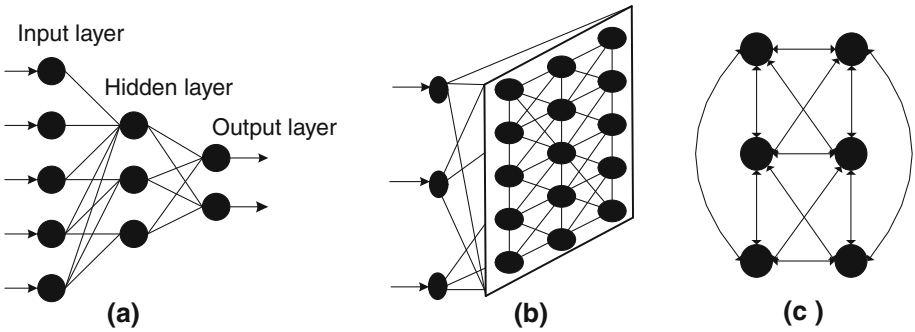

Fig. 1 **a** three layers feed forward. **b** Kohonen topology. **c** Hopfield topology

literature and from commerce, but they are not taken into consideration in the existing classification approaches of ANN hardware. These have motivated us to review most of existing classification approaches and their synthesis. Then, we propose a new classification approach of ANN hardware. The main feature of the proposed approach is that it takes into account most of the consensual elements of existing approaches in one hand and the evolution of the VLSI technology and the design techniques in the other hand.

The paper is structured as follows: In Sect. 2 a general presentation of ANNs is given. Section 3 deals with the presentation of different approaches used for the classification of neural hardware. In Sect. 4, a synthesis of the different classification approaches is given, and then in Sect. 5 we propose a new approach for classification of neural hardware. In Sect. 6, examples of neural hardware implementations are given and finally a conclusion is given.

2 General presentation of artificial neural networks

The purpose of this section is to give a general presentation of ANNs to help understanding the basic hardware architecture of the neuron. It is well known that an artificial neural network is a computing system that combines a network of highly interconnected processing elements "PEs" or neurons. Inspired by the physiology of the human brain, the PEs combine mathematical algorithms, to carry out information processing through their state response to stimuli. A neural network can be characterized by a few key properties (Titri et al. 1999): (1) network topology, (2) forward procedure, (3) training/learning procedure and (4) the post-training procedure.

2.1 Network topology

Topologically, an ANN can be seen as a direct graph with no self-loops, where each vertex or node represents a processing element "PE" and each edge is assigned a number called the weight of the edge or synaptic connection. Of practical interest, we can mention two networks topologies schemes: the multilayered networks and the un-layered networks. Figure 1a shows the popular three layered back propagation network topology. Figure 1b illustrates a typical Kohonen topology and Fig. 1c illustrates the Hopfield network, which is an un-layered topology. These networks differ not only by the topology but also by the learning and the training procedures.

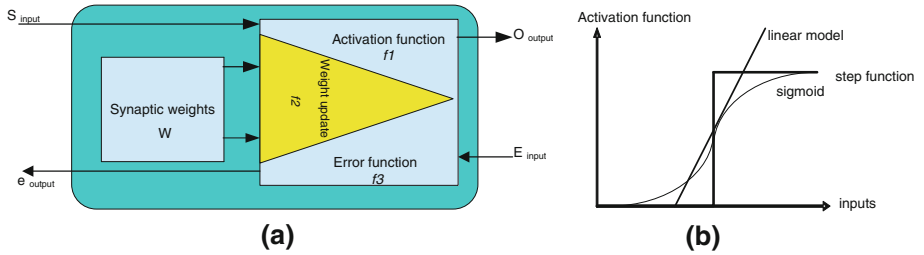

Fig. 2 **a** neuron's model. **b** ANNs activations functions

To give a general view, consider the general neuron's model of Fig. 2a, as proposed in Treleaven et al. (1989). This model comprises three specific functions: $f1$, $f2$ and $f3$, a table of weight: W, and a set of inputs: S_{input}, E_{input} and output: O_{output}, e_{output} signals. A neuron receives the input state, S_{input}, from the neurons of the previous layer and forwards its state to the output. Similarly, a set of input error, E_{input}; and an output error, e_{output}, provides feedback in the network. $f1$ stands for the *activation function*, $f2$ for the *weight updating* function, and $f3$ for the *error calculation* function.

2.2 Forward procedure

This procedure is specified by the function $f1$, which comprises the propagation rule *net* and the activation function, T:

$$net = f(S_{\text{input}}, \text{W}) \tag{1}$$

$$O_{\text{output}} = \text{T}\,(net) \tag{2}$$

In its simplest form, the propagation rule calculates the weighted sum of the input, modified by an offset, θ that defines the neuron bias:

$$net = \sum S_{\text{input}} \cdot W - \theta \tag{3}$$

The non-linear activation function of the propagation rule calculates the neuron state. Common activation functions include the step function, the sigmoid function and the linear one (Fig. 2b).

2.3 Training/learning procedure

Training consists of presenting a set of patterns (examples) to the network. Learning procedures are either supervised or unsupervised. In supervised learning, the system submits a training pair consisting of an input pattern and the target output to the network. The most used algorithm in this class is the back propagation learning algorithm. Unsupervised learning procedures classify input patterns without requiring information on the target output. The Kohonen algorithm deals with this category. Learning algorithms generally involve two functions, in addition to the activation function, the error calculation function, $e = f3\,(s, E, W)$, which controls the updating of weights and the function, $\Delta W = f2\,(S, e, W)$, which updates the weights until the error function converges to an acceptable minimum. Thus, each algorithm has its own learning rule.

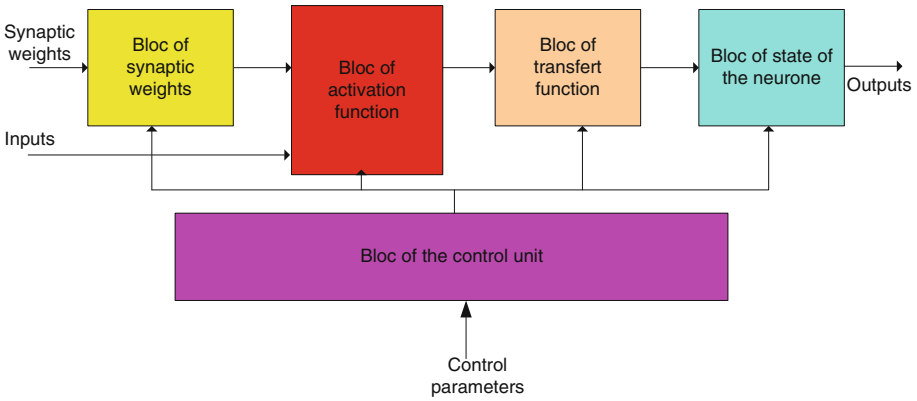

Fig. 3 Basic architecture of the neurone

2.4 Post-training

After training, propagation rule of the forward procedure is re-used to compute the network's outputs according to the final synaptic weight values obtained in the previous phase. A neural network can be used in its end application only after it has been trained.

A practical implementation can use either "on-chip" training or "off-chip" training.

"On-chip" training circuits require implementation of the activation function; the error calculation function and the weight updating function (recall procedure and the learning procedure). In the " off-chip " training circuits, the final synaptic weights are pre-calculated using a software tool. In that case, the final weights are loaded into memory and only the forward phase is used in post training operation. Thus, the hardware implementation makes it possible to compute the ANN outputs for given classes of inputs. But, after implementation, the neural network is no longer capable to learn new classes.

2.5 General architecture of the neuron

An investigation in the literature has revealed that the block diagram of Fig. 3 is suitable as a basic architecture equivalent to the mathematical model of artificial neuron (Liao 2001; Anderson and McNeill 1992).

The block of the activation function performs the sum of products of the Eq. (3). This block is always implemented within the circuit. Other blocks, such as the state of the neuron, synaptic weights and transfer functions can be implemented inside or outside the circuit. Calculation of these blocks can be achieved by a computer. These are called "on-chip" or "off-chip" implementation. Transfer of data between blocks is achieved by the control block which is always implemented "on chip". The parameters are used to control the circuit by a computer. For the multilayer perceptron (Cichocki and Unbehauen 1994) and the Hopfield network (Gascuel et al. 1991), the transfer function can be represented by a threshold function, a ramp function or a sigmoid function. For the Boltzmann Machine (Alspector 1991), the transfer function is represented by a threshold function to which a noise signal is added. For the Kohonen network (Melton et al. 1992), what is calculated by the activation bloc, corresponds to the Euclidian distance between the input and the synaptic weights vector. The bloc of transfer function which implements the minimum Euclidean distance determines the indices of the neurons in the self-organizing maps (Peiris et al. 1994). The states of the

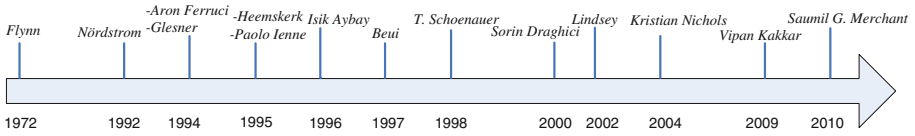

Fig. 4 Different approaches for classification of neural hardware

neurons and weights can be stored in digital or analog form. The weights can be loaded statically or dynamically.

3 Different approaches for the classification of neural hardware

During our research, we have identified thirteen (13) classification approaches from 1992 to 2010; we have classified them according to the year and to the first authors of the published paper, as follows (Fig. 4):

- 1992: NÖRDSTROM classification
- 1994: AARON FERRUCI classification and
- : GLESNER classifications
- 1995: PAOLO IENNNE classification and
- : HEEMSKERK classifications
- 1996: ISIK AYBAY classification
- 1997: BEIU classification
- 1998: T. SCHOENAUER classification
- 2000: SORIN DRAGHICI classification
- 2002: LINDSEY classification
- 2004: KRISTIAN NICHOLS classification
- 2009: VIPAN KAKKAR classification
- 2010: SAUMIL G. Merchant classification approach

In what follows, the different approaches for the classification of the neural hardware are presented. To make the text clear and consistent, we conducted a process in which we first present the ANN classification approach, then the classification criteria are highlighted, and finally examples of achievements cited in each classification approach of ANN hardware.

3.1 NÖRDSTROM classification approach

Presentation Nordstrom and Svensson (1992) conducted a study on the use of parallel machines in the implementation of neural networks. Starting from the four architectural classes defined by Flynn (1972), namely SISD, SIMD, MISD and MIMD architectures, Nordstrom and Svensson showed that SIMD-type architecture is the most appropriate for implementing neural networks with parallel machines.

Classification criteria Classification of neurocomputers was based on the number of parallel processors and the complexity of the processor. The number of parallel processors, N, is related to the degree of parallelism and is defined as follows:

- *Massively parallel*, $N \geq 2^{12}$
- *highly parallel*, $2^8 < N \leq 2^{12}$
- *moderately parallel*, $2^4 \leq N \leq 2^8$
- *barely parallel* $2 < N \leq 2^4$

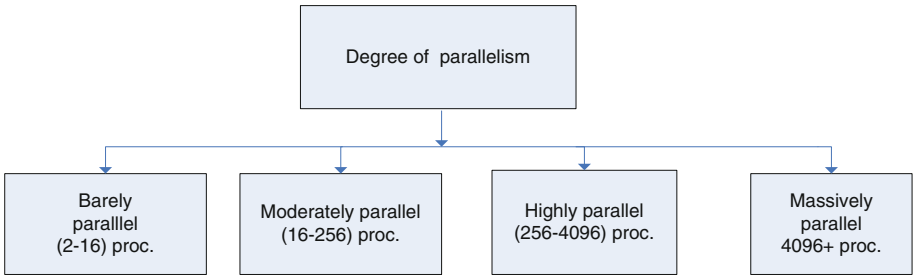

Fig. 5 Classification of neural hardware according to NÖRDSTROM

The complexity of the processors is related to the communication between processors in a SIMD machine.

Figure 5 shows the different classes used.

Examples

- In the class of *massively parallel* implementations, the following implementations are cited: AAP-2 (Watanabe et al. 1989), CM-2 (Hillis and Steel 1986), Mas-Par (MP-1) (Christy 1990) and DAP (Hunt 1989).
- In the class of *highly parallel* implementations, the following implementations are cited: REMAP (Nordstrom et al. 1991)], L-Neuro (Duranton and Sirat 1990), CNAPS (Hammerstrom 1990), GF11 (Beetem et al. 1987), and UCL-Neurocomputer (Bavan et al. 1988)
- In the class of *moderately parallel* implementations, the following implementations are cited: Sandy/8 (Kato et al. 1990), RAP (Morgan et al. 1990) and WARP (Kung 1987).
- In the class of *barely parallel*, the following implementations are cited: Hitachi-WSI (Yasunaga et al. 1989) and Siemens (Ramacher and Wesseling 1990).

Advantages and disadvantages The advantage of the classification proposed by Nordstrom and Svensson is that, they have defined and quantified the degree of parallelism based on the number of processors. However, other architectural properties such as analog design, digital and hybrid designs do not appear in their classification approach.

3.2 AARON FERRUCI classification approach

Presentation This approach was proposed by Aaron Ferruci, at the beginning of 1994 (Ferrucci 1994). He defined a classification of neurocomputers based on the following (Fig. 6):

Classification criteria

- *Data representation*: Synaptic weights and input/output values can be encoded as analog values, digital or stochastic bit streams. In the case of digital data representations, floating point or fixed point are used.
- *Interconnect strategy* that can be of type SIMD or MIMD
- *Arithmetic precision* that varies from one implementation to another depending on the targeted application.
- *Technology*: which can be either VLSI, DSP, FPGAs or Wafer scale integration (WSI)
- *Mapping the algorithm* which represents the degree of parallelism used. Four types of parallelism are cited: the parallelism of neurons, the parallelism of synapses, the parallelism of layers and the training examples parallelism.

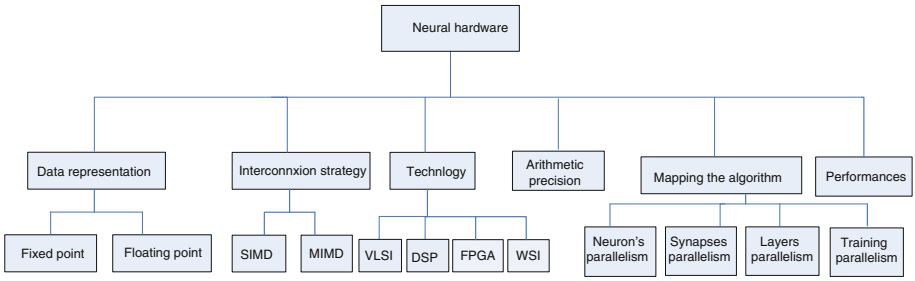

Fig. 6 Classification of neural hardware according to AARON FERRUCI

- *Performances metrics* which can be expressed in either*MCPS* (Millions of Connections per Second) in the case of off -chip implementation or MCUPS (Millions of Connections Updates per Second) in the case of on chip implementation.

Examples

– In his classification, Aron Ferruci was interested in the classification of the following neurocomputers: CM-2 (Hillis and Steel 1986)], CNAPS (Hammerstrom 1990), GF11 (Beetem et al. 1987), Hitachi-WSI (Yasunaga et al. 1989), IPSC/80 (Jackson and Hammerstrom 1991), Richo-LSI (Eguci et al. 1991), SPERT (Wawrzynek et al. 1993) and GANGLION (Cox and Blanz 1992).

Advantages and disadvantages The major drawback of this study is that the author was interested only in the classification of hardware implementations which are specific to the backpropagation algorithm.

3.3 GLESNER classification approach

Presentation In 1994, a completely different classification approach was proposed (Glesner et al. 1989; Glesner and Pochmuller 1994).

Classification criteria

- *Biological evidence* that represents the degree of imitation of biological systems
- *Mapping of the hardware*: that considers the parallelism of neurons, synapses or layer parallelism
- *Implementation technology* that can be digital, analog or hybrid.

Examples No examples available

Advantages and disadvantages The authors introduced a new classification criteria related to biological evidence; however we could not have enough information about this work.

