Engineering Applications of Artificial Intelligence ∎ (∎∎∎∎) ∎∎∎–∎∎∎

Contents lists available at ScienceDirect

Engineering Applications of Artificial Intelligence

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

Survey Paper

Fuzzy systems, neural networks and neuro-fuzzy systems: A vision on their hardware implementation and platforms over two decades ☆

G. Bosque [a,*], I. del Campo [b,**], J. Echanobe [b,**]

[a] Department of Electronic Technology, University of the Basque Country, Bilbao, Vizcaya 48013, Spain
[b] Department of Electricity and Electronics, University of the Basque Country, Leioa, Vizcaya 48940, Spain

ARTICLE INFO

Article history:
Received 28 February 2013
Received in revised form
7 February 2014
Accepted 13 February 2014

Keywords:
Fuzzy inference systems
Neural networks
Neuro–fuzzy systems
μ P–DSP
μ C–PLC
VLSI–ASIC–FPGA–FPAA

ABSTRACT

In recent decades, and in order to develop applications covering several areas of knowledge, different researchers have been performing hardware implementations around paradigms such as fuzzy systems, neural networks or systems resulting from the hybridization of the previous two systems, known as neuro–fuzzy systems. Applications have been performed on different types of devices and/or platforms.

The point of view of this paper is focused on a hardware taxonomy (devices where the applications have been implemented) and highlights the characteristics of the different applications covering the aforementioned paradigms done over the last two decades, and the beginning of the current decade. Special mention is made up of reconfigurable devices.

1. Introduction

Since the decade of the 1940s taking into account that in 1943, McCulloch and Pitts (1943) introduced the model of a neuron, to the present day, different paradigms have emerged: these are fuzzy systems, neural networks, genetic algorithms and hybrid systems, these last composed by a combination of the above. They are all encompassed in a more generic concept called Soft Computing.[1] These paradigms have been consolidated in various facets of human knowledge and in several applications in the fields of education, industry, consumption and research.[2] Fuzzy systems or fuzzy inference systems (FISs), based on fuzzy sets and fuzzy logic, work with imprecise reasoning and linguistic rules obtained from the information provided by an expert, leading to systems tolerant of imprecision. Artificial neural networks (ANNs) are computational structures that model the physiological behavior of neurons and connections. The neurons store the knowledge by means of training (learning), obtaining adaptive systems for their environment and tolerant of faults in some of their neurons. Genetic algorithms, based on the theory of evolution, apply genetic inheritance algorithms such as reproduction, crossover, and mutation, to a specific problem.

The scope of the different topics is extensive. Fuzzy systems have proved very useful in support of decision making in areas such as economics (Aluja, 2004; Dompere, 2004; Aliev, 2008) and medicine (Szmidt and Kacprzyk, 2001; Bates and Young, 2003;

☆This work was supported in part by the Spanish Ministry of Science and Innovation and European FEDER funds under Grant TEC2010-15388 and by the Basque Country Government under Grants IT419-10, S-PC11UN012, and S-PC12UN016.

* Corresponding author. Tel.: +34 946014461.
** Corresponding authors.
E-mail address: guillermo.bosque@ehu.es (G. Bosque).
URL: http://www.ehu.es/guillermo.bosque (G. Bosque).

[1] Term coined by Lotfi Zadeh in a course at the University of Berkeley in 1993, to define the computation that emulates the human mind's ability to reason and learn in an environment of uncertainty and imprecision. Subsequently, the meaning of this term has become more widely used.

[2] As interesting historical data, S. Grossberg (USA), T. Kohonen (Finland), and S.-I. Amari (Japan) create the "International Neural Network Society" (INNS) in the spring of 1987. The first conference on neural networks, "IEEE First International Conference on Neural Networks" was held in June 1987 and the number 1, volume 1 of the journal "Neural Networks" appears in 1988. The number 1, volume 1 of the magazine "Fuzzy Sets and Systems" appears in 1978 and the number 1, volume 1 of the journal "IEEE Transactions on Fuzzy Systems" appears in 1993.

http://dx.doi.org/10.1016/j.engappai.2014.02.008
0952-1976

Table 1
Summary which shows the application examples of different paradigms.

Paradigms	Articles		Applications
	Total	Partial	
Fuzzy systems	24	9	Economy (Aluja, 2004; Dompere, 2004; Aliev, 2008) Medicine (Szmidt and Kacprzyk, 2001; Bates and Young, 2003; Chen et al., 2010) Control (Wang and Langari, 1994; Kiriakidis, 1998; Abou, 2011)
Neural networks		8	Robotics (Bekey and Goldberg, 1993; Rao, 1995; Zou et al., 2006) Image (Carpenter and Grossberg, 1992; Egmont-Petersen et al., 2002; Hong et al., 2009 Speech recognition (Othman and Riadh, 2008; Lippman, 1988)
Neuro–fuzzy		7	Pattern recognition (Ray and Ghoshal, 1997; Pal and Mitra, 1999) Robotics (Rusu et al., 2003; Wongsuwarn and Laowattana, 2006) Nonlinear system identification (Babuska and Verbruggen, 2003; Panchariya et al., 2004) Adaptive signal processing (Li and Tsai, 2006; Chabaa et al., 2009)

Chen et al., 2010). Moreover, the application has been remarkable as an alternative control strategy to the classic control techniques for complex systems, which are difficult to model or whose dynamic is poorly understood (Wang and Langari, 1994; Kiriakidis, 1998; Abou, 2011). Neural networks, given their learning ability and adaptability, are applied in areas such as robotics (Bekey and Goldberg, 1993; Rao, 1995; Zou et al., 2006), image processing (Carpenter and Grossberg, 1992; Egmont-Petersen et al., 2002; Hong et al., 2009), and speech recognition (Othman and Riadh, 2008; Lippman, 1988). Within the hybrid systems the neuro–fuzzy systems combine both paradigms; on one hand the system of linguistic rules generated by an expert, on the other hand the learning ability of neural networks applied to this system. The applications include pattern recognition (Ray and Ghoshal, 1997; Pal and Mitra, 1999), robotics (Rusu et al., 2003; Wongsuwarn and Laowattana, 2006), nonlinear system identification (Babuska and Verbruggen, 2003; Panchariya et al., 2004), adaptive signal processing (Li and Tsai, 2006; Chabaa et al., 2009), etc.

Table 1 shows the summary of the aforementioned application examples. The aforementioned references are a sample of a large set of references.

It can briefly be mentioned that one of the main reasons that influenced the success of fuzzy systems, neural networks and neuro–fuzzy systems is their ability to approximate continuous nonlinear functions. In this area within the fuzzy systems, the following works can be cited: Wang (1992), Kosko (1994), Zeng and Singh (1996), Rovatti (1998), Kreinovich et al. (2000), Cao et al. (2001), and Landajo et al. (2001). With regard to neural networks, the following contributions should be highlighted: Stinchcombe and White (1989), Cotter (1990), Hornik (1991), Attali and Pagès (1997), and Castro et al. (2000). On neuro–fuzzy networks, the following references are highlighted: Buckley (1993), Castro (1995), Jang et al. (1997), Nauck and Kruse (1999), Wang and Wei (2000), and Wu et al. (2010). Table 2 shows these contributions.

The different applications of soft computing algorithms have been materialized on different supports over time, depending on the technologies of the moment. The following have been used: general-purpose processors, dedicated processors, dedicated coprocessors, specific designs with Very Large Scale Integration (VLSI) integration scales, either analog or digital or mixed,[3] up to the current reconfigurable hardware (HW) devices. Use has also been made up of multi-processor platforms for systems that require high speed computation and supporting parallel processing, as in the case of neural networks. The use of one or other support has been conditioned by different requirements, among others, power consumption, processing speed, size, portability and cost.

It is in this evolution where reconfigurable device consolidation of type Field Programmable Gate Array (FPGA)[4] (Altera; Xilinx; Lattice; Actel, www.actel.com; Atmel; Cypress; Quicklogic; Achronix) and its high integration, has allowed the implementation of solutions in the environment of soft computing versus focused solutions on specific HW of type Application Specific Integrated Circuit (ASIC)[5] (Atmel; Avago; Elmos; Lsi). Also, it is possible to implement embedded processors (hard-core) (Xilinx, 2007; ARM, 2001; Actel-ARM) and instantiated (soft-core) (Xilinx, 2008; Altera, 2004; Actel, www.actel.com/products/mpu/coremp7/) with other HW functional blocks on these devices. This leads to the so-called System-on-a-Programmable Chip (SoPC) being able to integrate complete systems on one device. All these make the search for efficient implementations in the area of fuzzy inference systems in neural networks and neuro–fuzzy networks on these devices particularly attractive. This paper aims to provide a historical overview of this search for the aforementioned systems.

The paper is structured as follows: Section 2 reviews the first HW implementations of fuzzy systems and neural networks. Section 3 shows a generic classification of HW implementations of the fuzzy, neural and neuro–fuzzy systems. Sections 4–6 review the implementations of fuzzy, neural and neuro–fuzzy systems respectively on different types of devices. Finally, Section 7 summarizes the conclusions of the work. The bibliography shows the books and articles consulted throughout this work.

2. First implementations of fuzzy, neural and neuro–fuzzy systems

The last two decades have been marked by a great evolution in the field of computer hardware for fuzzy, neural and neuro–fuzzy systems. The solutions have been provided on analog HW, digital HW and mixed digital/analog HW. The digital hardware is what has presented the most important development due to the consolidation of programmable or reconfigurable devices, mainly in the FPGAs. The high integration density and the power introduced by the parallel structures achieved by this technology have enabled implementations of fuzzy inference systems with a high number of fuzzy rules, neural networks with a large number of layers and neurons, including learning algorithms, and finally, neuro–fuzzy systems based on fuzzy rules and endowed with learning mechanisms of the same type as those used in neural networks.

[3] This paper employs the term "mixed" instead of "hybrid" to avoid confusion with the term applied to "hybrid system".

[4] Invented by Xilinx in 1984.

[5] The company Ferranti, around 1980, produced the first gate array known as ULA (Uncommitted Logic Array – Matrix Logic not Fixed) and can be regarded as the pioneer in this technology.

Please cite this article as: Bosque, G., et al., Fuzzy systems, neural networks and neuro-fuzzy systems: A vision on their hardware implementation and platforms over two decades. Eng. Appl. Artif. Intel. (2014), http://dx.doi.org/10.1016/j.engappai.2014.02.008

Table 2
Summary of works on approximation of nonlinear functions.

	Articles		Application: Approximation of nonlinear functions
	Total	Partial	
Fuzzy systems	18	7	Wang (1992), Kosko (1994), Zeng and Singh (1996), Rovatti (1998), Kreinovich et al. (2000), Cao et al. (2001), Landajo et al. (2001)
Neural networks		5	Stinchcombe and White (1989), Cotter (1990), Hornik (1991), Attali and Pagès (1997), Castro et al. (2000)
Neuro–fuzzy		6	Buckley (1993), Castro (1995), Jang et al. (1997), Nauck and Kruse (1999), Wang and Wei (2000), Wu et al. (2010)

The first digital fuzzy processing device was implemented by Togai and Watanabe (1986a) in 1985[6] at Bell Labs AT&T.[7] It was a VLSI device with two inputs and one output capable of running 80,000 FLIPS (Fuzzy Logical Inference Per Second). Prior to the appearance of the previous device, Yamakawa (1988) had built the first analog device for fuzzy control, based on bipolar transistors but this was not presented until 1988. The driver was able to evaluate 1 MFIPS (Mega Fuzzy Inference per Second) including defuzzification or 10 MFIPS without it. These two projects represent the beginning of a race for higher processing speeds, smaller devices and lower consumption in the field of digital and analog hardware. A broader view can be found in Basterretxea and del Campo (2009). In Section 4 the fuzzy implementations are presented.

The introduction of the neuron model of McCulloch and Pitts (1943)[8] and subsequent application of the learning method proposed by Hebb (1949) are two milestones, based on which, Minsky and Dean in 1951 developed the first learning machine named SNARC (Stochastic Neuro-Analog Reinforcement Computer) (Minsky).[9] It is an analog computer on which is implemented a neural network with learning capacity, composed of 40 processing elements (neurons) and wherein each processing element requires 6 vacuum tubes (Tsoukalas and Uhrig, 1997).

One of the first digital architectures that perform a neural computation occurs in 1987 and is known as NetSim. It is presented in the work "A chipset for high speed simulation of neural network systems" (Garth, 1987). Within this architecture there are two devices, "solution engine" and "communication handler". The first performs the neuronal computing and the second performs the routing of neural activations. It also requires an external memory for storing weights, achieving a performance of 4 MCPS (Millions of Connections Per Second). The two devices mentioned, the memory and a local processor, constitute the complete architecture of NETSIM card. A neuro-computer system consists of connecting multiple NETSIMs to a general purpose computer (Burr, 1995).

The two architectures mentioned open two paths of artificial neural network implementations, the first in an analog context and the second in a digital context. In Section 5 a classification of ANNs implementations is considered.

As a final comment about ANNs, it is interesting to note the efforts made by the company Intel Corporation to introduce, in 1989, the analog neuronal processor ETANN (Electrically Trainable Artificial Neural Network) (Holler et al., 1989), presenting the processor in the work named "An Electrically Trainable Artificial Neural Network (ETANN) with 10 240 'Floating Gate' Synapses" at the conference IJCNN'89. In this article some key aspects are highlighted of the architecture such as emphasis on speed via parallelism, both feed forward and feedback connections on chip, analog voltage input and output, a large pin count package, and the option to operate in a fast static mode or in a useful clocked mode with multiplexed external buses. The weights are stored on a EEPROM (Electrically Erasable Programmable Read Only Memory). The learning process, the company point out, is off-chip for flexibility and cost reduction.

The fusion of fuzzy systems with neural networks, as discussed above, leads to the neuro–fuzzy hybrid systems. Section 6 presents a taxonomy of implementations of neuro–fuzzy systems.

Among the proposed taxonomies for fuzzy, neural and neuro–fuzzy implementations, a greater emphasis is given to fuzzy implementations because it is an area that has undergone great experimentation in technologies by researchers, leading to a variety of solutions.

3. Generic classification of HW implementations

Taking into account the difficulty of performing a taxonomy of the different HW implementations, implementations are considered to be of the type: (a) *analog*, (b) *digital*, and (c) *mixed* and within each of them the hardware where the application resides, may have different characteristics:

- *Dedicated Integrated Circuits*: Devices designed specifically to implement a system. These are usually full-custom devices (analog, digital or mixed), and often ASICs.[10]
- *Programmable Integrated Circuits*: In this proposed structure, it is considered that a programmable integrated circuit is a commercial hardware that can be reconfigured by a user by means of any specific technique. These are commercial devices such as FPGAs, where the applications, as in the dedicated ones, are completely HW. One aspect that makes these devices particularly attractive is their ability to reprogramme and the existence of EDA tools that facilitate their design. A recent type of programmable device called FPAA (Field Programmable Analog Array)[11] can be also mentioned, which emerged as a commercial option in the 2000s, but in a very low integration density and with few applications made in the area at hand.
- *Commercial processors*: These processors deploy an application SW that characterizes the system.[12] The devices are μ Ps, μ Cs, DSPs, etc. and around these are performed different architectures for the development of the aforementioned systems. In the industrial control environment, PLCs(Programmable Logic

[6] Was published in April of 1986.

[7] After this work, in September, the article (Togai and Watanabe, 1986b) was published. It is taken from an article of AT&T Bell Laboratories.

[8] The program showed that a Turing machine could be implemented in a finite network of conventional neurons and the neuron was the basic logic unit of the brain.

[9] Although in the construction Dean takes part, the publication is carried out only by Minsky.

[10] Understanding that any VLSI circuit is itself an ASIC that is used when the term refers to a VLSI integration technology but keep the nomenclature that has historically been used in a large number of publications. This is due to personal mode of approaching VLSI designs but attempts to develop CAE and CAD tools for the design of VLSIs and is opposed to the systematic structure in the design of ASIC circuit devices designed in VLSI technology. Analog and digital implementations performed on full-custom circuits have sought to increase efficiency in terms of speed and the use of silicon.

[11] The main companies that manufacture such devices are Lattice and Anadigm.

[12] Let us bear in mind that this article wants to highlight the devices where the applications have been implemented.

Please cite this article as: Bosque, G., et al., Fuzzy systems, neural networks and neuro-fuzzy systems: A vision on their hardware implementation and platforms over two decades. Eng. Appl. Artif. Intel. (2014), http://dx.doi.org/10.1016/j.engappai.2014.02.008

Controllers) are also incorporating utilities in some applications. The programming flexibility offered by these devices and/or architectures as well as their cost has favored the implementation of systems on them.

Table 3 shows the classification.

Table 3
Classification of HW implementations for soft computing systems.

Types of implementations	analog	dedicated integrated circuits :	VLSI/ASICs
		programmable integrated circuits :	FPAAs
		commercial processor :	– – –
	digital	dedicated integrated circuits :	VLSI/ASICs
		programmable integrated circuits :	FPGAs
		commercial processor :	μPs–μCs–DSPs
	mixed	dedicated integrated circuits :	VLSI/ASICs
		programmable integrated circuits :	– – –
		commercial processor :	– – –

4. HW implementations for fuzzy systems

Referring to Section 3, the taxonomy of the different fuzzy implementations will be of the type: (a) *analog*, (b) *digital*, and (c) *mixed* and within each of them consider that the hardware where the application resides may have the characteristics such as Table 3 shows. Let us consider the following concerning the HW:

- *Dedicated Integrated Circuits*: Special attention has focused on increasing the processing speed of the fuzzy HW.
- *Programmable Integrated Circuits*: Referring to the FPAAs, although the commercial appearance mentioned in the 2000s, there are also some works in the 1990s.
- *Commercial processors*: Several manufacturers have introduced fuzzy instructions in their products. This allows flexibility in the programming of these devices and increases the speed, favoring the implementation of fuzzy systems on them.

4.1. Analog fuzzy implementations

The high speed, low silicon area occupation, low consumption and high parallelism are decisive reasons for implementing FISs on an analog hardware. In such implementations, solutions are focused on how the circuits work, distinguishing between circuits working in current mode, voltage mode or hybrid mode (transconductance). A fourth category is presented in such implementations; it works with circuit-switched, both in tension and current.

One of the benefits of such implementations is that they have a natural connection with different sensors and/or actuators as almost all of them have voltage mode (e.g., range ± 10 V) or current mode (eg., 4 .. 20mA), that is, converters A/D (Analog/Digital) or D/A (Digital/Analog) are not required.

4.1.1. Dedicated Integrated Circuits

Current-mode circuits: This architecture is shown to be most suitable for basic fuzzy operations because it requires few transistors. Addition and subtraction are performed by simple wire connections and *max–min* operators are performed with simple circuits (Baturone et al., 1994; Lemaitre et al., 1994). Another advantage is that they can work with very low voltage. The main disadvantage is that they work in current mirror mode to replicate the outputs, being his fan-out of 1, i.e., they can only connect to a single output circuit. Table 4 shown these references.[13]

As an example, the following can be cited (*):

- Lemaitre et al. (1994) in the article "Analysis and design of CMOS fuzzy logic controller in current mode" present a CMOS architecture consisting of current mirrors of 2 input variables and one output with 9 rules. This architecture reaches 10 MFIPS on a standard CMOS technology of 1.2 μm, consuming 2 mA at 5 V. In addition, this work presents a compiler on silicon SCOFIC destined for the automatic synthesis of the above-mentioned circuit.

Voltage-mode circuits: These have the advantage of connecting more than one input or output without resorting to adaptive signal circuitry. The first device is the aforementioned of Yamakawa (1988); subsequently Yamakawa (1993) also presented another device in bipolar technology. On CMOS technology, fuzzy controllers with higher speeds and lower power consumption (Peters et al., 1995), achieving speeds of 2 MFIPS, were subsequently proposed. Other researchers have addressed the design of MOS technology operating in subthreshold mode of floating gate transistor, yielding a very low consumption of the blocks.[14] They also have the ability to store information in the MOSFET gates (Marshall and Collins, 1997). Other references are Miki and Yamakawa (1995) and Guo et al. (1996). Table 5 shows these references.

The following contributions can be commented from the references (*):

- Yamakawa (1993) in the article "A fuzzy inference engine in nonlinear analog mode and its applications to fuzzy control" presented a bipolar device technology with the feature to work every transistor coupled by an emitter, in a configuration called by Yamakawa ECFL (Emitter Coupled Fuzzy Logic), variant of the ECL (Emitter Coupled Logic) and working in the active region. This gave high stability to the design in aspects such as temperature (2% of full scale in the range of -55 °C to $+125$ °C) and voltage (less than 0.1% at 5 V supply). The low output impedance achieved with this configuration allowed a high fan-out. The velocity attained by the inference engine is 1 μs, i.e., of 1 MFIPS and defuzzification is performed in 5 μs. To check the suitability of the design it is applied to a particular case of double inverted pendulum.
- Peters et al. (1995) in the article "A novel analog fuzzy controller for intelligent sensors" proposed an architecture where the generation of the membership functions, the inference system and the defuzzification required a lower surface than the controllers of the moment. The example was implemented with 13 rules but admits several hundreds. The tests performed show a speed in the process of inference

[13] Where some details are not explicit in the works referenced, these are expressed as '–'.

[14] Several orders of magnitude with respect to the designs operating in strong inversion.

Please cite this article as: Bosque, G., et al., Fuzzy systems, neural networks and neuro-fuzzy systems: A vision on their hardware implementation and platforms over two decades. Eng. Appl. Artif. Intel. (2014), http://dx.doi.org/10.1016/j.engappai.2014.02.008

Table 4
HW implementations of fuzzy systems – Analogics Dedicated Integrated Circuits – current-mode.

Year	Applications	Ref.	Type of FS	In	No. of in MFs	Type of out MFs	Out	No. of out MFs	Type of out MFs	No. of rules	Defuzz.	Speed	Technology
1994	Max. operator	Baturone et al. (1994)	IF–THEN**	2/3	–	–	1	–	–	–	–	100 ns	CMOS 1.6 μm
1994	Electromagnetic fields	Lemaitre et al. (1994)*	IF–THEN	2	–	–	1	–	–	9	–	10 MFLIPS	CMOS 1.2 μm

* Articles that will be comment.
** When the articles do not mention a specific system.

Table 5
HW implementations of fuzzy systems – Analogics Dedicated Integrated Circuits – voltage-mode.

Year	Applications	Ref.	Type of FS	In	No. of in MFs	Type of in MFs	Out	No. of out MFs	Type of out MFs	No. of rules	Defuzz.	Speed	Technology
1993	Generic	Yamakawa (1993)*	IF–THEN	3	–	Triang., trapez.	1	–	Triang.	n	COG[a]	1 MFLIPS	ECL–ECFL
1995	Automotion sensors	Peters et al. (1995)*	Mamdani	2	7–7	Bell and others	1	–	–	13 (100↑)	COA[b] (div. free)	2 MFLIPS	CMOS 2.4 μm
1995	Generic	Miki and Yamakawa (1995)	IF–THEN	3↑	n–n–n	Triang., trapez.	1↑	7	–	$4 \times m$	COG	0.6 MFLIPS	BiCMOS 2.0 μm
1996	Mobile robot	Guo et al. (1996)	Mamdani	3	3–3–3	Triang., trapez.	1	5	Singleton	13	COG (div. free)	6 MFLIPS	CMOS 2.4 μm
1997	Sensors	Marshall and Collins (1997)*	IF–THEN	3↑	3–5–5	Triang.	1↑	7	Triang.	75	COG	10 KFLIPS ↑	CMOS 2.0 μm

[a] Center of gravity.
[b] Center of area.

of 0.5 μs, i.e., of 2 MFIPS. They presented an application to the comfort in a vehicle, taking variables such as the irregularity and unevenness of the land. The chip is designed in CMOS process of 2.4 μm.

- Marshall and Collins (1997) in the article "Fuzzy logic architecture using floating gate subthreshold analogue devices" obtained a system with 75 rules and the parameters stored in programmable floating gate transistors. The circuit consumes 500 μW occupying an area less than 5 mm^2. It is considered suitable for integration into intelligent sensors. A potential disadvantage compared to other architectures that do not work in subthreshold mode is the lowest speed achieved (the order of tens of kHz compared with the MHz of these latter). The chip is designed in CMOS process of 2 μm.

Circuits operating in transconductance: Inputs in voltage and outputs in current. Many circuits in voltage mode operate in transconductance for the treatment of the membership functions. They are based on differential pairs of transistors operating in weak inversion (Landlot, 1996), or strong inversion (Baturone et al., 1994), to obtain membership functions of nonlinear softer profile. Other designs use amplifiers OTA (Operational Transconductance Amplifier) and capacitors as basic blocks (Indue et al., 1991; Tsukano and Inoue, 1995). The latter are more structured but occupy more silicon area than those based on differential transistor pairs. Table 6 shows these references.

In the cited references, the following can be pointed out (*):

- Tsukano and Inoue (1995) in the work "Synthesis of operational transconductance amplifier-based analog fuzzy functional blocks and its applications" presented the synthesis of different blocks of a FIS operating with OTAs. The circuit design of the membership functions is performed in a complementary manner – the *max* circuit operation is performed in a complementary manner $\overline{max}$, and the defuzzification is performed by the method COA with OTAs in multiplication and aggregation configuration. All the functional blocks are simulated in SPICE and the performed circuit has 2 inputs/1 output (singleton) and 3×3 rules on a standard CMOS process. Acquired inference speed is around 15 MFRPS (Mega Fuzzy Rules per Second), but the comparison with a typical fuzzy controller has been performed at 1.6 MFLIPS, and the power dissipation of this controller is 28.2 mW.
- Landlot (1996) in the article "Low-power analog fuzzy rule based on a linear implementation MOS transistor network" presented an architecture based on CMOS transistors operating in weak inversion. This allows them to define the pseudo-voltage and pseudo-conductance of a CMOS transistor such that the pseudo-conductance of each transistor is controlled linearly (following Ohm's law) by the pseudo-voltage activation of transistor. Following this behavior, the value of a membership function is the pseudo-conductance proportional to the voltage applied to the transistor. Moreover by powering the circuit to 1.8 V, consumption achieved is 850 nW and the response time is less than 400 μs. The performed circuit has 2 inputs, 80 rules and 5 outputs. The device is designed in 2 μm CMOS technology.

Circuits switched or circuits discretized: The purpose of these circuits is to increase the accuracy and programmability with respect to conventional analog designs but maintain a high processing speed with lower occupation areas. The switching circuit is controlled by a clock. The major disadvantage of these circuits is that the basic operations are not performed with transistors but with operational amplifiers or comparators that occupy more silicon area. Some publications of this type of circuits appeared in the 1990s. Table 7 shows these publications. Two basic design techniques are presented, (a) switched capacitor (SC: switched capacitors) working in voltage mode and (b) switched current (SI: switched intensity) working in current mode.

Please cite this article as: Bosque, G., et al., Fuzzy systems, neural networks and neuro-fuzzy systems: A vision on their hardware implementation and platforms over two decades. Eng. Appl. Artif. Intel. (2014), http://dx.doi.org/10.1016/j.engappai.2014.02.008

Table 6
HW implementations of fuzzy systems – Analogics Dedicated Integrated Circuits – transconductance.

Year	Applications	Ref.	Type of FS	In	No. of in MFs	Type of in MFs	Out	No. of out MFs	Type of out MFs	No. of rules	Defuzz.	Speed	Technology
1991	Membership generation	Indue et al. (1991)	MAX/MIN operation	3	–	Triang., trapez.	1	–	–	–	–	82 ns	CMOS (SPICE simul.)
1994	Max. input	Baturone et al. (1994)	IF–THEN	2/3	–	–	1	–	–	–	–	100 ns	CMOS 1.6 μm
1995	Generic	Tsukano and Inoue (1995)*	IF–THEN	2	–	$\overline{Triang.}$, $\overline{trapz.}$	1	9	Singleton	9	COA	15 MFRPS	CMOS –
1996	Sensors	Landlot (1996)*	Fuzzy rules	2	–	Bell, S and Z-shaped	5	–	Singleton	80	COG	–	CMOS 2 μm

Table 7
HW implementations of fuzzy systems – Analogics Dedicated Integrated Circuits – switched.

Year	Applications	Ref.	Type of FS	In	No. of in MFs	Type of in MFs	Out	No. of out MFs	Type of out MFs	No. of rules	Defuzz.	Speed	Technology
1993	Generic	Huertas et al. (1993)*	IF–THEN	*n*	–	S and Z-shape	–	–	–	–	COG	–	CMOS –
1994	Generic	Fattaruso et al. (1994)	IF–THEN	8	56	Triang., trapez.	4	28	–	32	COG	16 MFLIPS	CMOS 0.8 μm
1997	Controllers	Çilingiroglu et al. (1997)*	IF–THEN	4	64	S and Z-shape, triang., trapez.	2	7	Singleton	16	Weighted average	85 KFLIPS	CMOS 1.2 μm

Of the references, will comment on the following (*):

- Huertas et al. (1993) presented the work "A fuzzy controller using switched-capacitor techniques", where a sequential microcontroller is implemented based on a switched capacitor. The process is driven by a clock that controls the so-called operating cycle (consisting of N clock cycles). This cycle involves three processes performed in three blocks: a block generator of the membership function, a block generator of the rules (includes a ROM analog) and a defuzzification block. The internal data processing is serial which results, along with the introduction of switched capacitors, in a smaller area of silicon.
- Çilingiroglu et al. (1997) in the article "Sampled-analog implementation of application-specific fuzzy controllers" presented a fuzzy controller with 4 inputs, 16 rules and 2 outputs. The input acquisition is performed by means of amplifiers S/H. The membership functions are ramp (positive and negative), triangular and trapezoidal. The inference module is done with as many identical cells as there are rules. The cells are connected to a node, called "competition node", such that all cells have mutually inhibitory connections between them so that only a single node is activated. The defuzzification is performed by the method of weighted average of the singletons of the rules. The process speed is limited to 85 ks/s due to the absence of buffers at the outputs of the design presented.

4.1.2. Programmable Integrated Circuits

There are few references in analog implementations in the area of FPAA devices. Perhaps in the foreseeable future most works will be carried out on FPAA devices. The references are shown in Table 8.

In the cited references, the following three implementations will be considered.

- Pierzchala et al. (1994) presented one of the earliest works in this area, "A field programmable analog array for continuous, fuzzy, and multi-valued logic applications". The interest of the article lies in showing that this technology can be used for the implementation of a wide range of multi-valued logic, fuzzy logic, and continuous logic circuits. In the case of a fuzzy controller, the controller is implemented with m input variables and n fuzzy inference rules. Fuzzy membership function is implemented as a trapezoidal transfer function. The activation values of the rules are multiplied by centroid values of the fuzzy rules consequent. The defuzzified output variable is produced by a two-quadrant divider. The device has been developed under a bipolar transistor array technology and operates from ± 3.3 V or ± 5 V with frequencies up to several hundred MHz.
- do Amaral et al. (2002) in the work "Towards evolvable analog fuzzy logic controllers" proposed the implementation of Fuzzy Logic Controllers, based on building blocks, developed on an FPAA. They also presented an intrinsic evolution example of a building block addressed to a membership function, by means of Genetic Algorithms which automatically reconfigure the FPAA. The development relies on a platform named PAMA (Programmable Analog Multiplexer Array) consisting of a FPAA and a multifunction I/O board connected to a PC through a PCI Bus. The PC sends to the multifunction I/O board the string of chromosomes and each gene configures the select signals of a particular analog multiplexer. With this code, the FPAA generates the membership function.
- Ionita and Sofron (2004) in the work "Field-programmable analog filters array with applications for fuzzy inference systems" proposed a flexible membership function synthesis method based on the configurable filter blocks using the frequency-response curves of the analog filters (the filter is one of the basic configurable analog blocks (CABs) in the FPAA). A membership function is defined in terms of frequency on the x-axis and the response (gain) on the y-axis. The gains of the filter are in the fuzzy domain $[0, 1]$. The remaining blocks are implemented with programmable analog structures of the FPAA. The FIS is a Mamdani system which implements *max–min* operators with transistors. As in the previous work, this article aims to implement evolutionary algorithms for tuning the membership functions.

Please cite this article as: Bosque, G., et al., Fuzzy systems, neural networks and neuro-fuzzy systems: A vision on their hardware implementation and platforms over two decades. Eng. Appl. Artif. Intel. (2014), http://dx.doi.org/10.1016/j.engappai.2014.02.008

Table 8
HW implementations of fuzzy systems – Analogics Programmable Integrated Circuits.

Year	Applications	Ref.	Type of FS	In	No. of in MFs	Type of in MFs	Out	No. of out MFs	Type of out MFs	No. of rules	Defuzz.	Speed	Device
1994	Generic	Pierzchala et al. (1994)*	IF–THEN	n	–	Trapez.	1	–	–	N_R	COG	–	FPAA[a]
1998	Generic survey	D-Mello and Gulak (1998)	–	–	–	–	–	–	–	–	–	–	Analogix, IMP, etc.
2002	Generation of MFs	do Amaral et al. (2002)*	IF–THEN	n	–	Z-shape, triang., others	1	–	–	–	–	–	FPAA
2004	Filter config.	Ionita and Sofron (2004)*	Mamdani	n	–	Γ-function	1	–	–	N_R	–	–	FPAA

[a] When the work does not cite the reference of the FPAA.

4.2. Digital fuzzy implementations

Digital implementations have greater immunity against factors such as noise, temperature or voltage variation, among others. In contrast, the process speed tends to be lower than in analog devices although evolution in integration technologies has changed this scenario.

4.2.1. Dedicated Integrated Circuits

In these implementations the focus is mainly on structuring the rules of the FIS, if they are running parallel or sequentially.