3.4 PAOLO IENNE classification approach

Presentation In 1995, PAOLO IENNE (Ienne 1995) proposed a classification of neural hardware according to two criteria namely: the *flexibility* and *performances*.

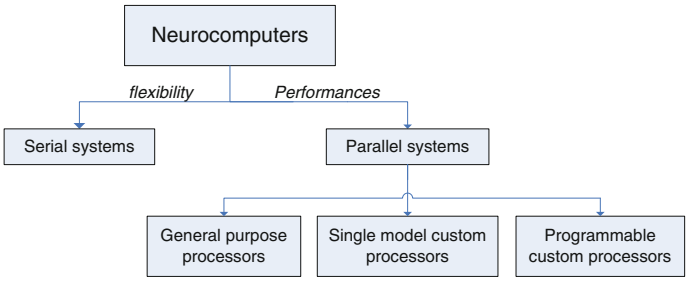

Fig. 7 Classification approach according to PAOLO IENNE

Classification criteria

- *Serial systems*
- *Parallel systems using general purpose processing elements*
- *Parallel systems using single- model custom processing elements*
- *Parallel systems using programmable custom processing elements*

Figure 7 shows the various classes proposed by PAOLO IENNE.

Examples

– An application example of *serial systems* using simulation software is described in (Driancourt 1994).
– *Parallel systems using general purpose* circuits are used to describe implementations using multiple general processors mounted in parallel. The SPRINT transputer- based-systolic-arrays computer (De Groot and Parker 1989) and the MUSIC "Multi-processor System with Intelligent Communication" (Muller et al. 1995) are examples of such systems. The problem with such implementations is that when the number of processors is increased, communication between them becomes slow.
– *Parallel systems using single-model custom processing* elements are used to describe VLSI implementations that exploit the parallelism of large networks of neurons. In this category the author distinguishes between:
-*VLSI systems specific to one type of algorithms* such as the chip Ni1000 from Intel (Nestor 1994), the Richo-LSI chip (Eguci et al. 1991)and Hitachi WSI neurocomputer developed by Hitachi (Yasunaga et al. 1989).
– *Programmable systems* that can support implementation of *several types of algorithms* such as the CNAPS (Hammerstrom 1990) neurocompter, the Synapse chip (Ramacher et al. 1991) and the Mantra-I system (Viredaz and Ienne 1993).

Advantages and disadvantages Paolo Ienne recommends combining good software approaches with good hardware approaches for algorithm parallelization in order to make a good compromise between performances and flexibility, but the classification was limited to digital neurocomputers only.

3.5 HEEMSKERK classification approach

Presentation In 1995, Jan N. H. HEEMSKERK (Heemskerk 1995) proposed a classification approach (Fig. 8):

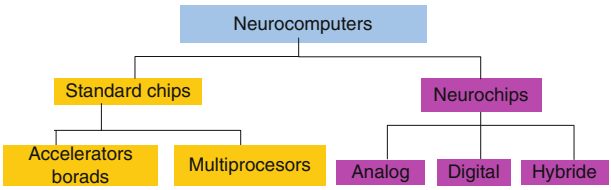

Fig. 8 Classification of neural hardware according to HEEMSKERK

Classification criteria

- *Neurocomputers built from standards chips* in which sequential accelerator boards are distinguished from multiprocessors implementations
- *Neurocomputers built from Neurochips*

In the standard chip implementations *Analog, digital and hybrid* implementations are classified. HEEMSKERK's classification was based on the MCPS and MCUPS performance measures to compare between different implementations available at that time.

Examples

– In the class of *accelerator cards* the following implementation are cited: The ANZA-plus card developed by "Heicht Neilson cooperation" (USA) (Trealeven 1989) and which is based on the Motorola processor NC68020 and the Motorola MC68881 coprocessor. The NT6000 (Lindsey et al. 1998) developed by the Neural Technologies limited company and the California Scientific Software. The card is equipped with a TMS320 DSP and an NISP (Neural Instruction Set Processor). Others neurocomputers such as the Ni1000 (Nestor 1994) developed by Intel (USA), the IBM NEP map developed by the IBM company (Hanson et al. 1987), and the Neuro Turbo board developed by the Nagoya Institute of Technology-Japan (Onuki et al. 1993) are cited in this category.
– In the class of *neurocomputers* from standard processors and multiprocessors, the following systems are cited: the WISARD system developed by the Imperial College of London (UK) (Aleksander et al. 1984), The Sandy/8 Developed by Fujitsu-Japan laboratory (Viredaz and Ienne 1993), the COKOS co-processor from the university of Tubingen-Germany (Speckman et al. 1993), the system TI-Netsim developed by Texas Instruments and the University of Cambridge -UK (Garth 1987).
– In the class of *Neurochips* the following systems are cited: SYNAPSE (Ramacher and Ruckert 1991; Ramacher et al. 1993), CNAPS (Hammerstrom 1990), WSI-Hitachi -Japan (Yasunaga et al. 1989), L-Neuro 1.0 (Hammerstrom 1991), UTAK1 developed by the University of Catalonia (Spain) (Castillo et al. 1992), Mantra-I (Viredaz and Ienne 1993) and IBM ZISC036 (David et al. 1999).

Advantages and disadvantages Heemskerk introduced new concepts and criteria and his classification approach covered a large number of ANNs implementations (Analog, digital and hybrid). However, the classification approach was general because it does not emphasis on the design techniques for each class.

3.6 ISIK AYBAY classification approach

Presentation In 1996, ISIK AYBAY (Aybay et al. 1996), developed an analysis in which he reported that existing classification approaches do not show all the characteristics of a hardware implementation. Some features may be lost and the reader spends a lot of time to compare between two implementations.

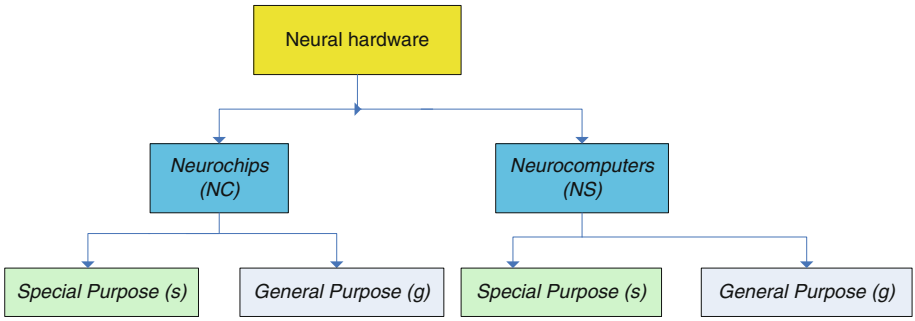

Fig. 9 Classification of neural hardware according to Aybay Isik

Classification criteria

- *Type of neural hardware* that can be a "*Neurochip* (NC) or a" *Neurocomputer* (NS)
- *Implementation type* which means if the circuit implements several neural network algorithms "*General Purpose*" or a specific algorithm "*Special Purpose*"

Figure 9 shows the classification according to ISIK AYBAY.
In addition to these criteria the author defines several attributes for classification, namely:

– *The input signal* (I): digital (d) analog (a) hybrid (h)
– *The activation block* (A): digital (d), analog (a)
– *The storage of synaptic weights* (W): digital (d), analog (a)
– *Storage of neurons* (S): digital (d), analog (a)
– *The transfer function* (T): digital (d), analog (a)
– *The type of learning* (L): "On-chip" (o), or "off-chip" (f)

Examples According to the classification criteria and attributes, the following examples of circuits are cited and classified as follows:

– The CNAPS (Hammerstrom 1990) is a "*general purpose neurocomputer* (NSgd) having the following characteristics: Id / Ad / WDO / SDO / tdo / Lo.
– The circuit ETANN (Holler et al. 1989), is a *general purpose hybrid Neuro-Chip* (NCgh) having the following characteristics: Ia / Aa / WDO / Sao / Tao / Lf.

Advantages and disadvantages The ISIK AYBAY approach introduces new classification criteria that highlight the architectural properties of the hardware in question. However it does not emphasis on the measure of performances.

3.7 BEUI classification approach

Presentation Because of the huge diversity of realizations, in 1997 BEIU proposed a historical classification of *neurocomputers* based on the following

Classification criteria

- *Publication date*
- *Technological changes*

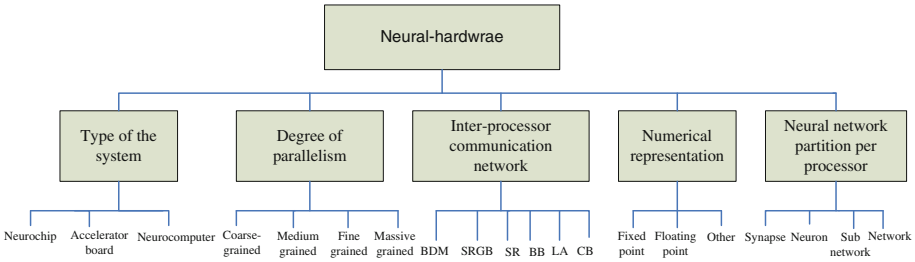

Fig. 10 Classification of neural hardware according to T. Shcoenauer

Examples

– He started the classification of the first historical *neurocomputer,* MARK-III (Hecht-Nielsen 1990), which appeared in 1984, passing through architectures based on accelerator cards, to DSP circuits, transputers, systolic array processors, and RISC processor. For this purpose, fifty achievements of *neurocomputers* were classified by the author from 1984 to 1996 (Beiu 1997).

Advantages and disadvantages In this approach, foremost research effort has been provided for the classification of different neural hardware. Nevertheless, the study concerns only digital implementations of neural networks. Analog and hybrid (analog / digital) realizations were not considered by the author.

3.8 SHCOENAUER classification approach

Presentation In 1998, T. Schoenauer and al. have proposed a classification approach based on the following criteria (Schoenauer et al. 1998)

Classification criteria

- *Type of the system* which can be: "*Neurochip*" "*Accelerator board*" or "*Neurocomputer*".
- *Inter-processor communication network* which can be "*Bidimensional Mesh*", "*Systolic Ring*", "*Systolic Ring with Global Bus*", "*Broadcast Bus*", "*Linear Array*" or " *Crossbar*".
- *Degree of parallelism* which can be "*Coarse-grain*", "*Medium Grained* ", "*Fine-Grained*", and "*Massive*".
- *Numerical representation* which can be *fixed*, *floating point* or other.
- *Typical neural network partition per processor*: Which can be of type: "*Synapse*", "*Neuron*", "*Sub-network*" and "*Network*"

Figure 10 shows the classification of ANN hardware according to Shcoenauer.

Examples

– Three examples of ANN hardware realizations were cited by the author; these are the CNAPS (Hammerstrom 1990), the SYNAPSE-1 (Ramacher and Ruckert 1991; Ramacher et al. 1993) and the NESPINN system (Jahnke et al. 1996). CNAPS is a very versatile and powerful platform for ANN, as long as the on-chip memories of the processing nodes can store the connectivity. SYNAPSE is an even more powerful neurocomputer consisting of vector processors in a systolic array. System complexity, however, restricts user friendly programming. More than the other two neurocomputers, the NESPINN-design exploits more aggressively the characteristics of a certain class of neural networks: spiking neural networks.

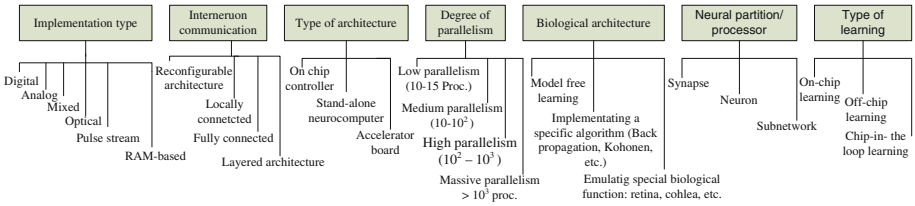

Fig. 11 Classification of neural hardware according to Sorin Draghici

Advantages and disadvantages According to the criteria mentioned above, the classification proposed by T. Shcoenauer focuses on hardware implementations using multiple processors communicating with each other. Design aspects such as analog and hybrid does not appear in this approach.

3.9 SORIN DRAGHICI classification approach

Presentation In 2000, Draghici (2000) proposed a classification based on the following

Classification criteria

- *Implementation type*, which regroups the following classes:*analog, digital, mixed, optical, pulse- stream and RAM -based* implementations
- *Interneuron communication paradigm*, which regroups the following classes: *layered architectures*, *fully connected networks*, *reconfigurable architecture* and *locally connected network*
- *Type of the architecture*, which regroups the following classes: chip with *on chip controller*, *stand alone neurocomputer* and *accelerator board*
- *Degree of parallelism,* which regroups the following classes: *low level parallelism*, *medium parallelism*, *High parallelism*, *massive parallelism*
- *Purpose of the implementation*, which regroups the following classes, *model free learning*, *general purpose implementing a specific paradigm*: back propagation, radial basis function, Boltzman machine, Kohonen, etc. and finally, *neural networks emulating special biological functions*: retina, cochlea and visual flow computation.
- *Partitioning the neural network per processing element and the type of learning*.
- *Type of learning,* which regroups the following classes: *on-chip*, *off- chip* and *chip-in-the loop* learning algorithms

Figure 11 shows classification criteria of ANNs according to Sorin Draghici.

Examples

– Several references were cited for each classification criterion, among them: ETANN (Holler et al. 1989; Hasler et al. 1995), ART-1 (Xie and Jabri 1991; Murray and Smith 1987), the Probalistic logic nodes (PLN) (Kane and Paquin 1993), the Goal seeking Neuron (GSN) (Fisher et al. 1991), Probalistic RAM- PRAM (Gorse et al. 1994; Schneider and Card 1991; Mead and Ismail 1989; Shima et al. 1992; Silviott et al. 1987; Jabri and Flower 1992)

Advantages and disadvantages The author presented several criteria for classifying neural networks, however and contrarily to the previous approaches he focused on analog hardware implementations only. Also, there is not a clear criterion for classification: for example; reference (Schneider and Card 1991) was cited as an example for *interneuron communication paradigm* class and the *implementation purpose* class.

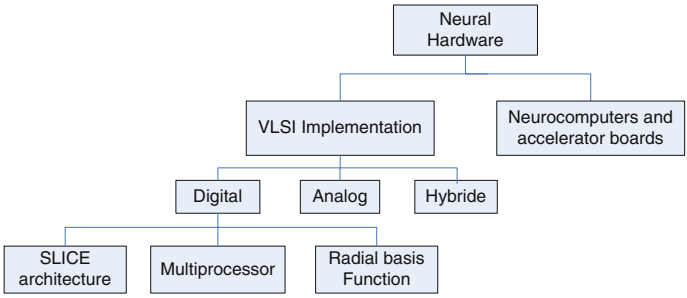

Fig. 12 Classification of neural hardware according to CLARK LINDSEY

3.10 CLARK LINDSY classification approach

Presentation In 2002, Lindsey and Thomas Lindbald (Lindsey 2002) proposed a classification of neural hardware using the following criteria (Fig. 12):

Classification criteria

- *VLSI implementations* in which circuits are classified into analog, digital and hybrid. In the case of digital circuits, the following architectures are cited: Slice Architecture, Architecture-based multiprocessor and radial basis function.
- *Accelerator boards and neurocomputers*

Examples

– In the class of VLSI implementations, the following circuits are cited: CNAPS (Hammerstrom 1990), IBMZISC036 (David et al. 1999), Intel ETANN (Holler et al. 1989), Nestor Ni1000 (Nestor 1994), Phelps L-Neuro (Duranton and Sirat 1990), Richo RN-200 (Eguci et al. 1991).
– In the class of accelerator boards and neurocomputers, the following circuits are cited: Accurate Automation and ISA Cards (Cox et al. 1995), Adaptive solutions CNAPS-ISA (McCartor 1991), Nestor PCI and VME Cards (Nestor 1994), BrainMaker Accelerators (Dvorak 1991), Siemens MA16 chip and Synapse-1 (Ramacher et al. 1991)

Advantages and disadvantages Clark Lindsey proposed a similar classification to that of Heemskerk, however he used the terminology "VLSI implementation" instead of "Neurochip" which was used by Heemskerk in his classification approach.

3.11 KRISTIAN NICHOLS classification approach

Presentation In 2004, KRISTIAN NICHOLS (Nichols 2004) proposed a classification of neural hardware according to the criteria below (Fig. 13):

Classification criteria

- *Learning/training algorithm* that identifies in one hand, if for a given algorithm, the learning type is "on chip" or "off chip" and on the other hand, if the circuit can implement a single type of algorithm or several types of algorithms.

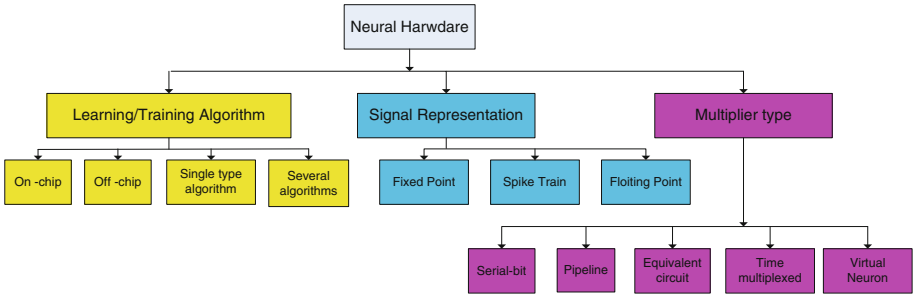

Fig. 13 Classification of neural hardware according to KRISTIAN NICHOLS

- *Signal representation* which means implementations using fixed-point representation or those currently using a representation of pulse "Spike Train."
- *Multiplier Reduction scheme* used for the hardware implementation of neural networks. For this the author reported five types of multiplier architectures, namely the "*bit-serial multiplier*", "*the pipeline multiplier*", "*signal representation that eliminates the need of multipliers*", use of a "time *Multiplexed algorithm*" and use *of" virtual neurons*".

The *bit-serial multiplier* calculates only one bit at a time. In such an implementation, the multiplier can scale up any range of precision while its area remains static. However, multiplication grows quadratically, $O(n^2)$, with the length of signal representation used. To overcome this problem, the pipeline multiplier is used. Elimination of the multiplier is obtained by using a typical signal representation that allows the substitution of the multiplier circuit by simple logic functions.

The *time multiplexed* approach is used to reduce the number of multipliers required to implement a neural network algorithm.

The concept of *virtual neurons* is identical to the concept of virtual memory in the case of computers. For this, an FPGA prototyping board is first chosen as the base platform, then all the parameters of the neural network (synaptic weights, input / output activation function, etc.) are stored in external memory and the rest of the architecture (mainly the multiplier) is implemented into the FPGA. The advantage of this approach is that the maximum number of neurons that can be supported depends only on the size of the external memory available. However the time lost to access the memory is a serious problem which must be taken into consideration.

Examples

– In the class "*Learning/training algorithm*", the following circuits are cited: RRANN (Eldredge and Hutchings 1994), RENCO (Beuchat et al. 1998), ACME (Glesner and Pochmuller 1994), FAST (Peres-Uribe 1999), and REMAP (Nordstrom et al. 1991).
– In the class *"Signal representation*", the following examples are cited: CAM-Brain Machine (De Garis and Korkin 2002), ECX card (Skrbek 1999), RRANN (Eldredge and Hutchings 1994) and the work of Holt and Baker (Holt and Baker 1991).
– In the class "*Multiplier Reduction scheme"*, the following circuit references are cited: (Eldredge 1994; Tavenik and Linde 1995), REMAP (Nordstrom et al. 1991), FAST (Peres-Uribe 1999; Nordstrom 1995) and RRANN (Eldredge and Hutchings 1994; Beuchat et al. 1998; De Garis and Korkin 2002; Skrbek 1999; Elmasry 1994)

Advantages and disadvantages Nichols introduced new criteria related to the learning algorithm as the "on-chip learning" and "off-chip learning". He also used the "the multiplier

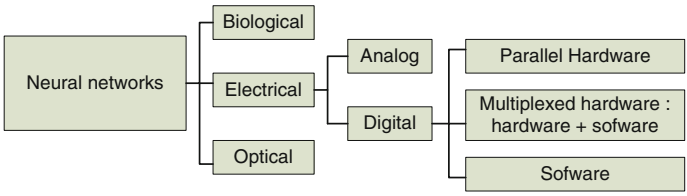

Fig. 14 Classification of neural hardware according to Vipan Kakkar

reduction scheme" as a criterion for classification. Nevertheless, his study focused on FPGA implementations only.

3.12 VIPAN KAKKAR classification approach

Presentation In 2009, Vipan Kakar proposed a comparative study between analog and digital neural networks (Kakkar 2009). These latter were classified according the following

Classification criteria

- Biological implementation,
- Optical implementation and
- Electrical implementation.

In the class of electrical implementation, neural networks were classified into analog and digital. The comparison was based on the following features: power consumption, area and weight storage. Figure 14 shows the proposed classification approach.

Examples

– Implementation examples were not clearly highlighted by the author.

Advantages and disadvantages Analog, digital and mixed design techniques were considered. However recent advances in the development of ANN implementations, such as system on chip and embedded systems were not considered.

3.13 SAUMIL G. MERCHANT classification approach

Presentation In 2010, Merchant and Peterson (2010) proposed a classification approach based on the following

Classification criteria

- Digital implementation,
- Analog implementation
- Hybrid implementation

The digital hardware implementations have been grouped into FPGA and ASIC implementations and classified according to design issues such as *data representation*, *design flexibility, on-chip/off-chip learning*, and *transfer function implementation* (Fig. 15).

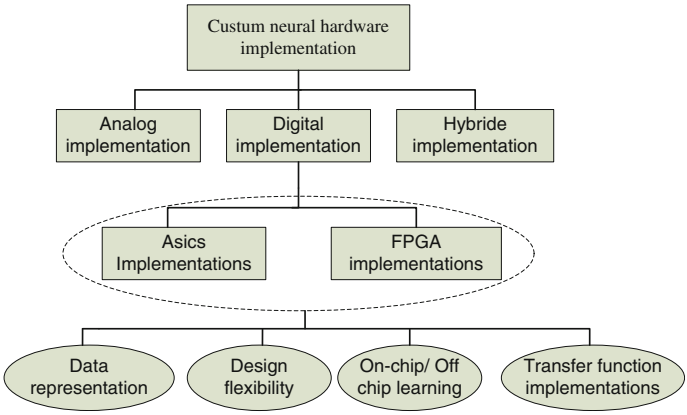

Fig. 15 Classification of neural hardware according to Saumil G. Merchant

Examples

- In the class of digital implementation the following circuit references are cited: (Ayala et al. 2002), SYNAPSE (Ramacher and Ruckert 1991; Moussa et al. 2006), CNAPS (Hammerstrom 1990), MY-NEUPOWER (Sato et al. 1993)
- In the class of analog implementation the following circuit references are cited: (Tang et al. 1993; Linares-Barranco et al. 1993), ETANN (Holler et al. 1989; Schmitz et al. 2003)
- In the class of hybrid implementation, the following references are cited (DeYong et al. 1992; Passos Almeida and Franca 1993)

Advantages and disadvantages Both analog, digital and hybrid implementation were considered. In the class of digital implementation, design features related to ASICs and FPGA implementation of ANN such as data representation, flexibility, etc. were considered. However, design issues related to analog and hybrid were not considered. Also, new design techniques and approaches, such as systems on chip, were not considered in the classification approach proposed by Saumil G. merchant

4 Synthesis of different classification approaches

Based on the different classification approaches mentioned above:

- In the first classification approach, Nordstrom identified four level of granularity that can be used to efficiently parallelize a neural network on a massively parallel machine. His classification allows comparison between different ANN hardware implementations available at that time. Nevertheless, the author was interested only on the classification of digital implementations of neural networks on parallel machines.
- Aron Ferruci introduced new criteria for classification of neural hardware such as data representation, precision, technology used and the mapping of the algorithm. The major drawback of this study is that the author was only interested in the classification of hardware implementations specific to the backpropagation algorithm.
- Glesener introduced a new criterion named the biological evidence. Unfortunately we were unable to collect enough information about his work.

- Paolo Ienne defined criteria related to performance and flexibility of neural hardware. His classification allowed distinction between specific systems dedicated to one type of algorithm and programmable systems dedicated to several types of algorithms. Nevertheless, the classification concerned only digital implementation.
- Hemskerk introduced the concepts of "Standard chips" and "Neurochips". His classification was more general, because unlike the previous approaches, analog and hybrid implementations were considered in the classification.
- Isik Aybay distinguished between "Neurochip" and "Neurocomputers". Also, he introduced new attributes that highlight the architectural features of the hardware in question. However his approach does not seem to give value to performances measures.
- Beiu proposed a classification approach based on the publication date and technological changes of the neural hardware. Nevertheless, the author was interested only in digital implementations. Achievements like analog and hybrid (analog / digital) do not appear in the classification of Beiu.
- Schonauer proposed a classification in which he gathered all the criteria proposed by Aybay Isik, Arron Ferruci and Nordstrom. However, his classification focused only on hardware implementations using several processors communicating with each other. Design aspects of analog and hybrid do not appear.
- Sorin Draghici, presented several criteria for classifying neural networks, however and contrarily to the previous approaches he focused on analog hardware implementations only.
- Clark Lindsey proposed a similar classification of Heemskerk's, however he used the terminology "VLSI implementation" instead of "Neurochip" which was used by Heemskerk in his classification aproach.
- Nichols introduced new criteria related to the learning algorithm as the "on-chip learning" and "off-chip learning". He also used the "multiplier reduction scheme" as a criterion for classification. Nevertheless, his study focused on FPGA implementations only.
- Vipan Kakkar focused on custom implementation of neural networks. Both analog, digital and mixed design techniques were considered. However recent advances such as system on chip and embedded systems were not considered.
- Saumil G. Merchant considered analog, digital and hybrid implementation. In the class of digital implementation, design features related to ASICs and FPGA implementation of ANN such as data representation, flexibility, etc. were considered. However, design issues related to analog and hybrid were not considered. Also, new design techniques and approaches, such as systems on chip, were not considered in the classification approach proposed by Saumil G. merchant.

In conclusion, the survey of the different classification approaches of neural hardware shows that there is not a clear consensus on the criteria for the classification and on the different definitions of these criteria. For example, Aron Ferruci used the term "interconnection strategy" to designate the types of SIMD architectures, MIMD, Ring, Systolic, etc., while Nordstrom uses the term "complexity of processors" and T. Shchoenauer used the terminology" Inter-processor Communication Network. Also the term "neurocomputer" is used to define either implementation on a "standard chip" or "neurochip" by Heemskerk; while T. Schoenauer uses the same term to define a subclass of neural hardware. In this case, the term neurocomputer refers to a neural system built of software and hardware. That is to say, each approach has brought a new criterion for classification compared to its previous one depending on the area of interest of the author.

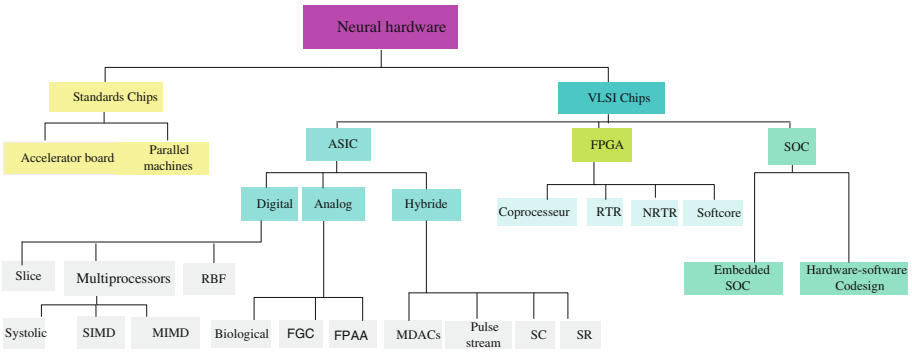

Fig. 16 Proposition of a new approach for the classification of neural hardware

Review of all approaches leads to the conclusion that they are complementary to each other. However, the VLSI technology is still evolving and new design techniques have been introduced to implement neural hardware. This lead to introduce a new classification approach based on new criteria, in the next section.

5 Proposition of a new approach for the classification of neural hardware

Presentation Subsequent to the examination of the previous classification approaches of neural hardware, we propose an approach that takes into account in one hand the consensual classification criteria and in the other hand the evolution of the VLSI technology, and the design techniques.

Figure 16 shows the proposed approach for classification of neural hardware.

Classification criteria First, the circuits are classified according to the criteria:

- *Standard Chips*
- *VLSI Chips.*

In the class of *Standard Chips* are classified implementations based on

- *Accelerator boards*
- *Parallel machines*.

In the class of *VLSI Chips*, are classified implementations based on

- *ASICs*
- *FPGAs*
- Systems on Chip "*SOC*".

This classification is made according to the evolution of the design technology of integrated circuits and design techniques.

In the class of *ASICs* are classified realizations of ANNs circuits based on :

- *Digital implementation,*
- *Analog implementation*
- *Hybrid implementation.*

In the class of *digital implementation* are classified the following architectures:

- *Slice architecture*
- *Multiprocessor architecture*
- *Systolic architecture*
- *SIMD architecture*
- *MIMD architecture*
- *RBF architecture*

In the class of *analog implementations*, we identified the following:

- Circuits dedicated to implement a *biological function* (synaptic retina, cochlea)
- Circuits to overcome issues related to *Synaptic weights* (Floating gate CMOS :FGC)
- Circuits to overcome issues related to *Neuron's activation function* implementations such as the charge coupled devices.

In the class of *hybrid implementation*, we have identified the following ANN based implementations techniques:

- Multiplying Digital to Analog Converters "MDAC",
- PULSE STREAM,
- Switched Capacitors "SC" and
- Switched Resistors "SR".

In the class of *FPGAs*, we have identified the following:

- Implementations where the FPGA is used as a *Coprocessor*
- Implementations that exploit *Run Time Reconfiguration* (RTR)
- Implementations that use a softCore in which neural network is described using the VHDL or VERILOG language.

In the class of System on-chip "SOC", we have identified implementations where the processor is embedded inside the FPGA "embedded system on-chip" and those for which the processor is outside the FPGA "Hardware-Software codesign".
Examples of hardware implementations of ANNs are presented in the following section.

6 Examples of neural hardware implementations

In this section, a description of different neural hardware implementations types is presented, followed by examples of well known circuits.

6.1 Standards chips based implementations

Implementations based on standard chips can support a wide variety of neural networks models. Figure 17 shows the overall architecture. The structure consists of a matrix of identical parallel processors interconnected through a parallel broadcast bus. Each processing unit "PU" on the bus executes a section of the "virtual" network. To program the neurocomputer, the virtual Processing Elements "PEs" are partitioned across the local memories of the physical processors. Updating a PE implies broadcasting the update through the bus. Units that need access to that information store the update in the system state memory. Standard chips can be divided in two categories: accelerator boards based implementations and multiprocessors based parallel implementations.