Parallel architectures rules: Each rule has a data path. When using memories any profile of the membership functions can be stored but the memory increases exponentially with the resolution, and therefore, tends to lower resolutions. The direct consequence of this decision is the reduction of the functional performance of the system (del Campo and Tarela, 1999; del Campo et al., 2001). The first digital fuzzy device was presented by Togai and Watanabe (1986b), in the article, "Expert system on a chip: an engine for real-time approximate reasoning". The rules are executed in parallel and each rule is executed in series. The set of rules is stored in a ROM and the membership functions are implemented on LUTs. This device requires a large silicon area. It also imposes a certain number of inference rules of type *max–min*, and their scalability is limited. Process speed is 80 KFLIPS.

Table 9 shows some relevant approaches.

Let us highlight the following works (*):

- Watanabe and Dettloff (1991) in the paper "VLSI fuzzy chip and inference accelerator board systems" presented a fuzzy chip working in a VMEbus environment. The maximum speed is 36 MHz. Included on the chip is a programmable rule set memory, an optional input fuzzification operation by look-up table, a *max–min* paradigm fuzzy inference processor, and an optional output defuzzification operation using a centroid algorithm. The design has a reconfigurable architecture implementing either 51 rules with 4 inputs and 2 outputs, or 102 rules with 2 inputs and 1 output. Separately addressed status registers allow programmed control of the fuzzy inference processing and chip configuration. All the rules operate in parallel, generating new outputs over 150,000 times per second. The implementation is done with four components: a fuzzifier, a rule memory, an inference mechanism, and a defuzzifier on a single chip. Each input and output data item is 6 bits. In the simplest configuration, sensors and 6 bit A/D converters drive fuzzy chip inputs, and controllers (adjusters) receive the output of 6 bit D/A converters directly connected to the fuzzy chip outputs. The chip is designed in a CMOS process of 1 μm.
- Jacomet and Walti (1996) in the work "A VLSI fuzzy processor with parallel rule execution" presented an architecture with the following advantages: high speed, a fuzzy system optimization with individual rule weights and a non-restricted membership function shape. There is a restriction; the membership functions are overlapped to 2. The processor presents 4 inputs and 1 output and two defuzzification methods, maximum and COA. The fuzzification process is carried out with a Look-up Table (LUT) and the membership functions are codified on 6 bits. Working with a 50 MHz of clock frequency, the processor executes 4 inputs and 64 fuzzy rules with a COA defuzzification method within 1.16 μs (58 clock cycles). An internal pipeline architecture allows an input sample rate of 740 ns (37clock cycles). This corresponds to a performance of 86 MFRPS (Mega Fuzzy Rules per Second) including the COA defuzzification calculation. The chip is designed in a CMOS process of 0.7 μm.
- Ascia and Catania (1997) in the work "A dedicated parallel processor for fuzzy computation" presented a fuzzy processor that processes only the active rules and optimizes the membership functions. These are approximated by segments and the height is proportional to a fixed value H (multiple of 2), so, to store the shape of the fuzzy set, it is sufficient to store the abscissa of end points and the height. The clock is 60 MHz and if the number of active rules is 16 then the performance obtained is 2.5 MFLIPS. The chip is designed in a HCMOS process of 0.1 μm.
- Falchieri et al. (2002) in the article "Very fast rate 2-input fuzzy processor for high energy physics" presented a fuzzy processor described using VHDL language. The processor has been developed, applying a parallel-pipeline architecture executing only the active rules. The FIS is a Sugeno order zero with 2 inputs (7 bits), 1 output (7 bits), 8 membership functions for each input (4 bits – overlapped with the two adjacent), 64 fuzzy rules (9 bits) and 128 crisp fuzzy sets (7 bits) for the output value. Due to the fact that the membership functions are overlapped, the number of active rules is 4. The membership functions reside on a LUT which allow, firstly, an increase in the fuzzification speed process and secondly, a high flexibility in defining membership function shapes. Taking into account that the processor works at a clock frequency of 133 MHz, the four active rules are processed in 30 ns. The defuzzification process corresponds to the Sugeno method. The total process requires 16 pipeline stages (16 clocks: 120 ns). The article performs two applications: (a) area recognition problem in boundary regions and (b) area recognition problem in physics experiments. In the last application, the task is to recognize and measure a given ellipsoidal area which is related to the charge of incident nuclei (propagation of cosmic rays). The chip has been realized using Alcatel Mietec 0.35 μm CMOS VLSI digital technology.
- Huang and Lai (2005) presented the article "A high-speed VLSI fuzzy inference processor for trapezoid-shaped membership functions". The main achievement of this processor is that the inference speed always remains constant, independently of the number of active rules. The membership functions are codified

Table 9
HW implementations of fuzzy systems – Digital Dedicated Integrated Circuits – parallel rules.

Year	Applications	Ref.	Type of FS	In	No. of in MFs	Type of in MFs	Out	No. of out MFs	Type of out MFs	No. of rules	Defuzz.	Speed	Technology
1989	Generic	Dettloff et al. (1989)	IF–THEN	4	–	Triang., trapez.	2	–	–	51	COA	580 KFLIPS	CMOS 1 μm
				2			1			102			
					4 bit		6 bit	–					
1991	Navigation	Watanabe and Dettloff (1991)*	IF–THEN	4	64	Any shape (LUT)[a]	2	–	–	51	COG	150 KFLIPS	CMOS 1 μm
				2			1			102			
				6 bit	4 bit		6 bit	4 bit					
1996	Generic	Patyra et al. (1996)	Any	4	–	Triang., trapez.	2	–	Triang.	–	Any	–	CMOS 0.35 μm
					4/6 bit		4/16 bit	–	Trapez.				
1996	Generic	Jacomet and Walti (1996)*	IF–THEN	4	7–7–7–7	Any shape (LUT)	1	8	–	64	COA, maximum	86 MFRPS	CMOS 0.7 μm
				8 bit	6 bit		8 bit	6 bit					
1997	Generic	Ascia and Catania (1997)*	IF–THEN	8	64	Segmented (α-level sets) 8 bit	1	–	–	256	Yager	2.5 MFLIPS	HCMOS 1 μm
2002	Area recognition, cosmic rays	Falchieri et al. (2002)*	Sugeno	2	8–8	Any shape (LUT)	1	128	-	64	Sugeno	33.3 MFLIPS	CMOS 0.35 μm
				7 bit	4 bit		7 bit	7 bit					
2005	Generic	Huang and Lai (2005)*	IF–THEN	2	No limit	Trapez.	1	64	Crisp	64	COG	7 MFLIPS	CMOS 0.35 μm
				8			4					4.5 MFLIPS	
				6/8 bit	4/6 bit		6/8 bit	4/6 bit					
2009	Active rule detection	Hamzeh et al. (2009)*	IF–THEN	*N*	*n*	Any shape	–	–	–	$(n)^N$	–	–	–
2013	Fuzzification	Javadi et al. (2013)*	IF–THEN	*N*	*n*	Any linear piecewise shape	*m*	–	–	–	–	–	CMOS
							–	–					0.13 μm
				7, …, 12 bit	7 bit		-	-					

[a] Look-up table.

on 6 bits, the rules are parallel and each rule pipelined. The maximum clock rate is 112 MHz and the fuzzy inference processor works with 64 rules with fuzzified inputs at the speed of 7 MFLIPS. The article performs a comparison between different devices: DSP TSM320C6201, FPGA Altera Flex10K, FPGA Xilinx XC2V1000 and a MIPS Processor with fuzzy instructions; although the scenario varies slightly, the comparison is favorable to the VLSI. The chip is designed in a CMOS process of 0.35 μm.

- Hamzeh et al. (2009) in the paper "Computationally efficient active rule detection method: algorithm and architecture" presented a theoretical study and the simulation results of ascalable architecture in terms of the number of inputs, number of membership functions and their bit widths. Also, this architecture is flexible in terms of membership function shape. The success of this study lies in finding the active membership functions at a low cost by very simple operations. Therefore the active rules are found with a minimal computational cost. This methodology introduces two blocks which are integrated in a conventional architecture, the blocks are the active rule detector (ARD) and the active rule queue (ARQ). The last block saves the active rules detected by the first block. This method analyzes the maximum number of processing elements (PEs) working in parallel to increase the speed, the delay introduced and the area occupied by this structure. Hamzeh et al. highlight that "*one of the most important contributions of our proposed algorithm is that the area and delay of its corresponding architecture is independent of the number of overlapping membership functions*".
- Javadi et al. (2013) in the article "A hardware oriented fuzzification algorithm and its VLSI implementation" introduced a new HW method for fuzzification which maximizes the effective resolution with a slightly higher implementation cost (slightly higher number of gates than a normal fuzzification method). The article has performed the study with trapezoidal membership functions but they highlight that the algorithm is extensible to any piecewise linear membership function. The DPF fuzzification algorithm method and a normal fuzzification method have been compared, showing the superiority of the DPF in the comparison; different word lengths for the input variables (WL: 7 bits .. 12 bits), slopes (named Coefficient-Length – CL: 7 bits) and a factor named Fraction-Length (FL: 0 bits .. 5 bits) are introduced. The last factor introduces a fraction in the input variable (integer+fraction). The design has been developed by means of VHDL language and synthesized on 0.13 μm CMOS library using a Mentor Graphics Leonardo Spectrum tool.

Architectures of sequential processing rules: The base of the rules are stored in memories and the membership functions are generated by a circuit shared by the rules. They are more flexible than the previous point but their flexibility is detrimental to the parallelism of fuzzy systems. The number of clock cycles is proportional to the number of rules. In systems with a high

Please cite this article as: Bosque, G., et al., Fuzzy systems, neural networks and neuro-fuzzy systems: A vision on their hardware implementation and platforms over two decades. Eng. Appl. Artif. Intel. (2014), http://dx.doi.org/10.1016/j.engappai.2014.02.008

number of inputs the size of the memory increases exponentially with the dimension of the entry. In this regard, and in order to optimize memory usage, an alternative organization to this is proposed. Table 10 shows some relevant approaches.

The following works deserve to be mentioned:

- Eichfeld et al. (1992) in the article "Architecture of a CMOS fuzzy logic controller with optimized operator organization and design" presented the architecture of a fuzzy controller integrated into a multipurpose controller. The target is to implement the memory of rules in a minimal memory space. The controller algorithm is of type Mamdani *max–min* and the defuzzification is a Center of Gravity (COG) method. The controller has 4 inputs, one output and can store up to 4096 rules. The membership functions overlapped by pairs, so there are a total of 16 rules active for a given input vector. These are executed in 16 clock periods. The simulation assumes a 0.6 μm CMOS process but there is no information about the speed.
- Yubazaki et al. (1998) in the article "Fuzzy inference chip FZP-0401A based on interpolation algorithm" performed a new fuzzy inference chip and proposed a new high-speed linear interpolation algorithm applied to the Product-Sum-Gravity method (Mizumoto, 1991). The chip supports a fuzzy system with 4 inputs and 1 output of 16 bit. Each input allows up to 7 membership functions (limitation of the memory capacity of Look-Up-Table) and the shape must be a normalized trapezoid or a triangle, and the sum of the memberships of an input must be 1.0. The consequent part has up to 255 labels and the maximum number of rules is 2401 (7^4). Due to the shape of the membership functions and the consequents, a linear equation is performed that allows us to interpolate the outputs. The chip allows us to load the fuzzy parameters from parallel ROM or serial ROM into the internal RAM whenever it is turned on, and works in a floating point for improving computation precision (24 bits: 8 exponent and 16 mantissa). The inference speed of the chip is about 0.48 MFLIPS (2100 ns), with a clock frequency of 14.2857 MHz, independent of the number of fuzzy rules and membership functions. The chip is designed in a CMOS process of 0.6 μm.
- Evmorfopoulos and Avaritsiotis (2002) in the article "An adaptive digital fuzzy architecture for application-specific integrated circuits" presented a generic fuzzy logic system implemented on an ASIC applied to window-type air conditioners. There is one analog input (8 bit ADC) from a thermistor sensor (ambient temperature) and two digital inputs from two up/down buttons that define the temperature setting (resolution: 0.5 °C; range: 16–32 °C). The outputs are two PWM (Pulse-width Modulation) to two DC motors (compressors) and one digital on/off for a relay. The membership functions are Gaussians but they do not specify the maximum number of membership functions. In the application there are 5. The clock frequency is 32 kHz. The chip is designed in a CMOS process of 0.7 μm.

4.2.2. Programmable Integrated Circuits

In this classification, FPGAs will be emphasized because they can be programmed through circuit design using graphic or, preferably, HW description languages like VHDL (Very High Speed Hardware Design Language) or Verilog. Further development of FIS has been focused on the FPGAs because of their ability to reconfigure and the low "time to market". For example, consulting the "Web of Knowledge – Web of Science" the number of works on FPGAs are around 340 between 1990 and 2012. Fig. 1 shows this production.

Some relevant approaches in this kind of implementations are shown in Tables 11–13.

In the cited references, the following twelve implementations can be point out (*):

- Manzoul and Jayabharathi (1992) presented in the work "Fuzzy controller on FPGA chip" a fuzzy controller expressed as Boolean equations on a FPGA. The device used is XC3020-50-PC68 of Xilinx® Xilinx to 50 MHz. It is a MISO (Multiple Input Single Output) system type; the system of IF–THEN rules is processed externally to the FPGA and the Boolean equations obtained are loaded into the device. The speed achieved is 50 MFLIPS.
- Hung and Zajak (1995) presented the implementation of a fuzzy inference system on a FPGA in the article "Design and

Table 10
HW implementations of fuzzy systems - Digital Dedicated Integrated Circuits – sequential rules.

Year	Applications	Ref.	Type of FS	In	No. of in MFs	Type of in MFs	Out	No. of out MFs	Type of out MFs	No. of rules	Defuzz.	Speed	Technology
1992	Generic	Eichfeld et al. (1992)*	Mamdani	4 5 bit	7–···–7 3 bit	Trapz., triang.	1 5 bit	8 3 bit	Trapz., triang.	4096	COG	–	CMOS 0.6 μm
1995	Generic	Eichfeld et al. (1995)	IF–THEN	256 6 bit	7–..–7 3 bit	Any shape	64 6 bit	8 3 bit	Crisp	16,384	COG	10 MFRPS	CMOS 1.0 μm
1998	Generic	Yubazaki et al. (1998)*	Prod–sum-grav.	4 16 bit	7–..–7	Trapz., triang.	1 16 bit	255	Any shape	2401	COG	0.48 MFLIPS	CMOS 0.6 μm
1998	Generic, radar	Cardarilli et al. (1998)		2 1 8 bit	Max: 16/ In 4	Trapez.	1 1 8 bit	Max: 7 4	Trapez.	Max: 256 4	COG (LUT)	100 KFLIPS (with 20 rules) 400 KFLIPS	CMOS 1.0 μm
2002	Generic, air cond.	Evmorfopoulos and Avaritsiotis (2002)*	Sugeno	3 8 bit	5-4-5	Gaussian (LUT)	2 8 bit	9/5	Singletons	25/20	Sugeno	–	CMOS 0.7 μm

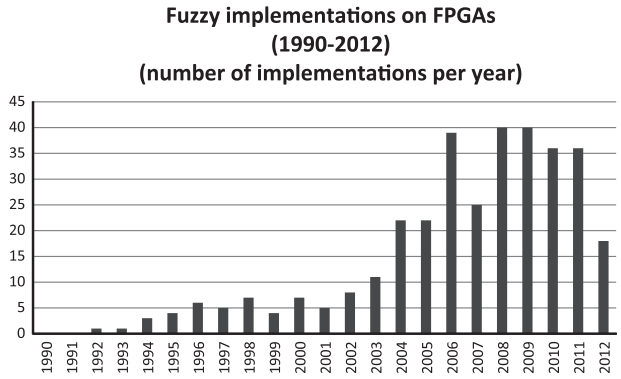

Fig. 1. Fuzzy implementations on FPGAs.

implementation of a hardware fuzzy inference system", based on generalized *modus ponens* rules, *max–min* composition and defuzzification COG method. The implementation has been performed with two inputs (3 triangular membership functions per input) and 1 output (9 singleton membership functions). The speed achieved is 1.67 MFLIPS without pipelining and 9 MFLIPS fully pipelined. The defuzzification is supported by an external EPROM (Cypress CY7C251). The FPGA is XC4008-6 of Xilinx®.

- Hollstein et al. (1996) in the article "Computer-aided design of fuzzy systems based on generic VHDL specifications" presented a development tool for performing parallel processing architectures or sequential rules. Each FIS consists of three distinct modules: fuzzification, rules of inference and composition/defuzzification. Modules can be described in C or VHDL. The defuzzification follows the MOA (Midpoint of Area) method. An example of a fuzzy controller is implemented on a Xilinx® FPGA, XC4006. It consists of 2 inputs with 5 membership functions, 11 rules and 1 output and controls the direction of a truck trailer. Each new output is estimated at 42 μs. The result is compared with other devices on which the FIS is implemented with a TMS320 DSP that require 150 μs and with the ASIC of Togai FC110, the calculation is performed in 32 μs.
- Blake et al. (1998) in the article "The implementation of fuzzy systems, neural networks and fuzzy neural networks using FPGAs" presented three approaches to the Soft Computing but the article will focus on the FIS. The interest of the article is, taking a non-linear function of three variables, to compare the approximation capacity between the architecture on a FPGA and the architecture on Matlab. The membership functions are triangular and the operator is *min*. The results are poor due to only choosing 2 membership functions, however the hardware results offer a similar degree of accuracy as the results achieved in Matlab. The FPGA is XC4008 of Xilinx®.
- D'Amore et al. (2001) in the article "A two-input, one-output bit-scalable architecture for fuzzy processors" presented an automatic synthesis of a fuzzy system with scalability, with either the bits of the I/O variables or the bits of the membership functions of the I/O. The synthesis process is performed using the VHDL code. It implements the classic problem of parking a truck, needing 35 rules, 2 inputs (5 and 7 membership functions respectively – triangular and trapezoidal) and 1 output (7 singleton membership functions). The implementation is performed on an Altera® (Altera) FPGA Flex10K and performs a comparison with a different number of bits of I/O (8–10–12–14–16) and different number of bits of membership functions (5–7–9–11–13) to give updates of the output between 3.0 μs and 6.3 μs, depending on the number of bits chosen.
- Raychev et al. (2005) in the work "VHDL modelling of a fuzzy co-processor architecture" presented a hardware accelerator for fuzzy calculations. The co-processor is connected to an external μ P and to a RAM (fuzzy inference rules database). The operations are realized by means of a 16-bit floating-point format (sign: 1 bit; exponent: 5 bit; mantissa: 10 bit). All functionality has been previously simulated. This architecture has been implemented on a FPGA design platform XC4005.
- Hung (2007) in the article "Using FPGA technique for design and implementation of a fuzzy inference system" made the implementation of a fuzzy inference system with *max–min* compositional rule with the COG being the defuzzification method applied. To increase the speed at defuzzification a Lookup-Table is used, thus eliminating the multiplication and division. The knowledge base is stored in an external EPROM, Cypress CY7C251 and the FPGA is the XC4008-6 of Xilinx®. The proposed architecture applies it to two inputs and one output, achieving speeds of 3.3 MOCS (Million Operations of Control per Second) for a control clock of 10 MHz.
- Lizárraga et al. (2008) in the article "Modeling and simulation of the defuzzification stage using Xilinx system generator and Simulink" focused attention on the defuzzification stage, using the Height method (Driankov et al., 1996) and performed a comparison between two systems: Simulink with System Generator and Matlab/Simulink, using three platforms: Simulink from Mathworks, Xilinx ISE and Xilinx System Generator. The fuzzy controller has two inputs and one output. The inputs and the output have five membership functions (two trapezoidal and three triangular) and are represented by 8 bits. They proposed the use of a modified high performance fixed point architecture for positive numbers, and the final stage made the conversion to real numbers. The target is a Spartan 3 with a System Generator.
- Thareja et al. (2009) in the work "Configurable fuzzy logic coprocessor for small scale food preparation" showed a practical application for an oven control which cooks steaks. The input membership functions are described using triangular and trapezoid shapes and the output membership functions are described using triangular shapes. There are two FIS: the first has two inputs (thickness-doneness) and one output (seconds) with 25 inference rules, and the second has two inputs (current temperature and temperature change) and one output (heat needed) with 15 inference rules. There are five ranges of temperature between 246 °C and 274 °C. The inference engine is a Mamdani-type. The clock rate achieved is 179.9 MHz. The Configurable Fuzzy Logic Coprocessor (CFLC) designs were synthesized using Altera Quartus II 9.0. The general purpose processor selected, to interconnect the CFLC via a coprocessor interface, is the SPARC V8e based LEON3 processor. This processor is an IP core (Gaisler et al.). The article established a comparison with two other platforms based on a general purpose processor Tensilica Xtensa Baseline Processor (TXP) (Tensilca), one with the CFLC and the other with a Fuzzy Logic Tie (FLT). The results are favorable for the CFLC. The article does not mention the FPGA used.
- Fung et al. (2009) in the article "FPGA-based adaptive fuzzy backstepping control for a micro-positioning Scott–Russell mechanism" presented a fuzzy controller with an error feedback mechanism applied to a micro-positioning Scott–Russell type. The actuator was piezoelectric. The displacement sensor range is 100 μm, and the accuracy is 10 nm. The controller is developed in VHDL and implemented on Stratix II device from Altera®. On this FPGA the NIOS processor is also instantiated.
- Hsu et al. (2010) in the article "Chip-implementation of a self-tuning nonlinear function control for DC–DC converters" proposed a model-free STNFC design method suitable for real-time practical applications. The system is applied to a DC–DC converter based on a FPGA controlling the duty-ratio of PWM modulator in the DC–DC converter. The article highlights the following points: STNFC is a system without heavy computational loading, the

Please cite this article as: Bosque, G., et al., Fuzzy systems, neural networks and neuro-fuzzy systems: A vision on their hardware implementation and platforms over two decades. Eng. Appl. Artif. Intel. (2014), http://dx.doi.org/10.1016/j.engappai.2014.02.008

Table 11
HW implementations of fuzzy systems – Digital Programmable Integrated Circuits (1).

Year	Applications	Ref.	Type of in MFs	In	No. of in MFs	Type of out MFs	Out	No. of out MFs	Type of out MFs	No. of rules	Defuzz.	Speed	Device
1992	Controller	Manzoul and Jayabharathi (1992)*	IF–THEN	2	4–4	Any	1	4	Any	6	–	50 MFLIPS	XC3020-50
1994	Plants classification	Halgamuge and Glesner (1994)	IF–THEN	Max: 128 3	Cluster 3–3–3	Sigmoid (PWL)[a]	Max: 4 1	Tuned 3	Trapez.	Max: 256 7	CBADD[b]	1.25 MFRPS	XC4006
1995	Controller	Hung and Zajak (1995)*	GMP[c]	2 6 bit	3–3 4 bit	Triang.	1 8 bit	9	Singleton	9	COG (LUT)	1.67 MFLIPS/ 9 MFLIPS	XC4008-6
1996	CAD truck control, etc.	Hollstein et al. (1996)*	IF–THEN	2 4 128 8 bit	5–5 4–..–4 128–..–128 4 bit	Triang.	1 3 4 8 bit	– 4 bit	Triang.	11 16 256	MOA[d]	42 μs 1.25 MFRPS	XC4006 XC4005 XC4006
1998	General	Blake et al. (1998)*	T–S	3	2–2–2	Triang.	1	–	Singleton	8	T–S	–	XC4008
2000	Truck control	Kim (2000)		2 8 bit	5–7 (LUT)	Triang., trapez.	1 8 bit	7	Triang.	27	COG	–	XC4013
2001	Automatic synthesis: park truck	D'Amore et al. (2001)*	IF–THEN	2 8/10/12	5–7 /14/16 bit	Trapez., triang.	1 Idem In	7	Singleton	35	Weighted average	3/3.6/4.5 5.4/6.3, μs (ref. no. of bits)	Flex 10K
2003	Car parking	Li et al. (2003)	IF–THEN	2 8 bit	5–5 3–3 5 bit	Triang.	1 1 4 bit	5 3	Singleton	25 9	Weighted average	–	EPF6024ACT144
2004	Controller	Cabrera et al. (2004)	IF–THEN	2 3 6/8 bit	3–3 5–5–5 5/8 bit	Any	1 6/8 bit	5 7 5/8 bit	Singleton	9 125	Fuzzy mean	–	XC4005XL XC2S200E
2005	Evolutionary FS	Mermoud et al. (2005)	IF–THEN	4 8 bit	3–..–3 4 bit	Trapez., triang.	1 8 bit	4	Rectang.	20	COA	–	Xilinx-Virtex (simulated)
2005	Co-processor	Stefano and Giaconia (2005)	IF–THEN (pipeline)	2 8 bit	Max: 16–16 3–3	Triang.	1 8 bit	Not fixed 9	Singleton	Not fixed 9	COG	– 0.4 μs	Spartan3-200 (SoPC)

[a] Piecewise linear.
[b] Customizable basic defuzzification distributions.
[c] Generalized modus ponens.
[d] Midpoint of area.

parameter-learning algorithm is designed based on the Lyapunov stability theorem to guarantee the system stability, there are successful applications of the STNFC system to control the forward DC–DC converter, and finally, the proposed STNFC methodology can be easily extended to other DC–DC converters. The FIS has one input (three membership functions – two trapezoidal and one triangular) and one output (three singleton), and uses three fuzzy rules. The device is Altera Cyclone II EP2C20F484 at 50 MHz with a μ P NIOS embedded. The algorithm is realized in the NIOS with on-line learning. Comparing the performance with other controls such as PI, FC, NFC and STNFC (without learning), the results are favorable to the STNFC with learning.

- Kung et al. (2011) in the work "Simulink/Modelsim co-simulation and FPGA realization of speed control IC for PMSM drive" implemented a fuzzy-control based speed control IC for a Permanent Magnetic Synchronous Motor (PMSM). Firstly the HW Co-Simulation attending to the development environment Matlab–Simulink (from Mathworks) is performed, and second, the algorithm code control, written in VHDL, is simulated on ModelSim. The code validated is downloaded on the FPGA. There are two input variables (error of angular speed and error change) and one output variable (speed voltage control). The fuzzy controller uses a singleton fuzzifier, a triangular membership function (7 for each variable: 2 trapezoidal and 5 triangular), product inference rule and a central average defuzzifier method. The chip adopted is Altera CycloneII EP2C35 with a Nios II embedded processor. The difference between the simulation and the experimental results are very similar. It presents a good speed following response without overshooting.

4.2.3. Commercial processors

In this paragraph general purpose processors will be considered with the classical repertoire instructions as well as general

Please cite this article as: Bosque, G., et al., Fuzzy systems, neural networks and neuro-fuzzy systems: A vision on their hardware implementation and platforms over two decades. Eng. Appl. Artif. Intel. (2014), http://dx.doi.org/10.1016/j.engappai.2014.02.008

Table 12
HW implementations of fuzzy systems – Digital Programmable Integrated Circuits (2).

Year	Applications	Ref.	Type of FS	In	No. of in MFs	Type of in MFs	Out	No. of out MFs	Type of out MFs	No. of rules	Defuzz.	Speed	Device
2005	Co-processor (Generic)	Raychev et al. (2005)*	IF–THEN	–	–	–	–	–	–	–	COG, max-average	–	XC4005
				16 bit	FP[a]		16 bit	FP					
2006	General	Barriga et al. (2006)	IF–THEN	2	3–3	Triang.	1	5	Singleton	9	–	35 ns	XC2S200E
					7–7					49		108 ns	
					6 bit		6 bit						
2006	Climate control	Castaneda-Miranda et al. (2006)	IF–THEN	4	7–..–7	Triang.	2	7	Triang.	16 (actives)	Centers average	77 KFLIPS	A54SX32A (SoC – Actel)
					12 bit		12 bit						
2006	Context switching	Cao et al. (2006)	FIM[b]	2	$v-w$	Triang.	1	–	Singleton	$v \times w$	Aggregation	17 MFLIPS	XC2V2000
					5–5			5		25			
				6 bit	4 bit		–	3 bit					
2007	Controller	Hung (2007)*	GMP	2	3–3	Simetric (LUT)	1	9	Singleton	9	COG (div. free)	3.3 MOCS[c]	XC4008-6
				6 bit	4 bit		8 bit	4 bit					
2007	Robotic (car park)	Sánchez-Solano et al. (2007)	IF–THEN	5	–	Triang (unrestricted)	2	–	–	6	FM[d], etc.		Spartan-3 (SoPC)
				–	–		–	–					
2007	Simplicial PWL	Echevarria et al. (2007)	PWL fuzzy	2	–	–	1	–	–	–	COG (subcell)	70 MHz/ 140 MHz	Virtex II Pro
				8 bit	–		–	–					
2008	Incremental controller	Millán et al. (2008)	IF–THEN	1	5	Trapez., triang.	1	5	Trapez., triang.	5	–	–	Simulink/Xilinx System Generator
				–	–		–	–					
2008	Bit-serial arithmetic	Dick et al. (2008)	Bit-serial min/max	2	3–3	LUT	1	9	Singleton	9	Centroid	5.26 MFLIPS	EP1S80B956C6
				6 bit	3 bit		–	–					
2008	Elevator systems	Muñoz et al. (2008)	IF–THEN	5	3–3–7–3–3	Trapez., triang.	2	–	–	9	–	–	Spartan 3
				–	–		–	–					
2008	Clinical diagnostic	Chowdhury et al. (2008)	IF–THEN	6	3–..–3	Trapez., triang.	1-mux4	–	Crisp	4	–	0.292 μs	EP1K6Q240C8
				–	–		7+4 bit	–					

[a] Floating point.
[b] Fuzzy inference mapping.
[c] Million operations of control per second.
[d] Fuzzy mean.

purpose processors which have introduced fuzzy processing instructions or incorporate a fuzzy coprocessor. The DSP (Digital Signal Processor) devices will also be examined.

General purpose processors: These are based on simple and widespread architectures, of low-cost but present low running speeds. The implementation of a fuzzy system is performed fully SW, allowing a very high generalization. It should be emphasized that this solution is still valid today in all environments where speed is not critical for implementation. Table 14 shows some relevant approaches.

The following works will be highlighted (*):

- Jackson (1994) in the article "Fuzzy logic vs. traditional approaches to the design of microcontroller-based systems" made the comparison of a fuzzy controller in a closed loop with a PID controller and with a pole/zero compensation controller. The implementations are made on the μ Cs of Motorola® MC68HC16 and MC68332 16-bit and 32-bit respectively. The system to be controlled is a motor positioning by pulses width modulated (PWM). The article only provides the results of a previous simulation of the three drivers. The settling times obtained for a step function is 85 ns, 80 ns and 60 ns, respectively, with sampling times of 0.5 ms. The first two controllers perform a smooth approximation, while the third shows an overflow (overshoot) at 30 ms. The article highlights the point that the fuzzy controller requires less engine power performance than the other two. In addition, it is not necessary to have theoretical knowledge of closed loop control.
- Binfet and Wilamowski (2001) in the work "Microprocessor implementation of fuzzy systems and neural networks" presented two classes of control: fuzzy and neural. The work will focus on the first. The fuzzy system is implemented on the 8-bit μ C 68HC711E9, from Motorola, with an internal clock frequency of 2 MHz. The control can use three different

Table 13
HW implementations of fuzzy systems – Digital Programmable Integrated Circuits (3).

Year	Applications	Ref.	Type of FS	In	No. of in MFs	Type of in MFs	Out	No. of out MFs	Type of out MFs	No. of rules	Defuzz.	Speed	Device
2008	Defuzz.	Lizárraga et al. (2008)*	IF–THEN	2 8 bit	5–5	Trapez., triang.	1 8 bit	5	Trapez., triang.	–	Height method	–	Spartan 3
2009	Oven control	Thareja et al. (2009)*	Mamdani	2 2	5–3 5–5	Triang., trapez.	1 1	20 5	Triang.	25 15	COG	–	RC10 (Celoxica)
2009	Micro-positioning	Fung et al. (2009)*	IF–THEN	2 12 bit	5	Triang.	1 12 bit	5	Singleton	25	COA	–	EP2S60F1020 (Stratix II-SoPC)
2010	DC–DC converters	Hsu et al. (2010)*	IF–THEN	1 10 bit	3	Trapez., triang.	1 10 bit	3	Singleton	3	COG Lyapunov (on-line)		EP2C20F484 (SoPC)
2010	Mobile robots	Tzafestas et al. (2010)	T–S	2 12 bit	9–9 8 bits	Trapez., triang.	1 12 bit	5t 8 bit	Singleton	81	Weight average	126.76 ns	Spartan 3 (SoC)
2010	Crack detection	Arati et al. (2010)	IF–THEN	3 2 bit	3–3–3 –	Triang.	1 2 bit	3 –	Triang.	6	–	–	FPGA (generic)
2011	Compensator (dc-servomot.)	Hsu et al. (2011)	IF–THEN	3 10 bit	3	Triang.	1 10 bit	3	Singleton	3	COG	–	3C16 (Cyclone III)
2011	Solar energy	Chekired et al. (2011)	IF–THEN	2 –	5–5 –	Trapez., triang.	1 –	5 –	Trapez., triang.	25	COA	–	V2MB1000 Virtex-II
2011	Solar energy	Messai et al. (2011)	IF–THEN	2 12 bit	5–5	Trapez., triang.	1 12 bit	5	Triang.	25	COG		XC2V1000
2011	Speed control	Kung et al. (2011)*	IF–THEN	2 16 bit	5–5	Trapez., triang.	1 16 bit	–	Singleton	–	Central average	–	EP2C35 (SoPC)

membership functions: trapezoidal, triangular, and Gaussian. It presents two different defuzzification processes: Zadeh and Takagi–Sugeno (T–S). The aim of this work is to emulate a control surface employing trapezoidal and triangular membership functions. In fact the work performs a comparison between different fuzzy systems (Zadeh and T–S) and different neural networks. The fuzzy and neural implementations show that the control surfaces obtained from neural controllers do not exhibit the roughness of fuzzy controllers. Table 15 shows the comparison between four fuzzy systems (input: 2; membership function: 7 per input; output: 1; membership function: 7).

- Nhivekar et al. (2011) in the article "Implementation of fuzzy logic control algorithm in embedded microcomputers for dedicated application" implemented a fuzzy inference unit and algorithms for fuzzification, rule evaluation and defuzzification of a fuzzy closed-loop control system on a μ C Atmega8 from Atmel. The fuzzy logic controller is applied to a temperature control system, with unknown dynamics, in real time and has an internal set-point temperature. One input (temperature) and one output (temperature control) are presented. The number of membership functions for the input is 5 (trapezoidal: 2; triangular: 3) and the number of membership functions for the output is 5 (triangular). The defuzzification is obtained by a weighted average technique and is a numeric (crisp) value that determines the firing angle of the Triac used to drive the heater.

Processors that include some fuzzy instructions in the instruction set: The fuzzy operations are performed at high speed and the total speed of the process is a compromise between general-purpose processors and those which incorporate a fuzzy coprocessor. The mode of operation is similar to the above but obtains fuzzy systems faster. This type of processor enables high generalization and medium cost. Table 16 shows these kinds of implementations. For example, let us comment the first reference in which Watanabe and Chen, in 1993, showed that the speed of a fuzzy control program increased by 2.57 times by adding two instructions, *max* and *min*, in the instruction set of a RISC processor. In 1996 Motorola® introduced the family μ Cs 68HC12[15] of 16-bit and incorporated an instruction set characteristic of fuzzy algorithms. Among the companies that are currently still manufacturing these

[15] This microcontroller family has been withdrawn by Motorola in 2007 and no further references of current Motorola μ Cs or μ Ps containing a subset of fuzzy instruction.

Please cite this article as: Bosque, G., et al., Fuzzy systems, neural networks and neuro-fuzzy systems: A vision on their hardware implementation and platforms over two decades. Eng. Appl. Artif. Intel. (2014), http://dx.doi.org/10.1016/j.engappai.2014.02.008

Table 14
HW implementations of fuzzy systems – Digital Commercial Processors.

Year	Applications	Ref.	Type of FS	In	No. of in MFs	Type of in MFs	Out	No. of out MFs	Type of out MFs	No. of rules	Defuzz.	Speed	Device
1994	Motor control	Jackson (1994)*	IF–THEN	2	7–7	Triang., trapez.	1	3	Singleton	49	Mamdani	–	μ C MC68HC16, μ C MC68332
				–	–		–	–					
1997	Cache management	Hammami (1997)	Mamdani	3	–	–	1	–	–	18	Weighted average (LUT)	–	μ P R4000
				2		–	1			34		–	
				4		–	2			81		–	
				64 bit			64 bit						
2001	Control	Binfet and Wilamowski (2001)*	Zadeh, T–S	2	7–7	Trapez., triang., Gaussian	1	7	–	–	Zadeh, T–S	See Table 15	μ C MC68HC711
				8 bit			8 bit						
2010	Mobile robot	Feng et al. (2010)	IF–THEN	3	–	Hyper-ellipsoid	2	–	Singleton	5	Weighted average	–	Simulated
				–	–		–	–					
2011	Temp. control	Nhivekar et al. (2011)*	IF–THEN	1	5	Trapez., triang.	1	5	Triang.	5	Weighted average	–	μ C Atmega8
				8 bit			8 bit						

Table 15
Error comparison between four fuzzy systems.

Type of fuzzy system	Type of in MFs	Processing time (ms)	Error (SSE)
Zadeh	Trapez.	1.95	908.4
Zadeh	Triang.	1.95	644.4
T–S	Trapez	28.5	296.5
T–S	Triang	28.5	210.8

devices, ST can be cited, which makes the μ C ST52513G of 8-bit[16] and the ST FIVE 508 of 8 bits. At this point, it is important to note that large manufacturing companies of μ Ps or μ Cs still show no interest in the architecture of this type or, those who did, are no longer interested, with the exceptions mentioned.

The following works will be highlighted (*):

- Ungering et al. (1994) in the article "Architecture of a fuzzy-processor based on an 8-bit microprocessor" presented an architecture based on μ P 6502 of MOS Technology, Inc.[17] and performs an expansion of the ALU (Arithmetic Logic Unit) used only for the defuzzification, introducing fuzzy firmware that performs basic operations. This change represents an increase of hardware size by 20% but achieves speeds 6 times higher than the implementation on the μ P 6502 standard. The inferences are calculated by implementing a microcode on a EPROM and the defuzzification is performed with the COG method. The system supports 15 inputs (8 membership functions for every input) and 8 outputs (8 singletons for each output). The article does not mention the type of membership functions but the figures are triangular and trapezoidal. The prototype has been implemented on two FPGAs (XILINX 3090).
- Watanabe et al. (1996) presented the work "Evaluation of *min*/*max* for fuzzy information processing instructions," which is a slight revision of the previously referenced work (Watanabe and Chen, 1993). They keep the speed increased 2.57 times by introducing the two statements, *max* and *min*, to the instruction set of the MIPS® R3000A RISC processor. The proposed modification is verified on image processing. The paper presents a simulation of these two operators on the MIPS processor and a comparison of an ASIC full-custom fuzzy controller operating at 10 MHz (Dettloff et al., 1989). The results are 48 FLIPS for R3000A, 122 FLIPS for R3000A (with *min*/*max*) and 158 KFLIPS for the ASIC.[18]
- Salapura (2000) presents a RISC processor in the article, "A fuzzy RISC processor" as an extension of RISC processors of the MIPS® company Computer Systems Inc.[19] A basic instruction set adds four fuzzy instructions (fuzzification, rule evaluation, *min* and *max*), with the new design being called MIPS-F. The FIS implemented are of Mamdani type and the defuzzification is performed by the Yager method (COG modified). The membership functions are triangular and trapezoidal but the article does not mention any restriction on other membership functions. The article points out that the membership degrees are packed in multiple fuzzy sets in a single processor word using the subword parallelism technique (Lee, 1996) and using 8-bit for encoding membership degrees and 4-bit for set identification. This technique allows a word to contain membership information for two overlapping fuzzy sets (on 32-bit processors). The development is carried out based on the description on VHDL of the core and realized a prototype on FPGA, with the final development of an ASIC. A comparison of three applications between MIPS and MIPS-F is performed. In the first application (gas heater) there were 151 cycles and 110 cycles respectively (approximately 1.37 faster), in the second application (Asynchronous Transfer Mode (ATM) communications network) there were 249 cycles and 168 respectively (approximately 1.48 faster) and in the third application (video encoding) 557 cycles and 301 cycles are obtained (approximately 1.85 faster) respectively.

Fuzzy coprocessors: The incorporation of fuzzy coprocessors leads to high speed fuzzy operations, being limited mainly by the

[16] At present no longer recommended for new developments.
[17] A commercial level Rockwell was the second most important source of this processor.
[18] We think that there are some mistakes in the Table V of the paper referred.
[19] Later known as MIPS Technologies. Models have been the R2000, R3000, R4000, etc.

Please cite this article as: Bosque, G., et al., Fuzzy systems, neural networks and neuro-fuzzy systems: A vision on their hardware implementation and platforms over two decades. Eng. Appl. Artif. Intel. (2014), http://dx.doi.org/10.1016/j.engappai.2014.02.008

Table 16
HW implementations of fuzzy systems – Digital Commercial Processors with fuzzy instructions.

Year	Applications	Ref.	Type of FS	In	No. of in MFs	Type of in MFs	Out	No. of out MFs	Type of out MFs	No. of rules	Defuzz.	Speed	Device
1993	Generic	Watanabe and Chen (1993) (see Watanabe et al., 1996)	IF–THEN	–	–	–	–	–	–	51	–	48/122 FLIPS	μ P R3000A
1994	Control	Ungering et al. (1994)*	IF–THEN	Max: 15 4 7 bit	Max: 8 7–..–7	Trapez., triang.	Max: 8 1 8 bit	Max: 8×8 7	Singleton	(N_R) 40	COG	–	μ P M6502
1996	Generic, image processing	Watanabe et al. (1996)*	IF–THEN	– 6 bit	– 4 bit	–	– 6 bit	– 4 bit	–	51	–	48/122 FLIPS	μ P R3000A
2000	Gas heating, image processing, communications, etc.	Salapura (2000)*	Mamdani	2 3 3 8 bit	3–3 3–3–3 4–3–4	Triang. Trapez., triang.	1 1 1 8 bit	5 7 5	Triang. Trapez., triang. Trapez.	9 18 48	COA	MIPS-F vs. MIPS 37–85%↑	μ P MIPS

communication of data between the processor and the coprocessor. This architecture allows for high generalization. The cost of this type of architecture is not very high. The company Omron® introduces, among others, the coprocessor FP5000 (Shimizu et al., 1992), and for the family of PLC C200 the module FZ001 (Omron). With regard to coprocessors, the same considerations must be applied as those in the previous section, for the low commercial value of these devices for large manufacturing companies μ Ps or μ Cs. Table 17 shows these kinds of implementations.

The following works will be highlighted (*):

- Eichfeld et al. (1993) presented the work "An 8 b fuzzy coprocessor for fuzzy control", which proposed a design to be integrated into a μ C. The features are, 256 inputs max, 64 outputs max, 16,384 number of rules max and a peak performance of 7.9 MFRPS with a clock of 20 MHz. The work presents an example with 4 inputs, 1 output and 1000 rules, obtaining a calculation time of 126.5 μs with a clock of 10 MHz. The prototype is connected to a μ C and the knowledge base memory (KBM) is located outside the coprocessor. The design method was VHDL logic synthesis and the integration technology was 1.0 μm CMOS.
- Eichfeld et al. (1998) presented the work "Applications of SAE 81C99x fuzzy coprocessors", based on Siemens coprocessors SAE 81C99A of 8-bit and SAE 81C991 of 12-bit. The 8-bit coprocessor provides a standard interface to connect to μ Cs or μ Ps. It has an on-chip memory KBM (Knowledge Base Memory), enabling storage of a total of 16,384 rules. The membership functions of inputs and outputs can be programmed freely on an area of 8 bits × 6 bits. There are three defuzzification methods such as COG, FM (First Maximum) and FL (Last Maximum). With regard to benefits in speed, with a clock frequency of 20 MHz, a system with 4 inputs, 1 output and 80 rules is resolved in 21 μs (3.8 MFRPS). In this work, the first coprocessor is applied to the brake system of motor vehicles (Brake by Wire systems (BBW)) and also involves a testbench, comparing the architecture "processor–coprocessor" with a general purpose microcontroller (16-bit Siemens μ C C167) which implements the SW of a high dimensionality fuzzy system. The three fuzzy systems, based on 81C99A, are shown in Table 18. The result is clearly superior in the architecture that incorporates the coprocessor. The 12-bit coprocessor presents 4096 inputs (programmed freely on an area of 12 bits × 12 bits), 1024 outputs (programmed freely on an area of 16 bits × 12 bits), 131,072 rules. The defuzzification methods are the same as those used on the 8-bit coprocessor. A system with 4 inputs, 1 output and 80 rules is resolved in 16 μs (5 MFRPS). Finally, with this coprocessor an image processing application (image segmentation) was performed and a comparison was made with a SUN Sparc20 workstation. Processing 40,000 pixels, the processing time is less than 1 min, while with a HW system with four coprocessors, the processing time is less than 0.2 s at 25 MHz.
- Garcia and Pedro (1996) presented an update of the Coprocessor ORBEX, at work "First applications of the Orbex coprocessor: control of unmanned vehicles" (Garcia and Pedro, 2000). In this application, the coprocessor works with a Ciryx processor and performs the control of movement of an electric vehicle that travels a circuit of 1 km in length.

Commercial DSPs: Few works have focused on the fuzzy logic on these devices. Table 19 shows the different implementations.

The following work will be considered (*):

- Bal et al. (2004) in the article "Fuzzy logic based DSP controlled servo position control for ultrasonic motor" highlighted that the ultrasonic motor (USM) is highly nonlinear, the control characteristics are complicated and the motor parameters are time-varying due to increases in temperature and changes in motor drive operating conditions, such as driving frequency, source voltage and load torque. It is for these reasons that a fuzzy logic control integrated into the position control loop is implemented. The number of inputs is two (position error and speed) and one output (frequency). The output control is a PWM signal. The number of membership functions is 7 (trapezoidal: 2 and triangular: 5) for both inputs and for the output 7 (singleton). The article shows the robustness and the performance (step speed – ramp responses – periodical step position responses) of the proposed drive and control system. The device chosen is a digital signal processor (DSP). The DSP is TMS320F243 from Texas Instruments® company.

4.3. Mixed fuzzy implementations

Mixed fuzzy implementations present an alternative to integrally analog circuits, combining the capabilities of digital and analog solutions. The fuzzy inference engine is implemented on

Please cite this article as: Bosque, G., et al., Fuzzy systems, neural networks and neuro-fuzzy systems: A vision on their hardware implementation and platforms over two decades. Eng. Appl. Artif. Intel. (2014), http://dx.doi.org/10.1016/j.engappai.2014.02.008

Table 17
HW implementations of fuzzy systems - Digital Fuzzy Coprocessors.

Year	Applications	Ref.	Type of FS	In	No. of in MFs	Type of in MFs	Out	No. of out MFs	Type of out MFs	No. of rules	Defuzz.	Speed	Technology
1993	Automotive	Eichfeld et al. (1993)*	IF–THEN	Max: 256 4 8 bit	–	Arbitrary (LUT)	Max: 64 1 8 bit	–	–	Max: 16,384 1000	COG, MOM[a]	7.9 MFRPS (peak perform.)	CMOS 1.0 μm
1996	Machine tool, sensors, etc.	Eichfeld et al. (1996)	IF–THEN	4096 4 12/22 bit	15,360/ in – 12 bit	Arbitrary (LUT)	1024 1 16 bit	16,384/ out – 12 bit	Arbitrary –	131,072 80	COG, FM, LM[b]	– 5 MFRPS	CMOS 0.8 μm
1998	Automotive, image, sensors, etc.	Eichfeld et al. (1998)*	IF–THEN	Max: 256 4 Max: 4096 4	Max: 64 – Max: 1024 –	Arbitrary (LUT)	Max: 64 1 Max: 1024 1	– –	– –	Max: 16,384 80 Max: 131,072 80	COG, FM, LM	– 3.8 MFRPS – 5 MFRPS	CMOS 1.0 μm CMOS 0.8 μm
2000	Unmanned vehicles	Garcia and Pedro (2000)*	IF–THEN	3 –	– –	Trapez.	2 –	– –	– –	–	–	–	–

[a] Mean of maximum.
[b] First/last max.

Table 18
System with the 81C99A for the testbench.

System	Inputs	Outputs	Rules	Processing time (μs)		Speed factor: μ C / (Copr. + μ C)
				Copr. + μ C	μ C	
1	2	1	9	33.8	96.8	3
2	4	1	72	43.8	768.0	18
3	5	1	288	113.0	3350.0	30

Table 19
HW implementations of fuzzy systems – Digital DSPs.

Year	Applications	Ref.	Type of FS	In	In MFs	Type of in MFs	Out	Out MFs	Type of out MFs	No. of rules	Defuzz.	Speed	Device
2004	Real-time	Frías-Martínez and Fernández-Hernández (2004)	T–S	2	7–7	Trapez., triang.	1	7	Trapez., triang.	49	COG	8.6 MFLIPS	TMS320C6201
												5 MFLIPS	TMS320C6701
				3	7–7–7							4.4 MFLIPS	TMS320C6201
												3.5 MFLIPS	TMS320C6701
				4	7–..–7							1.5 MFLIPS	TMS320C6201
				8 bit			8 bit					2.2 MFLIPS	TMS320C6701
2004	Servo control	Bal et al. (2004)*	IF–THEN	2	7–7	Trapez., triang.	1	7	Singleton	49	COG	–	TMS320F243
				–	–		–	–					

the analog part, giving the system a high parallelization of inferences with low silicon area, high speed and low power consumption. The digital part allows the programming and storage system parameters. These systems are described in Table 20.

From these referenced works, should be mentioned the following (*):

- Miki and Yamakawa (1995) presented in the article "Fuzzy inference on an analog fuzzy chip" (this article is based on Miki et al., 1993), a fuzzy controller consisting of two devices. The first focused on the rules and the second on the defuzzification. The device of rules contains 4 rules may be connected in parallel to increase their number without affecting the process speed. The rules are processed in parallel and the inference mechanism is *max–min*. The membership functions are evaluated analogically. The basic controller has 3 input variables and one output variable but their architecture allows parallel devices to have $3n$ inputs and n outputs. The rules are stored

Table 20
HW implementations of fuzzy systems – mixed.

Year	Applications	Ref.	Type of FS	In	No. of in MFs	Type of in MFs	Out	No. of out MFs	Type of out MFs	No. of rules	Defuzz.	Speed	Technology
1993	Generic (rules and defuzz. chips)	Miki et al. (1993) (see Miki and Yamakawa, 1995)	IF–THEN	3 5 bit	11–11–11 3 bit	S and Z-shaped, triang., trapez.	1 5 bit	7 3 bit	Singleton	$4 \times n_{RuleChip}$ (parallel)	COG	0.63 MFLIPS .. .. 1.4 MFLIPS	BiCMOS 2 μm (rule chip) Bipolar 3 μm (defuzz. chip)
1995	Generic	Miki and Yamakawa (1995)*	IF–THEN	$3n$ –	(4–4–4)n 8 bit	Triang., trapez.	$1n$ –	7 3 bit	–	N_R (parallel)	COG	0.6–1 MFLIPS	BiCMOS 2 μm Bipolar 3 μm
1995	Generic	Ramirez-Angulo et al. (1995)	IF–THEN	5	5–..–5	Triang., trapez.	3	–	Singleton	3 (parallel, serial)	COG	–	CMOS 2 μm
1997	Generic	Baturone et al. (1997)	T–S	2	3–3	Triang,. trapez.	1	3	Singleton	9 (parallel)	T–S	–	CMOS 2.4 μm
1998	Generic	Bouras et al. (1998)*	IF–THEN	3 –	5–5–5 –	Triang., trapez.	1 –	5 –	Triang., trapez.	25 (parallel)	COG	0,52 MFRPS	CMOS 0.7 μm
2002	Real-time	Evmorfopoulos and Avaritsiotis (2002)	Sugeno	4 8 bit	5–5 4–5	Gaussian	1 1 8 bit	9 5	Singleton	25 20 (parallel)	T–S	–	CMOS 0.7 μm
2005	Generic	Amirkhanzadeh et al. (2005)*	T–S	2 –	3–3 –	Triang., trapez.	1 –	5 –	Singleton	9 (parallel)	T–S	15.87 MFLIPS	CMOS 0.35 μm
2007	Generic	Yosefi et al. (2007)*	IF–THEN	2 –	4–4 –	Gaussian, triang., trapez.	1 –	7 7 bit	Singleton	16 (parallel)	COA	16.6 MFLIPS	CMOS 0.35 μm

digitally on a memory that can be accessed externally and internally updated, allowing to perform drivers dynamically adaptive to be adjustable the parameters. The defuzzification method is Center of Gravity (COG). Process speed is 1 MFLIPS regardless of the defuzzification and above 0.6 MFLIPS with defuzzification. The chip is designed in 2 μm BiCMOS technology.

- Bouras et al. (1998) presented the work "Mixed analog–digital fuzzy logic controller with continuous-amplitude fuzzy inferences and defuzzification", in VLSI CMOS technology of 0.7 μm, where the inputs and outputs are analog and the programming of the controller is performed digitally. The analog values of I/O are evaluated without A/D and D/A devices. It has 3 inputs, one output and 25 programmable rules. The inference mechanism is the *max–min* and the specific value of the output is obtained through the COG method, with the integrals being evaluated continuously. The membership functions are triangular and trapezoidal and the memory that stores their parameters are capacitors. The controller was performed by designing two devices, a drive inference mechanism and the defuzzification. The inputs are acquired sequentially and the *min* value of the currents associated with the corresponding membership function is also calculated sequentially, obtaining the current value associated with each rule which is introduced to block defuzzification. This block contains an integrator, a divider, a multiplier and a converter I/V. The total process time is 48 μs.
- Amirkhanzadeh et al. (2005) presented the work "A mixed-signal current-mode fuzzy logic controller", in which the analog part is dedicated to network computing and the digital part to programming the device. The membership functions implemented, triangles and trapezoids, are digitally programmed and the calculation of the functions is performed by MOS transistors on current-mode. The output is performed on voltage mode. The implemented device has 2 inputs, one output, 3 membership functions for input and 5 consequents. The number of rules is 9. The delay between input/output is 63 ns, i.e., the process speed is 15.87 MFLIPS. The chip is designed in 0.35 μm CMOS standard technology.
- Yosefi et al. (2007) in the work "Design of a new CMOS controllable mixed-signal current mode fuzzy logic controller (FLC) chip" presented an integrated circuit which implements a fuzzy controller chip with mixed-signal input (two) and digital output (one), while the internal blocks (fuzzifier, inference and defuzzifier) are realized by current mode analog circuits. The fuzzifier circuit is capable of generating three types of membership function shapes: Gaussian, trapezoidal and triangular, using a simple MOS transistor model for strong inversion. The defuzzifier block generates one crisp output (control) variable, 1 of the 16 (number of rules) truth-values transmitted from the inference block. The defuzzification strategy method is COA and it is also called "weighted average of singleton". The output implements an improved circuit based on a successive approximation technique that provides a current mode A/D converter with 7-bit resolution accuracy. The speed achieved is 16.6 MFLIPS. The chip is designed in 0.35 μm CMOS standard technology.

4.4. Conclusions and complementary readings of HW implementations for fuzzy systems

The development of fuzzy systems on a dedicated analog HW has been the focus, mainly in the 1990s, of a large number of researchers. These devices were implemented to work in either

Please cite this article as: Bosque, G., et al., Fuzzy systems, neural networks and neuro-fuzzy systems: A vision on their hardware implementation and platforms over two decades. Eng. Appl. Artif. Intel. (2014), http://dx.doi.org/10.1016/j.engappai.2014.02.008

Table 21
Speed and technology of fuzzy systems on analog, digital and mixed-dedicated circuits.

	Analog		Rules	Digital		Mixed			
	MFLIPS	Tech. μm CMOS		MFLIPS	Tech. μm CMOS	MFLIPS	Tech. μm		
							BiCMOS	Bipolar	CMOS
Current	10	1.2..1.6	Parallel	0.086..33.3	0.13..1.0	0.63..16.6	2	3	0.35..2.4
Voltage	0.01..6	2.0..2.4	Sequential	0.1..10	0.6..1.0	–	–	–	–
Transcond.	15	1.6..2.0							
Switched	0.085..16	0.8..1.2							

Table 22
Summary (rest) of the main features of fuzzy systems on analog, digital and mixed-dedicated circuits.

	Type of MFs[a]		Defuz.[a]	Current	Power
	In	Out			
Analog	Triang., trapez., S, Z-shaped, bell	Triang., singleton	COG, COA, (div. free[b]), weighted average	≈ nA (Baturone et al., 1994), 20 μA (Lemaitre et al., 1994)	550 μW (Marshall and Collins, 1997), 8.8 mW (Indue et al., 1991), 550 mW (Guo et al., 1996)
Digital	Triang., trapez., LUT (any shape), Gaussian	Triang., trapez., singleton	COG, COA, Sugeno, Yager	–	200 mW (Falchieri et al., 2002), 800 mW (Dettloff et al., 1989)
Mixed	Triang., trapez., S, Z-shaped, Gaussian	Singleton, triang., trapez.	COG, T–S, COA	–	10.49 mW (Amirkhanzadeh et al., 2005), 13.4 mW (Yosefi et al., 2007)

[a] In order to the preferences of the articles consulted.
[b] Sometimes the methods COG and COA are based on division free.

current, voltage, transconductance or switched mode. The differences between these implementations lie in speed, required power, size, integration, stability, accuracy or conformation of the membership functions, among others. All works consulted provide benefits in some of the parts that compose a fuzzy system compared to devices operating in a different mode. It should be noted that the commercial appearance of the FPAAs in the 2000s has opened a new scenario in the analog implementation of fuzzy systems, although this is yet to be consolidated.

Digital implementations on a dedicated HW have brought greater immunity to external factors (e.g. noise or fluctuation of the power supply) than analog implementations. The speed has been achieved by implementing parallel rules and optimizing the sequential rules. In systems with parallel rules, the membership functions are stored in memory. This grows when the accuracy increases so it tends to give low accuracy systems. In sequential implementations, both the membership functions and the rules are stored in memory. By increasing the number of inputs, the memory grows exponentially, and therefore techniques have been proposed to minimize this effect.

Mixed systems present their inferences on the analog area and programming, and parameters on the digital area. Architectures have been performed mainly with parallel rules.

Considering the speed and the technology, Table 21 shows the data extracted from the works consulted regarding analog, digital and mixed-dedicated circuits.

Table 22 shows the rest of the main features of dedicated circuits. At this point, let us emphasize that it is very difficult to extract data on current or power consumption of the devices consulted. Few articles clearly express this information. In this respect, only may be mentioned a few sparse data (the works are cited) as Table 22 shows.

The consolidation of FPGAs in recent years has been mainly due to their reconfiguration capability, the increase of their density and the existence of a wide commercial offer. This has made either analog, digital or mixed-dedicated designs to have been almost discontinued. The advent of programmable circuits, such as FPGAs, has opened up a wide range of options for implementing the rules, membership functions or reduction in defuzzification HW (eg. elimination of multiplication and division).

The use of processors in systems that do not require high speed is the most economical alternative regarding dedicated or configurable devices (FPGAs). The introduction of fuzzy instructions in commercial processors or coprocessors fuzzy design increases the processing speed significantly. In commercial processors with fuzzy instructions the speed increase is approximately between 1.4 and 6 compared to commercial processors without fuzzy instructions: MIPS-F vs. MIPS and μ P 6502-F vs. μ P 6502 respectively (in the works consulted). However, it is noted that commercial firms have mostly halted the manufacture of processors with fuzzy instructions. With respect to coprocessors, there is high speed processing of fuzzy rules although only one comparison with commercial processors is given in the works consulted: μ C C167 + 81C99A vs. μ C C167 and the speed factor is 3, 18 and 30 depending on the dimensionality of three fuzzy systems. Finally, some PLC manufacturers have adopted the use of fuzzy coprocessors in systems oriented to industrial control. Table 23 shows the main features of fuzzy systems on Analog and Digital programmable devices.

With regard to complementary readings, the following works deserve to be mentioned: regarding the implementations of fuzzy systems on FPGAs, Sulaiman et al. (2009) presented the review "FPGA-based fuzzy logic: design and applications—a review" up to 2008. Of the works focused on realizing a vision of the evolution of fuzzy systems, three works are highlighted: "Electronic hardware for fuzzy computation" (Basterretxea and del Campo, 2009), "Fuzzy hardware: architecture and applications" (Kandel and Langholz, 1998) and "VLSI architecture of fuzzy logic hardware implementation: a review" (Murshid et al., 2011). The first conducts a review of the evolution of fuzzy HW since its beginnings up to 2008, the second shows the evolution of fuzzy implementations on different platforms HW, and the third gives an overview, up to 2010, of the processors and controllers implemented on a VLSI device. The scope is interesting because

Please cite this article as: Bosque, G., et al., Fuzzy systems, neural networks and neuro-fuzzy systems: A vision on their hardware implementation and platforms over two decades. Eng. Appl. Artif. Intel. (2014), http://dx.doi.org/10.1016/j.engappai.2014.02.008

Table 23
Summary of the main features of fuzzy systems on analog, digital and mixed programmable circuits.

Device	Type	Type of MFs[1]		Defuz.[1]	Speed	
		In	Out		FLIPS	Time
FPAA		Triang, trapez., Z-shaped, Γ-function	–	COG	–	–
FPGA		Triang, trapez., sigmoid	Singleton, triang, trapez.	COG, COA, weighted average, MOA, T–S, etc.	77k, 1.25M..9M, 17M..50M	42 μs, 0.29 μs..6.3 μs, 35 ns..127 ns
Commercial processor	General purpose	Triang, trapez., Gaussian, Hyper ellipsoid	Singleton, triang.	Weighted average, Mamdani, T–S	–	1.95 ms, 28.5 ms
	Fuzzy instructions	Triang., trapez.	Triang., trapez., singleton	COG, COA	48, 122	–
	Coprocessor	Arbitrary (LUT), trapez.	Arbitrary	COG, FM, LM, MOM	3.8M..7.9M	–
	DSP	Triang., trapez.	Triang., trapez., singleton	COG	1.5M..8.6M	–

[1] In order to the preferences of the articles consulted.

different architectures of the blocks involved in a processor or controller fuzzy are presented. Finally, and more recently, the article "Fuzzy hardware: a retrospective and analysis" shows an overview up to 2012[20] (Hernandez-Zavala and Camacho-Nieto, 2012). The review refers to the period between 1985 and 2011.

5. HW implementations for neuronal networks

In the following section several implementations of ANNs will be presented on different technologies made in the last two decades. The focus will be on implementations, as can be seen in Section 3, in (a) *analog*, (b) *digital*, and (c) *mixed*, leaving out of this approach the so-called optical implementations, chains of pulses, etc. The classification is the same as shown in Table 3.

5.1. Analog neural implementations

Until the mid 1990s, there were a number of reasons that favored analog circuit solutions against the digital solutions. One reason was determining the high speed analog circuits compared to digital circuits. Another reason was the simple implementation of adders and multipliers with a small number of transistors. A further reason was the inherent parallelism and high connectivity of this circuit. In recent years, due to advances in integration technologies for digital circuits, the difference in speed has diminished dramatically. However, analog implementations of HW remain of interest but there is less production of works in this field.

5.1.1. Dedicated Integrated Circuits

Within these implementations it is worth mentioning the references shown in Table 24.

Six papers are highlighted to show this kind of implementation (*):

- In 1991, Mitsubishi introduced a device with unsupervised learning in CMOS technology of 1.0 μm, "A 336 – neuron 28 k-synapse self-learning neural network chip with branch-neuron-unit-architecture" (Arima et al., 1991b). This was a network of fully connected layers with feedback. It has 336 neurons and 28,224 synapses. The speed of operations is 10 GCPS (Giga Connections Per Second). This device can be expanded with several hundred interfacing devices. If it disposes of 200 devices, it can support up to 3300 neurons and 5.6 Msynapses.
- Satyanarayana et al. (1992) present, in the article "A reconfigurable VLSI neural network", an implementation of a general purpose ANN capable of reconfiguring the number of layers. This circuit has 1024 neurons and 1024 synapses organized in one, two or three layers. The circuit uses dynamically reconfigurable capacitors that can be refreshed in 74 ms CMOS technology is 0.9 μm.
- Morie and Amemiya (1994) presented, in the article "An all-analog expandable neural network LSI with on-chip backpropagation learning", a prototype device with on-chip learning implementing a backpropagation algorithm called "Contrastive backpropagation learning", working in real time. The device has 9 neurons and 81 synapses. Due to the feedback connections, the article considers it appropriate to implement recurrent ANNs. The circuit is fabricated using a 1.3 μm double-polysilicon CMOS process.
- Sun et al. (2002) in the work "Analogue implementation of a neural network controller for UPS[21] inverter applications" presented a hardware inverter (PWM) with an analog neural network mainly constructed with operational amplifiers and resistors. An EPROM is added for weight storage (6 bits for the weight and 1 bit for the sign). The ANN performs a multifeedback-loop with the inputs being the currents and voltages. The ANN is trained off-line to ensure a fast transient response and low cost. Finally it performs a comparison with a PI controller and the results show lower distortion, better control of the output voltage and smaller overshoot in the output voltage. The implementation is on several integrated circuits.
- Yamasaki and Shibata (2003) presented the article "Analog soft-pattern-matching classifier using floating-gate MOS technology", focused on image recognition of geometric forms and digits made by hand, and being able to separate overlapping shapes. Unlike other implementations, it can "tune" externally determined parameters that affect the recognition of shapes. Also, it implements a robust algorithm for image representation. The circuit is fabricated in CMOS technology of 0.6 μm. It is interesting to highlight that this work was performed on analog technology in 2003, in order to obtain a high-speed image processing in real time. Consider that the signal compression algorithms are computationally expensive and beyond the state of the art for computational applications in real-time.

[20] In this article, there are several dates of publication, December 2011 and August 2012 and there is only one reference from 2012 but this article is originally from 1987.

[21] Uninterruptible Power Supply.

Please cite this article as: Bosque, G., et al., Fuzzy systems, neural networks and neuro-fuzzy systems: A vision on their hardware implementation and platforms over two decades. Eng. Appl. Artif. Intel. (2014), http://dx.doi.org/10.1016/j.engappai.2014.02.008

Table 24
HW implementations of neural systems – analogics dedicated.