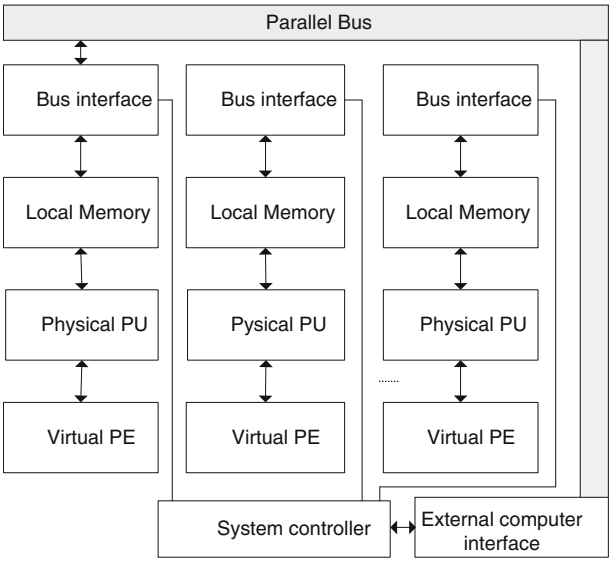

Fig. 17 Standard chip based implementation architecture

6.1.1 Accelerator boards based implementations

Accelerator boards are the most commercialized neural hardware, because they are cheap, widely available and easy to connect to a PC or a workstation. Often, they are accompanied with a user-friendly software tool. The speed that can be achieved is about one order of magnitude compared to sequential implementations. A drawback of this kind of neural hardware and software tools is their lack of flexibility because they do not offer many possibilities for setting up new paradigms or algorithms. Some examples of commercially neural accelerator boards are presented below:

ANZA Plus "HNC: Hecht-Neilson cooperation CA, USA"
This type of card contains the MC68881 processor and the NC68020 floating point coprocessor. Performance rates are given in Treleaven et al. (1989): 1 M virtual PEs, 1.5 M Connections, 1.500 CUPS, 6MCPS. Software: User Interface Subroutine Library (Trealeven 1989).

NT6000 "Neural Technologies Limited, HNC, and Neural California limited software"
The NT6000 (Hanson et al. 1987) is a plug-in PC card equipped with a TMS320 DSP circuit and a NISP (Neural Instruction Set Processor) which speed up neural processing to 2 MCPS. This type of board is supplied with a well developed software package which allows interfacing with other neural simulators like the BrainMaker and the NeuralWorks. The implemented neural network paradigms are, however, limited to the backpropagation and Kohonen maps, and little flexibility is left for the development of new paradigms.

Ni1000 Recognition Accelerator Hardware "Intel Corp., CA, USA"
The Ni1000 is a PC board specially developed for optical character recognition "OCR" applications. It can be programmed by the NestorACCESS software and mainly uses radial basis functions (RBF) and backpropagation algorithms. It gains a speed-up of 100–1,000 times compared to implementations on a single DSP or a typical host processor (Nestor 1994).

Neuro I and Neuro II Turbo card "Nagoya Institute of Technology, Mitec Corp., Japan"

The first version of Neuro-I Turbo, consists of 4 Motorola DSP circuits placed on a board, then a second version which speed up the original concept by a factor of 2 was proposed. The architecture was based on 16 Fujitsu DSPs: MB68220 arranged in a hypercube. The Neuro II Turbo card was based on a general purpose floating point DSP circuit which contains the Motorola DSP96002 circuit. The architecture was based on a matrix memory with a broadcast bus. The performances are 16.05 MCUPS (Onuki et al. 1993).

6.1.2 Parallel machines based implementations

Parallel machines are implementations based on the use of multiple arrays of processors: multiprocessors. Implementations can range from a simple low-cost architecture to architectures with sophisticated processors like the transputer and the DSP circuits. Compared with accelerator boards; these implementations offer greater flexibility of programming the neuronal functions and achieve high computing performances. However, programming such machines requires extensive knowledge of machine architecture and the detailed setting of the software. The best-known achievements are cited as follow:

The connection machine CM 1/2/5 "Thinking Machine Corporation (TMC) Cambridge, Massachusetts"

The connection machine (Hillis and Steel 1986) is one of the most popular parallel machines based implementation for neural computing. The first version was called the CM-1 and was developed during the period 1981–1986. The second version was called CM-2 and was launched in 1987. The CM-1 and CM-2 took the form of a cube 1.5 meters on a side, divided equally into eight smaller cubes. Each sub-cube was composed of 16 printed circuit boards and a main processor called a sequencer. Each printed circuit board was composed of 32 chips. Each chip was composed of a communication channel called a router, 16 processors, 16 RAMs. The CM-1 and CM-2 architecture were based on the SIMD communication strategy. Compared to the CM-1, the CM-2 added the WEITEK 3132 floating-point co-processor and more RAMs. The software for the CM-1 and CM-2 was based on the Lisp programming language. In 1992, the Thinking machine switched from the CM-2 to CM-5 which is MIMD architecture based on a fat tree network of SPARC RISC processors (Fig. 18).

The NETSIM system

The NETSIM system (Garth 1987) consists of a set of accelerator boards arranged in a 3D structure and controlled by a host computer as shown in Fig. 19. Each accelerator board is composed of:

- A standard microprocessor type 80188 and its associated memory program
- Two dedicated circuits: the simulation circuit and the communication circuit
- A memory for storing the synaptic weights, the number of input neurons and the number of neural networks per card.
- A Bios system.

The simulation circuit is used as a coprocessor to perform computation of different neural models. The communication circuit allows connection of the different boards. This circuit consists of a bus of 64 bits. The first 16 bits represent the destination address; the remaining 48 represent the data. The addressing scheme allows 15 NETSIM cards to be connected in three dimensions.

The host computer is used to set initial conditions and characteristics of the neural network.

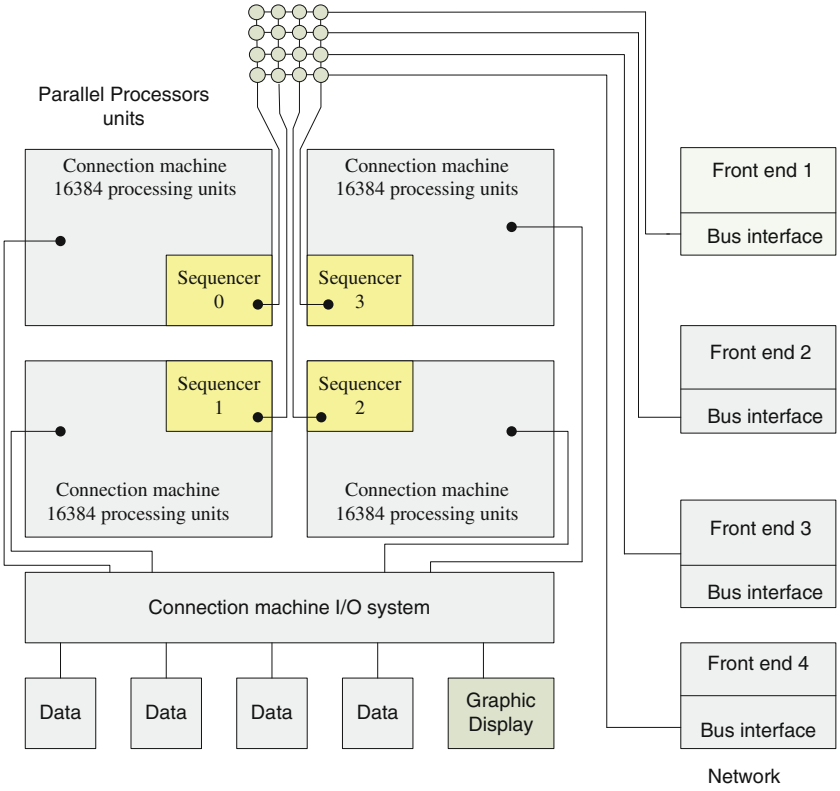

Fig. 18 Architecture of the connection machine (Hillis and Steel 1986)

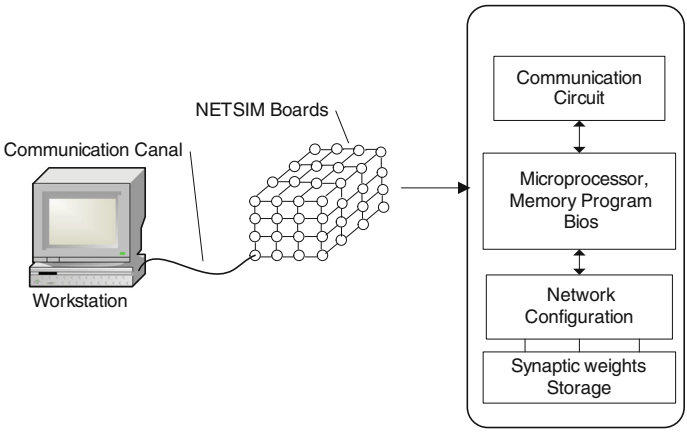

Fig. 19 Architecture of the NETSIM system (Garth 1987)

The IBM GF11

IBM GF11 (Beetem et al. 1987) is an experimental SIMD machine with 566 processors interconnected through a BENES network and achieving a peak performance of 11 gigaflops (Fig. 20). Each processor is capable of 20 million floating or fixed point operations per

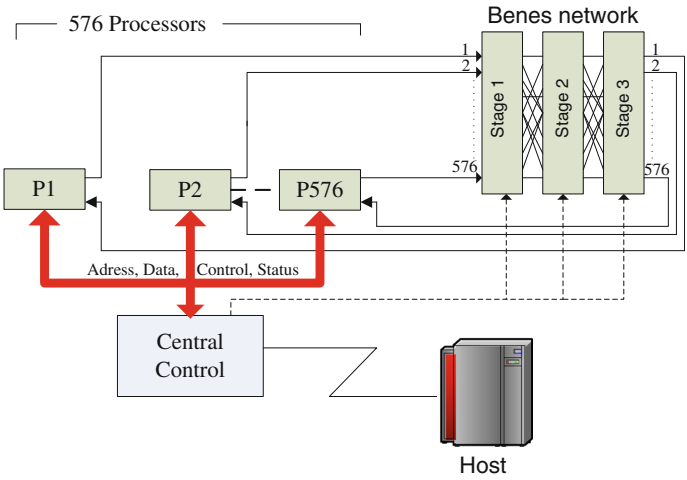

Fig. 20 Architecture of the IBM GF11 machine (Beetem et al. 1987)

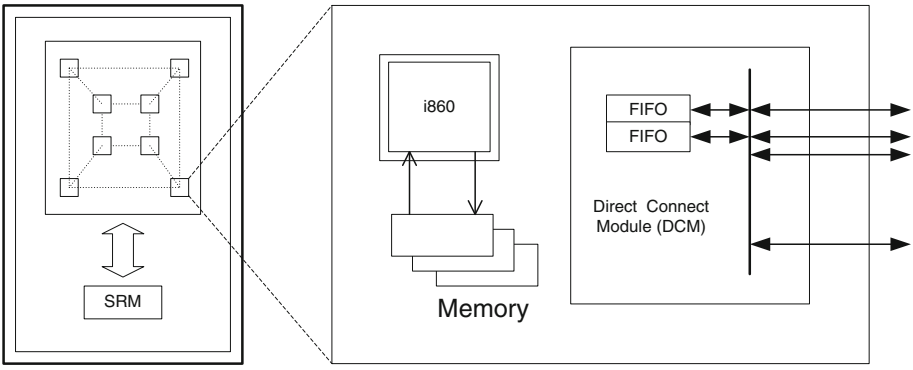

Fig. 21 The iPSC/860 machine (Jackson and Hammerstrom 1991)

second. The processors contain static memories of 16 Kbytes of static memory and 512 Kbytes of dynamic memory. The GF11 has been evaluated on the NETtalk, tool, using a network of 203-60-26 dimension. With 356 operational processors, the GF11 realizes 901 MCUPS.The authors estimate that with 566 operational processors, 1231 MCUPS can be reached.

The iPSC/860 machine

The iPSC/860 machine (Jackson and Hammerstrom 1991) has a MIMD parallel architecture with distributed memory. It consists of nodes interconnected using a hypercube topology. All the nodes are connected to the Service Resource Manager (SRM) which links the iPSC/860 with the external network and which also manages access to the disk (Fig. 21).

Each node consists of the Intel i860 processor with 16 Mega bytes of memory and a routing module (DCM) of 8 channels which allows emission and reception of messages: 1channel is reserved to system input /output (I / 0) and 7 channels to other nodes. These last ones allow a maximum degree of connectivity (total of 128 nodes or processors). Programs can be

written in a superset of C that supports inter-processor communication and other necessary functions.

It should be noted that the iPSC/860 is a very powerful parallel computing machine. However, it presents a disproportion between the speed and the communication time; for example, the data used in the NETtalk (Sejnowski and Rosenberg 1987) project was implemented on the iPSC/860+, using different numbers of processors. The IPSC/860 with 32 processors achieved 12 MCUPS, while the iPSC/860 with 1 processor, achieved 1 MCUPS!

6.2 Implementation of neural hardware on VLSI chips

There are two main approaches to VLSI implementation of neural networks (Ramacher et al. 1991):

- Implementations that integrate the learning phase in the same circuit "on-chip training"

Implementations that incorporate only the generalization phase. In such circuits, the learning is done in software which generates the synaptic weights "off chip training". This kind of implementation is the most used, and the circuits are made very efficient, however, they are specific to one application and cannot be adapted to other applications.

Three major design approaches can be considered in VLSI implementation: ASIC, FPGA and SoC. The choice of one approach over another one depends on several factors such as speed, accuracy, flexibility, programmability, degree of parallelism degree, transfer function and data storage.

6.2.1 Implementation on ASIC

6.2.1.1 Analog implementations Analog circuits are simple and very fast (10–100 times faster than digital circuits), but their implementation on silicon is tedious and requires a good knowledge of the physical laws and mathematical models of transistors, basic circuits, as well as a good experience of layout design. The analog circuits have attractive features that can be directly used to implement neural networks. As shown in Fig. 22, the operational amplifier can be used to implement the transfer function of a neuron and the synaptic weights can be

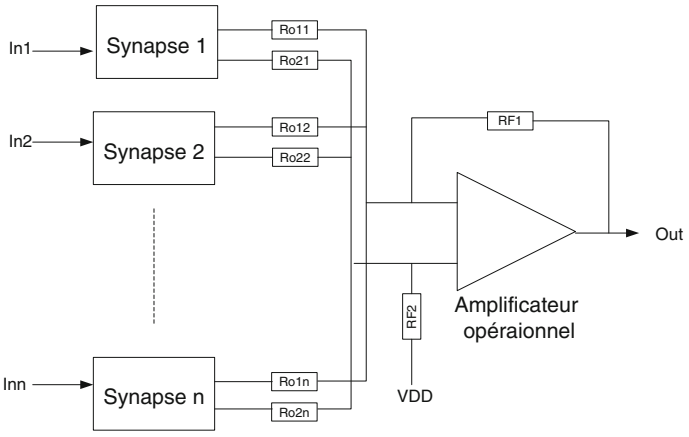

Fig. 22 Architecture of an analog neuron

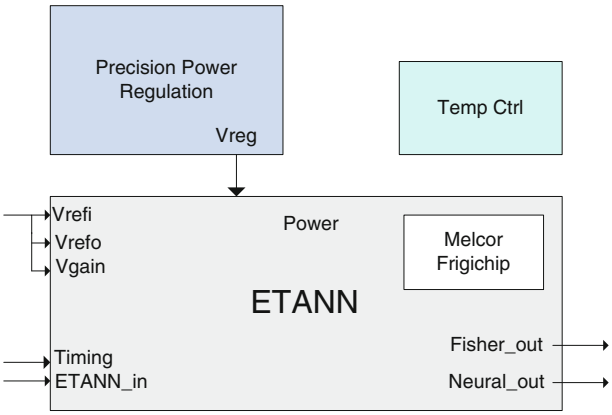

Fig. 23 The ETANN Chip (Holler et al. 1989)

implemented using resistances or their equivalent circuit (Prange 1991). From a characteristic point of view, the drawback of analog circuits is their sensitivity to noise, temperature, voltage supply; they are less accurate and consume a large amount of area.

From a design point of view, the resistors are not adjustable (programmable), which limits the implementation of the neural network approach only to "off-chip training". Other techniques are used to overcome this problem, such as the floating gate CMOS resistors "FGC" (Holler et al. 1989), the Charge coupled device "CCD" technology (Goser et al. 1989), the use of field programmable analog arrays (FPAAs) and the use of biological model.

Examples of analog neurochips are described below:

The ETANN circuit (Intel Inc., CA, USA) (Floating Gate CMOS)
The Electrically trainable Analog Neural Network (ETANN-801770NX) is the first analog neurochip developed (Holler et al. 1989). The circuit considers "off chip training" implementation of up to two layers of an ANNs. It accepts up to 64 inputs (neurons) and 1,024 synaptic weights and can be configured for 64 neurons in the hidden layer and 64 neurons in the output layer. Each neuron is connected to 16 fixed internal bias voltages.