Year	Applications	Ref.	Type of ANN	In	Out	Hidden	Activat. function	No. of neurons	No. of synapses	No. of layers	Learning	On/off chip	Speed	Technology
1991	Learning, classification	Jabri and Flower (1991)	Feedforward, recurrent	2	1	2	–	–	–	3	Weight perturbation	On	–	Discrete
			Perceptron	21	5	10				3				
1991	General	Arima et al. (1991b)*	Self-organization	–	–	–	–	336	28,224		Boltzmann	On	10 GCPS	CMOS 1.0 μm
1992	General	Arima et al. (1992)	Feedback	–	–	–	–	400	40,000	–	Boltzmann	On	2 TCPS	CMOS
				–	–	–	–	10	1068	400–6–4			80 GCUPS	0.8 μm
1992	Voice recognition	Shima et al. (1992)	Multi-layer	–	–	–	Sigmoid	24	24 × 24	–	Backprop, Hebbian	On	36 GCUPS	CMOS 0.8 μm
1992	General	Satyanarayana et al. (1992)*	General	4	4	–	Sigmoid	1024	1024	1, 2 o 3	Weight perturbation	On	–	CMOS 0.9 μm
1993	Image, signal processing	Varrientos et al. (1993)	Cellular NN	–	–	–	–	–	–	–	–	–	–	CMOS -
1994	General	Morie and Amemiya (1994)*	Recurrent	2	1	–	Sigmoid	9	81	1	Contrastive backprop	On	0.1–1 MCUPS	CMOS 1.3 μm
2002	Inverter PWM	Sun et al. (2002)*	Multifeedback	5	1	3	Sigmoid, linear (out)	9	–	3	Backprop. (Off-line)	Off	–	A.Os
2003	Signal compression	Yamasaki and Shibata (2003)*	–	1	1	3	Bell, Gaussian	–	–	3	Tunable		–	CMOS 0.6 μm
2008	Surface recognition	Gatet et al. (2008)*	MLP[a]	2	1	3	Sigmoid	–	–	3	Backprop, Levenb.-Marqua.	Off	–	CMOS 0.35 μm
2012	Sigmoid function	Khodabandehloo et al. (2012)*	Resistive-type	1	1	–	Sigmoid	1	–	–	–	–	–	CMOS 90nm

[a] Multilayer perceptron.

Table 25
HW implementations of neural systems – Programmable Integrated Circuits.

Year	Applications	Ref.	Type of ANN	In	Out	Hidden	Activat. function	No. of neurons	No. of synapses	No. of layers	Learning	On/off chip	Speed	Device
1998	Survey	D-Mello and Gulak (1998)	CNN[a]	–	–	–	–	–	–	–	–	–	–	Analogix, Zetex, etc.
2004	Survey	Sekerli and Butera (2004)*	Morris–Lekar	–	–	–	–	–	–	–	–	–	–	AN221E04
2005	Creation a neuron circuit	Rocke et al. (2005)	McCulloch–Pitts	–	–	–	Comparator	–	–	–	–	–	–	AN221E04
			MLP	–	–	–	Sigmoid	–	–	–	–	–	–	
			Spicking	–	–	–	Comparator	–	–	–	–	–		
2006	Classific.	Dong et al. (2006)	Feedforward	2	1	5	Piecewise linear			3	Backprop. (Matlab)	Off	6 MCPS	AN221E04
2008	Classific.	Grzechca et al. (2008)*	RBF	5	1	27	–	–	–	3	–	–	–	AN221E04
2012	Motor control	Kamala-Kannan et al. (2012)*	–	2	1	2	–	–	–	3	–	–		FPAA

[a] Cellular Neural Network.

- Gatet et al. (2008) in the paper "Analog neural network implementation for a real-time surface classification application" presented a real-time surface recognition, with a laser beam, using a phase-shift rangefinder which drives an ANN performed by multilayer perceptron (2–3–1). The rangefinder provides two signals to the ANN, the filtered photoelectric signal amplitude and the signal proportional to the distance. The ANN discriminates four types of surfaces. The implementation of each neuron (3) of the hidden layer is on an ASIC of 44-pins in CMOS technology of 0.35 μm. The neuron of the output layer is performed on A.Os.
- Khodabandehloo et al. (2012) in the article "Analog implementation of a novel resistive-type sigmoidal neuron" presented a resistive-type neuron with low sensitivity to circuit variations to generate a sigmoid-like function for analog implementations of neural networks. The neuron is based on the transistor characteristics in both triode and saturation regions working in transconductance mode. Taylor series are used to approximate the sigmoid function and the obtained error is less than 1.3%. The design includes a study of the behavior taking into account the temperature changes between −55 °C and 125 °C. The voltage variations from the voltage value at 27 °C are

Please cite this article as: Bosque, G., et al., Fuzzy systems, neural networks and neuro-fuzzy systems: A vision on their hardware implementation and platforms over two decades. Eng. Appl. Artif. Intel. (2014), http://dx.doi.org/10.1016/j.engappai.2014.02.008

Table 26
HW implementations of neural systems – digital dedicated.

Year	Applications	Ref.	Type of ANN	In	Out	Hidden	Activat. function	No. of neurons	No. of synapses	No. of layers	Learning	On/ off chip	Speed	Technology
1990	General	Hammerstrom (1990)*	Any	203 8/16 bit	21	64	Sigmoid	–	–	3	Backpropag	On	1.6 TCPS	CMOS –
1993	Pattern recognition	Marchesi et al. (1993)*	MLP	64 6 bit	8 8 bit	64	Any (LUT)	–	–	3	Backpropag.	Off	–	VLSI –
2000	Survey: Matrix-vector multiplier	Szabo et al. (2000)	MLP CNN[a]	*N* 5,10,20,40 – 8/12/16 bit	*M* 5,10,20,40 – –	– – –	–	–	– –	–	Pre-trained	Off	–	ASIC – FPGA (generic)
2001	Image acquisition	Gregoretti et al. (2001)*	MLP (internal weights) Kohonen (internal weights) 8 bit	*n* 8	– –	– –	– –	$32 \times n$ 1×30	– –	– –	Adaptive Neighborhood	On	130 MCPS 60 MCUPS 110 MCPS 60 MCUPS	CMOS 0.8 μm
2003	Automotive control/ diagnosis	Hendry et al. (2003)*	Self-organizing 8 bit	16 16	– –	– –	– –	256 256 1024 4096	– –	– –	LVQ	Off	1.3 GCPS 11 GCPS 44 GCPS 176 GCPS	CMOS 0.68 μm 0.18 μm
2013	Optical template	Zamanlooy and Mirhassani (2014)*	3-layers	4 8 bit	2 5 bit	3	–	–	–	3	–	–	–	CMOS 0.18 μm

[a] Cellular neural network.

0.7% and 1.4% for −55 °C and 125 °C, respectively and for the saturated neuron, the values are changing to 0.5% and 0.2% for −55 °C and 125 °C, respectively. The layout of the neuron has been performed in 90-nm CMOS technology, with a power supply voltage of 1.2 V.

5.1.2. *Programmable Integrated Circuits*

Within these implementations the following references will be mentioned in Table 25, which will also include references to FPAA implementations.

The following references will be mentioned (*):

- Sekerli and Butera (2004) in the work "An implementation of a simple neuron model in field programmable analog arrays" presented the implementation of a minimal neuron model, the Morris–Lecar model (Morris and Lecar, 1981), on a FPAA. The differential equations of the model are solved using inverters, adders, multipliers and integrators that are available via the FPAA design software. The article highlights that such a model can be run accurately in real-time or many orders of magnitude faster than real-time. The neuron is implemented on three AN221E04 of Anadigm Inc®.
- Grzechca et al. (2008) presented the work "Diagnosis of specification parametric faults in the FPAA—the RBF neural network approach". The aim of the work is to diagnose a parametric fault in the Configurable Analog Blocks (CABs) of the FPAA, based on the output time domain response. A neural network with radial basis functions (RBFs) has been used for feature classification. The RBF-NN has one input layer, one hidden layer, and one output layer (5–27–1). The design under test is a low-pass filter and the output layer of the RBF-NN includes a linear function that indicates if the specifications meet the requirements. The device under test is the AN221E04 of Anadigm Inc®.
- Kamala-Kannan et al. (2012) in the paper "Sensorless control of SR drive using ANN and FPAA for automotive applications" presented a control of Switched Reluctance Motor (SRM) estimating the rotor position using the phase current and voltage. The ANN consists of three layers. The input layer has two inputs which are scaled current and flux linkage. The rotor position (θ) is the single output available in the output layer. The article performs a comparison between a conventional

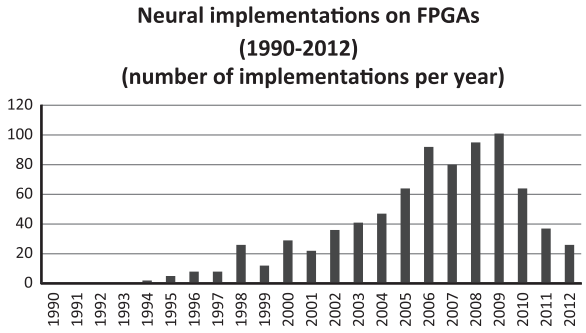

Fig. 2. Neural implementations on FPGAs.

machine with and without a sensor, and an SMC sensorless machine, measuring the speed response and applying different loads (20 N m and 10 N m). The speed error during load disturbance is much less in the conventional machine sensorless operation as compared to the SMC sensorless machine operation.

5.2. Digital neural implementations

Digital implementations have aroused great interest during the last two decades because of the high integration achieved and the increase of the execution speed, resulting in a large production of articles. These implementations have been done on full-custom VLSI circuits, ASICs and FPGAs. As in Section 4.2.2, further development has been focused on the FPGAs because of their ability to reconfigure and the low "time to market". With regard to research in general purpose neural processors on dedicated devices, it appears that it is a line of work that is being discontinued.

5.2.1. Dedicated Integrated Circuits

Here are some relevant approaches in this kind of implementation shown in Table 26.

Of all these references, the following will be mentioned (*):

- Hammerstrom (1990) presents a new chip oriented to neurocomputing in the work "A VLSI architecture for high-performance, low-cost, on-chip learning". Hammerstrom (1990) considers that this chip is *a general purpose, "microprocessor" of neurocomputing*. For this reason this work will be discussed in Section 5.2.3.
- Marchesi et al. (1993) in the article "Fast neural networks without multipliers" referred to an implementation of a multilayer perceptron network in which the weights are a power of 2 and therefore does not require multipliers but shift registers. They develop a backpropagation learning algorithm for the proposed network which is performed off-line. The advantage of this method is that it requires a smaller area of silicon and computing times are reduced. Design is applied to pattern recognition. Marchesi et al. (1993) note that this algorithm is suitable to be implemented on a generic VLSI digital device.
- From the past decade, in 2001, let us cite the work "A high speed VLSI architecture for handwriting recognition", where Gregoretti et al. (2001) presented a complex system for simultaneous image acquisition, processing, neural network emulation, and a straightforward interface with a hosting PC. The device is named HACRE and is a massively parallel VLSI image processor with the instruction set dedicated to execute both traditional image processing (such as filtering and image enhancement) and mathematical morphology (such as opening, closing, skeletonization) and several types of neuro–fuzzy networks (such as perceptrons, self-organizing maps, cellular networks, fuzzy systems). The device needs 1 external RAM, plus some glue logic for microprocessor interfacing. Larger-size configurations are obtained by cascading chips, as many external RAMs, some additional global interconnection logic and a host interface (PCI). This work develops a board with 4 HACRE-devices and all the aforementioned logic. The neural subsystem carries out a pre-recognition of the individual characters, based on an integrated segmentation and recognition technique. The maximum frequency of the device is 50 MHz and is obtained from a PCI Bus by means of a PLL. The chip has been implemented in a 0.8 μm CMOS technology.
- Zamanlooy and Mirhassani (2014) "Efficient VLSI implementation of neural networks with hyperbolic tangent activation function" proposes a new hybrid scheme to implement the activation function, which is based on a linear approximation in combination with bit-level mapping. They demonstrated that the proposed structure requires less output bits for the same maximum allowable error compared to the previously developed schemes. The approximation scheme was used for implementing a 4–3–2 network for an optical template matching application. The hardware implementation was coded using Verilog hardware description language and synthesized by Synopsis Design Compiler using 0.18 μm CMOS technology.

Some applications that require a smaller silicon area, lower power and higher clock speeds are implemented on ASICs. An example of this type of realizations is as follows:

- The work of Hendry et al. (2003), "IP core implementation of a self-organizing neural network", oriented to self-organizing neural networks, implements an array of 256 neurons. The IP soft-core designed is part of a SoC, which also implements a RISC processor. One area of interest mentioned in this work is in the automotive market, applying to the nonlinear control (camless engines) and diagnostic (virtual sensor technology) of the car. The target clock frequency is 200 MHz and for 16 element reference vectors (each element consisting of eight bits) this gives a classification time of 370 ns (2.7 Million classifications/s). The speed of operations is 11 GCPS. This design has been implemented in a 0.18 μm CMOS technology.

5.2.2. Programmable Integrated Circuits

Applying the same reasoning as in Section 4.2.2 and consulting the "Web of Knowledge – Web of Science", the number of works on FPGAs is around 830 between 1990 and 2012. Fig. 2 shows this production.

Here are some relevant approaches in this kind of implementation shown in Tables 27–31.

Of all these references, the following will be mentioned (*):

- Hikawa (1995) in the article "Implementation of simplified multilayer neural networks with on-chip learning" proposes a new architecture of a multilayer ANN without multiplication and learning on the device. The implementation is done on a FPGA but the article does not cite the reference. The multiplications are performed by shift registers, as in the previous work, and ANDs. In the calculation process of the network (forward-path) the neuron activation functions are performed with three states (0, 1/2, 1) operating over the displacement and the logical operator AND. The sum of the weighted inputs of the neuron is performed by a serial adder. As for the learning algorithm, it is based on the backpropagation algorithm operating in pulse mode to remove the multiplication; the number

Please cite this article as: Bosque, G., et al., Fuzzy systems, neural networks and neuro-fuzzy systems: A vision on their hardware implementation and platforms over two decades. Eng. Appl. Artif. Intel. (2014), http://dx.doi.org/10.1016/j.engappai.2014.02.008

Table 27
HW implementations of neural systems – Digital Programmable Integrated Circuits (1).

Year	Applications	Ref.	Type of ANN	In	Out	Hidden	Activat. function	No. of neurons	No. of synapses	No. of layers	Learning	On/ off chip	Speed	Device
1994	Run-time reconfiguration	Eldredge and Hutchings (1994)	Feed-forward	*n*	*m*	*h*	–	–	–	*l*	Backprop.	On	568 KCUPS	XC3090
				8 bit		–			–					
1995	Pattern recognition	Hikawa (1995)*	Multilayer	2	2	3	Sigmoid	7	17	3	Backprop. (pulse mode)	On	14.6 MCPS	FPGAs (generic)
				7×5	7	20..70		62..112	840..2940				10.9 MCUPS	
				1 bit		–			12 bit					
1996	Digital signal-image processing	Isshiki and Dai (1996)	3-layer	12	4	14	–	–	30	3	–	–	–	XC3164A
				8	4 8 bit	8			20					
1998	N-Queen problem	Abramson et al. (1998)*	Hopfield	–	–	–	–	–	–	–	–	–	–	XC4010
				–	–	–								XC4020
1998	Non-linear funct. approx.	Blake et al. (1998)	3-layer (feed-forward)	3	8	1	Sigmoid	12	25	3	–	–	–	XC4020
				9 bit	FP				9 bit FP					
1998	Classifying points inside/outside region	Hikawa and Sato (1998)	Multilayer	2	1	2	Threshold	5	6	3	Backprop. (simplified)	On	0.857 MCUPS	XC4010
				–	8 bit				16 bit FxP					
1998	Air pollution	Taright and Hubin (1998)	MLP	–	–	–	Sigmoid	–	–	3	Backprop.	–	–	FPGA Xilinx (generic)
				–	–	–			–					
1999	ECG[a] classifier	Izeboudjen et al. (1999)	3-layer	5	2	3	Sigmoid	10	21	3	Backprop.	Off	–	XC4020E
					8 bit				8 bit					
1999	Generic	Sato and Hikawa (1999)	Multilayer	2	2	3	Threshold	7	10	3	Backprop. (PWM)	On	–	XC4010P
					8 bit				16 bit FxP[b]					
1999	Reconfigurable computing	Zhu et al. (1999)	Multilayer	–	–	–	–	Max. 8	–	–	–	On	–	XC6216
					8 bit			Max. 32	8 bit		(on-line)			XC6264

[a] Electrocardiogram.
[b] Fixed point.

of pulses is proportional to the error obtained and acts on up–down counters containing the weights. When the device operates at 25 MHz, the performance achieved is of 14.6 MCPS and 10.9 MCUPS.

- Abramson et al. (1998) in the paper "FPGA based implementation of a Hopfield neural network for solving constraint satisfaction problems" described and solved the N-Queen problem using a Hopfield neural network (used to solve difficult optimization problems) to demonstrate and solve the potential of a custom computer based on FPGA technology. The paper highlights that in this architecture, first, the weights are small and can be represented using small integers. They also reduce the carry propagation delays. This means that the hardware responsible for the accumulation can be optimized for small integer values. Second, the vector product becomes a set of conditional additions without the need to perform any multiplication operations. Third, the interconnection between neurons is fixed and dictated by the nature of the constraints in the problem. The hardware platform is an Aptix AP4 that contains up to 16 Xilinx XC4010 devices. The synthesis is performed with Galileo Exemplar Logic's and V-System tools and the algorithm has been described in VHDL. The paper establishes a comparison with a Sun UltraSparc workstation running at 140 MHz and yielding 2–3 orders of magnitude speedup. Finally, the HW was optimized and the 8-Queen problem was tested on a XC4020 device.
- Omondi and Rajapakse (2002) published the work, "Neural networks in FPGAs" in which an approach is made to parallelism and arithmetic, the HW or SW implementation, and which finally examines a case of Independent Component Analysis (ICA) (Comon, 1994), implementing an independent component neural network (ICNN) over the Xilinx® XCV812C. The paper proposes three different nonlinear functions but the implementations are focused on two. There are two types of HW implementations, the first on Lookup-table (LUT) – synaptic inputs – and the second on combinational logic (CL). In the

Table 28
HW implementations of neural systems – Digital Programmable Integrated Circuits (2).

Year	Applications	Ref.	Type of ANN	In	Out	Hidden	Activat. function	No. of neurons	No. of synapses	No. of layers	Learning	On/ off chip	Speed	Device
2000	Systolic array	Gadea et al. (2000)	MLP	2	2	6–3	–	–	–	4	Backprop. (on-line)	On	81 MCPS	XCV400
						8 bit			16 bit				81 MCUPS	
2000	Neuron probabilistic deactivation	Restrepo et al. (2000)	FAST[a]	2	–	2/6/10/ 14	–	4/8/12/ 16	–	2	Unsupervised (correlation)	On	0.65 ms/ ../3.04 ms	XC4013E
					8 bit				8 bit					
2001	Generic (XOR, classification)	Hikawa (2001)	Multilayer (pulse-mode)	–	1	2	Nonlinear adder (voting circuit)	3	9	2	Backprop. (frequency modulation)	On	211 KCPS	FPGAs (generic)
					1 8 bit	4		5	14 9 bit				843 KCUPS	
2001	Glass problem	Girau (2001)	FPNA[b]	16	6 16 bit	8 FxP	Sigmoid	30	– 16 bit FxP	–	Backprop.	On	5 MCUPS .. ..1 GCUPS	XC40250
2001	Word recognition	Kim and Lee (2001)	RBF	1024	1024	1024	Gaussian	1024	–	1	Means and variances	Off	15 ms (recognition)	Flex 10K
					8/16 bit				–					
2002	ICA[c]	Omondi and Rajapakse (2002)*	ICNN[d]	*n*	*n*	–	Sigmoid	–	–	–	Gradient (probability density)	On	–	XCV812C
				–	–	–			–					
2002	Alphabet recognition	Yun et al. (2002)	MLP	256	5	32	–	293	512	3	Backprop.	Off	–	Virtex2 6000
				1 bit	5 bit	24 bit FxP			24 bit FxP					
2002	Detecting human hands	Krips et al. (2002)	Multilayer	3	1	3	Hyperbolic tangent (LUT)	4	12	3	Resilient propagation	Off	10 ms	XCV100
				8 bit	1 bit	–			–					

[a] Flexible adaptable-size topology.
[b] Field programmable neural array.
[c] Independent component analysis.
[d] Independent component neural networks.

first case, the values of the nonlinear functions proposed in the study are stored on RAM and, in the second case, the weights are stored on RAM and the functions are performed with functional blocks.

- Kim et al. (2003) "FPGA implementation of ICA algorithm for blind signal separation and adaptive noise canceling" applied to speech recognition in noise environments and echo. Algorithms of signal separation (blind signal separation) and algorithms of adaptive noise cancellation (adaptive noise canceling) were implemented. The device used was Altera® EP20K600EBC652-1.
- Ide and Saito (2006) presented the article "FPGA implementations of neocognitrons" applied to character recognition and biometric measures. It describes the implementation of a reconfigurable ANN on a parallel computer architecture based on FPGAs, called REOMP (Reconfigurable Orthogonal Multiprocessor Memory). On this architecture Neocognitrons are implemented. A Neocognitron ANN model is a feed-forward topology proposed by Fukushima (1982), based on the model of Hubel and Wiesel (1968) concerning the research of the vision from a biological point of view. The parallel architecture is based on a host processor that works as a Control Unit (CU) and 1–64 nodes consisting of Reconfigurable Processors (RPs) connected to an orthogonal structure of memory modules. The development was performed with the Altera® Quartus II tool. There is no mention of the device.
- Bastos et al. (2006) presented the work "FPGA implementation of neural network-based controllers for power electronics applications", where an ANN was implemented to control a buck converter (step-down DC/DC) based on the behavior of a SACT controller (synergetic approach to control theory). The ANN chosen had the structure 4–4–1 and was trained to have the high-performance characteristics of the SACT controller. The phase of simulation was performed using the System Generator of Xilinx® and the design of the ANN was performed using the Neural Network tool of Matlab®. The device used was a Virtex-II Pro of Xilinx®.
- Ferreira et al. (2007) presented the paper "A high bit resolution FPGA implementation of a FNN with a new algorithm for the activation function", which mainly presents the calculus in the floating point and where the activation function is performed with piece-wise linear approximation. The training of the Feedforward Neural Network is off-line and is carried out in Matlab on a PC and the weights are sent from the PC to FPGA. The device used is the FPGA Cyclone EP1C20F324C7 from Altera®.
- Hu et al. (2008) in the work "Key issues of FPGA implementation of neural networks" give an overview of the different parts and methods involved in the design of the ANNs such as data representation, inner-products computation, implementation of activation function, storage and update of weights, nature of learning algorithm and design constraints.
- Orlowska-Kowalska and Kaminski (2009) in the article "Application of MLP and RBF neural networks in the control structure of the drive system with elastic joint" presented a neural

Table 29
HW implementations of neural systems – Digital Programmable Integrated Circuits (3).

Year	Applications	Ref.	Type of ANN	In	Out	Hidden	Activat. function	No. of neurons	No. of synapses	No. of layers	Learning	On/ off chip	Speed	Device
2003	BSS and ANC[a] (ICA-based)	Kim et al. (2003)*	RBF	2+4	2	–	–	–	–	–	Stochastic	On	–	EPK20K600
					12 bit				–					
2003	Weight clustering optimization	Noory and Groza (2003)	Any	9	1	–	Distributed arithmetic	–	–	–	DASP	On	–	Flex 10K
				8 bit		–	Serial/parallel		8 bit					
2003	One-dimensional systolic array	Hung and Wang (2003)	Systolic architecture	–	–	–	Ramp	–	–	–	Iterations	On	–	Virtex 2000E
				–	–	–								
2003	Speech recognition	Ortigosa et al. (2003)	MLP extended	220	10	24	Sigmoid	254	–	3	Backprop.	On	12.5 μs	Virtex 2000E
					8 bit				8 bit					
2005	Internal data formats	Prieto et al. (2005)	3-layers	3	12	6	Logarithmic or tangential sigmoid, linear (out)	–	–	3	Backprop.	On	–	FPGAs (generic)
				16 bit	S/M[b]	–			16 bit S/M					
2006	Charact. recognit., biometric measures	Ide and Saito (2006)*	Neocognitron	Complex matrix	distribution cells	in	–	–	–	–	Algorithm (own)	On	–	REOMP[c] (1..64 reconf. processors)
				–	–	–			–					
2006	Buck converter (SACT[d])	Bastos et al. (2006)*	3-layers	4	1	4	Sigmoid	9	–	3	Backprop.	On	–	Virtex II Pro
					16 bit				16 bit					
2006	Floating point arithmetic	Sahin et al. (2006)	3-layers	2	1	3	Sigmoid	6	9	3	–	–	–	Spartan II
					32 bits FP				32 bits FP					2S200
2006	Neuron model	Huitzil (2006)	Neuron	Each neuron have 8 input max.			Sigmoid	max:150	–	–	–	–	87 MCPS	XC2V2000-5
				–	–	–			9 bit FxP					

[a] Blind signal separator and adaptive noise canceling.
[b] Sign-magnitude.
[c] Reconfigurable orthogonal memory multiprocessor.
[d] Synergetic approach to control theory.

estimator of the torsional torque and the load-machine speed of the drive system with an elastic joint, using multi-layer perceptron (MLP) and radial basis function (RBF) NN in the open-loop and closed-loop control structure. The paper demonstrates that RBF-based neural estimators can give better accuracy of the load speed and torsional torque estimation and better results in the successive vibration damping. The simulation results with the Neural Network Toolbox of Matlab/Simulink software were obtained with MLP estimators, while the RBF NN were implemented in a Netlab package. The article proposes the implementation on a FPGA and indicate that the RBF estimator requires many more hardware resources than a simple MLP network.

- Shoushan et al. (2010) in the work "A single layer architecture to FPGA implementation of BP artificial neural network" presented the design of a back propagation ANN and constructed an application for classifying the defects of the carbon fiber reinforced plastic. A one dimensional systolic array of the finite impulse response (FIR) filter for the backpropagation algorithm is introduced. All calculated parameters are stored on a RAM and the implementation of excitation function (sigmoid) is performed on Look-up tables. The design is implemented on two FPGAs and a comparison between the resources used is performed. The FPGAs are Cyclone and Cyclone II of Altera® using the Quartus II tool.
- Mekki et al. (2010) in the article "FPGA-based implementation of a real time photovoltaic module simulator" proposed a multilayer perceptron (MLP) for simulation and implementation of a real time PV-module on FPGA. The evaluation of the performance of a PV-module is based only on meteorological

Table 30
HW implementations of neural systems – Digital Programmable Integrated Circuits (4).

Year	Applications	Ref.	Type of ANN	In	Out	Hidden	Activat. function	No. of neurons	No. of synapses	No. of layers	Learning	On/off chip	Speed	Device
2007	Mobile robotic vehicles	Pearson et al. (2007)	Speaking NN (LIF neurons (a)) (α, β-neuroprocessor: 10 NPE)	α: 384	384	–	–	112/NPE	912/NPE	–	–	–	–	XC2V1000F
				β: 2256	2256			64/NPE	1024/NPE					
					16 bit				16 bit					
2007	Classification	Patel et al. (2007)	MLP (bit-stream)	2	1	2	Stochastic side effect	3	6	2	Backprop. (bit-stream)	On/Off	–	APEX200
				4	3	4		7	28	2				
				8 bit	–	–			–					
2007	PWL (activation function)	Ferreira et al. (2007)*	Feedforward NN	1–16	1–16	1–16	Hyperbolic tangent (PWL), linear (Out)	–	–	3–17	–	Off	–	EP1C20F324C7
					32 bit FP				–		(Matlab)			
2007	Space vector modulator	Muthuramalingam et al. (2007)	Multilayer	1	3	6–6–6	Log & tan sigmoid	21	96	5	Backprop.	Off	–	XCV50HQ
				8/10/	12/ 16/32	bit			8..32 bit					
2007	Classification, image processing	Torres-Huitzil et al. (2007)	3-layer	2	3	15	Sigmoid	20	75	3	–	Off	130 MHz	XCV2000E
						10 bit FxP			–			(Matlab)	..	
			Diabolo	16	16	5	Linear	37	160	3			90 MHz	
						9/11/12 bit FxP			–					
2008	General issues	Hu et al. (2008)*	ANNs	–	–	–	Sigmoid, ramp, etc.	–	–	–	–	–	–	FPGAs (generic)
				–	–	–	8 bit		16 bit					
2008	Lossy compression digit. images	Kurdthongmee (2008)	Kohonen-SOM	3	1	–	–	–	256		Competitive unsupervised	On	24.2 MHz	XC2VP1000
					8+1 bit				–					

[a] Leaky integrate and fire.

Table 31
HW implementations of neural systems – Digital Programmable Integrated Circuits (5).

Year	Applications	Ref.	Type of ANN	In	Out	Hidden	Activat. function	No. of neurons	No. of synapses	No. of layers	Learning	On/ off chip	Speed	Device
2008	Prediction, identification	Lin and Tsai (2008)	Wavelet	2	1	4–4	Gaussian first derivative (LUT)	11	28	4	Particle swarm optimization	On	–	XC2V8000
				1	1	4–4		10	24	4				
2009	Two-mass drive system (torque-speed)	Orlowska-Kowalska and Kaminski (2009)*	MLP	2	2	5–6	Sigmoid Gauss linear (Out)	15	–	4	Levenb.-Marquar.	–	–	FPGAs (generic)
			RBF	2	2	40/70		44/74	–	3				
				–	–	–			–					
2011	Speed regulation	Hasanien (2011)*	Adaptive	3	1	4	Tansigmoid	8	–	3	Widrow–Hoff (on-line)	On	–	XC3S500E
				–	–	–			–					
2012	Electric network	Cárdenas et al. (2012)*	ADALINE	2	1	–	–	–	–	–	Widrow–Hoff (on-line)	On	–	XC2VP30
				12 bit	–	–			–					
2012	Pattern classification	Papadonikolakis and Bouganis (2012)		Max: 300		–	–	–	–	–	–	–	–	EP3SE260 Stratix III
				–	–	–			–					
2012	Handwritten alpha-digits recognition	Soleimani et al. (2012)	Spiking ANN	20×16	A..Z	–	–	–	–	–	STDP	–	–	XC2VP30 Virtex-II Pro
					20 bit	FxP			20 bit FxP					
2013	Image deblurring	Saadi and Bettayeb (2013)*	MLP	3	3	2	–	–	–	–	ABC[a]	On	–	Virtex5-LXT
					18 bit	FxP			18 bit FxP					
2013	Training RBFNN	Fan and Hwang (2013)*	RBF	4×4	1	8	Gaussian	–	–	–	LMS[b]	On	–	Cyclone III
				8×8	1	8								
				–	–	–			–					

[a] Artificial bees colony.
[b] Least mean square.

data such as air temperature and total solar radiation, and can be used for prediction of the PV electrical energy output under actual climatic conditions. The model has been developed and simulated under Matlab/Simulink and the optimal configuration has been written in VHDL on ModelSim and then implemented on a FPGA. The device is Virtex II (XC2V1000) from Xilinx.

- Hasanien (2011) in the article "FPGA implementation of adaptive ANN controller for speed regulation of permanent magnet stepper motor drives" studied the dynamic response of a PMSM under full load torque and under load disturbance. The study was performed simulated and experimentally. As other papers reviewed, the environment Matlab/Simulink is used for simulation. The ANN adaptation is the Widrow–Hoff algorithm. The input vector of the ANN controller consists of the reference ω_{ref}, the actual motor speed $\omega(t)$, and the previous output signal of the ANN controller $u(t-1)$. The previous output of the ANN is added to the inputs as a stabilizing signal. The ANN structure is $3 \times 4 \times 1$. The effectiveness of the proposed ANN controller in comparison with a PI controller; the speed response is very smooth and the ripple is reduced to a very small value in compared with the conventional PI controller. The device chosen is a FPGA XC3S500E Spartan-3E from Xilinx on the platform Spartan-3E Starter Kit board.
- Cárdenas et al. (2012) in the article "Development of a FPGA based real-time power analysis and control for distributed generation interface" presented the development and the experimental evaluation of a power control system for a single-phase grid-connected which has several energy sources connected. An ADALINE network is used for control and synchronization of the power of the electric network. Harmonics are calculated from current and voltage signals in real time and the number of the harmonics may vary (4/8/16/32/50). The number of inputs is 2 (voltage and current) and the number of outputs is 1. The learning is preformed by means of Widrow–Hoff algorithm. The article performs a comparison between the Adaline network and the FFT implemented off-line on Matlab. The results are similar. The analysis and control are implemented on a FPGA XC2VP30 from Xilinx.
- Soleimani et al. (2012) in the article "Biologically inspired spiking neurons: piecewise linear models and digital implementation" proposed PWL models with a fewer number of multipliers for implementations of spiking neural networks on FPGAs. The models replaced the operation "square" by a comparison or "absolute value"; this means that in digital implementations the multipliers are replaced by comparators which implies that they can implement a large number of neurons. The network is trained with a supervised and unsupervised learning algorithm. The results show 91.7% accuracy in

Please cite this article as: Bosque, G., et al., Fuzzy systems, neural networks and neuro-fuzzy systems: A vision on their hardware implementation and platforms over two decades. Eng. Appl. Artif. Intel. (2014), http://dx.doi.org/10.1016/j.engappai.2014.02.008

the recognitions and the implemented PWL models are significantly faster than the Izhikevich (2003) model (approximately 9.2 times on a Virtex II Pro) with a simple combinational multiplier. The device chosen is a FPGA Virtex-II Pro XC2VP30 from Xilinx.