The synaptic weights circuit is based on the Gilbert multiplier (Castro et al. 1993). The outputs from each synapse are summed and presented to a threshold amplifier which models the neuron. The circuit performs 2GCPS. Figure 23 shows the ETANN 80177NX board.

The silicon retina (biological model)
The silicon has an architecture inspired from Biological retina which is composed of a thin layer of neural tissue that lines the back wall inside the eye. Some of these cells act to receive light, while others interpret the information and send messages to the brain through the optic nerve as shown in Fig. 24a. The computation of the silicon retina is based on models of computation in distal layers, which include the cones, the horizontal cells, and the bipolar cells. The cones are the light detectors. The horizontal cells average the outputs of the cones spatially and temporally. Bipolar cells detect the difference between the averaged output of the horizontal cells and the input. The first vision chip which implemented a biological retina on silicon was proposed by Mahowald (1994). Then, (Mead and Ismail 1989) proposed an enhanced version. Figure 24b shows the hardware equivalent model. In this silicon retina, the cones have been implemented using parasitic phototransistors and MOS-diode logarithmic current to voltage converters. Averaging is performed using an hexagonal network of active

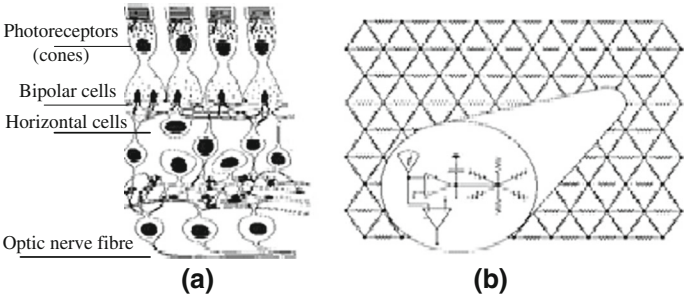

Fig. 24 **a** Biological synaptic retina **b** hardware equivalent model

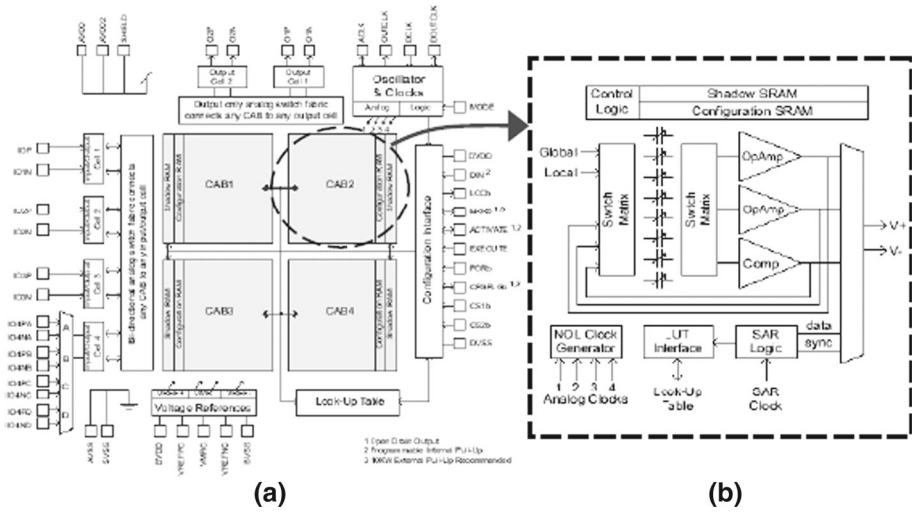

Fig. 25 **a** Anadigm® AN221E04 chip architecture. **b** Architecture of the Configurable analog bloc "CAB" (Rocke et al. 2008)

resistors. Other enhanced versions of the synaptic retina are proposed in Culurciello et al. (2003), Kameda and Yagi (2003) and Sanada et al. (2010).

Field Programmable Analog Arrays (FPAA)
In Rocke et al. (2008), Rock et al. have used a network of cascaded FPAAs to implement an evolved analog spiking neural network (SNN). The FPAA platform is based on the AN221E04 from Anadigm Inc. (AN221E04 2010). The Anadigm® AN221E04 is a reconfigurable analog device based on the switched capacitor technology. Figure 25a shows the schematic of the chip. Different circuit configurations are achieved through manipulation of switches between various circuit components inside four configurable analog blocks (CABs). Each CAB contains two op-amps, a comparator, and 8 variable capacitors (Fig. 25b). Performance of the evolved analog hardware was compared to that of software based SNN controller. Authors showed that the evolved hardware network display an improved tolerance to changing environments compared with networks which evolved only in simulation.

6.2.1.2 Digital implementations Compared with analog circuits, digital circuits are less sensitive to noise, variation in temperature and supply voltage. Moreover, these circuits are

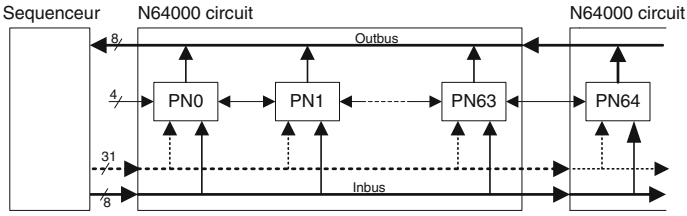

Fig. 26 CNAPS architecture (Hammerstrom 1990)

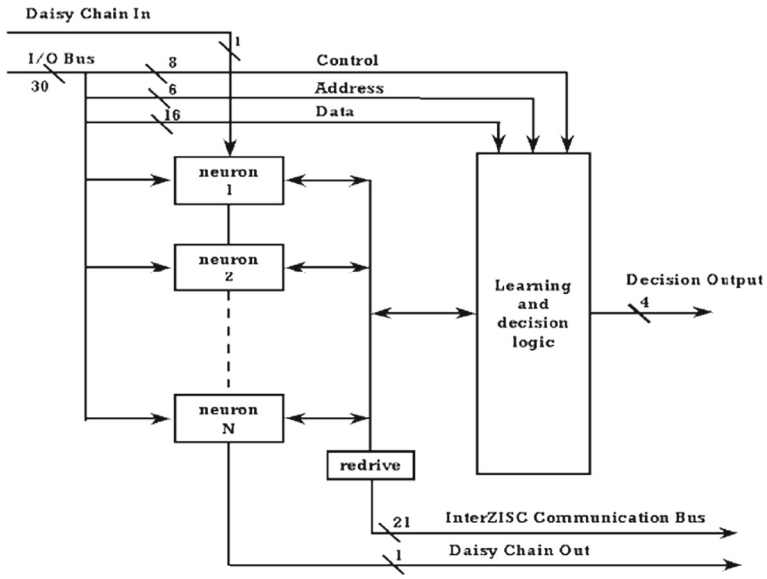

Fig. 27 Architecture of the IBM ZISC036 chip (David et al. 1999)

more accurate and can integrate large networks. Examples of digital implementations are cited below:

The CNAPS system (Adaptive Solutions, Inc., USA)
CNAPS (Connected Network of Adaptive Processors) is a multiprocessor SIMD based architecture. The basic building block of the CNAPS system is the neurochipN6400 (Hammerstrom 1990). As shown in Fig. 26, the N6400 itself consists of 64 processing elements (referred to as processing nodes PN). The N6400 chips are connected in cascade and communicate with a broadcast interconnect scheme. Each N6400 contains 64 processing units and 256 Kbytes weight memory (1–16 bits resolution). The CNAPS system supports the Kohonen and the backpropagation algorithms. For example, a single chip can perform 1.6 GCPS during the feed forward learning phase and 256 MCUPS during the updating phase of the backpropagation algorithm.

The ZISC036 IBM chip (IBM Micro Electronics, France)
The Zero Instruction Set Computer, *ZISC036*, (David et al. 1999) is the first chip developed by IBM Paris semiconductor for implementation of RBF-like neural network-type. The chip was designed for pattern recognition and classification in real-time. Figure 27 shows the general architecture of the ZISC processor. The first generation of ZISC chip contains 36

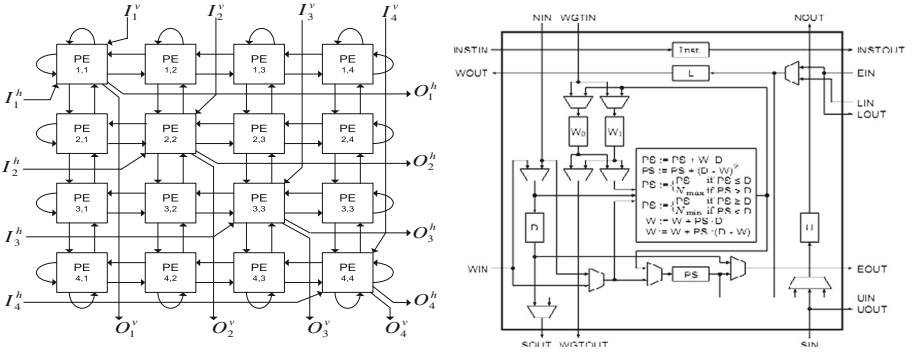

Fig. 28 Architecture of the MANTRA-I (EPFL) architecture (Viredaz and Ienne 1993)

independent cells that can be thought of as neurons or parallel processors. Each of these processors can compare an input vector of up to 64 bytes to a similar vector stored in the "cell" memory. If the input vector matches the vector in the memory, then the cell "fires".

Two important features of the ZISC processor are parallelism and scalability: a ZISC network can be expanded by adding more ZISC devices without suffering a decrease in recognition speed. With the evolution of technology, the ZISC chip contains 78 neurons per chip and can find a match among 1 million patterns in 1 s @ 50 MHz. An example of the use of the IBM ZISC processor is the CogniSight/cogniMen project which is a hardware neural system used for fish inspection (Menendez and Paillet 2008).

MANTRA-I (EPFL, Swiss)
Compared to the previous approaches where only neuron's parallelism was achieved, the MANTRA-I chip is one of the rare implementations which address synapse level parallelism (Viredaz and Ienne 1993) Developed at the Ecole Polytechnique Fédérale de Lausanne (EPFL), Mantra-I is aimed at a multi-model neural computer which supports several type of networks and paradigms. Figure 28 shows the architecture It consists of a 2-D array of 40×40 elements systolic processors called GENES-IV (Generic element for Neuro-Emulator Systolic Array) (Ienne 1993). The GENES chips are bit-serial processing elements that perform vector/matrix multiplication. The MANTRA machine is in principle very well scalable. Performances evaluations for the backpropagation algorithm composed of 1600 PEs are: 400 MCPS for the feed forward phase and 133 MCUPS for the updating phase.

6.2.1.3 Hybrid analog/digital implementations Hybrid or mixed design techniques aim to combine the advantages of digital and analog circuits. However, in order to combine these two techniques the analog circuit's process technology must be compatible with the digital process technology in question. There are several mixed analog/digital design techniques as follows:

Multiplying Digital-to-Analog Converters "MDAC"
In this technique the synaptic weights are digitally encoded and stored in RAM cells. And the inputs signals are encoded in analog form. Subsequently, the weights are transferred to the MDAC circuit which performs multiplication of the synaptic weights with the analog input signal. This approach can be done without the use of Analog/digital converters. However, it offers a high VLSI design cost for medium and large ANNs.

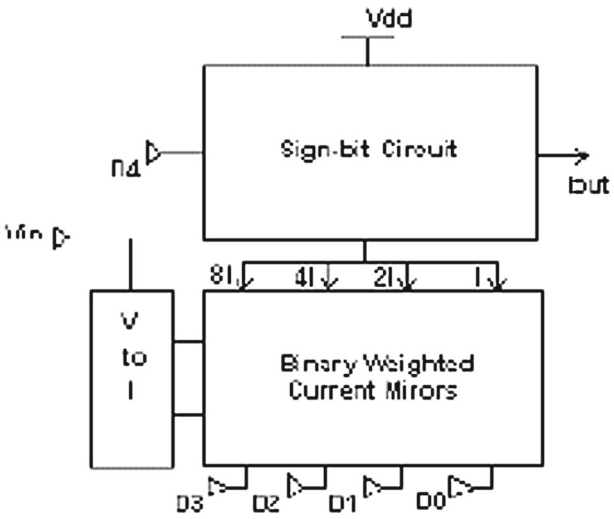

Fig. 29 Bloc diagram of the MDAC circuit (Sudarshan et al. 2010)

For example, in Sudarshan et al. (2010), a MDAC synapse neuron for analog neural networks was proposed. The block diagram of the MDAC synapse is shown in Fig. 29. The main components of the circuit are Sign bit circuit, voltage-to-current converter and binary weighted current mirrors. MDAC receives a 5-bit sign magnitude input from the weight memory (D0 to D4). D0 to D3 of the weight is given as input to the binary weighted current mirror circuit and the sign bit D4 is given as input to the sign bit circuit. The analog input voltage (Vin) is given to the V-to-I converter and the output obtained is a current value (Iout) which is proportional to the product of the 5-bit sign-magnitude digital weight and the analog input voltage (Vin). The schematic of the MDAC circuit is shown in Fig. 30. Transistors M1, M2 and M3 are used for the V-to-I conversion of the analog input. Transistors M4 to M15 form the binary-weighted current mirrors part of the circuit. Transistors M16 to M21 are used in the sign-bit circuit. The circuit is implemented in 0.1μm technology process.

Pulse stream implementations techniques

Pulse stream are a class of modulation techniques widely used in the field of electronic such as telecommunications. The first neural implementations which used pulse stream techniques were proposed in 1987 by Murray and Smith (1987, 1988) and Murray et al. (1991). In this type of implementations, the information is carried by a train of pulses. In turn, pulse stream neural networks can be subdivided into circuits using pulse amplitude modulation, width modulation, frequency modulation and phase modulation. This implementation is obtained by performing analog multiplication under digital control. The advantage of this technique is high noise immunity, insensitivity to signal attenuation, ease of multiplexing, low power and energy requirements. The analog multiplication is attractive in neural VLSI for reason of compactness, speed and lack of quantization effects compared to digital implementation which consumes more area and power. On the other hand, pulse stream implementations suffer from a few drawbacks when compared to analog implementations such as higher electromagnetic interferences (due to switching of binary waveforms); incompatibility with sub threshold operation of MOS in very low power systems; signal cross talk and larger size. Compared to digital implementations, the drawbacks are namely limited accuracy and higher noise sensitivity.

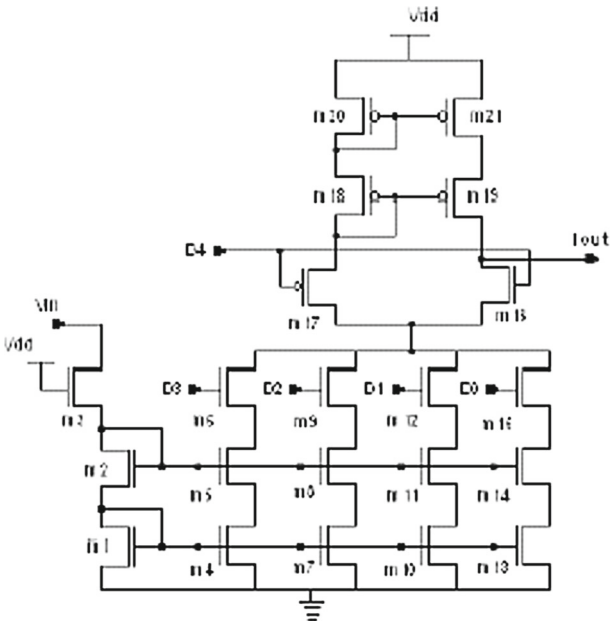

Fig. 30 Architecture of MDAC (Sudarshan et al. 2010)

Switched Capacitor (SC) techniques

ANNs based switched capacitor circuits are adequate for solving optimization problems, linear and non linear programming and speech recognition. In Rodriguez et al. (1990), a non linear "SC" neural network was proposed for optimization problems. In Bor and Wu (1996), the authors proposed an electronic cochlea circuit based on the "SC" technique.