- Saadi and Bettayeb (2013) in the article "ABC optimized neural network model for image deblurring with its FPGA implementation" try to improve radiological images degraded during acquisition and processing. An autoregressive moving average (ARMA) model is identified using an ANN. The network training is improved using a novel swarm optimization algorithm called Artificial Bees Colony (ABC), inspired from the foraging intelligence of honey bees. They developed a 3-input 3-output multilayer perceptron (MLP) with two hidden layers. The optimized ARMA–ANN model was simulated under ModelSim software, and the best configuration was implemented on a Virtex5-LXT FPGA chip of Xilinx using VHDL. The timing results of image deblurring executed on FPGA (fixed frequency about 62.5 MHz) and on Desktop PC indicate that execution time is largely better on FPGA.
- Fan and Hwang (2013) in the work "Efficient VLSI architecture for training radial basis function networks" presented a novel architecture for the training of radial basis function (RBF) networks. The architecture includes the circuits for fuzzy C-means that perform the training of centers in the hidden layer, and the circuit for the recursive Least Mean Square algorithm which trains the connecting weights in the output layer. The architecture is used as a hardware accelerator in a system on programmable chip (SOPC) for real-time training and classification. The RBF network was implemented using low cost FPGA devices of the Altera's Cyclone III family. Experimental results show that the proposed architecture is an effective alternative for on-chip learning applications requiring low area costs, high classification success rate, and high speed computation.

5.2.3. Commercial processors

In this section, the general purpose processors with the classical repertoire instructions as well as general purpose processors which have introduced fuzzy processing instructions or which incorporate a fuzzy coprocessor and platforms that incorporate general purpose processors will be considered. Also, the DSP (Digital Signal Processor) devices will be examined.

General purpose processors: Here are some relevant approaches in this kind of implementations shown in Table 32.

Of all these references, the following will be mentioned (*):

- Hammerstrom (1990) proposed a multiprocessor architecture SIMD (Single Instruction Multiple Data) to develop applications of artificial neural networks called "Adaptive solutions X1 chip" in the work "A VLSI architecture for high-performance, low-cost, on-chip learning", later known as "CNAPS". One of the design ideas of this architecture is that it is a general purpose application for the development of ANNs. As the article emphasizes "*The goal is an architecture that could be considered a general-purpose microprocessor for neurocomputing*". The device contains 64 processors, with an instruction bus of 32 bits, a global bus of 8-bit outputs, a global bus of 8-bit input and an inter-processor bus of 4-bit. It also contains a 4 kbyte SRAM memory for weights, 32 registers of 16 bits, a 8×16-bit multiplier, an adder of 32 bits, etc., for each processor. It can implement a wide variety of learning algorithms and signal processing algorithms. At 25 MHz, the performance data are (a) 1.6 GCPS in internal multiplications (in non-learning mode), (b) 12.8 GCPS for 1-bit weights and (c) 260 MCUPS (connections considering up-date learning) with back-propagation learning mode. It requires an external instructions sequencer that can work with multiple devices simultaneously.
- Leonhard et al. (1995) in the paper "ArMenX: a flexible platform for signal and image processing" presented a flexible platform for parallel processing composed of replicated nodes. Each node consists of a Transputer (Inmos T805), a FPGA (Xilinx XC4010), and a DSP (Motorola 56002). This structure allows for a high level neural network programming environment. The design has been implemented in three layers: (a) upper layer – message passing MIMD computer (T805) + RAM, (b) middle layer – a network of FPGAs (XC4010), and (c) bottom layer – DSP (M56002) + SRAM chips tightly coupled to the FPGAs. The paper describes several examples. One of these is an ANN with multilayer perceptron for handwritten digit recognition. The structure for the multilayer perceptron is 256 input neurons, 40 hidden neurons, and 10 output neurons. In this example, the transputer loads the input vector onto the RAM of the DSP, then the transputer interrupts its DSP and the DSP computes the first layer of the ANN. When this computation is finished, each DSP interrupts its transputer. The input vector for the next layer is the output vector computed during this step. Afterwards, each FPGA initiates a DMA (Direct Memory Access) session, taking the partial output from its DSP RAM, and sends it to the other nodes. The learning process consists of minimizing an energy function by updating the synaptic weight values. The error vector (difference between the overall output vector and a target vector) is retropropagated through layers, from output to input.

Commercial DSPs: The implementations on DSPs are also interesting. Some relevant approaches of this kind are shown in Table 33.

In this regard the following works should be mentioned (*):

- Card et al. (1998) in the article "Competitive learning algorithms and neurocomputer architecture" presented a reconfigurable parallel neurocomputer architecture applied to digital implementations of a competitive learning and self-organizing feature maps (SOFMs). The reconfigurable machine described employs DSP computations in a bus-based architecture with FPGAs primarily for reconfigurability. The architecture is based on nodes named Processing Elements (PEs) which consist of one DSP (primary processing unit) and three FPGAs (message decoder, preprocessor, and postprocessor). The estimation of the upper limit of the PEs is in the range of 16–64. Table 34 shows the performance of the DSP–FPGA machine compared with other representative workstations. The DSP device is a Motorola 96002 and the FPGA is a Xilinx 3090.
- Boquete et al. (2002) in the article "Hardware implementation of a new neurocontrol wheelchair-guidance system" presented a neurocontroller based on a new model of radial basis function (RBF) recurrent neural network, where the radial basis function is fed back by means of finite impulse response (FIR) filters. A Kalman filter identifies the wheelchair on-line and is used to propagate the control error towards the neurocontroller coefficients. The wheelchair is a two-input and a two-output system. Inputs: angle speed of the right-hand and left-hand wheel; outputs: linear and angle speeds. The control is divided into two levels: low level control (PID: drivers of the wheelchair's motors) and high level control (neural controller). The neurocontroller generates the signals for controlling the plant output and the PID controller ensures that the speed of each wheel corresponds as closely as possible to that sent by the neural controller. The low level control is performed on a FPGA from

Please cite this article as: Bosque, G., et al., Fuzzy systems, neural networks and neuro-fuzzy systems: A vision on their hardware implementation and platforms over two decades. Eng. Appl. Artif. Intel. (2014), http://dx.doi.org/10.1016/j.engappai.2014.02.008

Table 32
HW implementations of neural systems – digital general purpose processors.

Year	Applications	Ref.	Type of ANN	In	Out	Hidden	Activat. function	No. of neurons	No. of synapses	No. of layers	Learning	On/off chip	Speed	Device type
1990	General	Hammerstrom (1990)*	Several	203	21	64	Sigmoid	–	–	3	No	–	1.6 GCPS 12.8 GCPS	μ P X1
				8/16 bit	8 bit				1/8/16 bit S/M		Backprop., etc.	On	260 MCUPS	
1995	Image	Leonhard et al. (1995)*	MLP, Hopfield	256	10	40	–	–	–	3	Backprop.	On	–	Transputer–FPGA–DSP
				–	–	–			–					
1997	Cache management	Hammami (1997)	Backprop,	–	–	–	–	–	–	–	Backprop., LVQ	On	–	R4000
					64 bit	FP			64 bit FP		Recurrent			
2001	General	Binfet and Wilamowski (2001)	Cascade (fully connected),	2	1	–	Sigmoid, elliot, tang. hyp., etc.	4	–	2	Backprop.	On	–	μ C 68HC711E9
								6	–	2				
			MLP			6		7	–	2				
					8 bit				8 bit					

Table 33
HW implementations of neural systems – Digital Signal Processor.

Year	Applications	Ref.	Type of ANN	In	Out	Hidden	Activat. function	No. of neurons	No. of synapses	No. of layers	Learning	On/off chip	Speed	Device
1995	Birdsong recognition	McIlraith and Card (1995)	Back-prop.	10/19	6	12	–	28/37	–	3	Backprop.	Off	–	DSP –
					8 bit				8 bit					
1998	General	Card et al. (1998)*	SOFM competitive learning	–	–	–	Sigmoid	–	–	–	Comp. learning	On	2 MCUPS	M96002(DSP)
				–	–	–			–					X3090(FPGA)
2002	Wheelchair	Boquete et al. (2002)*	RBF	2	2	–	FIR filter	–	–	–	Gradient descent	On	–	TMS320C31
				–	–	–			–					
2003	Energy generation		DHP[a]	12	2	14	Sigmoid		–		Reinforcement (critic and action)			
		Venayagamoorthy et al. (2003)*		6	2	10			–	3		On	–	TMS320C6701
					32 bit	FP			32 bit FP					

[a] Dual heuristic programming.

Table 34
Performance comparison with several workstations.

Implementation	MCUPS
Current M96002 DSP–FPGA (20 MHz)	2
Projected DSP–FPGA machine (16 PEs)	29
Intel Pentium II (300 MHz)	13
Sun Ultra 1/170 UltraSparc (167 MHz)	17
Sun Ultra 30 Ultrasparc (296 MHz)	32
DEC AlphaStation 21164 (500 MHz)	36

Xilinx (XC4000) and the high level control is performed on a DSP. The DSP is TMS320C31 from the company Texas Instruments. Finally the article mentions that the user has various wheelchair-guidance options, joystick, breath-expulsion, voice, etc.

- Venayagamoorthy et al. (2003) presented the work "Implementation of adaptive critic-based neurocontrollers multimachine for turbogenerators in a power system" applied to turbines for power generation control, replacing the conventional automatic voltage regulators (AVRs) and turbine governors. The work shows the design of a hardware implementation of nonlinear excitation and turbine neurocontrollers based on dual heuristic programming (DHP) theory (a member of the adaptive critic designs (ADC) family) for turbogenerators in a multimachine power system. Werbos (1992) proposed ACDs as a new optimization technique of neural-network combining concepts of reinforcement learning and approximate dynamic programming. An experiment is performed on a microturbine (driven by a DC motor) which drives a microalternador. The DHP neurocontrollers are tested for dynamic and transient operation for the following three disturbances: (a) an inductive load addition along the transmission line, (b) an increase in the transmission line

Please cite this article as: Bosque, G., et al., Fuzzy systems, neural networks and neuro-fuzzy systems: A vision on their hardware implementation and platforms over two decades. Eng. Appl. Artif. Intel. (2014), http://dx.doi.org/10.1016/j.engappai.2014.02.008

impedance and (c) a temporary three-phase short circuit. The tests were carried out with different controller combinations and compared to the conventional controllers, the neurocontrollers performance never degraded during these tests and the DHP neurocontrollers consistently had better damping. Depending on the type of test carried out, the DHP neurocontrollers have settling times faster than those obtained with the other controllers by 2–10 s. The device used is the TMS320C6701 from Texas Instruments® operating at 160 MHz.

5.3. Mixed neural implementations

The most common mixed implementations are based on analog circuits that incorporate some type of digital memory to store the weights and which need a D/A converter for treatment. These avoid the use of capacitors because of limited accuracy. Another type of mixed implementations, typical of past years, is presented in the FPGAs containing converters A/D and D/A for the treatment of external signal input and output.[22]

5.3.1. Dedicated Integrated Circuits

Within these implementations, the following references could be mentioned in Table 35.

Among the various proposals shown in Table 35, the following will be cited:

- Lee and Sheu (1990) implemented a general purpose neural device at work, "A compact and general-purpose neural chip with electrically programmable synapses" containing 64 neurons and 4096 synapses programmable on DRAM with refresh cycles of 0.2 s and accuracy of 8 bits. The synapse weighting data are stored in the buffer memory and are used to periodically refresh the on-chip synapse voltage through a D/A converter. Since the neural computation and synapse weighting refresh can be conducted concurrently, no dead time for refreshing the synapse voltage is required. The work does not mention the speed of the device. The device was developed with scalable CMOS technology of 2.0 μm.
- Boser et al. (1991) in the paper "An analog neural network processor with programmable network topology" present an analog–digital mixed device, named ANNA (Analog Neural Network Arithmetic unit), that performs over 2000 multiplications and additions simultaneously. Computations are performed with 6-bit accuracy for the weights and 3 bits for the neuron states after A/D conversion. The number of inputs per neuron varies between 16 and 256. The weights are 4096 and are stored on capacitors which must be refreshed every 110 μs. The peak performance of the chip is 5 GCPS with a clock rate of 20 MHz. It has been implemented a neural network for optical character recognition (handwritten digits). This network contains over 133,000 connections, is evaluated in 1 ms and recognizes 1000 characters per second which corresponds to 130 MCPS. The paper highlights that this rate constitutes a speedup of two orders of magnitude over a DSP based implementation (20 characters per second with a DSP32C from AT&T). The chip was fabricated in a single-poly, double-metal 0.9 μm CMOS technology with 5 V power supply.
- Shima et al. (1992) proposed an architecture formed by two high speed VLSI devices, performed with CMOS technology 0.8 μm, "Neuro chips with on-chip back-propagation and/or Hebbian learning". The first device contains the matrix of synapses and the second the neuronal matrix. The matrix contains the weights of synapses incorporating local control mechanism of the weights. The neuronal matrix contains neurons with backpropagation learning algorithms and/or Hebbian learning. The network is formed by two devices containing 24 neurons and 576 synapses (24×24) with 8 bits of precision.
- Lu et al. (2001) in the work "A programmable on-chip BP learning neural network with enhanced neuron characteristics" presented a novel neuron circuit with programmable parameters, generating the sigmoidal function and their derivatives. The neuron is analogic and its analogic weights are digitally stored on a RAM by means of an ADC and recovered by means of a DAC. The weights are digitally updated by means of a 7-bit ADC and added to the current value of 12 bits. The new value is digitally stored on a RAM. The neuron operates in transconductance, built with strong-inversion biased transistors. The computation rate is 12 MCPS. The work performs two experiments in order to determine the behavior of the neuron: (a) a non-linear partition problem and (b) a $\sin(x)$ function approximation. The results show, in the first experiment, a convergence in 1 ms and a good approximation (unquantified but shown graphically) in the second experiment. The NN has been implemented with a standard 1.2 μm CMOS process and the simulation has been performed with the SPICE tool.
- Minkovich et al. (2012) in the article "Programming time-multiplexed reconfigurable hardware using scalable neuromorphic compiler" presented a programmable front-end composed of a neuromorphic compiler and a digital memory, based on the concept of synaptic time multiplexing (STM). The neuromorphic compiler automatically translates any given neural architecture to hardware (neuromorphic system). The neurons and synapses are stored on an analog core. The device also contains a digital memory and an analog memory (capacitors). The implementation has 10^6 neurons and 10^{10} synapses. The article provides a neural simulator feedback loop that allows improvement of the network partition and parallelization of the simulation. Also, among other advantages, it allows for synapses with larger synaptic conductances to have shorter path lengths and, thus, ensures more reliable transmission. The simulator can also provide firing rate information for each neuron. The architecture has been performed on 90 nm CMOS technology.
- Erkmen et al. (2013) published the work "A mixed mode neural network circuitry for object recognition application". They developed a general purpose Conic Section Function Neural Network (CSFNN) using full custom 0.5 μm CMOS technology. The feedforward computation units are all analog current mode circuits while the control unit and storage of the network parameters are digital. The digital part was composed of memories (EEPROM cells), decoders, and the control unit. The architecture of the CSFNN allows a maximum size of 16 inputs, a hidden layer with 16 neurons, and 8 outputs and was trained using hardware-in-the-loop techniques. The work demonstrates that the performance for typical object recognition problems is similar to that obtained using a software approach. Moreover, the forward propagation time was significantly reduced; it takes 0.24 μs for the CSFNN chip and about 1 ms on a Pentium4 with 2 GB RAM.

5.3.2. Programmable Integrated Circuits

The following proposal, reflected in Table 36, is a combination of dedicated (VLSI) and programmable (FPGA) circuits but their functionality is fully programmable.

Among the two proposals shown in Table 36, the following will be cited:

[22] In this case, it could be accepted that the incorporation of converters A/D and D/A does not substantially change the digital nature of the device.

Table 35
HW implementations of neural systems – digital mixed.

Year	Applications	Ref.	Type of ANN	In	Out	Hidden	Activat. function	No. of neurons	No. of synapses	No. of layers	Learning	On/off chip	Speed	Technology
1990	General	Lee and Sheu (1990)*	2-layer	64 (mux8to1) 64	64 (mux8to1) 64 Analog	– –	–	64 12	4096 144 8 bit	–	–	Off	–	CMOS 2.0 μm
1991	General, character recognition	Boser et al. (1991)*	Multilayer, Hopfield, LRF[a], etc.	16..256 64 64 × 1 16 × 16 Analog	Multiplexed ≥ 8 32 – – →	– 32–32 – – 3 bit	–	Multiplexed ≥ 8 160 64 – 6 bit	Multiplexed ≥ 4096 4096 – –	– 4	–	Off	5 GCPS 0.8 GCPS 2.1 GCPS 4.7 GCPS	CMOS 0.9 μm (ANNA chip)
1991	General	Arima et al. (1991a)	Fully feedback connection	25	100 Analog	–	–	125	– –	10k	Boltzmann	On	1 ∼ 2 μs	CMOS 1.0 μm
1992	General	Satyanarayana et al. (1992)	Program.	2 8	1 4 Analog	4 24	Sigmoidal	7 36	1024 8 bit	2 3	Weight perturbation	On	–	CMOS –
1992	Voice recognition	Shima et al. (1992)*	Multilayer	24 24	1 24 Analog	– 24	Sigmoidal	24 72	576 8 bit	1 3	Backprop., Hebbian	On	–	CMOS 0.8 μm
2003	General	Lu et al. (2001)*	MLP, –	2 1	1 1 Analog	– 5	Sigmoidal	– –	24 12 bit	2 3	Backprop	On	12 MCPS	CMOS 1.2 μm
2012	Biological neuron	Minkovich et al. (2012)*	Neuromorphic system	– –	– –	– –	–	10^6	10^{10} –	– –	STDP[b]	On	–	CMOS 90 nm
2013	Object recognition	Erkmen et al. (2013)*	CSFNN[c]	16	8 Analog	16	–	–	– 8 bit	3	HW-in-the-loop[d]	off	0.24 μs	CMOS 0.5 μm

[a] Local receptive fields.
[b] Spike timing-dependent plasticity.
[c] Conic section function neural network.
[d] Hardware-in-the-loop techniques.

Table 36
HW implementations of neural systems – Digital Programmable Integrated Circuits.

Year	Applications	Ref.	Type of ANN	In	Out	Hidden	Activat. function	No. of neurons	No. of synapses	No. of layers	Learning	On/ off chip	Speed	Technology/ device
1991	Image analysis	Sackinger et al. (1991)	Multilayer	16..256	–	–	–	–	4096↑	–	–	On (DSP)	30 MCPS .. .. 400 MCPS	CMOS 0.9 μm (ANNA) DSP32C
1996	Text location, Opt. character recognition, etc.	Sackinger and Graf (1996)*	Convolutional, LRF, multilayer, CNN, etc.	12 bit 512×512	$\geq 8 \times 2$	Analog –	Sigmoid, others	$\geq 8 \times 2$	6+4 bit $\geq 256 \times 8 \times 2$	–	–	–	10/20 GCPS (peak)	CMOS 0.9 μm (ANNA $\times$ 2) X4005 $\times$ 4
				–	–	–			6 bit					

Table 37
Summary of the main features of neuronal networks on analog, digital and mixed-dedicated circuits.

Circuit	Activat. function[a]	Learn[a]	Speed		Tech. μm CMOS	Current	Power	
			CPS	CUPS			mW	W
Analog	Sigmoid, bell, Gaussian	Weight perturbation, backprop., Boltzmann	10M .. 1T	10M .. 80T	(90 nm), 0.35 .. 1.3	35 μA..15 mA	6, 53.3 .. 320	3, 4.5
Digital	Sigmoid, any (LUT)	Backprop., adaptive, LVQ	110M, 130M, 1.3G, 1.6G	60M, 0.66G	0.18 .. 0.8	–	11.2	1, 3
Mixed	Sigmoid	Backprop., weight perturbation, Boltzmann, etc.	12M, 5G	36G	(90 nm), 0.5 .. 2	–	200, 250	1.5

[a] In order to the preferences of the articles consulted.

- Sackinger and Graf (1996) (AT&T Bell Labs) presented in the article "A board system for high-speed image analysis and neural networks", a system based on the ANNA device (Boser et al., 1991). It is applied to optical character recognition, noise removal, etc. It consists of a card containing two fully programmable analog VLSI devices (ANNA neural-network) and four Xilinx® FPGAs, 4005-55PG156, memory and interfaces. The card industry is presented as a "double Europe" (6U) and under VME Bus. ANNA devices are performed by CMOS technology of 0.9 μm. Each ANNA device evaluates 8 neurons with up to 256 synapses per neuron in 4 clock cycles (4×50 ns). The card features vary from hundreds of MCPS and 2 GCPS depending on the topology of the network and the card has a recognition speed of 1000 characters per second. The computational peak performance is 10 GCPS.

5.4. Conclusions and complementary readings of HW implementations for neuronal networks

The development of neural networks on an analog dedicated HW has been given as a result of various factors. These are speed (a trend mainly reflected in the 90 s) of adders and multipliers for the low number of transistors required and a final important reason that these architectures can keep the parallelism inherent in neural networks. As regards the learning algorithms, there are two trends in the implementation: learning on-circuit or off-circuit (in the latter case, in order to increase the speed of the network). The most widely used algorithms are backpropagation and weight perturbation. As in the case of fuzzy systems, the emergence of the FPAA in the 2000s, to which researchers are beginning to pay attention, must also be noted, but there are few contributions. Digital designs have introduced more stability against factors such as noise, temperature or voltage variation and, finally, easily achievable cascadability.

Regarding mixed devices, generic devices have been developed where the weights are stored on a digital memory, requiring D/A converters for processing. There are also implementations on FPGAs incorporating A/D and D/A. Single-chip or multi-chip developments are tending towards high performance computing structures.

Considering the main features on dedicated devices, Table 37 shows the summary of the data extracted from the works consulted: analog, digital and mixed-dedicated circuits.

The increase in the integration density, in the last decade, has modified the design trend targeting digital devices. Dedicated processors have been discontinued, tending to techniques which are more structured, as employed in the development of the ASICs. Here a small area for integration is required, as well as a greater tendency to use the FPGA, due to the high parallelism and their ability for reconfiguration. Regarding learning algorithms, as in dedicated devices, there are two trends: learning on-line and off-line. Some implementations perform on-chip learning. The most widely used algorithm is pure or modified (e.g. pulse mode eliminates multipliers) backpropagation. The off-line learning is performed off-chip and weights are loaded into the FPGA.

There are some contribution on DSPs but it is the FPGAs that show the tendency to continue in implementing neural networks. In addition, the FPGAs present low "time to market". Table 38 shows the main features of neural networks on analog and digital programmable devices.

With regard to complementary readings, the following works should be mentioned. On analog hardware, the article by Draghici (2000) "Neural networks in analog hardware—design and implementation issues" should be consulted. It gives an interesting view of analog hardware and the pros and cons regarding digital solutions until 2000. Also, it is interesting to cite the article of Misra and Saha (2010) "Artificial neural networks in hardware: a survey of two decades of progress" which examines

Please cite this article as: Bosque, G., et al., Fuzzy systems, neural networks and neuro-fuzzy systems: A vision on their hardware implementation and platforms over two decades. Eng. Appl. Artif. Intel. (2014), http://dx.doi.org/10.1016/j.engappai.2014.02.008

Table 38
Summary of the main features of neural networks on analog and digital programmable circuits.

Device	Type	Activat. function[a]	Learn[a]	Speed		
				CPS	CUPS	Freq. Hz
FPAA		Comparator, sigmoid PWL	Backprop. (Matlab)	6M	–	–
FPGA		Sigmoid, threshold, Gaussian, hyperbolic tangent, ramp, tansigmoid	Backprop., gradient, unsupervised, resilent prop., stochastic, comparative unsupervised, Widrow–Hoff, etc.	211k, 14.6M .. 87M	568k .. 857k, 5M .. 81M, 1G	24.2M .. 90M
Commercial processor	General purpose	Sigmoid, hyperbolic tang.	Backprop.	1.6G, 12.8G	260M	–
	DSP	Sigmoid, FIR filter	Backprop., gradient, comp. learning, reinforcement	–	2M	–

[a] In order to the preferences of the articles consulted.

the implementations of ANNs on different HW, and also some papers with analog designs, until 2010. The book "FPGA implementations of neural networks" (Omondi and Rajapakse, 2006) is also interesting.

In a specific area such as image processing, the revision made by Egmont-Petersen et al. (2002) "Image processing with neural networks—a review" is also of interest. It analyzes a period of 15-years until 2000.

6. HW implementations for neuro–fuzzy systems

The hybrid systems, composed by neural networks and fuzzy systems, are divided into two system types: on one hand the type FNN (Fuzzy–Neural Network) (Buckley and Hayashi, 1994; Chen and Teng, 1995; Zengqi and Zhidong, 1996) that can be formed by an ANN capable of handling fuzzy information, and on the other hand, the type NFS (Neural–Fuzzy System) (Takagi and Hayashi, 1991) formed by a fuzzy system with neural structure and learning ability. This last case still presents a set of interpretable rules after learning because it remains as an FIS. It is on these latter that the largest number of hardware implementations has been performed in different application areas. As in Section 5, the implementations are (a) *analog*, (b) *digital*, and (c) *mixed*. The classification of these systems is the same as that adopted in Table 3.

6.1. Analog neuro–fuzzy implementations

6.1.1. Dedicated Integrated Circuits

In such implementations there are hardly any relevant papers, given the complexity in the design of a dedicated analog device. Table 39 shows these papers.

Of all of these, the following are highlighted (*):

- Rodriguez-Vazquez and Vidal-Verdù (1996) presented an analog hardware to implement neuro–fuzzy systems, "Hardware implementation of neuro/fuzzy systems as mixed-signal chips". It is based on CMOS transistor technology and the membership functions are performed on blocks operating in transconductance mode. The inputs are voltage and the outputs are current. This hardware includes a learning method based on a variation of gradient descent, instead of computing derivatives using the method known as perturbation of the weights (Jabri and Flower, 1992).
- Conti et al. (1998) in the work "A current-mode neuro–fuzzy network" proposed a new current-mode circuit for the implementation of the membership functions. The circuit is a current driven double differential pair biased in strong inversion, and furthermore incorporates a multiplier for inferences and one divider for normalization. As an application example of the architecture proposed, it implements a controller of an air conditioning system. The number of rules is three and the number of membership functions is three (two trapezoidal and one triangular), one input (temperature) and one output (heater); the output has three membership functions of rule consequents. The device was developed with CMOS technology of 0.8 μm.
- Sultan and El-Sayed (2000) presented the work "Analog VLSI implementation of adaptive neuro–fuzzy inference systems", where they discuss the implementation of an ANFIS with on-chip learning. The system is trained with the perturbative stochastic learning method applied to predict a chaotic time series (Mackey Glass). The stochastic algorithm specifies incremental updates in the weights using a stochastic approximation to true gradient. The weights are stored on capacitors. Two membership functions are assigned to each input, there are 4 inputs (four past samples), and thus there are sixteen rules. Trapezoidal membership functions are employed and implemented using CMOS current-mode techniques. The project has been simulated on SPICE with CMOS technology of 1.2 μm.
- Wang and Jin (2006) in the article "Neuro–fuzzy system with high-speed low-power analog blocks" presented several voltage-mode analog CMOS circuit blocks for the fuzzy system. The neuro–fuzzy system is based on a Mamdani model. This has a Gaussian-like membership function circuit, a minimization circuit, and the defuzzification circuit not using division. The system has two-input/one-output and with a complexity of 25 rules. The number of membership functions is 5 for each input. The defuzzification is the COA method. The values of weight ω_i are generated by Matlab. Due to the compact architecture and the low supply voltage the circuit consumes only 1 mW. The operation speed of inference in the designed neuro–fuzzy system is 5 MFLIPS with the power-consumption of only 1 mW. All the blocks have been fabricated in SMIC 0.18−μm mixed-signal CMOS technology. This work presents a comparison between this implementation and four previous works.

6.1.2. Programmable Integrated Circuits

Undetected.

6.1.3. Commercial processors

Undetected.

Please cite this article as: Bosque, G., et al., Fuzzy systems, neural networks and neuro-fuzzy systems: A vision on their hardware implementation and platforms over two decades. Eng. Appl. Artif. Intel. (2014), http://dx.doi.org/10.1016/j.engappai.2014.02.008

Table 39
HW implementations of neuro–fuzzy systems – analogics dedicated.

Year	Applications	Ref.	Type of NFS	In	No. of in MFs	Type of in MFs	Out	No. of out MFs	Type of out MFs	No. of rules	Defuzzification	Learning	On/ off chip	Speed	Technology
1995	General	Rodriguez-Vazquez and Vidal-Verdu (1995)*	T–S	3	–	Bell	1	–	Singleton	4	T–S	Weights perturbation	On	5 MFLIPS	CMOS 1.5 μm
1997	General	Moreno et al. (1997)	T–S	3	2	–	1	–	–	8	T–S	–	–	1–5 MHz	CMOS 0.8 μm
1997	Low-cost embedded	Conti et al. (1997)	AINN[1]	1	3	Triang.	1	3	Triang.	3	COG	–	–	–	CMOS (Simulat. SPICE)
1998	Air conditioned	Conti et al. (1998)*	AINN	1	3	Trapz., triang.	1	3	Triang.	3	–	–	–	–	CMOS 0.8 μm (Simul. spectra)
1999	General	Vidal-Verdu et al. (1999)	T–S	3	Clustering	Trapz., triang., bell	1	4	Singleton	4	T–S	Weights perturbation	Off	5 MFLIPS	CMOS 1.6 μm
2000	Chaotic prediction	Sultan and El-Sayed (2000)*	ANFIS	4	2	Trapez.	1	16	Singleton	16	ANFIS	Stochastic gradient	On	–	CMOS 1.2 μm
2000	Fuzzy partition MFs	Conti et al. (2000)	3-layer	*n*-dimensional	*m*	Trapez	–	–	–	–	–	–	–	–	CMOS (Simulat. cadence)
2004	Current-mode circuits	Wang and Jin (2004)	5-layer	–	–	Gaussian	–	–	Gauss	–	COG	–	–	–	CMOS 0.35 μm (Simulat. HSPICE)
2006	General	Wang and Jin (2006)*	Mamdani	2	5	Gaussian	1	–	–	25	COA (div. free)	Matlab	Off	5 MFLIPS	CMOS 0.18 μm

[1] Approximative identity neural network.

6.2. Digital neuro–fuzzy implementations

In this subsection the same comments can be made as those made in Section 5.2.

6.2.1. Dedicated Integrated Circuits

Undetected.

6.2.2. Programmable Integrated Circuits

Here are some relevant approaches of this kind of implementations presented in Tables 40 and 41.

The following will be cited (*):

- Baturone et al. (2001) in the article "VLSI design of universal approximator neuro–fuzzy systems" presented two strategies for generating code for a type NFS Takagi–Sugeno zero-order. One strategy is based on memory for storage of parameters and the other is based on the fuzzification and calculation of the activation of the rules. The first strategy, through synthesis tools both Sinopsis as Xilinx®, generates code for Xilinx® FPGAs and the second also generates VHDL code and code for Xilinx® FPGAs. They also allow both on-line and off-line learning.
- Sameh and Samir (2005) in the article "Generic floating point library for neuro–fuzzy controllers based on FPGA technology" implemented FSOM networks (Fuzzy Self Organizing Map) and ANFIS (Adaptive Neuro Fuzzy Inference System) on a FPGA through the FPGA-Advantage tool from Mentor Graphics. Calculations are performed using floating point FPU (Floating Point Unit) as a library module. Learning is off-line and off-chip due to the difficulty of implementation. The training is performed on Matlab and values are dumped on the device.
- Bosque et al. (2006) in the work "Efficient hardware/software implementation of a neuro–fuzzy system on a SoPC" presented the implementation of a neuro–fuzzy ANFIS type on a FPGA. The implementation is done by creating a partition HW/SW on the FPGA. SW partitioning algorithms implement a hybrid learning network for a system of two inputs and one output. On the HW partition feed-forward network is implemented. After the completion of the learning process, the feed-forward network is executed in 4 clock cycles. Operating at 67 MHz gives a time of 60 ns per inference. The chosen device belongs to the family of Altera® Excalibur and the processor is a hard-core of ARM family.
- Fujimoto et al. (2007) (there are two works, one presented in 2006 and the other in 2007) in the work "An implementation of the neuro–fuzzy inference circuit" implemented a back-propagation learning algorithm on the FPGA hardware, achieving a high speed through parallel processing. The membership functions are triangular, normalized and overlapping pairs. The neuro–fuzzy inference circuit is described with a hardware description language Verilog-HDL. The work does not mention which FPGA they use.
- Echanobe et al. (2008) in the work "An adaptive neuro–fuzzy system for efficient implementations" described an ANFIS

Table 40
HW implementations of neuro–fuzzy systems – digital (1).

Year	Applications	Ref.	Type of NFS	In	No. of in MFs	Type of in MFs	Out	No. of out MFs	Type of out MFs	No. of rules	Defuzz.	Learning	On/off chip	Speed	Device
2001	Generic	Baturone et al. (2001)*	T–S	2	3–5	Triang.	1	15	Singleton	15	T–S	Weight perturb., gradient-descent	On/off, chip-in-the-loop		FPGA Xilinx (generic)
				–	–	–	–	8 bit							
2004	Generic	Bosque et al. (2004)	ANFIS	2	4–4	Triang.	1	16	Singleton	16	ANFIS	Hybrid (off-line)	On	69.5 MFRPS (57.5 ns)	EPXA4-F672C1 μP ARM922T (SoPC)
					5–5			25		25					
						16 bit FxP									
2005	Generic	Sameh and Samir (2005)*	FSOM	2	-	Triang., bell	1	-	Singleton	–	Sum of fire rules	Hybrid	Off (Matlab)	–	FPGA (generic)
			ANFIS	3	–		1	–							
				32/14 bit FP											
2006	Generic	Bosque et al. (2006)	PWM–ANFIS	2	5–5	Triang.	1	25	Singleton	25	ANFIS	Hybrid (off-line)	On	67 MFRPS (59,7 ns)	EPXA1 μP ARM922T (SoPC)
						2 bit FxP									
2006	Real-time learning	Fujimoto et al. (2006, 2007)*	IF–THEN	2	5–5	Triang.	1	–	–	–	Sum average	Self-tuning (conse-quents)	On	13.2 ms	FPGA (generic)
					9–9									28.9 ms	
				12 bit FxP		9 bit FxP	–								
Year	Applications	Ref.	Type of NFS	In	Out	Hidden (neurons)	Activat. function	No. of neurons	No. of synapses	No. of layers	–	Learning	On/off	Speed	Device
2006	Pattern recog., system identif., image process.	Pedram et al. (2006a, 2006b)	LOLIMOT[a] (T–S–K type)	n	1	m	Gaussian, weighted sum (out)	$m+1$	$(n+1)\times m$	2	–	Divide and conquer	On	8-b: 92.5 MHz 16-b: 76.4 MHz	EP1S10F
														32-b: 63.3 MHz	EP1S40F
						8–16–32 bit									

[a] Local Linear Model Tree.

Table 41
HW implementations of neuro–fuzzy systems – digital (2).

Year	Applications	Ref.	Type of NFS	In	No. of in MFs	Type of in MFs	Out	No. of out MFs	Type of out MFs	No. of rules	Defuzz.	Learning	On/ off chip	Speed	Device
2007	Generic architecture	Kala and Srinivas (2006)	IF–THEN	4	–	Any	2	–	Any	1024	–	–	–	–	CoolRunner (PLD)
				8 bit	4 bit		8 bit	–							
2007	Car-driving system	Kao et al. (2007)	IF–THEN (adaptive)	2	–	Triang.	1	–	–	–	–	Backprop.	On	–	FPGA (generic)
						14 bit									
2007	Asynchronous pipeline	Lin et al. (2007)	Feedforward	2	–	Triang.	1	–	–	–	Weighted average	Backprop.	On	192.31 kHz	FPGA (generic)
				–	–		–	–							
2008	Generic	Bosque et al. (2008)	PWM–ANFIS	n	m	Triang.	1	p	Singleton	m^n (activ: 2^n)	ANFIS (div.-free)	Hybrid (On/off-line)	On	–	FPGA (generic)
				2	7–5		1	35		Activ: 4					
				3	5–5–5		1	125		Activ: 8					
						8/16/32 bit									
2008	Generic	Echanobe et al. (2008)*	ANFIS	2	10–10	Triang.	1	100	Singleton	Activ: 4	ANFIS	Hybrid	Off	29 MFIPS	EP2S15
				3 4	7–7–7 6–..–6			343 1296							
					8 bit	FxP									
2008	General	Nagamine et al. (2008)*	IF–THEN	2	5–5	Triang.	1	–	–	16	Average	Gradient (div.-free)	On	37.34 MHz	XCS300E
					16 bit	FxP				34					
2008	Generation of MFs	Fujimoto et al. (2008)	IF–THEN	2	$3\uparrow-3\uparrow$	Triang.	1	–	–	$16\uparrow$	-	Backprop.	–	–	FPGA (generic)
2008	Digital predistorter	Zhai et al. (2008)	ANFIS	1	–	Bell generalized 12 bit	1	3	Linear	3	Sum	Hybrid (off-line)	Off	–	Altera –

network implemented on an FPGA introducing certain restrictions on the membership functions to simplify the network HW. Functions are triangular, remaining normalized and with overlapping pairs. The parameters of the network reside in the memory of the FPGA. The implementation was done on a Stratix II device via Altera® Quartus II tool. The rate achieved for an inference is 34 ns or, in other words, 29 MFIPS for a clock frequency of 116.54 MHz.

- Nagamine et al. (2008) in the article "Fuzzy inference models appropriate for digital circuit" proposed two models of learning a fuzzy inference system and the implementation of one of them on an FPGA. The two models are based on the model *division-free*: *division-free* strict and *division-free* with learning *ensemble*. Both models allow more effective implementation in terms of size and processing speed. The main disadvantage is that they are less accurate than the conventional division, although the second surpasses the first in accuracy. The membership functions are triangular and the restriction is imposed to give normalized and overlapping pairs. The model implemented is the *division-free* strict and implementation is done on a Spartan2E Xilinx® device by means of ISE tool. The article performs a comparison of speeds and obtained a clock speed of 37.34 MHz in the case of *division-free* and of 9.18 MHz for classical calculation, which shows a rate 4 times higher for the proposed model.
- Sundarambal et al. (2009) in the article "VHDL implementation of neuro–fuzzy based adaptive bandwidth controller for ATM networks" aim to maintain a quality of service (QoS) in ATM (Asynchronous Transfer Mode). They proposed a neuro–fuzzy adaptive bandwidth controller developed in VHDL. The architecture comprises the following subsystems: estimation of Cell Loss Ratio (CLR) (2 inputs: trapez., 1 output: trapez.), an Adaptive Congestion Controller (ACC) (3 inputs: trapez., 1 output: trapez., triang.), an Adaptive Bandwidth Estimator (ABE) (3 inputs: trapez., 2 output: singleton.), and finally, an Adaptive Admission Controller (AAC) (3 inputs: trapez., triang., 1 output: trapez., triang.). It is interesting to note that the MFs are generated by a Kohonen learning algorithm. It is implemented on an Altera® Cyclone FPGA EP1C6Q240C8 using the Quartus II tool.
- Aldair and Wang (2010) in the article "FPGA based adaptive neuro fuzzy inference controller for full vehicle nonlinear active suspension systems" introduced a Fraction Order $PI^{\lambda}D^{\mu}$ (FOPID) controller, an Evolutionary Algorithm which trains the parameters of the FOPID and an ANFIS for controlling the nonlinear active suspension system. The inputs are the position error and error rate, the output is the control of a hydraulic cylinder and the membership functions are bell shaped. The data obtained from the FOPID are the reference for the ANFIS design (is a subcontroller of the ANFIS). The tool for Simulation has been Matlab/Simulink and the tool for the simulation results has been ModelSim XE. In this last case, the implementation has been written in VHDL. The article presents a comparison between the responses of Matlab and the FPGA, which are

Table 42
HW implementations of neuro–fuzzy systems – digital (3).

Year	Applications	Ref.	Type of NFS	In	No. of in MFs	Type of in MFs	Out	No. of out MFs	Type of out MFs	No. of rules	Defuzz.	Learning	On/ off chip	Speed	Device
2008	Identify a nonlinear syst., truck backing	Juang and Tsao (2008)	Type-2 SONFS	2	–	Interval sets	1	–	Mamdani	6	Sum average	Kalman filter	Off	54 MHz	XC4V1X60
				3	– 8 bit		1 13 bit	–		5					
2009	Communications	Sundarambal et al. (2009)*	5-layer	2	2-3	Trapez.	1	3	Trapez., triang.	6	Weighted average	Kohonen	On	–	EP1C6
				3	2–2–3			5							Cyclone
				3	3–2–3		2	6–6	Singleton	9					
				3	3–5–3	Trapez., triang. 8 bit	1	4	Trapez., triang.	23					
2009	Identification and prediction	Lin and Lee (2009)	Recurrent	2	–	Gaussian	2	–	–	12	Sum	Simultaneous perturbation	On	–	XC2V6000
			NFN	3		16 bit	1 FxP			15					
2009	Obstacle avoidance	Mahyuddin et al. (2009)	Cooperative NFS	3	4–4–4	Trapez., triang.	2	–	–	40	LOM[a]	Backprop. (on-line)	On	–	Cyclone II (SoPC)
2010	Suspension systems	Aldair and Wang (2010)*	ANFIS	2	–	Bell	1	–	Singleton	–	ANFIS	Hybrid	On	–	XC3S700AN
2010	Nonlinear funct. generator	Saldaña and Cardenash (2010)	ANFIS	2	5–5	Triang.	1	25	Singleton	9	ANFIS	Hybrid	Off	10 μs	XC3S200
						8/16 bit	FxP								
2011	Classification, prediction (chaotic signal)	Lin and Lee (2011)*	4-layer	2	–	Gaussian (LUT)	1	–	–	–	Sum	Simultaneous perturbation	On	12.6 MHz	XC2V6000
				1		10 bit	1 FxP								
2012	Ambient intelligence	del Campo et al. (2012)	PWM–FIS	4	3–..–3	Triang.	1	16	Singleton		Sum	Hybrid	On/ off	100 MHz	XC5VSX50T (SoPC)
					5–..–5 8 bit		32 bit	8 bit							
2013	Motor control	Chou et al. (2013)*	IF–THEN and RBF NN	2	7–7	Triang.	1	–	Singleton	49 (4 activ.)	Central average	Gradient descent	On	7.36 μs	Cyclone
						16 bit									EP2C70

[a] Largest of Maximum.

similar. The work has been performed on the FPGA Spartan XC3S700AN from Xilinx and the platform has been the Spartan 3E starter Kit board.

- Lin and Lee (2011) in the article "Implementation of a neuro–fuzzy network with on-chip learning and its applications" utilized the simultaneous perturbation method as a learning algorithm. The advantage of this optimization method is its simplicity. The method can estimate the gradient using only values of the error function. The membership functions are Gaussian but approximated with digital techniques using a second order function without an upper saturation region (Blake et al., 1998). The article analyzes the XOR problem and a prediction of a chaotic signal problem, performing a comparison between a software implementation and the hardware implementation (FPGA). The results are similar. The device is a FPGA Xilinx Virtex-II XC2V6000-4FF1152C.
- Chou et al. (2013) in the article "Optimized FPGA design, verification and implementation of a neuro–fuzzy controller for PMSM drives" presented a neural fuzzy controller (NFC) for speed loop of permanent synchronous motor (PMSM) whose parameters are adjusted by a RBF NN. The NFC is implemented on a FPGA, describing the system by means of VHDL language. The number of inputs is two (error of the rotor speed and error change), the number of outputs is one (speed rotor) and the membership functions are 7 (triangular overlapped by 2) for each input. The RBF NN is applied in order to identify in real-time the plant dynamic for providing exact plant information to the learning algorithm of NFC. To simplify the computation of the RBF, the Gaussian function (exponential) of the hidden layer is approximated by means of Taylor series (12th order). The system is co-simulated by Matlab/Simulink and the implementation is on a FPGA Cyclone EP2C70 of Altera. The execution time of the NFC on this device is 7.36 μs.

6.2.3. *Commercial processors*

The same considerations apply as those in Section 4.2.3.

General purpose processors: Here are some relevant approaches of this kind of implementations presented in Table 43.

The following will be cited (*):

- Porto et al. (2002) in the work "A neuro–fuzzy approach for hardware modeling of nuclear fusion reactors plasma"

Table 43
HW implementations of neuro–fuzzy systems – commercial processor.

Year	Applications	Ref.	Type of NFS	In	No. of in MFs	Type of in MFs	Out	No. of out MFs	Type of out MFs	No. of Rules	Defuzz.	Learning	On/off chip	Speed	Device type
2002	Nuclear fusion plasma	Porto et al. (2002)*	IF–THEN	6	2–..–2	Triang.	1	–	Singleton	64	Minimum inference	Mean quadratic (on-line)	On	–	μ P ST52x420
2007	Gas detection	Kim et al. (2007)*	ARTMAP, ART	1 1 bit	– –	–	3 –	– –	–	–	–	ARTMAP, ART	On	–	μ C MSP430F1611
2012	Motor control	Dongale et al. (2012)*	Mamdani	2 –	7–7 –	Trapez., triang. –	1 –	7 –	Trapez., triang.	49	–	Matlab (Narma-L2)	Off	–	μ C 16F877A

designed a behavior predictor of a nuclear fusion plasma. The work performs a modelization of the nonlinear relation existing among the six forcing magnetic fields and the geometric displacement of the high-temperature plasma. The membership function shape is triangular and there are two for each input, leading to a fuzzy model with 64 rules. The output is a crisp value resulting from the minimum inference method. The system has been implemented on a μ P ST52x420 with fuzzy instructions and the FUZZYSTUDIO 4.1 tool has been used for importing the fuzzy algorithms previously developed on the AFM (Adaptive Fuzzy Modeler) software tool of STMicroelectronics.

- Kim et al. (2007) in the work "An intelligent wireless electronic nose node for monitoring gas mixtures using neuro–fuzzy networks implemented on a microcontroller" designed a neuro–fuzzy system which classifies and quantifies binary gas mixtures (NH_3 and H_2S). The classification is performed using a pattern recognition algorithm using the fuzzy ART and ARTMAP neural networks. The measures are transmitted to a Laptop Computer. The software has been developed and simulated on Matlab and the gas monitoring on LabVIEW. The device used is the μ C MSP430F1611 from Texas Instruments®.
- Dongale et al. (2012) in the article "AC induction motor control —a neuro–fuzzy approach" presented a simulated (Matlab/Simulink) and an experimental implementation (μ C) of a neuro–fuzzy controller of an AC induction motor. The system model is Mamdani type and has two inputs (error and error change) and one output (speed). The inputs and output have 7 membership functions (triangular and trapezoidal) and the number of rules is 49. The model is implemented on Simulink (Matlab) using the neuro–fuzzy controller NARMA-L2; this model performs the network training. The controller generates the PWM sequence for a half bridge inverter. The hardware implementation includes a three phase half bridge inverter and a 0.5 HP, 1A motor. The device used is the μ C PIC 16F877A of 8-bit from Microchip.

Commercial DSPs: Some relevant approaches are shown in Table 44.

The following will be cited (*):

- Grabowski and Blaabjerg (1998) in the work "Direct torque neuro–fuzzy control of induction motor drive. DSP implementation" established that the control strategies guarantee very good dynamic and steady state characteristics with low sampling time and constant switching frequency. The NFS has two inputs (ε_ψ – flux error, ε_m – torque error) and two outputs ($|V_c|$ – reference voltage amplitude, φ_{V_c} – reference voltage angle). These outputs are the inputs of the vector modulation block by means of a dual port RAM. Each input has three membership functions (trapezoidal: 2, triangular: 1). The devices are Analog Devices SHARC ADSP-21062 (floating-point) and SIEMENS SAB 80C167 μ C which provides the modulation. The induction machine is a four-pole 4 kW ABB MBT 112M.
- Grabowski et al. (2000) in the paper "A simple direct-torque neuro–fuzzy control of PWM-inverter-fed induction motor drive" presented a Direct-Torque Neuro–Fuzzy Control (DTNFC) similar to an ANFIS net. The controller was tuned automatically by a gradient algorithm and the tuning is off-line. The controller has two input (three membership functions: 2 trapezoidal and 1 triangular) and two outputs. The system is controlled by two DSPs: TMS320C31 and TMS320P14. The first (main) processor implements the DTNFC control algorithm and the second provides the vector modulation. The board is equipped with four A/D converters (two 16-bit and two 12-bit), four digital-to-analog converters, and the input for an encoder. The induction motor is a 3 kW four-pole.
- Akcayol (2004) in the article "Application of adaptive neuro–fuzzy controller for SRM" presented an adaptive neuro–fuzzy inference system (ANFIS) to speed control of a switched reluctance motor (SRM). The ANFIS controller generates change in the inference current (Δ_i), based on speed error (ω_e) and change in the speed error (ω_{ce}). The membership functions are generalized bell and there are 5 per input. The ANFIS net must train 3 consequents per rule (p_i, q_i and r_i). The phase currents are sampled at every 100 μs regulating the phase current outputs. Two experiments are performed: (a) the motor is running at 500 rpm in steady state, applying a 10 N m load torque for 12 s and removed and (b) the motor is running at 1500 rpm in steady state, applying a 10 N m load torque for 12.4 s and removed. In both cases the overshoot and oscillations are negligible. The algorithm has been implemented on a DSP TMS320F240 of Texas Instruments. It is a 16-bit fixed point and has a 50 ns instruction cycle. The motor is a 8/6 (Stator pole/Rotor pole) of 5.5 HP.
- Elmas et al. (2008) in the paper "A neuro–fuzzy controller for speed control of a permanent magnet synchronous motor drive" performed an ANN to adjust input and output parameters of membership functions in a FLC. The NFC has two inputs and one output, inputs: speed error (e_ω) and the derivative of the speed error ($\dot{e}_\omega$), output: the change of the control current (ΔI^*). The membership functions are Gaussian but the paper does not mention how many were used. The output layer performs the defuzzification using the COA method. The neuro–fuzzy network is trained using an off-line backpropagation learning algorithm. One experiment is performed comparing the results between the NFC controller and a Proportional–Integral (PI) controller. The speed is 1500 rpm under a nominal load of 1 N m. Both controllers closely follow the reference speed, but the PI controller presents overshoot and oscillation in the speed curves. The motor is a three phase PMSM of 400 W. The DSP is the TMS320F240.

Please cite this article as: Bosque, G., et al., Fuzzy systems, neural networks and neuro-fuzzy systems: A vision on their hardware implementation and platforms over two decades. Eng. Appl. Artif. Intel. (2014), http://dx.doi.org/10.1016/j.engappai.2014.02.008

Table 44
HW implementations of neuro–fuzzy systems – Digital Signal Processor.

Year	Applications	Ref.	Type of NFS	In	No. of in MFs	Type of in MFs	Out	No. of out MFs	Type of out MFs	No. of rules	Defuzz.	Learning	On/ off chip	Speed	Device
1997	General	Bona et al. (1997)	Any	–	–	Any	–	–	–	–	–	Any	On	–	TMS320C31 AMINAH
				–	–		–	–							
1998	Motor drive	Grabowski and Blaabjerg (1998)*	5-layer	2	3-3	Trapez., triang.	2	–	–	9	Normaliz.+ vector adder	–	–	–	ADSP21062
															μ C SAB80C167
				12 bit	–		–	–							
1998	Control, educational	Bona et al. (1998)	Any	–	–	Any	–	–	–	–	–	Any	On	–	TMS320C31
				12 bit	–		12 bit	–							AMINAH
2000	Motor drive	Grabowski et al. (2000)*	DTNFC	2	3-3	Trapez., triang.	2	–	–	9	Normaliz.+ vector adder	Gradient	Off	–	ADSP320C31
															TMS320P14
				12/ 16 bit	–		–	–							
2002	Speed tracking	Chau and Chung (2002)	5-layer	1	–	Bell	2	–	Bell	21	COA	Backprop.	On	–	TMS320F240
2003		Chau et al. (2003)		–	–		–	–							
2004	Motor drive	Akcayol (2004)*	ANFIS	2	5–5	Bell	1	–	–	25	ANFIS	Gradient +least square	On	–	TMS320F240
					16 bit	FxP									
2007	Motor drive	Uddin and Wen (2007)	5-layer	2	3–3	Gaussian	1	No	–	–	–	Backprop. (on-line)	On		TMS320F240
					64 bit	FP		–							
2008	Motor drive	Elmas et al. (2008)*	4-layer	2	–	Gaussian	1	–	–	–	COA	Gradient (off-line)	On	–	TMS320F240
				–	–		–	–							
2011	Motor drive	Choi and Jung (2011)*	T–S	2	–	Gaussian	1	–	–	–	Weighted average	LMI	On	–	TMS320F28335
				12 bit	–		12 bit	–							
2012	Speech recognition	Ekhtiyar et al. (2012)*	MLP, ANFIS	13	–	Gaussian	1	–	–	48	–	Hybrid	On	–	DM6437-EVM
					16 bit	FxP									

- Choi and Jung (2011) in the article "Takagi–Sugeno fuzzy speed controller design for a permanent magnet synchronous motor" conclude that in terms of linear matrix inequalities (LMIs), sufficient conditions are given for the existence of a Takagi–Sugeno Controller. Choi and Jung highlight that this method provides a systematic approach to stability analysis and NFC design. The inputs are the rotor angular speed (ω – obtained by means of the rotor position (θ)) and the acceleration ($\dot{\omega}$) obtained by the technique named fuzzy acceleration observer (in terms of linear matrix inequalities (LMIs)). One experiment is performed comparing the results between the NFC and a Proportional–Integral (PI) plus nonlinear compensation controller. The load torque is 1 N m and the speed increases from 125.7 rad/s to 251.3 rad/s and then decreases from 251.3 rad/s to 125.7 rad/s. The results show that the NFC presents a minimal overshoot in the speed response as a zero steady-state error, and the speed error may vary up to about 8%. The PI presents a big overshoot and a long settling time and the speed error changes by more than 15% during the transient. Simulations were performed using Matlab/Simulink and the experiments were carried out using a TMS320F28335 floating point DSP controlling a motor with 12 poles and 1 HP.
- Ekhtiyar et al. (2012) in the paper "Model based neuro–fuzzy ASR on Texas processor" presented an algorithm for speech recognition, performing two kinds of classifiers, MLP and ANFIS. The algorithm is based on the algorithm MFCC (Mel-Frequency Cepstral Coefficients) which employs the Discrete Fourier Transform (DFT). The DSP is the DM6437-EVM of Texas Instrument.

Programmable Logic Controllers: Table 45 shows the detected reference.

- In 1994, Siemens provided neuro–fuzzy control to their Programmable Logic Controllers (PLCs), "Fuzzy control and neural networks industrial applications in the world of PLCs" (Wegmann, 1994), for a wide variety of applications. An initial interest was focused in environmental applications.

6.3. Mixed neuro–fuzzy implementations

In this subsection the same observations made previously apply.

Please cite this article as: Bosque, G., et al., Fuzzy systems, neural networks and neuro-fuzzy systems: A vision on their hardware implementation and platforms over two decades. Eng. Appl. Artif. Intel. (2014), http://dx.doi.org/10.1016/j.engappai.2014.02.008

Table 45
HW implementations of neuro–fuzzy systems – Programmable Logic Controller.

Year	Applic.	Ref.	Type of NFS	In	No. of in MFs	Type of in MFs	Out	No. of out MFs	Type of out MFs	No. of rules	Defuzz.	Learning	On/off chip	Speed	Device
1994	Industrial control	Wegmann (1994)*	–	–	–	–	–	–	–	–	–	Backprop.			SIMATIC

Table 46
HW implementations of neuro–fuzzy systems – mixed (1)

Year	Applications	Ref.	Type of NFS	In	No. of In MFs	Type of In MFs	Out	No. of Out MFs	Type of Out MFs	No. of Rules	Defuzz.	Learning	On/Off Chip	Speed	Technology
1993	Home	Wakami et al. (1993)*	IF-THEN	3	3-3-3	Triang.	1	-	-	27	COG	Gradient	On	-	-
				8	Variable	Variable	1			9		descent			
				8 bit	-		-	-							
1994	Robotic	Reyneri et al. (1994)	CPWM(a)	*n*	-	-	*m*	-	-	-	-	Several	On/Off	140 MCPS	CMOS 1.5 μm (CINTIA)
						8 bit						methods			
1994	Speech	Han (1994)	-	64	8-..-8	Several	32	-	-	Synapses	-	-		10 μs	CMOS 1.2 μm (URAN-I)
	recognition			6 bit	Analog		6 bit			135,424					
1995	General	Chiaberge et al. (1995)*	MLP,	32		-	any	32	1056	-		-	-	140 MCPS	CMOS 1.5 μm
			RBF,	Analog or CPWM	Analog	-	-		-						
			WRBF(b)												
1997	General	Han (1997)*	-	64	-	Arbitrary	32	-	-	-	-	-	-	> 100 GCPS	CMOS 0.8 μm
1998	Buck	Gomariz et al. (1998)	ANFIS	3	2-2-2	Triang.	1	-	-	8	COG	Hybrid	Off	500 ns	CMOS 0.8 μm
	converter			2	2-2		1	-	-	4					
						6 bit									
1999	General	Wilamowski et al. (1999)*	T-S	2	-	Trapez.,	1	-	-	64	Weighted	-	-	-	MOSIS 2 μm
			(modified)			Gaussian					sum				
					Analog			6 bit							
1999	General	Sadati and Mohseni (1999)	IF-THEN	4	-	Trapez.	1	-	-	$30..10^2$	COG	-	Off	300 ns	CMOS 1.2 μm
						Analog				$10^2..10^3$	(ANN: MLP)				

(a) Coherent Pulse Width Modulation / (b) Weighted Radial Basis Function.

6.3.1. *Dedicated Integrated Circuits*

There are some interesting approaches such as those shown in Tables 46 and 47.

Thee following will be highlighted (*):

- The company Matsusita, in 1993, performed neuro–fuzzy developments in home-oriented electrical applications in the article "Recent applications of fuzzy logic to home appliances" (Wakami et al., 1993), for example, refrigerators, air conditioning, etc. In air conditioning, control neural networks and fuzzy rules provided by experts are used. There are heat sensors which attend a fuzzy segmentation algorithm for a thermal image compression. This allows the identification of the number of persons and situation in a room to create a more comfortable environment. The article does not mention the technology and the integration scale of the sensor that performs the application.
- Chiaberge et al. (1995) presented the article "A pulse stream system for low-power neuro–fuzzy computation" where a VLSI neuro–fuzzy processor (known as AMINAH) working with the Coherent Pulse Width Modulation (CPWM) technique (Reyneri et al., 1993) is introduced. There is a conversion from analog inputs to CPWM signals (A/CPWM), and back to analog outputs (CPW/A). This system can operate with three different cases of the Weighted Radial Basis Functions (WRBFs) algorithm. The structure of the neural network chip is based on a synaptic array of 1056 synapses (32×32, plus 32 thresholds) with 32 common inputs, and 32 neurons. Neurons are programmable, therefore the user can vary network topology, as well as the shape and steepness of the transfer function. The new synapses have an unlimited weight retention time (due to self-refresh). The article highlights that this mixed HW approach simplifies the design of synapses and neurons and provides advantages over fully analog and fully digital implementations (high noise immunity, higher accuracy, low power dissipation, etc.). The comparison with other techniques (the same CMOS technology and the same channel size) is carried out in Reyneri (1995), these are PWM, PRM, SPM, analog and digital. Among the different factors (synapses size, MCPS, computation energy, response time and power dissipation), the article highlights the "response time" which is 5–10 μs compared with 50–200 μs, > 100 μs, 20–100 μs, 2–50 μs respectively (the digital implementation does not present this factor). The chip computes about 140 MCPS with an average power dissipation of about 10 mW, at a frequency $f_o \approx 139$ kHz (clock frequency). The chip is manufactured using a 1.5 μm CMOS digital technology.

Please cite this article as: Bosque, G., et al., Fuzzy systems, neural networks and neuro-fuzzy systems: A vision on their hardware implementation and platforms over two decades. Eng. Appl. Artif. Intel. (2014), http://dx.doi.org/10.1016/j.engappai.2014.02.008

Table 47
HW implementations of neuro–fuzzy systems – mixed (2).

Year	Applications	Ref.	Type of NFS	In	No. of in MFs	Type of in MFs	Out	No. of out MFs	Type of out MFs	No. of rules	Defuzz.	Learning	On/off chip	Speed	Technology
1999	Switching power conv.	Alarcon et al. (1999)	T–S	3	2–2–2	Bell-shaped	1	8	Singleton	8	COG	Hybrid	On	–	CMOS
				Analog	6 bit		Analog	6 bit							0.8 μm
2003	DC–DC (PWM)	Navas-Gonzalez et al. (2033)*	T–S	2	2–2	S- and Z-shaped	1	4	Singleton	4	Interpolated	–	Off	500 ns	CMOS
					8–8		1	64		64 Select: bit					0.7 μm
						Analog									
2009	Object detection	Kim et al. (2009)	–	8	–	Gaussian: one-, two-dimensional (Digital weights)	1	–	–	–	Comparator	Run-time optimization	On	22.9 GOPS (objects detected)	CMOS 0.13 μm
					Analog										
2010	Object recognition	Oh et al. (2010)*	VANFIS[a]	3	3–3–3	Trapez., triang. Gaussian, etc.	1	–	–	27	–	Adap. Pertur. Algorith. (on-line)	On	1 MFLIPS	CMOS
														54 MCUPS	0.13 μm
						Analog						(Digital: 16 bit)			
2011	General	Daneshwar et al. (2011)*	ANFIS	2	4–4	Trapez., triang. Gaussian	1	16	Singleton	16	COA	Hybrid (on-line)	On	18.18 MFLIPS	CMOS 0.35 μm (Simul. Hspice)
					Analog		4 bit	Analog							
2011	Object detection	Oh et al. (2011)	MLP, RBFN, RNN	–	–	–	–	–	–	–	–	Global/local learning accelerator	–	49.14 GOPS	CMOS 0.13 μm (IRIS)
						Analog						(Digital: 32 bit)			
2013	Object recognition	Oh and Yoo (2013)*	VANFIS	3	2–2–2	Bell	1	8	Singleton	8	T–S	Parameter perturbation	ANFIS	1 MFLIPS	CMOS
				3	3–3–3			27		27					0.13 μm
						Analog						(Digital: 12 bit FxP)			

[a] Versatile adaptive neuro–fuzzy inference system.

- Han (1997) presented the work "Hardware implementation of neuro–fuzzy system with the analogue–digital hybrid neural chip", where he proposed circuits based on mixed analog and digital operations, including a linear voltage-controlled MOSFET resistance working either linearly or by pulse operation. The membership functions are obtained by means of a pulse processing and analog operation. Any numbers of membership functions can be evaluated in parallel during a single cycle. The VLSI is fully programmable and achieves hundreds of GCPS (Giga Connections per Second). The name adopted for the circuit is URAN-I and has been built in 0.8 μm CMOS technology.
- Wilamowski et al. (1999) in the article "neuro–fuzzy architecture for CMOS implementation" implemented an inference system of modified Takagi–Sugeno. He incorporates the phases of fuzzification-normalization over the 2 inputs and output normalization. The membership functions can be trapezoids or Gaussians, and admit 8 membership functions per variable. These functions are arranged so that only 4 rules are active for each input variable. The total number of rules is 64 and their activation level is set digitally by means of 6 bits acting on an array of transistors operating in current-mirror mode. The circuit has been built in VLSI technology 2 μm MOSIS.
- Navas-Gonzalez et al. (2003) in the article "neuro–fuzzy chip to handle complex tasks with analog performance" showed and implemented a strategy that exploits the local feature of ANNs to preserve the advantages of analog implementations such as short delay, power consumption and area. Since ANNs provide the output from just a few sets of nodes, these nodes have been implemented in an analog core and make it dynamically programmable to compute the output for any input vector. The circuit identifies the inputs and performs an input space partition with a set of A/D converters. These converters perform a coarse clustering to get the set of basic functions that determine the output. This means that only simple A/D converters (as low as 3-bit) are needed for every input dimension. The output of the converters is used to address a data base which stores the programming data for the analog core. The proposed controller is based on a zero-order Takagi–Sugeno fuzzy system with two inputs and one output with the advantage of providing two rules per input. The controller is named MFCON. The chip was simulated with HSPICE and designed with Design Framework II. The MFCON controller prototype has been integrated in a single-poly, double-metal CMOS 0.7 μm technology implementing 64 rules and the in/out delay is 500 ns.
- Oh et al. (2010) presented in the work "A 1.2 mW on-line learning mixed mode intelligent inference engine for robust object recognition", with 94% average classification accuracy within 1 μs operation. This circuit provides 5 distinctive shapes of function (such as Gaussian, trapezoidal, triangular, s-shape and z-shape) by combining three parameters. With an inference accelerator running at 200 MHz, the analog VANFIS (Versatile Adaptive Neuro–Fuzzy Inference System) core is measured to have a $\sim$1 μs operation speed, which achieves 1 MFLIPS and maximum 54 MCUPS as a neuro–fuzzy system. It is implemented in 0.13 μm CMOS process and achieves 1.2 mW power consumption.
- Daneshwar et al. (2011) in the article "Hardware implementation of an adaptive mixed signal neuro fuzzy system using high speed and low power analog CMOS circuits" highlighted, as in other studies, the advantages of analog implementations (high speed, simplicity, etc.) and proposed a new programmable fuzzifier circuit. This was based on mixed-signal input, a *min* circuit for inference and a multiplier/divider circuit for a defuzzifier block and a current-mode high speed flash analog to digital (A/D) converter with 4 bit resolution. The fuzzifier circuit can obtain Gaussian, trapezoidal and triangular membership functions and the neuro–fuzzy system is an ANFIS. The defuzzification strategy is COA applied to the singletons (weighted average of singleton). The chip has been designed and implemented using HSPICE. The inference speed of the system is about 18.18 MFLIPS. The controller simulated in 0.35 μm CMOS standard technology.
- Oh and Yoo (2013) "1.2 mW online learning mixed-mode intelligent inference engine for low-power real-time object recognition processor" presented a mixed-mode hardware implementation of the Versatile Adaptive Neuro–Fuzzy Inference System (VANFIS). The system is mainly composed of a current-mode analog VANFIS core, the digital learning controller, and analog-to-digital converter (ADC) and digital-to-analog converter (DAC) arrays. The circuit is implemented in 0.13 μm CMOS process. It achieves 94% classification accuracy for the test database with a performance of 1 million fuzzy inferences per second (MFLIPS), and consumes only 1.2 mW for average power and 5.3 mW for peak power.

6.3.2. Programmable Integrated Circuits

Undetected.

6.4. Conclusions and complementary readings of HW implementations for neuro–fuzzy systems

In general, both analog and mixed line present a small number of contributions. Digital line is where there is the largest number of contributions and shows the trend in the evolution of future developments.

Regarding the specific designs, in analog devices, the inputs are in voltage and the outputs in current, implementing the membership functions with blocks working in transconductance. The learning algorithms are based on different lines from those which are oriented to the derivative (e.g. hybrid or perturbation of the weights). In mixed designs, some industrial approaches applied to both the home and industry can be observed. Mixed designs provide, among other benefits, reduced consumption and increased speed. Where the comparisons are shown, the results match or improve in one or more orders of magnitude to the other techniques. The treatment of the weights and the rules are made both analog and digital. In the first case, the weights are stored on arrays of transistors working in current mirrors, digitally programmed and in the second, on condensers sequentially refreshed.