Switched Resistor Techniques (SR)

"SC" and " MDAC" techniques are expensive and require a great deal of area. To overcome these problems the switched resistor technique is proposed. To do so, a MOS transistor can be used for the realization of the switch. This realization can be achieved by applying a clock signal to the gate of the MOS transistor. Figure 31a shows the MOS transistor and its equivalent resistance model is depicted in Fig. 31b, where:

$$R_{on} = \frac{V_s - V_d}{I_{on}} \tag{4}$$

where V_s and V_d, are the source and drain voltages of the MOS transistor, respectively. I_{on}, represents the current of the transistor. Figure 31c shows the clock signal.

In Tsividis and Anastassiou (1987), the author shows that the resistance of the transistor, controlled by the clock, is written as:

$$R = R_{on}\frac{T}{\tau} \tag{5}$$

where, T represents the period of the clock signal and τ the pulse duration.

Thus using this technique, one can make a programmable resistance representing the synaptic weights, which is an alternative to the problem weight programming or modification in analog design.

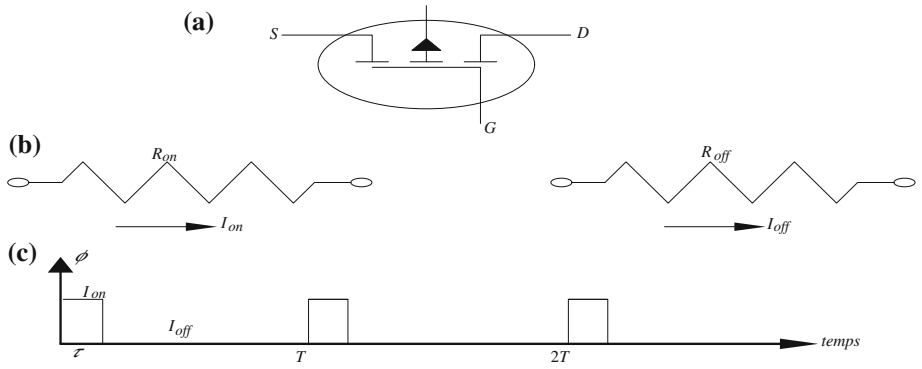

Fig. 31 **a** MOS transistor, **b** Equivalent model, **c** Clock signal

The clock can be realized by programmable logic arrays "PLA" (Blair 1992).

The disadvantage of the approach is the noise due to parasitic capacitances between the substrate and the source in one hand and the substrate and the drain in the other hand.

FPGA implementation of ANNs The first use of FPGA circuits in the implementation of neural networks dates back to the early 1990s'. Often the term *reconfigurable computing for ANNs* is cited in the literature (Dehon 2000). This includes all the techniques that accelerate the performances of an application or a given neural algorithm, above those obtained by general purpose standard chips circuits.

In general, FPGAs circuits can achieve a better compromise between the flexibility of standard chips and the performances obtained using ASICs implementations.

The first FPGAs devices, such as the Xilinx XC3000 and XC4000 families, were very limited in term of performances and integration density, which prevented the implementation of neural networks of large size. Nowadays, with the advances in the development of both of the microelectronic technology and the specific CAD tools, FPGAs have progressed in an evolutionary and revolutionary way. The evolution process has allowed faster and bigger FPGAs, better CAD tools and better technical support. The revolution process concerns the introduction of high performances multipliers, Microprocessors and DSP functions. This has a direct incidence to FPGA implementation of ANNs and a lot of research has been carried to investigate the use of FPGAs in ANNs implementations, such as those cited in Omandi and Rajapakse (2006). The main research topics concern the exploitation of the configuration and reconfiguration of FPGAs devices, the use of FPGAs as hardware accelerators (coprocessors), and the development of ANNs softcores. In what follows we present a state of the art on each topic.

6.2.2.1 Using the configuration and reconfiguration of FPGAs Most of FPGA implementations of neural networks aim to exploit either the configuration or the reconfiguration features. Identifying the purpose of the configuration can help to understand the motivation behind each type of implementation. Generally the style of configuration fails into two groups, static configuration and dynamic reconfiguration.

- The static configuration is the process of loading data of a specific design into one or more FPGAs which define the functional operation of internal blocs and their interconnection.

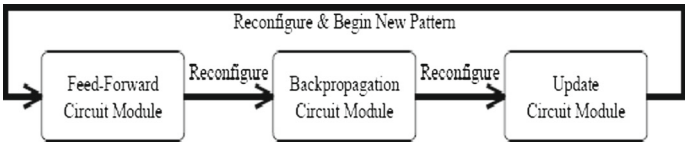

Fig. 32 Partial dynamic reconfiguration: RRANN (Eldredge 1994)

- The dynamic reconfiguration of FPGAs (partial or global) is a solution that permits to use the smallest FPGA and to reconfigure it several times during the processing (Lysaght 1991).

Static configuration
In this class, direct configuration and reconfiguration of the FPGA chip is used on an unlimited basis for different ANNs implementations strategies in order to run simulations and proof of concepts in a very short time. The GANGLION project (Cox and Blanz 1992) is a good example of rapid prototyping of neural networks which encountered a great success.

GANGLION implements a multi layered neural network consisting of (12-14-4) neurons, and is used in image processing. The network in question was implemented on two Xilinx FPGAs XC3090 device-families.

Dynamic reconfiguration
In this class, are presented methods that enhance the integration density per unit area through partial reconfiguration of FPGA. Thus, it is possible to achieve a time multiplexing of an FPGA for the implementation of each sequential phase of a neural network algorithm. Methods of improving the density involve reconfiguring the FPGA. Three techniques are proposed: module based design; difference based and bit stream manipulation. With such techniques, good performances are obtained if the reconfiguration time is small compared to total execution time.

In Eldredge (1994) Eldredge et al., have proposed an approach called RRANN "Run Time Reconfiguration Artificial Neural Network" which consists of dividing the backpropagation algorithm into three modules: the Feed-forward, the backpropagation and the weight Updating module as shown in Fig. 32. First, the Feed-forward module is implemented on the XC3090 FPGA device family; hereafter the circuit is reconfigured to implement the second module and so on. Thus, the implementation of each module is executed in the same FPGA resources.

The authors reported that only 1 neuron per FPGA could be implemented using the static configuration and 6 neurons per FPGA, using the run time reconfiguration approach. Thus the use of RRANN saves 500% of integration density.

In James-Roxby and Blodget 2000, James Roxby implemented a 4-8-8-4 neural network on the Virtex-XCV600BG560 FPGA device family, using constant coefficient multipliers which represent the synaptic weights. James Roxby shows that synaptic weights can be reprogrammed each 69 µs. The same idea was adopted by Zhu et al. (1999).

Upegui et al. (2005) presented a methodology to define the topology of neural networks based on a genetic algorithm and which is implemented using the partial reconfiguration properties of the Virtex-II FPGA device. Figure 33a, shows the principle of partial reconfiguration in Virtex-II and Fig. 33b shows the layout of the reconfigurable network topology. In this figure, each reconfigurable module represents à layer. Each module communicates solely with its neighbor modules through a bus macro. For each module, there is a pool of different possible configurations. A simple genetic algorithm determines which configuration is going to be active.

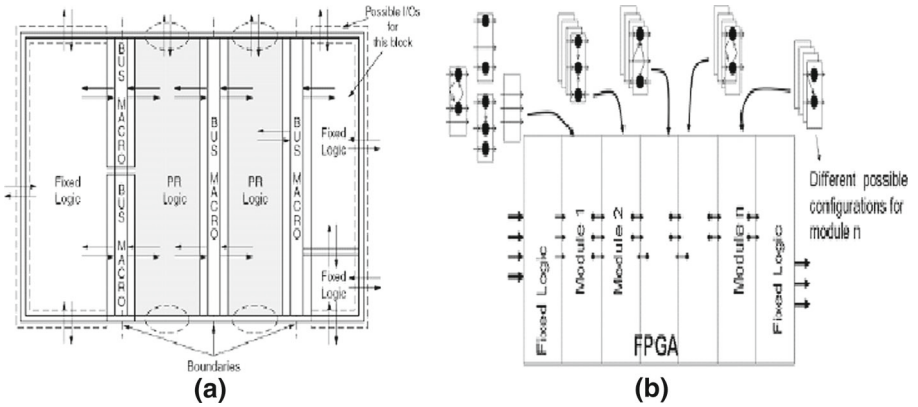

Fig. 33 Andrés Upegui approach: **a** principle of partial reconfiguration **b** Layout of the reconfigurable blocs (Upegui et al. (2005))

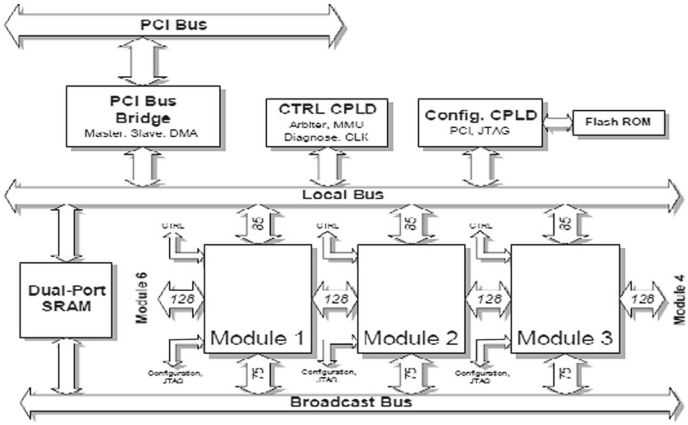

Fig. 34 RAPTOR2000 architecture (Porrmann 2002)

6.2.2.2 Using FPGAs as coprocessor and accelerator cards Porrmann (2002) presented a prototype system called RAPTOR2000 dedicated to the implementation of neural networks in general and especially the Kohonen Self Organizing Map. RAPTOR2000 is an accelerator card composed of three to six application specific modules (ASM) as shown in Fig. 34. The local bus allows internal communication between ASM modules and between the host PC and the ASM. Another "Broadcast Bus" can be used for simultaneous communication between ASMs. In addition, an SRAM is accessible via the various ASM "Broadcast bus". Other functions such as memory management, bus arbitration, error detection and JTAG configuration are integrated in both CPLD circuits. The PCI bus connects RAPTOR2000 to the host computer or the workstation. RAPTOR2000 is also equipped with a daughter card used for neural networks simulations.

Each ASM can be equipped with several types of FPGA VIRTEX circuits, emulating complex circuits from 400,000 to 2.5 million logic gates. The configuration of an ASM can be achieved either by the host computer or through another PCI bus or another ASM. Thus, there is a possibility to configure the FPGA with data located somewhere in the system. The

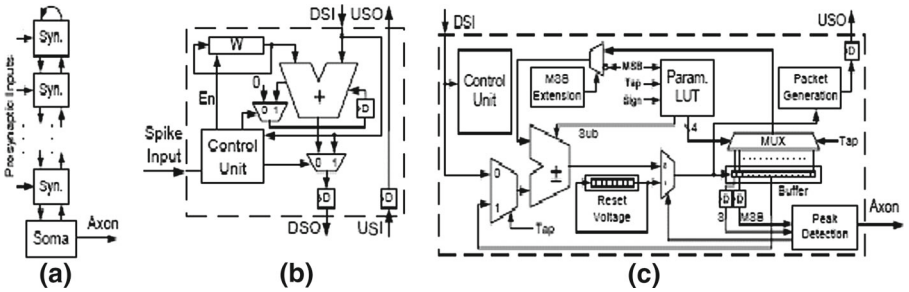

Fig. 35 Heterogeneous spiking neural networks; **a** Neuron architecture, **b** Synapse architecture, **c** Soma architecture (Shayani et al. 2008)

reconfiguration time is less than 20 ms. The Authors reported that, by using five VIRTEX-E FPGA circuits, RAPTOR2000 was capable of performing neural networks which are 190 times faster than those implemented in software.

Shayani et al. (2008) have proposed a digital neuron model suitable for evolving and growing heterogeneous spiking neural networks on FPGAs by introducing novel flexible dendrite architecture and a new soma model based on Piecewise-Linear Approximation of the quadratic integrate and fire model. Figure 35a shows the general architecture of the neuron which consists of a chain of synapse units (syn) and a soma unit connected in daisy chain architecture, Fig. 35b shows the synapse architecture and Fig. 35c the soma architecture. The authors show that a neural network of 161 neurons and 1610 synapses achieves 85% of resource utilization and a maximum clock frequency of 160 MHz, using the Viretx-5 XC5VLX50T FPGA device. The authors showed that this model is 43% faster than the one presented by Schrauwen and Van Campenhout (2006).

6.2.2.3 Using soft-core descriptions The term Soft-Core is used to designate a VHDL/Verilog or other high level description language. Such descriptions allow for flexibility, reuse and a high degree of technology independence. Nevertheless, the resulted circuit is often not optimized in terms of area and speed performances.

Among the works that have addressed ANNs soft core descriptions, one can mention the Soft TOTEM NC3003 chip (Lee et al. 1997) and the work presented by Diepenhorst et al. (1999).

Soft TOTEM NC3003 is a soft core processor, developed and marketed by the Italian company *NeuriCam*, in collaboration with the Italian University *TRONTO* and the English University *KENT* (Lee et al. 1997). The first circuit, NC3000 TOTEM, was designed in 1999 using the full custom design approach and was considered as a hard core. Subsequently, and in order to keep independence with technology and in due time to market constraints; a soft core was proposed in 2002, hence the name Soft TOTEM NC3003.

The NC3003 is designed to implement the multilayer perceptron, but it can also be used to implement certain functions of signal processing such as filtering, convolution, multiplication and accumulation. Figure 36 shows the simplified architecture of the circuit. The main features are summarized as follows:

The architecture is of SIMD type, optimized to perform multiplication and accumulation "MAC" operations in one clock cycle. There are 32 MAC processing units connected in parallel.

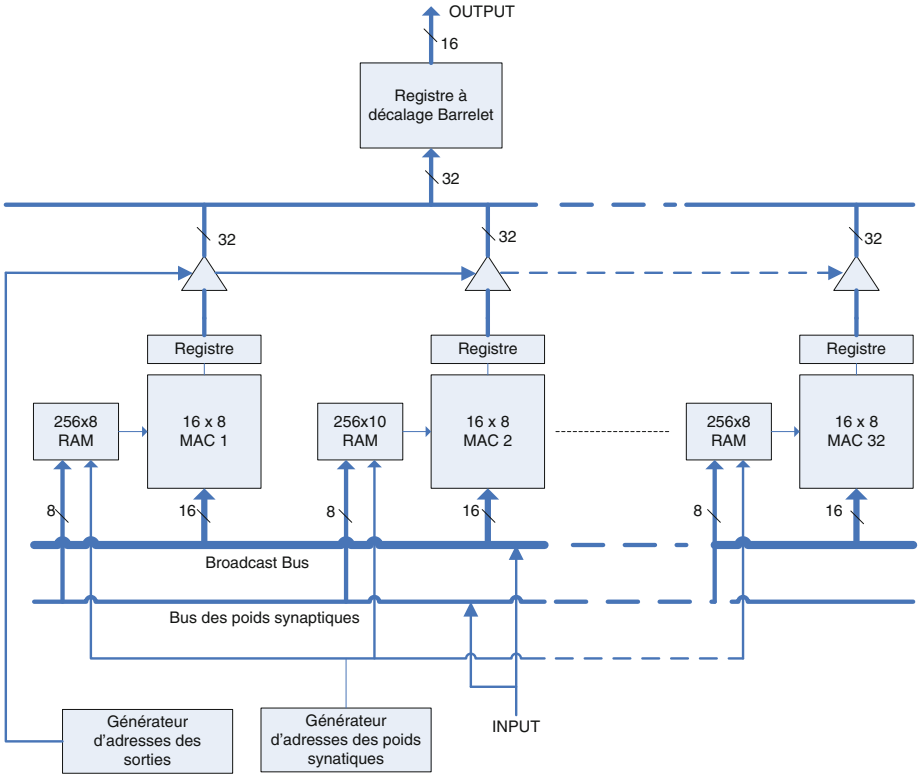

Fig. 36 Architecture of the SoftTOTEM NC3003 (Lee et al. 1997)

- The data are represented in fixed point and complement of 2.
- The synaptic weights are stored in 256 × 8 bits memory
- Data have a limited width format: 16 bit for inputs, 8 bits for synaptic weights and 16 bits for outputs
- The Barrel shift register (Gigliotti 2004) is used to truncate the result.
- The NC3003 can implement a multilayer network of up to 32 neurons with performance of 750 million MAC operations @ 25 MHz. According to the Datasheet, the NC3003 is portable into ASIC or Xilinx FPGA circuit (NC3003, http://www.neuricam.com).

Another example of ANN Soft-core description is the Virtual Neural Processor (VNP) developed by Diepenhorst et al. (1999). Figure 37 shows the main steps of this approach.

The first step is to establish a model of the neuron and the neural network in question. From this model, a VHDL description at the behavioral level is generated, through VHDL code generation. Subsequently, the behavioral description is converted to a structural description of the neural network using a library of elementary logic gates. The last step is to introduce the structural description obtained into a synthesis tool for targeting an ASIC or an FPGA implementation.

Two other examples of ANNs soft-core are described in references (Munoz et al. 2008; Pearson et al. 2005).

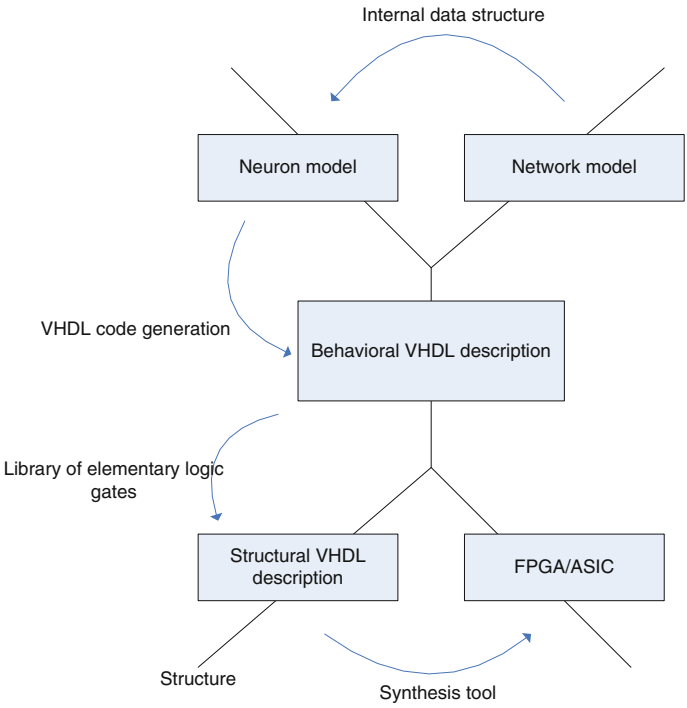

Fig. 37 The Virtual neural processor (VNP) approach, Diepenhorst et al. (1999)

6.2.2 Use of embedded systems on chip and HW/SW co-design to implement ANNs

Wolf et al. 2001 have proposed a system on chip (SOC) based architecture to implement the multilayer perceptron (MLP). Figure 38 shows the system architecture. It consists of the Altera Nios processor which is connected to an ANN (Altera: http://www.altera.com).

The Nios processor implements the forward and learning/training phases of the MLP. The ANN hardware architecture implements only the post training phase of the MLP. Thus the output of the neural network is connected to the input of the processor which performs the synaptic weights update phase. The new values are communicated to the neural network and the same operation is repeated until convergence is achieved.

The hardware architecture of the network model has been designed using Altera's Quartus tool. Each neuron is composed of four multipliers which perform multiplication of each entry with the corresponding synaptic weights. Thus, the obtained results are added to the bias values and finally the transfer function delivers the neuron's output. The software part of the algorithm (learning and training) was implemented using the C language. It is executed by the Nios processor.

To compare the performance of the proposed SOC approach with existing ones, three types of tests were conducted: the first one concerns the implementation of all phases of the MLP algorithm into a dedicated FPGA hardware without the Nios processor. The second test considers implementation of the MLP algorithm entirely on the Nios processor and the last one concerns the implementation of the MLP on an Intel Pentium III based processor.

The results obtained showed that the Intel Pentium-III processor allows an execution time of 23 us @ 550 MHz, the Nios processor gives a result of 144 μs @ 40 MHz, the SOC provides

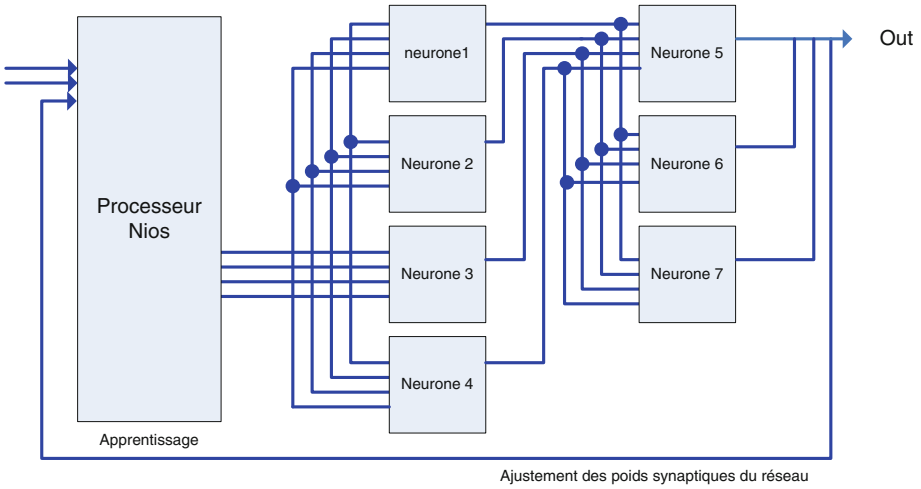

Fig. 38 ANN SOC architecture based on the Altera Nios processor (Wolf et al. 2001)

a runtime of 600ns @ 40 MHz and dedicated FPGA hardware MLP allows an execution time of 32ns @ 40 MHz. It is obvious that the last implementation approach is the most efficient. However, it is difficult to implement and lacks flexibility. The use of the SOC approach can achieve a compromise between the flexibility of software implementations and the performance obtained by using dedicated hardware. The whole SOC has been implemented on the Altera FPGA APEX20k200E.

Another example which uses embedded systems and HW/SW Co design for implementing ANNs was proposed in Uker and Alkar (2006). Figure 39 shows the architecture of the proposed system.

The embedded system is formed by applying the four major changes:

- partitioning the algorithm to be implemented into smaller pieces,
- allocating those partitions to the microprocessors and hardware units,
- scheduling the times at which functions are executed,
- Mapping the generic algorithm description into an implementation on a particular set of components, either as software suitable for a given microprocessor or a logic device which can be implemented from the given hardware libraries.

Based on this principle, the system has been partitioned into two parts: software and hardware. A task manager installed in the host computer, acts as a master controller. Learning is implemented in software in the host computer to determine the synaptic weights that satisfy the convergence criteria. For a specific architecture of the MLP, the RS232 interface ensures the transfer of synaptic weights to the circuit (FPGA XCV600E) acting as a co-processor. The SRAM is used to store the synaptic weights. Once the software unit determines the appropriate synaptic weights, the feed forward operation is executed on the processing units (PUs) to determine the desired output of the network. A finite state machine (FSM) is implemented on FPGA for the synchronization of the feed-forward propagation. The processing units (PU) are arranged in grid architecture to ensure the vector-matrix multiplication. Each (PU) consists of a multiplier accumulator circuit (MAC). The activation function is implemented using (LUT).

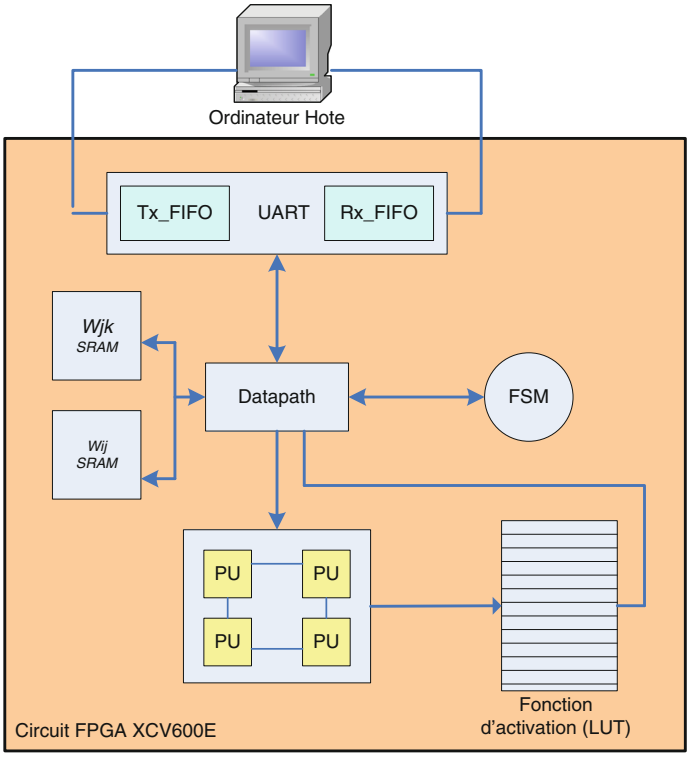

Fig. 39 HW/SW co-design approach for ANN implementation (Uker and Alkar 2006)

The proposed approach was compared with several implementations such as the RRANN (Eldredge 1994), and Hitachi—WSI (Yasunaga et al. 1989). The results showed that the proposed circuit achieves performance of 327 MCPS. These results are better compared to the RRANN (722 KCUPS) and Hitachi—WSI (138 MCPS).

7 Conclusion

Through our new classification approach of neural hardware we have presented the evolution in the design of ANNs during the last three decades. These designs have evolved from standard chips to embedded systems on chip. The choice between one type and another depends on the application field and the physical/technological constraints. In each implementation type a number of issues must be taken into consideration by the designer. These issues can be enumerated as follows:

- The parallelism that allows real-time processing
- Flexibility and adaptability of the circuit
- Floating point representation
- Exploitation of dynamic reconfiguration in the latest FPGA devices
- The use of high level language such as system-C for the IP-soft core descriptions and HW/SW co-design of embedded systems

- Application of the design reuse concept to overcome the gap between the evolution in technology and the design complexity.
- Dedicated platforms for neural hardware implementation

Despite the evolution in technology and the design techniques, these problems are still posed and therefore they constitute guidelines for future research work and investigation.

References

Aleksander I, Thomas W, Bowden P (1984) WISARD, a radical new step forward in image recognition. Sensor Rev 4(3):120–124

Alspector J (1991) .VLSI architecture for neural networks: concepts, applications, and implementations, vol 1 Prentice-Hall, Englewood Cliffs 180–213

AN221E04 Datasheet (2010) Dynamically reconfigurable FPAA with Enhanced I/O, DS030100-U006b. Available in the internet at: http://www.anadigm.com (accessed November 2011)

Anderson D, McNeill G (1992) Artificial neural networks technology. DACS State-of-the-Art Report, Contract Number F30602-89-C-0082

Ayala JL et al. (2002) Design of a pipelined hardware architecture for real-time neural network computations. In: Proceedings of the 45th midwest symposium on circuits and systems MWSCAS'02 Tulsa, Okla, USA, vol 1, pp 419–422

Aybay I, Cetinkaya S, Halici U (1996) Classification of neural network hardware. Neural Network World. IDG Co 6:11–29

Bavan P, Lee MS, Trealeven P (1988) A simple VLSI architecture for Neurocomputing. In: Proceedings of the International Neural Network Society. First annual Meeting, Boston, Massachusetts, p 398

Beetem J, Denneau M, Weingarten D (1987) the GF11 parallel computer, experimental parallel computing architectures. J J Dongarra Elsevier science Publisher, North-Holland

Beiu V (1997) Digital integrated circuit implementations. IOP Publishing Ltd and Oxford University Press Handbook of Neural Computation release 97/1, pp 1–34

Beuchat JL, Haeni JO, Sanchez E (1998) Hardware reconfigurable neural networks. IPPS/SPDP'98 Workshop no 10 Orlando- Florida, vol 1388, pp 91–98

Blair GM (1992) PLA design for single-clock CMOS. IEEE J Solid State Circuit 27(8):1211–1213

Bor JC, Wu CY (1996) Analog electronic cochlea design using multilexing switched capacitor circuits. IEEE Trans Neural Netw 7(1):155–166

Bower MJ, Beeman D (1998) The book of GENESIS: exploring realistic neural models with the GEneral NEural SImulation System. Springer, New York, ISBN 978-0387949383

Castillo FJ, Cabestany J, Moreno JM (1992) The dynamic ring architecture. In: Proceedings of the ICANN-92-Brighton UK. Elsevier, Amsterdam, pp 1439–1442

Castro HA, Tam SM, Holler MA (1993) Implementation and performance of an analog nonvolatile neural network. Analog Integr Circuits Signal Process 4(2):97–113

Christy P (1990) Software to support massively parallel computing on the MasPar MP-1. In: Proceedings of COMPCON, pp 29–33

Cichocki A, Unbehauen R (1994) Neural networks for optimization and signal processing. Wiley, London

Cox CE, Blanz WE (1992) GANGLION: a fast field programmable gate array implementation of a connectionist classifier. IEEE J Solid State Circuits 3:288–299

Cox CE, Mathia K, Saeks R (1995) Learning flight control and LoFLYTE. WESCON'96 Microelectronics Communication Technology Producing Quality Products Mobile and Portable Power Emerging Technologies. doi:10.1109/WESCON.1995.485490

Culurciello E, Cummings RE, Boahen KA (2003) A biomorphic digital image sensor. IEEE J Solid-State Circuits 38(2):281–294

David R, Williams E, De Tremiolles G, Tannhof P (1999) Description and practical uses of IBM ZISC036. Proc SPIE 3728:198–211

De Garis H, Korkin M (2002) The Cam-Brain machine: an FPGA based hardware tool which evolves a 1000 neuron net module in seconds and updates a 75 million neuron artificial brain for real time robot. Neurocomput J 42(1–4):35–68

De Groot AJ, Parker SR (1989) Systolic implementation of neural networks. SPIE, The International Society for Optical Engineering. In: Bromley K (ed) High speed computing II 1058, pp 182–190

Dehon A (2000) The density advantage of configurable computing. IEEE Comput 33(5):41–49

Demuth H, Beale M (1992) Neural network toolbox for use with MATLAB. User Guide Version-4