Both in analog and mixed design, where it is explicit, the chosen input membership functions are, mainly, triangular or trapezoidal. Others are Gaussian or bell. The output membership functions are, mainly, singleton.

Regarding dedicated devices, Table 48 shows a summary of the main data extracted from the works consulted: analog and mixed circuits (digital works are not detected).

Regarding the programmable devices, in the digital line, there is a wide use of FPGAs. From the point of view of learning, some implementations performed the learning off-chip, loading the parameters to the HW of the FPGA, and others performed the learning on-chip, either on their own hardware or on a μP (hard-core or soft-core) block. In this case, the learning can be either off-line or on-line, with the hybrid and the backpropagation being the most common methods. However other learning methods such as "division-free" are also introduced. Regarding the input membership functions, there is a trend to use of the triangular for the simplicity introduced into the calculus, but Gaussian or trapezoid membership functions are not discarded. The output membership

Please cite this article as: Bosque, G., et al., Fuzzy systems, neural networks and neuro-fuzzy systems: A vision on their hardware implementation and platforms over two decades. Eng. Appl. Artif. Intel. (2014), http://dx.doi.org/10.1016/j.engappai.2014.02.008

Table 48
Summary of the main features of neuro–fuzzy systems on analog and mixed-dedicated circuits.

Circuit	Type of MFs[a]		Defuz.[a]	Learn[a]	Speed			Power	Tech. μm
	In	Out			FLIPS	CPS	Time	mW	CMOS
Analog	Triang, trapez., Gaussian, bell	Singleton	T–S, COG, ANFIS	Weights perturbation, stochastic gradient	5M	–	0.2 μs..1 μs	1, 613	0.18, 0.35 .. 1.6
Mixed	Triang, trapez., S, Z-shaped Gaussian, bell	Singleton triang., Gaussian	COG, COA, weighted sum, interpol. comp., T–S	Hybrid, gradient descent, adapt. perturbation, param. perturbation, etc.	1M	140M, 100G	300 ns, 500 ns, 10 μs	1.2, 2.8, 13.5..20	0.13, 0.35, 0.7..2

[a] In order to the preferences of the articles consulted.

Table 49
Summary of the main features of neuro–fuzzy systems on Digital Programmable Circuits.

Device	Type	Type of MFs[a]		Defuz.[a]	Learn[a]	Speed		
		In	Out			FLIPS	Time	Freq. (Hz)
FPGA		Triang, trapez., Gaussian, bell	Singleton linear, Mamdani	ANFIS, T–S, weights or sum or central average, sum of fire rules	Hybrid, backprop., gradient descent, weights or simultaneous perturbat., division-free, Kalman filter, divide and conquer	2.9M, 67M, 69.5M	0.2 μs..1 μs	–
Commercial processor	General purpose	Triang, trapez.	Singleton, triang., trapez.	Minimum inference	Mean quadratic, ART, ARTMAP, Matlab	–	–	–
	DSP	Triang., trapez., Gaussian, bell	Bell	COA, ANFIS, weights average normaliz.+ vector adder	Hybrid, backprop., gradient descent, LMI	–	7.36 μs..10 μs 13.2 ms..28.9 ms	192.31k, 12.6M 37.34M..100M
	PLC	–	–	–	Backpropagation	–	–	–

[a] In order to the preferences of the articles consulted.

functions are, mainly, singleton. Curiously, there is a development system that implements a VHDL code generation and code (bit-stream) for FPGAs. Table 49 shows the main data extracted from the works consulted: digital programmable devices.

Finally, to conclude this review of papers on neuro–fuzzy implementations, it can be said that the FPGAs offer better performance for their versatility, integration level and speed. The inclusion of a processor on the FPGA gives it greater flexibility to implement solutions that incorporate some type of SW treatment on neuro–fuzzy networks. The areas of analog design and mixed design seem to be discontinued in favor of the consolidation of the reconfigurable digital technology.

With regard to complementary readings, several papers and books have addressed the remarkable evolution that has taken place in the neuro–fuzzy world. Within the literature produced, can be highlighted the work of Tsoukalas and Uhrig (1997), "Fuzzy and neural approaches in engineering", where, in addition to developing theoretical aspects, there is also a review of method implementations in ANNs fuzzy (fuzzy neurons) and neuronal methods in fuzzy systems. From a general point of view, Liao (2005) in the article "Expert system methodologies and applications" gives an overview of expert systems applications from 1995 to 2004 based on 166 articles that include ANN and FIS applications.

7. Conclusions

This section is an overview of the different summaries previously viewed in Sections 4.4, 5.4 and 6.4. The highlights of the different topics of soft computing are presented and, in order to facilitate reading, have omitted the tables.

Fuzzy systems: Fuzzy inference systems have been implemented on several platforms, differing between analog, digital or mixed-on commercial or dedicated circuit implementations. Let us highlight the conclusions reflected in Section 4.4.

The development of fuzzy systems on a dedicated analog HW have been implemented to work in either current, voltage, transconductance or switched mode. The differences lie in speed, required power, size, integration, stability, accuracy or conformation of the membership functions, among others. It should be noted that the commercial appearance of the FPAAs in the 2000s, opened a new scenario in the analog implementation of fuzzy systems although this is yet to be consolidated.

Digital implementations on a dedicated HW have brought greater immunity to external factors than analog implementations. The speed has been achieved by implementing parallel rules and optimizing sequential rules. In systems with parallel rules, the membership functions are stored in memory. This grows when the accuracy increases so it tends to give low accuracy systems. In sequential implementations, both the membership functions and the rules are stored in memory. By increasing the number of inputs, the memory grows exponentially, and therefore techniques have been proposed to minimize this effect. The advent of programmable circuits, such as FPGAs, have opened up a wide range of options for implementing the rules, membership functions or reduction in defuzzification HW.

Mixed systems present their inferences on the analog area, and programming and parameters on the digital area. Architectures have been performed mainly with parallel rules.

Please cite this article as: Bosque, G., et al., Fuzzy systems, neural networks and neuro-fuzzy systems: A vision on their hardware implementation and platforms over two decades. Eng. Appl. Artif. Intel. (2014), http://dx.doi.org/10.1016/j.engappai.2014.02.008

As a final comment on dedicated fuzzy system implementations, the consolidation of FPGAs in recent years has been mainly due to their reconfiguration capability and increasing their density. These factors have caused both analog or digital dedicated designs to have been almost discontinued. The few implementations that are still done on dedicated devices present an interest which is mainly academic.

The use of processors in systems that do not require high speed is the most economical alternative regarding dedicated or configurable devices (FPGAs). The introduction of fuzzy instructions in commercial processors or fuzzy design coprocessors significantly increases the processing speed. However, it is noted that commercial firms have mostly stopped the manufacture of processors with fuzzy instructions. With respect to coprocessors, there is high speed processing of fuzzy rules. Some PLC manufacturers have adopted the use of fuzzy coprocessors in systems oriented to industrial control.

With respect to the membership functions in all of these systems, the triangular is mainly the trend for the inputs, and for the outputs, mainly triangular and singleton.

Neural networks: Neural networks present a similar analysis. The conclusions presented in Section 5.4 show the same trends as fuzzy inference systems. The development of neural networks on an analog HW has happened as a result of various factors, these are speed of adders and multipliers for the low number of transistors required and a final important reason that these architectures can keep the parallelism inherent in neural networks. As regards the learning algorithms, two trends are shown: learning on-circuit and off-circuit. In the latter case, this is in order to increase the speed of the network. It must be noted, as in the case of fuzzy systems, the emergence of the FPAA in the 2000s to which researchers begin to pay attention.

In the last decade, the increase in the integration density, has modified the design trend targeting digital devices. Dedicated processors have been discontinued, tending to techniques which are more structured, such as those employed in the development of the ASICs. Here a small area for integration is required, as well as a greater tendency to use the FPGA, due to the high parallelism and their ability to reconfigure. Regarding learning algorithms, there are two trends: learning on-line and off-line. Some implementations perform on-chip learning. The algorithm used is pure or modified backpropagation. The off-line learning is performed off-chip and weights are loaded into the FPGA.

Some contribution on DSPs has been observed but is the FPGAs that show the tendency to continue in implementing neural networks.

Regarding mixed devices, generic devices have been developed where the weights are stored on a digital memory, requiring D/A converters for processing. There are also implementations on FPGAs incorporating A/D and D/A. Single-chip or multi-chip developments are tending towards high performance computing structures.

Ii is worth highlighting the implementation on ASIC devices when the size of the circuit needs to be minimal. This is aimed at industrial applications.

With respect to the activation functions in all of these systems, the trend is mainly the sigmoid.

Neuro–fuzzy systems: With regard to the neuro–fuzzy systems, the conclusions reflected in Section 6.4 show that these systems require the same treatment as the two previous systems.

In analog devices, the inputs are in voltage and the outputs in current, implementing the membership functions with blocks working in transconductance. The learning algorithms are based on different lines of those oriented to the derivative.

In the digital line, there is a wide use of FPGAs. From the point of view of learning, some implementations performed the learning off-chip, loading the parameters to the HW of the FPGA, and others performed the learning on-chip, either on own hardware or on a μP (hard-core or soft-core) block. In this case the learning can be off-line or on-line, with the hybrid and the backpropagation being the most common methods, but other learning methods such as "division-free" have also been introduced. Regarding the membership functions, there is a trend to the use of the triangular due to the simplicity introduced into the calculus but Gaussian or trapezoid membership functions have not been discarded. In these works, as a singularity, there is a development system that implements a VHDL code generation and code for FPGAs.

Regarding mixed designs, some industrial approaches can be observed, applied to both the home and industry. Mixed designs provide, among other benefits, reduced consumption and increased speed. Where comparisons are shown, the mixed design matches or improves the other techniques in one or more orders of magnitude. The treatment of the weights and the rules are made both by analogue and digital form. In the first case the weights are stored on arrays of transistors working in current mirror, digitally programmed and in the second on condensers sequentially refreshed. Where it is explicit, the chosen membership functions are trapezoidal or Gaussian.

As in the previous systems, FPGAs are what make the trend in the evolution of neuro–fuzzy systems implementations.

With respect to the membership functions in all of these systems, as in the fuzzy systems, the triangular is mainly the trend for the inputs, and for the outputs, mainly triangular and singleton.

General: The implementation on FPGAs of the previous paradigms of soft computing systems, presents a common design style which is the use of hardware description languages (HDL), with the most used being the VHDL, followed by Verilog HDL. One of the generalized interests is the low " time to market" presented by FPGAs. Similar considerations also apply to the FPAAs when its presence in new implementations has been consolidated.

As a final comment, the emergence and consolidation of FPAA devices in the previous decade opens a scenario for future implementations on analog devices. An added advantage to this technology is that it makes the A/D and D/A unnecessary. Currently, the low density of FPAAs and/or the non-inclusion of processors does not allow implementations such as some of those reviewed in this paper. At this point it is interesting to highlight the appearance of a new analog processor based on FPAAs (Fu et al., 2010) presented at the 10th IEEE International Conference on Solid-State and Integrated Circuit Technology (ICSICT) in November 2010.

References

Abou, S., 2011. Fuzzy-logic-based network for complex systems risk assessment: application to ship performance analysis. Accid. Anal. Prev. 45, 305–316.

Abramson, D., Smith, K., Duke, D., 1998. FPGA based implementation of a Hopfield neural network for solving constraint satisfaction problems. In: Euromicro Conference, vol. 2. IEEE Xplore, pp. 688–693.

Achronix. URL: ⟨www.achronix.com⟩.

Actel. URL: ⟨www.actel.com⟩.

Actel. URL: ⟨www.actel.com/products/mpu/coremp7/⟩.

Actel-ARM. URL: ⟨www.actel.com/products/mpu/CortexM1/⟩.

Akcayol, M., 2004. Application of adaptive neuro–fuzzy controller for SRM. Adv. Eng. Softw. 35, 129–137.

Alarcon, E., Madrenas, J., Moreno, J., Cosp, J., Gomariz, S., Guinjoan, F., Poveda, A., 1999. Mixed-signal implementation of a discrete-time sequential takagi–Sugeno neuro–fuzzy controller. In: 6th International Conference on Mixed-Signal Design of Integrated Circuits and Systems (MIXDES99), pp. 389–395.

Aldair, A., Wang, W., 2010. FPGA based adaptive neuro fuzzy inference controller for full vehicle nonlinear active suspension systems. Int. J. Artif. Intell. Appl. 1 (4), 1–15.

Aliev, R., 2008. Modelling and stability analysis in fuzzy economics. Appl. Comput. Math. 7 (1), 31–53.

Altera. URL: ⟨www.altera.com⟩.

Please cite this article as: Bosque, G., et al., Fuzzy systems, neural networks and neuro-fuzzy systems: A vision on their hardware implementation and platforms over two decades. Eng. Appl. Artif. Intel. (2014), http://dx.doi.org/10.1016/j.engappai.2014.02.008

Altera, 2004. Nios 3.0 CPU (v2.2). Technical Report. Altera, October.
Aluja, J., 2004. Fuzzy Sets in the Management of Uncertainty. Studies in Fuzzyness and Soft Computing, vol. 145. Springer-Verlag.
Amirkhanzadeh, R., Khoei, A., Hadidi, K., 2005. A mixed-signal current-mode fuzzy logic controller. Int. J. Electron. Commun. (AEÜ) 59, 177–184.
Anadigm. URL: ⟨http://www.anadigm.com/fpaa.asp⟩.
Arati, M., Singh, H., Meitzler, T., 2010. Soft computing approach to crack detection and FPGA implementation. Mater. Eval. 68 (11), 1263–1272.
Arima, Y., Mashito, K., Okada, K., Yamada, T., Maeda, A., Kondoh, H., Kayano, S., 1991a. A self-learning neural network chip with 125 neurons and 10k self-organization synapses. IEEE J. Solid-State Circuits 26 (4), 607–611.
Arima, Y., Mashito, K., Okada, K., Yamada, T., Maeda, A., Notani, H., Kondoh, H., Kayano, S., 1991b. A 336-neuron 28k-synapse self-learning neural network chip with branch-neuron-unit-architecture. IEEE J. Solid-State Circuits 26 (11), 1637–1644.
Arima, Y., Murasaki, M., Yamada, T., Maeda, A., Shinohara, H., 1992. A refreshable analog VLSI neural network chip with 400 neurons and 40k synapses. IEEE J. Solid-State Circuits 27 (12), 1854–1861.
ARM, 2001. Arm922t Technical Reference Manual (rev. 0). Technical Report. ARM Lim.
Ascia, G., Catania, V., 1997. A dedicated parallel processor for fuzzy computation. In: IEEE International Conference on Fuzzy Systems, vol. 2, pp. 787–792.
Atmel. URL: ⟨www.atmel.com⟩.
Attali, J.-G., Pagès, G., 1997. Approximations of functions by a multilayer perceptron: a new approach. Neural Netw. 10 (6), 1069–1081.
Avago. URL: ⟨www.avagotech.com⟩.
Carpenter, G.A., Grossberg, S. (Eds.), 1992. Neural Networks for Vision and Image Processing. The MIT Press.
Chen, H., Fuller, S.S., Friedman, C., Hersh, W. (Eds.), 2010. Medical informatics: knowledge management and data mining in biomedicine. In: Integrated series in information systems. Springer.
Babuska, R., Verbruggen, H., 2003. Neuro–fuzzy methods for nonlinear systems identification. Annu. Rev. Control 27, 73–85.
Bal, G., Bekiroglu, E., Demirbas, S., Colak, I., 2004. Fuzzy logic based DSP controlled servo position control for ultrasonic motor. Energy Convers. Manag. 45, 3139–3153.
Barriga, A., Sanchez-Solano, S., Brox, P., Cabrera, A., Baturone, I., 2006. VHDL high level modelling and implementation of fuzzy systems. Fuzzy Log. Appl., 11–18
Basterretxea, K., del Campo, I., 2009. Electronic hardware for fuzzy computation. In: Laurent, A., Lessot, M.-J. (Eds.), Scalable Fuzzy Algorithms for Data Management and Analysis: Methods and Design, Information Science Reference, p. 27 (Chapter I).
Bastos, J., Figueroa, H., Monti, A., 2006. FPGA implementation of neural network-based controllers for power electronics applications. In: Applied Power Electronics Conference and Exposition (APEC 06). IEEE. pp. 1443–1448.
Bates, J.H.T., Young, M.P., 2003. Applying fuzzy logic to medical decision making in the intensive care unit. Am. J. Respir. Crit. Care Med. 167, 948–952.
Baturone, I., Barriga, A., Sanchez-Solano, S., 1994. Current-mode multiple-input max circuit. Electron. Lett. 30 (9), 678–680.
Baturone, I., Sanchez-Solano, S., Barriga, A., Huertas, J., 1997. Implementation of CMOS fuzzy controllers as mixed-signal integrated circuits. IEEE Trans. Fuzzy Syst. 5 (1), 1–19.
Baturone, I., Sánchez, S., Barriga, A., Jiménez, C., Senhadji, R., López, D., 2001. VLSI design of universal approximator neuro–fuzzy systems. In: XVI Conference on Design of Circuits and Integrated Systems (DCIS2001), pp. 358–363.
Bekey, G.A., Goldberg, K.Y. (Eds.), 1993. Neural Networks in Robotics, vol. 202. pp. 580. Kluwer academic publishers.
Binfet, J., Wilamowski, B., 2001. Microprocessor implementation of fuzzy systems and neural networks. In: International Joint Conference on Neural Networks (IJCNN 01), vol. 1, pp. 234–239.
Blake, J., Maguire, L., McGinnity, T., Roche, B., McDaid, L., 1998. The implementation of fuzzy systems, neural networks and fuzzy neural networks using FPGAs. Inf. Sci. 112, 151–168.
Bona, B., Carabelli, S., Chiaberge, M., Miranda, E., Reyneri, L., 1997. A neuro–fuzzy core for DSP-based controls of complex systems. In: IEEE International Conference on Computational Cybernetics and Simulation, vol. 4.
Bona, B., Carabelli, S., Chiaberge, M., Miranda, E., Reyneri, L., 1998. Hybrid neuro–fuzzy system for control of complex plants. In: IEEE International Symposium on Industrial Electronics (ISIE98), vol. 1, pp. 87–92.
Boquete, L., Martin, P., Mazo, M., Garcia, R., Barea, R., Rodriguez, F., Fernandez, I., 2002. Hardware implementation of a new neurocontrol wheelchair-guidance system. Neurocomputing 47, 145–160.
Boser, B., Sackinger, E., Bromley, J., Le Cun, Y., Jackel, L., 1991. An analog neural network processor with programmable topology. IEEE J. Solid-State Circuits 26 (12), 2017–2025.
Bosque, G., del Campo, I., Echanobe, J., Tarela, J., 2004. Implementación de un sistema neuro–fuzzy sobre un sopc. In: Congreso Español sobre Tecnologías y Lógica Fuzzy (Estylf 2004). European Society for Fuzzy Logic and Technology (EUSFLAT), pp. 587–592.
Bosque, G., del Campo, I., Echanobe, J., 2006. Efficient hardware/software implementation of a neuro–fuzzy system on a SOPC. In: Recent Advanced in Soft Computing (RASC).
Bosque, G., del Campo, I., Echanobe, J., Tarela, J., 2008. Modelling and synthesis of computational efficient adaptive neuro–fuzzy systems based on matlab. In: Artificial Neural Networks—ICANN 2008. Lecture Notes in Computer Science 5164, vol. 2. Springer, pp. 131–140.
Bouras, S., Kotranakis, M., Suyama, K., Tsividis, Y., 1998. Mixed analog-digital fuzzy logic controller with continuous-amplitude fuzzy inferences and defuzzification. IEEE Trans. Fuzzy Syst. 6 (2), 202–215.
Buckley, J., 1993. Sugeno type controllers are universal controllers. Fuzzy Sets Syst. 53 (1), 299–303.
Buckley, J., Hayashi, I., 1994. Fuzzy neural networks. Fuzzy Sets Syst. 66 (1), 1–13.
Burr, J., 1995. Digital Neural Network Implementations. Department of Electrical Engineering, Stanford University, pp. 1–47.
Cabrera, A., Sánchez-Solano, S., Brox, P., Barriga, A., Senhadji, R., 2004. Hardware/software codesign of configurable fuzzy control systems. Appl. Soft Comput. 4, 271–285.
Cao, S., Rees, N., Feng, G., 2001. Mamdani-type fuzzy controllers are universal fuzzy approximators. IEEE Trans. Comput. 123 (3), 359–367.
Cao, Q., Lim, H., Li, J., Ong, Y., 2006. A context switchable fuzzy inference chip. IEEE Trans. Fuzzy Syst. 14 (4), 552–567.
Cardarilli, G., Re, M., Lojacono, R., 1998. Vlsi implementation of a real time fuzzy processor. J. Intell. Fuzzy Syst.: Appl. Eng. Technol. 6 (3), 389–401.
Cárdenas, A., Guzmán, C., Agbossou, K., 2012. Development of a FPGA based real-time power analysis and control for distributed generation interface. IEEE Trans. Power Syst. 27 (3), 1343–1353.
Card, H., Rosendahl, G., McNeill, D., McLeod, R., 1998. Competitive learning algorithms and neurocomputer architecture. IEEE Trans. Comput. 47 (8), 847–858.
Castaneda-Miranda, R., Ventura-Ramos, E., Peniche-Vera, R., Herrera-Ruiz, G., 2006. Fuzzy greenhouse climate control system based on a field programmable gate array. Biosyst. Eng. 94 (2), 165–177.
Castro, J., 1995. Fuzzy logic controllers are universal approximators. IEEE Trans. Syst. Man Cybern. 25 (4), 629–635.
Castro, J., Mantas, C., Benitez, J., 2000. Neural networks with continuous squashing function in the output are universal approximators. Neural Netw. 13, 561–563.
Çilingiroglu, U., Pamir, B., Günai, Z., Dülger, F., 1997. Sampled-analog implementation of application-specific fuzzy controllers. IEEE Trans. Fuzzy Syst. 5 (3), 431–442.
Chabaa, S., Zeroual, A., Antari, J., 2009. Application of adaptive neuro–fuzzy inference systems for analyzing non-Gaussian signal. In: International Conference on Multimedia Computing and Systems, ICMCS '09, pp. 377–380.
Chau, K., Chung, S., 2002. Neuro–fuzzy speed tracking control of traveling-wave ultrasonic motor drives using direct pulse width modulation. In: IEEE Industry Applications Conference—Conference Record of the 2002.
Chau, K., Chung, S., Chan, C., 2003. Neuro–fuzzy speed tracking control of traveling-wave ultrasonic motor drives using direct pulse width modulation. IEEE Trans. Ind. Appl. 39 (4), 1061–1069.
Chekired, F., Larbes, C., Rekioua, D., Haddad, F., 2011. Implementation of a MPPT fuzzy controller for photovoltaic systems on FPGA circuit. Energy Procedia 6, 541–549.
Chen, Y.-C., Teng, C.-C., 1995. A model reference control structure using a fuzzy neural network. Fuzzy Sets Syst. 73, 291–312.
Chiaberge, M., Sologuren, E.M., Reyneri, L.M., 1995. A pulse stream system for low-power neuro–fuzzy computation. IEEE Trans. Circuits Syst. I. Fundam. Theory Appl. 42 (11), 946–954.
Choi, H., Jung, J.-W., 2011. Takagi-sugeno fuzzy speed controller design for a permanent magnet synchronous motor. Mechatronics 21, 1317–1328.
Chou, H.-H., Kung, Y.-S., Quynhb, N., Cheng, S., 2013. Optimized FPGA design, verification and implementation of a neuro–fuzzy controller for PMSM drives. Math. Comput. Simul. 90, 28–44.
Chowdhury, S., Chakrabarti, D., Saha, H., 2008. FPGA realization of a smart processing system for clinical diagnostic applications using pipelined datapath architectures. Microprocess. Microsyst. 32, 107–120.
Comon, P., 1994. Independent component analysis—a new concept? Signal Process. 36 (3), 287–314.
Conti, M., Orcioni, S., Turchetti, C., 1997. Analog neuro–fuzzy network for system modeling and control. In: Symposium on Intelligent Systems (AMSE-ISIS 97).
Conti, M., Orcioni, S., Turchetti, C., 1998. A current-mode neuro–fuzzy network. In: IEEE International Conference on Electronics, Circuits and Systems, vol. 1, pp. 427–430.
Conti, M., Crippa, P., Orcioni, S., Turchetti, C., Catani, V., 2000. Fuzzy controller architecture using fuzzy partition membership functions. In: 4th International Conference on Knowledge-Based Intelligent Engineering Systems and Allied Technologies, vol. 2, pp. 864–867.
Cotter, N., 1990. The Stone–Weiertrass theorem and its application to neural networks. IEEE Trans. Neural Netw. 1 (4), 290–295.
Cypress. URL: ⟨www.cypress.com⟩.
Daneshwar, M., Aminifar, S., Yosefi, G., 2011. Hardware implementation of an adaptive mixed signal neuro fuzzy system using high speed and low power analog CMOS circuits. J. Basic Appl. Sci. Res. 1 (10), 1415–1422.
D'Amore, R., Saotome, O., Kienitz, K., 2001. A two-input, one-output bit-scalable architecture for fuzzy processors. IEEE Des. Test Comput. 18 (4), 56–64.
del Campo, I., Tarela, J., 1999. Consequences of the discretization on the performance of a fuzzy logic controller. IEEE Trans. Fuzzy Syst. 7 (February (1)), 85–92.
del Campo, I., Tarela, J., Basterretxea, K., 2001. Digital Controller Implementation and Fragility. A Modern Perspective: Quantisation Errors in Digital Implementation of Fuzzy Controllers. Springer, pp. 253–274
del Campo, I., Basterretxea, K., Echanobe, J., Bosque, G., Doctor, F., 2012. A system-on-chip development of a neuro–fuzzy embedded agent for ambient-

Please cite this article as: Bosque, G., et al., Fuzzy systems, neural networks and neuro-fuzzy systems: A vision on their hardware implementation and platforms over two decades. Eng. Appl. Artif. Intel. (2014), http://dx.doi.org/10.1016/j.engappai.2014.02.008

intelligence environments. IEEE Trans. Syst. Man Cybern. Part B: Cybern. 42 (2), 501–512.
Dettloff, W., Yount, K., Watanabe, H., 1989. A VLSI fuzzy logic inference engine for real-time process control. In: Custom Integrated Circuits Conference, IEEE. pp. 12.4/1–12.4/5.
Dick, S., Gaudet, V., Bai, H., 2008. Bit-serial arithmetic: a novel approach to fuzzy hardware implementation. In: Fuzzy Information Processing Society, Annual Meeting of the North American. pp. 1–6.
D-Mello, D.R., Gulak, P.G., 1998. Design approaches to field-programmable analog integrated circuits. Analog Integr. Circuits Signal Process. 17, 7–34.
do Amaral, J.M., do Amaral, J., Santini, C., Tanscheit, R., Vellasco, M., Pacheco, M., 2002. Towards evolvable analog fuzzy logic controllers. In: NASA/DOD Conference on Evolvable Hardware (EH 02), pp. 123–128.
Dompere, K., 2004. Cost-Benefit Analysis and the Theory of Fuzzy Decisions: Identification and Measurement Theory, Studies in Fuzziness and Soft Computing, vol. 158. Springer-Verlag
Dongale, T., Kulkarni, T., Jadhav, S.R., Kulkarni, S., Mudholkar, R., 2012. AC induction motor control—a neuro–fuzzy approach. Int. J. Eng. Sci. Adv. Technol. 2 (4), 863–870.
Dong, P., Bilbro, G., Chow, M.-Y., 2006. Implementation of artificial neural network for real time applications using field programmable analog arrays. In: International Joint Conference on Neural Networks (IJCNN 2006), IEEE.
Draghici, S., 2000. Neural networks in analog hardware—design and implementation issues. Int. J. Neural Syst., 1–22.
Driankov, D., Hellendoorn, H., Reinfrank, M., 1996. An Introduction to Fuzzy Control, 2nd edition.
Echanobe, J., del Campo, I., Bosque, G., 2008. An adaptive neuro–fuzzy system for efficient implementations. Inf. Sci. 178 (May (9)), 2150–2162.
Echevarria, P., Martinez, M., Echanobe, J., del Campo, I., Tarela, J., 2007. Digital hardware implementation of high dimensional fuzzy systems. Appl. Fuzzy Sets Theory, 245–252
Egmont-Petersen, M., de Ridder, D., Handels, H., 2002. Image processing with neural networks—a review. Pattern Recognit. 35, 2279–2301.
Eichfeld, H., Löhner, M., Müller, M., 1992. Architecture of a CMOS fuzzy logic controller with optimized organisation and operator design. In: Proceedings of 1st International Conference on Fuzzy Systems, FUZ-IEEE. IEEE Computer Society Press, pp. 1317–1323.
Eichfeld, H., Kunemund, T., Klimke, M., 1993. An 8b fuzzy coprocessor for fuzzy control. In: IEEE International Conference Solid-State Circuits, pp. 180–181.
Eichfeld, H., Klimke, M., Menke, M., Nolles, J., Künemund, T., 1995. A general-purpose fuzzy inference processor. IEEE Micro 15 (3), 12–17.
Eichfeld, H., Kiinemund, T., Menke, M., 1996. A 12b general-purpose fuzzy logic controller chip. IEEE Trans. Fuzzy Syst. 4 (4), 460–475.
Eichfeld, H., Mertens, A., Brandmeier, T., Waidelich, F., Graumann, R., Schwab, M., 1998. Applications of SAE 81c99x fuzzy coprocessors. In: IEEE World Congress on Computational Intelligence, vol. 1, pp. 19–24.
Ekhtiyar, H., Sheida, M., Moghadam, S., 2012. Model based neuro–fuzzy ASR on Texas processor. Int. J. Comput. Sci. Issues 9 (1), 371–374.
Eldredge, J., Hutchings, B., 1994. Density enhancement of a neural network using FPGAs and run-time reconfiguration. In: FPGAs for Custom Computing Machines. IEEE, pp. 180–188.
Elmas, C., Ustun, O., Sayan, H., 2008. A neuro–fuzzy controller for speed control of a permanent magnet synchronous motor drive. Expert Syst. Appl. 34 (1), 657–664.
Elmos. URL: ⟨www.elmos.de/english⟩.
Erkmen, N.K.B., Vural, R.A., Yildirim, T., 2013. A mixed mode neural network circuitry for object recognition application. Circuits Syst. Signal Process. 32, 29–46.
Evmorfopoulos, N., Avaritsiotis, J., 2002. An adaptive digital fuzzy architecture for application-specific integrated circuits. Act. Passiv. Electron. Compon. 25, 289–306.
Falchieri, D., Gabrielli, A., Gandolfi, E., 2002. Very fast rate 2-input fuzzy processor for high energy physics. Fuzzy Sets Syst. 132, 261–272.
Fan, Z.-C., Hwang, W.-J., 2013. Efficient VLSI architecture for training radial basis function networks. Sensors 13, 3848–3877.
Fattaruso, J., Mahant-Shetti, S., Barton, J., 1994. A fuzzy logic inference processor. IEEE J. Solid-State Circuits 29 (4), 397–402.
Feng, H.-M., Chen, C.-Y., Horng, J.-H., 2010. Intelligent omni-directional vision-based mobile robot fuzzy systems design and implementation. Expert Syst. Appl. 37, 4009–4019.
Ferreira, P., Ribeiro, P., Antunes, A., Morgado, F., 2007. A high bit resolution FPGA implementation of a FNN with a new algorithm for the activation function. Neurocomputing, 71–77
Frías-Martínez, E., Fernández-Hernández, J.G.-R.F., 2004. Efficient fuzzy compiler for SIMD architectures. Appl. Soft Comput. 4 (3), 287–301.
Fu, W., Jiang, J., Qin, X., Yang, S., Yi, T., Hong, Z., 2010. A large-scale reconfigurable analog processor based on field programmable analog array technology. In: 10th IEEE International Conference on Solid-State and Integrated Circuit Technology (ICSICT), pp. 406–408.
Fujimoto, K., Sasaki, H., Masaoka, Y., Yang, R.-Q., Mizumoto, M., 2006. An implementation of the neuro–fuzzy inference circuit. In: 1st International Conference on Innovative Computing, Information and Control (ICICIC 06), vol. 2. IEEE, pp. 301–304.
Fujimoto, K., Sasaki, H., Masaoka, Y., Yang, R.-Q., Mizumoto, M., 2007. An implementation of the neuro–fuzzy inference circuit. In: 1st International Conference on Innovative Computing, Information and Control—Volume II (ICIC 07), vol. 3, pp. 1043–1052.
Fujimoto, K., Sasaki, H., Yang, R.-Q., Shi, Y., 2008. A design of neuro–fuzzy inference circuit with automatic generation of membership functions. Int. J. Innov. Comput. Inf. Control 4 (10), 2711–2719.
Fukushima, K., 1982. Neocognitron: a new algorithm for pattern recognition tolerant of deformations and shift in position. In: Patter Recognition, vol. 15, pp. 455–469.
Fung, R.-F., Weng, M.-H., Kung, Y.-S., 2009. FPGA-based adaptive backstepping fuzzy control for a micro-positioning Scott–Russell mechanism. Mech. Syst. Signal Process. 23, 2671–2686.
Gadea, R., Cerda, J., Ballester, F., Mocholi, A., 2000. Artificial neural network implementation on a single FPGA of a pipelined on-line backpropagation. In: 13th International Symposium on System Synthesis, pp. 225–230.
Gaisler, J., Catovic, E., Isomäki, M., Glembo, K., Habinc, S. URL: ⟨http://www.gaisler.com/products/grlib/grip.pdf⟩.
Garcia, R., Pedro, T.D., 1996. Un modelo de coprocesador borroso. In: ESTYLF'96—VI Congreso Español sobre Tecnologías Fuzzy.
Garcia, R., Pedro, T.D., 2000. First applications of the orbex coprocessor: control of unmanned vehicles. In: Mathware & Soft Computing, vol. 7, pp. 265–273.
Garth, S., 1987. A chipset for high speed simulation of neural network systems. In: IEEE 1st International Conference on Neural Networks, pp. 443–452.
Gatet, L., Tap-Béteille, H., Lescure, M., 2008. Analog neural network implementation for a real-time surface classification application. IEEE Sensors J. 8 (8), 1413–1421.
Girau, B., 2001. On-chip learning of FPGA-inspired neural nets. In: IEEE International Joint Conference on Neural Networks (IJCNN01), vols. 1–4, pp. 222–227.
Gomariz, S., Alarcon, E., ,Martinez, J.A., Poveda, A., Madrenas, J., Guinjoan, F. 1998. Minimum time control of a buck converter by means of fuzzy logic approximation. In: Proceedings of the 24th Annual Conference of the IEEE Industrial-Electronics-Society, vol. 2, pp.1060–1065.
Grabowski, P., Blaabjerg, F., 1998. Direct torque neuro–fuzzy control of induction motor drive. DSP implementation. In: Annual Conference of the IEEE Industrial Electronics Society (IECON 98), vol. 2, pp. 657–661.
Grabowski, P., Kazmierkowski, M., Bose, B., Blaabjerg, F., 2000. A simple direct-torque neuro–fuzzy control of PWM-inverter-fed induction motor drive. IEEE Trans. Ind. Electron. 47 (4), 863–870.
Gregoretti, F., Passerone, R., Reyneri, R., Sansoe, C., 2001. A high speed VLSI architecture for handwriting recognition. J. VLSI Signal Process. 28, 258–278.
Grzechca, D., Golonek, T., Rutkowski, J., 2008. Diagnosis of specification parametric faults in the FPAA—the RBF neural network approach. In: 2nd European Computing Conference (ECC 08), pp. 275–280.
Guo, S., Peters, L., Surmann, H., 1996. Design and application of an analog fuzzy logic controller. IEEE Trans. Fuzzy Syst. 4 (4), 429–438.
Halgamuge, S., Glesner, M., 1994. Neural networks in designing fuzzy systems for real world applications. Fuzzy Sets Syst. 65 (1), 1–12.
Hammami, O., 1997. Pipeline integration of neuro and fuzzy cache management techniques. In: IEEE International Conference on Fuzzy Systems, vol. 2. IEEE, pp. 653–658.
Hammerstrom, D., 1990. A VLSI architecture for high-performance, low-cost, on-chip learning. In: IJCNN International Joint Conference on Neural Networks, vol. 2, pp. 537–544.
Hamzeh, M., Mahdiani, H., Saghafi, A., Fakhraie, S., 2009. Computationally efficient active rule detection method: algorithm and architecture. Fuzzy Sets Syst. 160 (4), 554–568.
Han, I., 1997. Hardware implementation of neuro–fuzzy system with the analogue–digital hybrid neural chip. In: IEE Colloquium on Neural and Fuzzy Systems: Design, Hardware and Applications, vols. 1–4. IEE, London.
Han, I.S., 1994. Neuro-fuzzy hybrid hardware implementation with neural chip URAN. IEEE Int. Conf. Neural Networks 1, 3978–3981.
Hasanien, H., 2011. FPGA implementation of adaptive ANN controller for speed regulation of permanent magnet stepper motor drives. Energy Convers. Manag. 52, 1252–1257.
Hebb, D.O., 1949. The Organization of Behavior.
Hendry, D.C., Duncan, A.A., Lightowler, N., 2003. Ip core implementation of a self-organizing neural network. IEEE Trans. Neural Netw. 14 (5), 1085–1096.
Hernandez-Zavala, A., Camacho-Nieto, O., 2012. Fuzzy hardware: a retrospective and analysis. IEEE Trans. Fuzzy Syst. 20 (4), 623–635.
Hikawa, H., 1995. Implementation of simplified multilayer neural networks with on-chip learning. In: IEEE International Conference on Neural Networks, vol. 4, pp. 1663–1637.
Hikawa, H., 2001. Multilayer neural network with on-chip learning based on frequency-modulated pulse signals and voting neurons. Electron. Commun. Jpn. Part III. Fundam. Electron. Sci. 84 (1), 32–42.
Hikawa, H., Sato, K., 1998. Multilayer neural network with threshold neurons. IEICE Trans. Fundam. Electron. Commun. Comput. Sci. E81A (6), 1105–1112.
Holler, M., Tam, S., Castro, H., Benson, R., 1989. An electrically trainable artificial neural network (ETANN) with 10 240 'floating gate' synapses. In: International Joint Conference on Neural Networks (IJCNN), vol. 2. IEEE, pp. 191–196.
Hollstein, T., Halgamuge, S., Glesner, M., 1996. Computer-aided design of fuzzy systems based on generic VHDL specifications. IEEE Trans. Fuzzy Syst. 4 (4), 403–417.
Hong, W., Chen, W., Zhang, R., 2009. The application of neural network in the technology of image processing. In: Proceedings of the International Multi-Conference of Engineers and Computer Scientists, vol. 1, pp. 18–20.
Hornik, K., 1991. Approximation capabilities of multilayer feedforward networks. Neural Netw. 4, 251–257.