Denby B (1993) The use of neural networks in high energy physics. Neural Comput 5(4):505–549
DeYong MR, Findley RL, Fields C (1992) The design, fabrication, and test of a new VLSI hybrid analog-digital neural processing element. IEEE Trans Neural Netw 3(3):363–374
Diepenhorst M et al (1999) Automatic generation of VHDL code for neural applications. In: International Joint Conference on Neural Networks (IJCNN), vol 4, pp 2302–2305
Draghici S (2000) Neural NETWORKS IN analog hardware-design and implementation issues. Int J Neural Syst 1:19–42
Driancourt X (1994) Personnal communication. Neuristique Inc. France
Duranton M, Sirat JA (1990) Learning on VLSI: a general-purpose digital neurochip. Philips J Res 45(1):1–17
Dvorak JC (1991) Best of 1990: BrainMaker Professional, Version 1.5, PC Magazine, January 15. Available in the internet at: http://www.calsci.com/referenc.html (accessed November 2011)
Eguci H et al (1991) Neural network LSI chip with on-chip learning. In: International Joint Conference on Neural Networks, vol 1, pp 453–456
Eldredge JG, Hutchings BL (1994) RRANN: the run time reconfiguration artificial neural network. IEEE Custom Integrated Circuits Conference, San Diego, pp 77–80
Eldredge JG (1994) FPGA density enhancement of a neural network through run-time reconfiguration. Master thesis Department of Electrical and Computer engineering, Brigham Young University
Elmasry MI (1994) VLSI artificial neural networks engineering. Kluwer, Dordrecht
Ferrucci AT (1994) ACME: a field programmable gate array implementation of a self adapting and scalable connectionist network. Master Thesis University of California SANTA CRUZ
Fisher WA et al (1991) A programmable analog neural network processor. IEEE Trans Neural Netw 2:222–229
Flynn MJ (1972) Some computer organization and their effectiveness. IEEE Trans Comput 21:948–960
Garth SCJ (1987) A chipset for high speed simulation of neural network systems. In: Proceedings of the IEEE first International Conference on Neural Networks III, pp 443–452
Gascuel JD et al (1991) A digital CMOS fully connected neural network within circuit learning capability and automatic identification of spurious attractors. IEEE Conference on Euro ASIC, pp 247–250
Gigliotti P (2004) Implementing barrel shifters using multipliers. Application Note: XAPP195 (v1.1). Available in the internet at: http://www.xilinx.com (accessed on November 2011)
Glesner M, Huch M, Pochmuller V, Palm G (1989) Hardware implementations for neural networks. Workshop on Parallel Architectures on Silicon, pp 65–7
Glesner M, Pochmuller W (1994) Neurocomputers: an overview of neural networks in VLSI. Chapman & Hall, London, ISBN 0-412-56390-8
Goser K, Hilleringmann U, Rueckert U, Schumacher K (1989) VLSI technologies for artificial neural networks IEEE MICRO, pp 28–44
Gorse D, Taylor GJ, Klarkson TG (1994) Extended functionality for pRAMs. In: International Conference on Artificial Neural Networks, ICANN'94, pp 705–708
Hammerstrom D (1990) A VLSI architecture for high-performance, low cost, on-chip learning. Int Joint Conf Neural Netw 2:537–543
Hanson WA, Cruz CA, Tam JY (1987) CONE-computational network environment. In: Proceedings of IEEE First International Conference on Neural Networks III, pp 531–538
Hasler P, Diorio C, Minch BA, Mead C (1995) Single transistor learning synapses. Advances in neural information processing systems 2. MIT Press, Cambridge 817–824
Hecht-Nielsen R (1990) Neurocomputing. Addison-Wesley Publishing Company, Reading
Heemskerk JNH (1995) Neurocomputers for brain-style processing: design, implementation and application. PhD Thesis, Leiden University the Netherlands
Hillis WD, Steel GLJ (1986) Data parallel algorithms. Commun ACM 29(12):1170–1183
Holler M et al (1989) An electrically trainable artificial neural network (ETANN) with 1024 floating gate synapse. In: Proceedings of IACNN, pp 191–196
Holt JL, Baker TE (1991) Backpropagation simulations using limited precision calculations. Int Joint Conf Neural Netw 2:121–126
Hunt DJ (1989) AMT DAP: a processor array in a workstation environment. Comput Syst Sci Eng 4(2):107–114
Ienne P (1993) GENES IV: a bit serial processing element for a multi model neural network accelerator. In: Proceedings of the International Conference on Application Specific Array Processors IEEE Computer Society, pp 345–356
Ienne P (1995) Digital systems for neural networks. Digital signal processing technology CR57 of critical reviews series. SPIE Optical Engineering, pp 314–345
Jabri MA, Flower B (1992) Weight perturbation: an optimal architecture learning technique for Analog VLSI feed forward and recurrent multilayer networks. IEEE Trans Neural Netw 3(1):154–157

Jackson D, Hammerstrom D (1991) Distributing back propagation networks over the Intel iPSC Hypercube. IEEE Int Joint Conf Neural Netw 1:569–574
Jahnke A, Roth U, Klar H (1996) A SIMD/dataflow architecture for a neurocomputer for spike-processing neural networks (NESPINN), MicroNeuro'96, pp 232–237
James-Roxby P, Blodget BA (2000) Adapting constant multipliers in a neural network implementation. In: Proceedings of IEEE symposium on field-programmable custom computing machines, pp 335–336
Kakkar V (2009) Comparative study on analog and gital neural networks. Int J Comput Sci Netw Secur (IJCSNS) 9(7):14–19
Kameda S, Yagi T (2003) An analog VLSI chip emulating sustained and transient response channels of the vertebrate retina. IEEE Trans Neural Netw 14(5):1405–1412
Kane J, Paquin M (1993) POPART: practical optical implementation of adaptative resonance theory 2. IEEE Trans Neural Netw 4:695–702
Kato H et al (1990) A parallel neurocomputer architecture towards billion connection updates per second. Int Joint Conf Neural Netw 2:47–50
Kung HT (1987) The warp computer: architecture, implementation and performance. IEEE Trans Comput 36(12):1523–1528
Lee P et al (1997) Advances in the design of the TOTEM neurochip. Nucl Instrum Methods Phys Res Sect A Accel Spectrom Detect Assoc Equip 389:134–137
Liao Y (2001) Neural networks in hardware: a survey. Available in the internet at http://bit.csc.lsu.edu/~jianhua/shiv2.pdf (accessed November 2011)
Linares-Barranco B, Sanchez-Sinencio E, Rodriguez-Vazquez A, Huertas J. L (1993) A CMOS analog adaptive BAM with on-chip learning and weight refreshing. IEEE Trans Neural Netw 4(3):445–455
Lindsey C (2002) Neural networks in hardware: architectures, products and applications. Available in the internet at: http://www.particle.kth.se/~lindsey/HardwareNNWCourse/home.html (accessed November 2011)
Lindsey CS, Denby B, Lindblad T (1998) Neural network hardware. Available in the internet at: http://neuralnets.web.cern.ch/NeuralNets/nnwInHepHard.html (accessed November 2011)
Lippmann RP (1987) An Introduction to computing with neural nets. IEEE ASSP Mag 4(2):4–22
Lysaght P (1991) Dynamically reconfigurable logic in undergraduate projects. In: Moore W, Luk W (eds) FPGAs. Abingdon EE&CS Books, England
Mahowald M (1994) Analog VLSI chip for stereocorrespondence. In: Proceedings of IEEE International Symposium on Circuits and Systems, vol 6, pp 347–350
McCartor H (1991) Back propagation implementation on the adaptive solutions CNAPS neurocomputer chip. In: Lippmann R et al (eds) Proceedings of NIPS-3 advances in neural information processing systems, 3 edn, pp 1028–1031
McCulloch W, Pitts V (1943) A logical calculus of ideas immanent in nervous activity. Bull Math Biophys 5:115–133
Mead C, Ismail M (1989) Analog VLSI implementation of neural systems. Kluwer academic publisher, Boston 239–246
Melton M et al (1992) The TINMANN VLSI chip. IEEE Trans Neural Netw 3(3):375–384
Menendez A, Paillet G (2008) Fish inspection system using a parallel neural network chip and the image knowledge builder application. Artif Intell Mag 29(1):21–28
Merchant SG, Peterson GD (2010) Evolvable block-based neural network design for applications in dynamic environments. VLSI Design Hindawi Publishing Corporation. doi:10.1155/2010/251210
Morgan N et al. (1990) The RAP: a ring array processor for layered network calculations. In: Proceedings of the Conference on application specific array processors, Princeton, NJ, pp 296–308
Moussa M, Areibi S, Nichols K (2006) On the arithmetic precision for implementing back-propagation networks on FPGA: a case study. In: Omondi AR, Rajapakse JC (eds) FPGA implementations of neural networks. Springer, Berlin pp 37–61
Muller UA, Gunzinger A, Guggenb'uhl W (1995) Fast neural net simulation with a DSP processor array. IEEE Transactions on Neural Networks, pp 203–213
Munoz AR et al (2008) An IP core and GUI for implementing multilayer perceptron with a fuzzy activation function on configurable logic devices. J Univers Comput Sci 14(10):1678–1694
Murray AF, Smith AVW (1987) A novel computational and signaling method for VLSI neural networks. European solide state circuit conference, pp 19–22
Murray AF, Smith AVW (1988) Asynchronous VLSI neural networks using pulse stream arithmetic. IEEE J Solid State Circuits 23(3):688–697
Murray AF et al (1991) Pulse stream VLSI networks mixing analog and digital techniques. IEEE Trans Neural Netwo 2(2):193–204

NC3003-Digital Processor for Neural Networks (2011) Data sheet, Rel. 12/99 available in the internet at: http://www.digchip.com/datasheets/parts/datasheet/327/NC3003-pdf.php (accessed on November 2011)
Nestor Inc. Providence R.I Ni1000 (1994) Recognition Accelerator Datasheet
Nichols KR (2004) A reconfigurable computing architecture for implementing artificial neural networks on FPGA. Master Thesis, University of Guelph
Nordstrom T (1991) Sparse distributed memory simulation on REMAP3. Research Report N^o TULEA 1991:16, Luleå University of Technology, Sweden
Nordstrom T (1995) On line localized learning systems part II: parallel computer implementation. Research report TULEA 1995:02, Division of Computer Science and Engineering, Lulea University of Technology, Sweden
Nordstrom T, Svensson B (1992) Using and designing massively parallel computers for artificial neural networks. J Parallel Distrib Comput Special Issue Neural Comput Massively Parallel Process 14(3):260–285
Omandi AR, Rajapakse J (2006) FPGA implementations of neural networks. Kluwer, Dordrecht
Onuki J, Maenosono T et al (1993) ANN accelerator by parallel processor based on DSP. In: Proceedings of the IJCNN-93-Nagoya, pp 1913–1916
Passos Almeida A, Franca JE (1993) A mixed-mode architecture for implementation of analog neural networks with digital programmability. In: Proceedings of the International Joint Conference on Neural Networks (IJCNN'93), Nagoya, Japan, vol 1, pp 887–890
Pearson MJ et al (2005) Design and FPGA implementation of an embedded real-time biologically plausible spiking neural network processor. International Conference on Field Programmable Logic and Applications, pp 582–585
Peiris V, Hochet B, Declercq M (1994) Implementation of a fully parallel Kohonen map: a mixed analog/digital approach. IEEE Int Conf Neural Netw 4:2064–2069
Peres-Uribe A (1999) Structure adaptable digital neural networks. PhD thesis 2052, Logic System laboratory, Computer Science department, Swiss federal Institute of Technology- Lausanne
Porrmann M (2002) Implementation of artificial neural networks on a reconfigurable hardware accelerator. 10th Euromicro Workshop on Parallel, Distributed and Network-based Processing, pp 243–250. doi:10.1109/EMPDP.2002.994279
Prange SJ (1991) Architectures for a biology-oriented neuroemulator. VLSI design of neural networks. Kluwer, Dordrecht 83–102
Ramacher U, Ruckert U (1991) VLSI design of neural networks. Kluwer, Dordrecht
Ramacher U, Wesseling M (1990) Systolic synthesis of neural networks. Int Neural Netw Conf 2:572–576
Ramacher U et al (1991) Design of a first generation neurocomputer. VLSI design of neural networks. Kluwer, Norwell 271–310
Ramacher U, Raab W et al (1993) Multiprocessor and memory architecture of the neurocomputer SYNAPSE-1. In: Proceedings of the 3rd International conference on microelectronics for neural networks (Micro Neuro), Edinburgh, pp 227–231
Rocke P, McGinley B, Maher J, Morgan F, Harkin J (2008) Investigating the suitability of FPAAs for evolved hardware spiking neural networks. In: ICES 2008, LNCS 5216, pp 118–129
Rodriguez A, Dominguez CR, Rueda A, Huertas RL, Sanchez-sinenco E (1990) Non linear switched capacitor neural networks for optimization problems. IEEE Trans Circuits Syst 37(3):384–398
Sanada A, Ishii K, Yagi T (2010) A robot vision system using a silicon retina. Brain Inspir Inf Technol 266:135–139. doi:10.1007/978-3-642-04025-2_23
Sato Y et al (1993) Development of a highperformance, general purpose neuro-computer composed of 512 digital neurons. In: Proceedings of the International Joint Conference on Neural Networks (IJCNN '93), Nagoya, Japan, vol 2, pp 1967–1970
Schoenauer T et al (1998) Digital neurohardware: principles and perspectives. Neural Networks in Applications, Magdeburg 101–106
Schmitz T, Hohmann S, Meier K, Schemmel J, Schurmann F (2003) Speeding up hardware evolution: a coprocessor for evolutionary algorithms. Evolv Syst Biol Hardw Lect Notes Comput Sci 2606:274–285
Schneider CR, Card HC (1991) CMOS mean field learning. Electron Lett 27(19):1702–1704
Schrauwen B, Van Campenhout J (2006) Parallel hardware implementation of a broad class of spiking neurons using serial arithmetic. In: Proceedings of ESANN, pp 623–628
Sejnowski TJ, Rosenberg CR (1987) Parallel networks that learn to pronounce English text. Complex Syst 1:145–168
Shayani H, Bentley PJ, Tyrrell AM (2008) An FPGA-based model suitable for evolution and development of spiking neural networks. European symposium on artificial neural networks advances in computational intelligence and Learning, pp 197–202. ISBN 2-930307-08-0
Shima T et al (1992) Neuro chips with on-chip back-propagation and/or hebbian learning. IEEE J Solid State Circuts 27:1868–1876

Silviott MA, Mahowald MA, Mead CA (1987) Real-time visual computations using analog CMOS processing arrays. In: Proceedings Of the stanford advanced research in VLSI conference. MIT Press, Cambridge
Skrbek M (1999) Fast neural network implementation. Neural Netw World 9(5):375–391
Speckman H, Thole P, Rosenstiel W (1993) COKOS: a co-processor for Kohonen's self organizing map. In: Proceedings of the ICANN-93-Amsterdam. Springer, London, pp 1040–1045
Sudarshan S, Santosh K, Pinjare SL (2010) MDAC Synapse-Neuron for Analog Neural Networks, 2, 523–527
Tang T, Ishizuka O, Matsumoto H (1993) Backpropagation learning in analog T-model neural network hardware. In: Proceedings of the International Joint Conference on Neural Networks (IJCNN '93), Nagoya, Japan, vol 1, pp 899–902
Tavenik M, Linde A (1995) A reconfigurable SIMD computer for artificial neural networks. Licentiate thesis No 189L. Department of Computer Engineering Chalmers, University of Technology, Goteborg, Sweden
Titri S, Boumeridja H, Lazib D, Izeboudjen N (1999) A reuse oriented design methodology for artificial neural networks implementation. Twelfth Annual IEEE International ASIC/SOC conference, pp 409–413
Treleaven P, Pacheco M, Vellasco M (1989) VLSI Architectures for Neural Networks. IEEE MICRO, pp 8–27
Trealeven PC (1989) Neurocomputers. Int J Neural Comput 1:4–31
Tsividis Y, Anastassiou D (1987) Switched-capacitor neural networks. Electron Lett 23(18):958–959
Uker A, Alkar AZ (2006) HW/SW Code sign of FPGA-based Neural Networks. Fifteenth Turkish Symposium on Artificial Intelligence and Neural Networks
Upegui A, Pena-Reyes CA, Sanchez E (2005) An FPGA platform for on-line topology exploration of spiking neural networks. Microprocess Microsyst Elsevier 29:211–223
Viredaz MA, Ienne P (1993) MANTRA- I: a systolic neurocomputer. In: Proceedings of the International Joint Conference on Neural Networks III, pp 3054–3057
Watanabe T et al (1989) Neural network simulation on a massively parallel cellular array processor: AAP-2. International Joint Conference on Neural Networks, Washington DC, vol 2, pp 155–161
Wawrzynek J, Asanovi K, Morgan N (1993) The design of a neuro microprocessor. IEEE Trans Neural Netw 4:394–399
Wolf DF et al (2001) Using embedded processors in hardware models of artificial neural networks. In: Proceeding of the Brazilian Symposium of Intelligent Automation
Xie Y, Jabri MA (1991) Training algorithms for limited precision feed forward neural networks. Technical report SEPAL
Yasunaga M et al (1989) A wafer scale integration neural network utilizing completely digital circuits. Int Joint Conf Neural Netw 2:213–217
Zell A, Korb T, Sommer T, Bayer R (1990) Recent developments of the snns neural network simulator. In: Proceedings of the applications of neural networks conference, SPIE 1294, pp 534–544
Zhu J, Milne GJ, Gunther BK (1999) Towards an FPGA based reconfigurable computing environment for neural network implementations. In: Ninth International Conference on artificial neural networks, vol 2, pp 661–666