Please cite this article as: Bosque, G., et al., Fuzzy systems, neural networks and neuro-fuzzy systems: A vision on their hardware implementation and platforms over two decades. Eng. Appl. Artif. Intel. (2014), http://dx.doi.org/10.1016/j.engappai.2014.02.008

Hsu, C.-F., Wang, S.-L., Li, M.-C., Lin, C.-M., 2010. Chip-implementation of a self-tuning nonlinear function control for dc–dc converters. Int. J. Intell. Comput. Cybern. 3 (1), 117–134.
Hsu, C.-F., Chiu, C.-J., Tsai, J.-Z., 2011. Auto-tuning PID controller design using a sliding-mode approach for dc servomotors. Int. J. Intell. Comput. Cybern. 4 (1), 93–110.
Huang, S.-H., Lai, J.-Y., 2005. A high-speed VLSI fuzzy inference processor for trapezoid-shaped membership functions. J. Inf. Sci. Eng. 21, 607–626.
Hubel, D., Wiesel, T., 1968. Receptive fields and functional architecture of monkey striade cortex. J. Physiol. 165, 215–243.
Huertas, J., Sanchez-Solano, S., Barriga, S., Baturone, I., 1993. A fuzzy controller using switched-capacitor techniques. In: Proceedings of the IEEE International Conference on Fuzzy Systems, pp. 516–529.
Hu, H., Huang, J., Xing, J., Wang, W., 2008. Key issues of FPGA implementation of neural networks. In: 2nd International Symposium on Intelligent Information Technology Application. IEEE Computer Society, pp. 259–263.
Huitzil, C., 2006. A bit-stream pulse-based digital neuron model for neural networks. In: ICONIP 2006, Part III, pp. 1150–1159.
Hung, D., 2007. Using FPGA technique for design and implementation of a fuzzy inference system. Int. Ser. Intell. Technol., Fuzzy Logic Intell. Syst., 271–288
Hung, D., Wang, J., 2003. Digital hardware realization of a recurrent neural network for solving the assignment problem. Neurocomputing 51, 447–461.
Hung, D., Zajak, W.F., 1995. Design and implementation of a hardware fuzzy inference system. Inf. Sci. 3, 193–207.
Ide, A., Saito, J., 2006. FPGA implementations of neocognitrons. In: Omondi, A., Rajapakse, J. (Eds.), FPGA Implementations of Neural Networks. Springer, Netherlands, pp. 197–224
Indue, T., Motomura, T., Matsuo, R., 1991. New OTA-based analog circuits for fuzzy membership functions and maximum operation. IEIC Trans. Commun. Electron. 74 (11), 3619–3621.
Ionita, S., Sofron, E., 2004. Field-programmable analog filters array with applications for fuzzy inference systems. In: International Conference on Hybrid Intelligent Systems (HIS 04), pp. 470–471.
Isshiki, T., Dai, W., 1996. Bit-serial pipeline synthesis for multi-FPGA systems with c++ design capture. In: IEEE Symposium on FPGAs for Custom Computing Machines, pp. 38–47.
Izeboudjen, N., Farah, A., Titri, S., Boudmeridja, H., 1999. In: Mendeley (Ed.), Digital Implementation of Artificial Neural Networks: From VHDL Description to FPGA Implementation, IWANN 99. Springer-Verlag, pp. 139–148.
Izhikevich, E., 2003. Simple model of spiking neurons. IEEE Trans. Neural Netw. 14 (6), 1569–1572.
Jabri, M., Flower, B., 1991. Weight perturbation: an optimal architecture and learning technique for analog VLSI feedforward and recurrent multilayer networks. Neural Comput. 3 (4), 546–565.
Jabri, M., Flower, B., 1992. Weight perturbation: an optimal architecture and learning technique for analog VLSI feedforward and recurrent multilayer networks. IEEE Trans. Neural Netw. 3, 154–157.
Jackson, A., 1994. Fuzzy logic vs traditional approaches to the design of microcontroller-based systems. In: IEEE Aerospace Applications Conference, pp. 19–33.
Jacomet, M., Walti, R., 1996. A VLSI fuzzy processor with parallel rule execution. In: 5th IEEE International Conference on Fuzzy Systems, vol. 1, pp. 554–558.
Jang, J., Sun, C., Mizutani, E., 1997. Neuro–Fuzzy and Soft Computing. Prentice Hall
Javadi, M., Mahdiani, H., Zeinali-Kh, E., 2013. A hardware oriented fuzzification algorithm and its VLSI implementation. Soft Comput. 17 (4), 683–690.
Juang, C.-F., Tsao, Y.-W., 2008. A type-2 self-organizing neural fuzzy system and its FPGA implementation. IEEE Trans. Syst. Man Cybern. Part B: Cybern. 38 (6), 1537–1548.
Kala, K., Srinivas, M., 2006. A generic architecture for intelligent system hardware. In: IEEE Asia Pacific Conference on Circuits and Systems APCCAS 2006, pp. 321–326.
Kamala-Kannan, C., Kamaraj, V., Paranjothi, S., 2012. Sensorless control of SR drive using ANN and FPAA for automotive applications. Energy Procedia 14, 1831–1836.
Kandel, A., Langholz, G. (Eds.), 1998. Fuzzy Hardware: Architecture and Applications. Kluwer Academic Publishers
Kao, T.-P., Yu, C.-C., Chen, T.-Y., Wang, J.-S., 2007. Hardware design of an adaptive neuro–fuzzy network with on-chip learning capability. In: ISNN 2007. Springer-Verlag, pp. 336–345.
Khodabandehloo, G., Mirhassani, M., Ahmadi, M., 2012. Analog implementation of a novel resistive-type sigmoidal neuron. IEEE Trans. Very Large Scale Integr. (VLSI) Syst. 20 (4), 750–754.
Kim, D., 2000. An implementation of fuzzy logic controller on the reconfigurable FPGA system. IEEE Trans. Ind. Electron. 47 (3), 703–715.
Kim, C., Lee, S., 2001. A digital chip for robust speech recognition in noisy environment. In: International Conference on Acoustics Speech and Signal Processing (ICASSP01), pp. 1089–1092.
Kim, C.-M., Park, H.-M., Kim, T., Choi, Y.-K., Lee, S.-Y., 2003. FPGA implementation of ICA algorithm for blind signal separation and adaptive noise canceling. IEEE Trans. Neural Netw. 14 (5), 1038–1046.
Kim, Y., Cho, J., Jeon, G., 2007. An intelligent wireless electronic nose node for monitoring gas mixtures using neuro–fuzzy networks implemented on a microcontroller. In: IEEE International Conference on Computational Intelligence for Measurement Systems and Applications (CIMSA 2007).
Kim, M., Kim, J.-Y., Lee, S., Oh, J., Yoo, H.-J., 2009. A 22.8 gops 2.83 mW neuro–fuzzy object detection engine for fast multi-object recognition. In: Symposium on VLSI Circuits, pp. 260–261.
Kiriakidis, K., 1998. Fuzzy model-based control of complex plants. IEEE Trans. Fuzzy Syst. 6 (4), 517–529.
Kosko, B., 1994. Fuzzy systems and universal approximators. IEEE Trans. Comput. 43 (11), 1329–1333.
Kreinovich, V., Nguyen, H., Yam, Y., 2000. Fuzzy systems are universal approximators for a smooth function and its derivatives. Int. J. Intell. Syst. 15, 565–574.
Krips, M., Lammert, T., Kummert, A., 2002. FPGA implementation of a neural network for a real-time hand tracking system. In: 1st IEEE International Workshop on Electronic Design, Test and Applications, pp. 313–317.
Kung, Y.-S., Quynh, N.-V., Hieu, N.-T., 2011. Simulink/modelsim co-simulation and FPGA realization of speed control IC for PMSM drive. Procedia Eng. 23, 718–727.
Kurdthongmee, W., 2008. A novel hardware-oriented Kohonen SOM image compression algorithm and its FPGA implementation. J. Syst. Archit. 54, 983–994.
Landajo, M., Rio, M., Perez, R., 2001. A note on smooth approximation capabilities of fuzzy systems. IEEE Trans. Fuzzy Syst. 9 (2), 229–237.
Landlot, O., 1996. Low-power analog fuzzy rule implementation based on a linear MOS transistor network. In: 5th International Conference on Microelectronics for Neural Networks end Fuzzy Systems, pp. 86–93.
Lattice. URL: ⟨www.latticesemi.com⟩.
Lee, R., 1996. Subword parallelism with max-2. IEEE Micro 16 (4), 51–59.
Lee, B., Sheu, B., 1990. A compact and general-purpose neural chip with electrically programmable synapses. In: IEEE Custom Integrated Circuits Conference, pp. 26.6.1–26.6.4.
Lemaitre, L., Patyra, M., Mlynek, D., 1994. Analysis and design of CMOS fuzzy logic controller in current mode. IEEE J. Solid-State Circuits 29 (3), 317–322.
Leonhard, G., Cousin, E., Laisne, J., LeDrezen, J., Ouvradou, G., Maubant, A., Thepaut, A., 1995. Armenx: a flexible platform for signal and image processing. In: Proceedings of the Society of Photo-Optical Instrumentation Engineers (SPIE), vol. 2607, pp. 71–82.
Li, C., Tsai, K.-B., 2006. Adaptive interference signal processing with intelligent neuro–fuzzy approach. In: ICCOMP'06 Proceedings of the 10th WSEAS International Conference on Computers, pp. 393–398.
Liao, S.-H., 2005. Expert system methodologies and applications—a decade review from 1995 to 2004. Expert Syst. Appl. 28, 93–103.
Li, T.-H., Chang, S.-J., Chen, Y.-X., 2003. Implementation of human-like driving skills by autonomous fuzzy behavior control on an FPGA-based car-like mobile robot. IEEE Trans. Ind. Electron. 50 (5), 867–880.
Lin, C.-J., Lee, C.-Y., 2009. FPGA implementation of a recurrent neural fuzzy network with on-chip learning for prediction and identification applications. J. Inf. Sci. Eng. 25, 575–589.
Lin, C.-J., Lee, C.-Y., 2011. Implementation of a neuro–fuzzy network with on-chip learning and its applications. Expert Syst. Appl. 38, 673–681.
Lin, C.-J., Tsai, H.-M., 2008. FPGA implementation of a wavelet neural network with particle swarm optimization learning. Math. Comput. Model. 47, 982–996.
Lin, C.-W., Wang, J.-S., Yu, C.-C., Chen, T.-Y., 2007. Synchronous pipeline circuit design for an adaptive neuro–fuzzy network. In: ICIC 2007. Springer-Verlag, pp. 164–173.
Lippman, R., 1988. Neural network classifiers for speech recognition. Linc. Lab. J. 1 (1), 107–124.
Lizárraga, G., Sepúlveda, R., Montiel, O., Castillo, O., 2008. Soft computing for hybrid intel systems. In: Modeling and Simulation of the Defuzzification Stage Using Xilinx System Generator and Simulink, vol. 154. Springer-Verlag, pp. 333–343.
Lsi. URL: ⟨www.lsi.com⟩.
Lu, C., Shi, B., Chen, L., 2001. A programmable on-chip BP learning neural network with enhanced neuron characteristics. In: IEEE International Symposium on Circuits and Systems (ISCAS 2001), vol. 3.
Mahyuddin, M., Wei, C., Arshad, M., 2009. Neuro–fuzzy algorithm implemented in altera FPGA for mobile robots obstacle avoidance mission. In: TENCON 2009. IEEE.
Manzoul, M., Jayabharathi, D., 1992. Fuzzy controller on FPGA chip. In: IEEE International Conference on Fuzzy Systems, pp. 1309–1316.
Marchesi, M., Orlandi, G., Piazza, F., Uncini, A., 1993. Fast neural networks without multipliers. IEEE Trans. Neural Netw. 4 (1), 53–62.
Marshall, G., Collins, S., 1997. Fuzzy logic architecture using subthreshold analogue floating gate devices. IEEE Trans. Fuzzy Syst. 5 (1), 32–43.
McCulloch, W., Pitts, W., 1943. A logical calculus of the ideas imminent in nervous activity. Bull. Math. Biophys. 5, 115–133.
McIlraith, A., Card, H., 1995. Birdsong recognition with DSP and neural networks. In: Communications, Power, and Computing, Conference Proceedings, vol. 2. IEEE.
Mekki, H., Mellit, A., Kalogirou, S., Messai, A., Furlan, G., 2010. FPGA-based implementation of a real time photovoltaic module simulator. Prog. Photovolt.: Res. Appl. 18, 115–127.
Mermoud, G., Upegui, A., Peña, C.-A., 2005. A dynamically-reconfigurable FPGA platform for evolving fuzzy systems. In: Cabestany, Prieto, Sandoval (Eds.), 8th International Work-Conference on Artificial Neural Networks (IWANN'2005). Springer-Verlag, pp. 572–581.
Messai, A., Mellit, A., Pavan, A.M., Guessoum, A., Mekki, H., 2011. FPGA-based implementation of a fuzzy controller (MPPT) for photovoltaic module. Energy Convers. Manag. 52, 2695–2704.
Miki, T., Yamakawa, T., 1995. Fuzzy inference on an analog fuzzy chip. IEEE Micro 15 (4), 8–18.
Miki, T., Matsumoto, H., Ohto, K., Yamakawa, T., 1993. Silicon implementation for a novel high-speed fuzzy inference engine: mega-fips analog fuzzy processor. J. Intell. Fuzzy Syst. 1 (1), 27–42.

Please cite this article as: Bosque, G., et al., Fuzzy systems, neural networks and neuro-fuzzy systems: A vision on their hardware implementation and platforms over two decades. Eng. Appl. Artif. Intel. (2014), http://dx.doi.org/10.1016/j.engappai.2014.02.008

Millán, I., Montiel, O., Sepúlveda, R., Castillo, O., 2008. Design and implementation of a hybrid fuzzy controller using VHDL. Stud. Comput. Intell., Soft Comput. Hybrid Intell. Syst., 437–446

Minkovich, K., Srinivasa, N., Cruz-Albrecht, J., Cho, Y., Nogin, A., 2012. Programming time-multiplexed reconfigurable hardware using scalable neuromorphic compiler. IEEE Trans. Neural Netw. Learn. Syst. 23 (6), 889–901.

Minsky, M. 1952. A Neural-Analogue Calculator Based upon a Probability Model of Reinforcement. Psychological Laboratories, Harvard University, Cambridge, MA.

Misra, J., Saha, I., 2010. Artificial neural networks in hardware: a survey of two decades of progress. Neurocomputing 74, 239–255.

Mizumoto, M., 1991. Min–Max-Gravity Method Versus Product-Sum-Gravity Method for Fuzzy Control. Fuzzy Systems Association. pp. 127–130.

Moreno, J., Madranas, J., Alarcon, E., Cabestany, J., 1997. Analog sequential architecture for neuro–fuzzy models. In: Artificial Neural Networks (ICANN'97). Springer, pp. 1199–1204.

Morie, T., Amemiya, Y., 1994. An all-analog expandable neural network LSI with on-chip backpropagation learning. IEEE J. Solid-State Circuits 29, 1086–1093.

Morris, C., Lecar, H., 1981. Voltage oscillations in the barnacle giant muscle fiber. Biophys. J. 35, 193–213.

Muñoz, D., Llanos, C., Ayala-Rincon, M., Els, R., 2008. Distributed approach to group control of elevator systems using fuzzy logic and FPGA implementation of dispatching algorithms. Eng. Appl. Artif. Intell. 21, 1309–1320.

Murshid, A., Loan, S., Abbasi, S., Alamoud, A., 2011. Vlsi architecture of fuzzy logic hardware implementation: a review. Int. J. Fuzzy Syst. 13 (2), 74–88.

Muthuramalingam, A., Himavathi, S., Srinivasan, E., 2007. Neural network implementation using FPGA: issues and application. Int. J. Inf. Technol. 4 (2), 86–92.

Nagamine, S., Shigei, N., Miyajima, H., 2008. Fuzzy inference models appropriate for digital circuit. In: The 23rd International Conference on Circuits/Systems, Computers and Communications (ITC-CSCC 2008).

Nauck, D., Kruse, R., 1999. Neuro–fuzzy systems for function approximation. Fuzzy Sets Syst. 101, 261–271.

Navas-Gonzalez, R., Vidal-Verdu, F., Rodriguez-Vazquez, A., 2003. Neuro–fuzzy chip to handle complex tasks with analog performance. IEEE Trans. Neural Netw. 14 (5), 1375–1392.

Nhivekar, G., Nirmale, S., Mudholker, R., 2011. Implementation of fuzzy logic control algorithm in embedded microcomputers for dedicated application. Int. J. Eng. Sci. Technol. 3 (4), 276–283.

Noory, B., Groza, V., 2003. A reconfigurable approach to hardware implementation of neural networks. In: Canadian Conference on Electrical and Computer Engineering, 2003. IEEE, pp. 1861–1864.

Oh, S.L.J., Yoo, H.-J., 2013. 1.2 m · w online learning mixed-mode intelligent inference engine for low-power real-time object recognition processor. IEEE Trans. Very Large Scale Integr. (VLSI) Syst. 21, 921–933.

Oh, J., Lee, S., Kim, M., Kwon, J., Park, J., Kim, J.-Y., Yoo, H.-J., 2010. A 1.2 mW on-line learning mixed mode intelligent inference engine for robust object recognition. In: IEEE Symposium on VLSI Circuits, pp. 17–18.

Oh, J., Park, J., Kim, G., Lee, S., Yoo, H.-J., 2011. A 57 mW embedded mixed-mode neuro–fuzzy accelerator for intelligent multi-core processor. In: IEEE International Solid-State Circuits Conference, pp. 130–132.

Omondi, A., Rajapakse, J., 2002. Neural networks in FPGAs. In: Proceedings of the 9th International Conference on Neural Information Processing (ICONIP'02), vol. 2.

Omron. URL: ⟨http://industrial.omron.es⟩.

Orlowska-Kowalska, T., Kaminski, M., 2009. Application of MLP and RBF neural networks in the control structure of the drive system with elastic joint. Int. J. Comput. Math. Electr. Electron. Eng. 28, 556–569.

Ortigosa, E., Canas, A., Ros, E., Carrillo, R., 2003. FPGA implementation of a perceptron-like neural network for embedded applications. In: 7th International Work Conference on Artificial and Natural Neural Networks, vol. 2687, pp. 1–8.

Omondi, A.R., Rajapakse, J.C. (Eds.), 2006. FPGA Implementations of Neural Networks. Springer.

Othman, A., Riadh, M., 2008. Speech recognition using scaly neural networks. World Acad. Sci. Eng. Technol., 253–258

Pal, S.K., Mitra, S., 1999. Neuro–Fuzzy Pattern Recognition: Methods in Soft Computing. Wiley-Interscience

Panchariya, P., Palit, A., Popovic, D., Sharma, A., 2004. Nonlinear system identification using Takagi–Sugeno type neuro–fuzzy model. In: 2nd IEEE International Conference on Intelligent Systems. IEEE, pp. 76–81.

Papadonikolakis, M., Bouganis, C.-S., 2012. Novel cascade FPGA accelerator for support vector machines classification. IEEE Trans. Neural Netw. Learn. Syst. 23 (7), 1040–1052.

Patel, N., Nguang, S., Coghill, G., 2007. Neural network implementation using bit streams. IEEE Trans. Neural Netw. 18 (5), 1488–1504.

Patyra, M., Grantner, J., Koster, K., 1996. Digital fuzzy logic controller: design and implementation. IEEE Trans. Fuzzy Syst. 4 (4), 439–459.

Pearson, M., Pipe, A., Mitchinson, B., Gurney, G., Melhuish, C., Gilhespy, I., Nibouche, M., 2007. Implementing spiking neural networks for real-time signal-processing and control applications: a model-validated FPGA approach. IEEE Trans. Neural Netw. 18 (5), 1472–1487.

Pedram, A., Jamali, M.R., Pedram, T., Fakhraie, S.M., Lucas, C., 2006a. Local linear model tree (LOLIMOT) reconfigurable parallel hardware. Trans. Eng. Comput. Technol. 13, 96–101.

Pedram, A., Jamali, M., Fakhraie, S., Lucas, C., 2006. Reconfigurable parallel hardware for computing local linear neuro–fuzzy model. In: International Symposium on Parallel Computing in Electrical Engineering (PARELEC 06).

Peters, L., Guo, S., Camposano, R., 1995. A novel analog fuzzy controller for intelligent sensors. Fuzzy Sets Syst. 70 (2–3), 235–247.

Pierzchala, E., Perkowski, M.A., Grygiel, S., 1994. A field programmable analog array for continuous, fuzzy, and multi-valued logic applications. In: IEEE International Symposium on Multiple-Valued Logic (ISMVL 2004), pp. 148–155.

Porto, D., Morabito, F., Branciforte, M., Calabro, A., 2002. A neuro–fuzzy approach for hardware modeling of nuclear fusion reactors plasma. In: International Conference on Electronics, Circuits and Systems, vol. 3, pp. 1255–1288.

Prieto, B., de Lope, J., Maravall, D., 2005. Reconfigurable hardware implementation of neural networks for humanoid locomotion. In: IWINAC'05, vol. II. Springer-Verlag, pp. 395–404.

Quicklogic. URL: ⟨www.quicklogic.com⟩.

Ramirez-Angulo, J., Treece, K., Andrews, P., Choi, T., 1995. Current-mode and voltage-mode VLSI fuzzy processor architecture. In: IEEE International Symposium on Circuits and Systems (ISCAS95), vol. 2, pp. 1156–1159.

Rao, D., 1995. Neural networks in robotics and control: some perspectives. In: International Conference on Industrial Automation and Control IEEE/IAS. IEEE, pp. 451–456.

Ray, K., Ghoshal, J., 1997. Neuro fuzzy approach to pattern recognition. Neural Netw. 10 (1), 161–182.

Raychev, R., Mtibaa, A., Abid, M., 2005. VHDL modelling of a fuzzy co-processor architecture. In: International Conference on Computer Systems and Technologies - CompSysTech, pp. 1–6.

Restrepo, H., Hoffmann, R., Perez-Uribe, A., Teuscher, C., Sanchez, E., 2000. A networked FPGA-based hardware implementation of a neural network application. In: IEEE Symposium on Field-Programmable Custom Computing Machines, pp. 337–338.

Reyneri, L.M., Chiaberge, M., Zocca, L., 1994. CINTIA: a neuro-fuzzy real time controller for low power embedded systems. In: Proceedings of the Fourth International Conference on Microelectronics for Neural Networks and Fuzzy Systems, vol. 1, pp. 392-403.

Reyneri, L.M., 1995. A performance analysis of pulse stream neural and fuzzy computing systems. IEEE Trans. Circuits Syst. II: Anal. Digit. Signal Process. 42 (10), 642–660.

Reyneri, L.M., Chiaberge, M., Corso, D.D., 1993. Using coherent pulse width and edge modulations in artificial neural systems. Int. J. Neural Syst. (IJNS) 4 (4), 309–316.

Rocke, P., Maher, J., Morgan, F., 2005. Platform for intrinsic evolution of analogue neural networks. In: International Conference on Reconfigurable Computing and FPGAs.

Rodriguez-Vazquez, A., Vidal-Verdu, F., 1995. Learning in neuro/fuzzy analog chips. In: IEEE International Symposium on Circuits and Systems (ISCAS 95), vol. 3, pp. 2325–2328.

Rodriguez-Vazquez, A., Vidal-Verdù, F., 1996. Hardware implementation of neuro–fuzzy systems as mixed-signal chips. In: 1st International Symposium on Neuro–Fuzzy Systems.

Rovatti, R., 1998. Fuzzy piecewise multilinear and piecewise linear systems as universal approximator in Sobolev norms. IEEE Trans. Fuzzy Syst. 6 (2), 235–249.

Rusu, P., Petriu, E.M., Whalen, T.E., Cornell, A., Spoelder, H.J.W., 2003. Behavior-based neuro–fuzzy controller for mobile robot navigation. IEEE Trans. Instrum. Meas. 52(4), 1335–1340.

Saadi, A.G.S., Bettayeb, M., 2013. Abc optimized neural network model for image deblurring with its FPGA implementation. Microprocess. Microsyst. 37, 52–64.

Sackinger, E., Graf, H., 1996. A board system for high-speed image analysis and neural networks. IEEE Trans. Neural Netw. 7 (1), 214–221.

Sackinger, E., Boser, B., Jackel, L., 1991. A neurocomputer board based on the ANNA neural network chip. In: Neural Information Processing Systems (NIPS 91), vol. 4, pp. 773–780.

Sadati, N., Mohseni, H., 1999. A VLSI neuro-fuzzy controller. Intell. Autom. Soft Comput. 5, 239–256.

Sahin, S., Becerikli, Y., Yazici, S., 2006. Neural network implementation in hardware using FPGAs. In: King, I., et al. (Eds.), ICONIP 2006, vol. III. Springer-Verlag, pp. 1105–1112.

Salapura, V., 2000. A fuzzy risc processor. IEEE Trans. Fuzzy Syst. 8 (6), 781–790.

Saldaña, H., Cardenash, C., 2010. Design and implementation of an adaptive neuro–fuzzy inference system on an FPGA used for nonlinear function generation. In: ANDESCON. IEEE.

Sameh, A., Samir, M., 2005. Generic floating point library for neuro–fuzzy controllers based on FPGA technology. AIML J. 5 (1), 5–13.

Sánchez-Solano, S., Cabrera, A.J., Baturone, I., Moreno-Velo, F.J., Brox, M., 2007. FPGA implementation of embedded fuzzy controllers for robotic applications. IEEE Trans. Ind. Electron. 54 (4), 1937–1945.

Sato, K., Hikawa, H., 1999. Implementation of multilayer neural network with threshold neurons and its analysis. In: International Symposium on Artificial Life and Robotics, vol. 3, pp. 170–175.

Satyanarayana, S., Tsividis, Y., Graf, H., 1992. A reconfigurable VLSI neural network. IEEE J. Solid-State Circuits 27 (1), 67–81.

Sekerli, M., Butera, R.J., 2004. An implementation of a simple neuron model in field programmable analog arrays. In: Proceedings of the 26th Annual International Conference of the IEEE EMBS. IEEE, pp. 4564–4567.

Shima, T., Kimura, T., Kamatani, Y., Itakura, T., Fujita, Y., Iida, T., 1992. Neuro chips with on-chip back-propagation and/or Hebbian learning. IEEE J. Solid-State Circuits 27 (12), 1868–1876.

Shimizu, K., Osumi, M., Imae, F., 1992. Digital fuzzy processor fp-5000. In: International Conference on Fuzzy Logic & Neural Networks, pp. 539–542.

Shoushan, L., Yan, C., Wenshang, X., Tongjun, Z., 2010. A single layer architecture to FPGA implementation of bp artificial neural network. In: 2nd International Asia

Please cite this article as: Bosque, G., et al., Fuzzy systems, neural networks and neuro-fuzzy systems: A vision on their hardware implementation and platforms over two decades. Eng. Appl. Artif. Intel. (2014), http://dx.doi.org/10.1016/j.engappai.2014.02.008

Conference on Informatics in Control, Automation and Robotics. IEEE, pp. 258–264.
Soleimani, H., Ahmadi, A., Bavandpour, M., 2012. Biologically inspired spiking neurons: piecewise linear models and digital implementation. IEEE Trans. Circuits Syst. 12, 2991–3004.
ST. URL: ⟨www.st.com/mcu⟩.
Stefano, A.D., Giaconia, C., 2005. An FPGA-based adaptive fuzzy coprocessor. Comput. Intell. Bioinspired Syst., 590–597
Stinchcombe, M., White, H., 1989. Multilayer feedforward networks are universal approximator. Neural Netw. 2, 359–366.
Sulaiman, N., Obaid, Z., Marhaban, M., Hamidon, M., 2009. FPGA-based fuzzy logic: design and applications—a review. IACSIT—Int. J. Eng. Technol. 1 (5), 491–503.
Sultan, A., El-Sayed, M., 2000. Analog VLSI implementation of adaptive neuro–fuzzy inference systems. In: IEEE International Conference on Electronics, Circuits and Systems, vol. 1, pp. 558–561.
Sundarambal, M., Chandrakala, D., Anbalagan, P., 2009. VHDL implementation of neuro–fuzzy based adaptive bandwidth controller for ATM networks. Int. J. Commun. Netw. Distrib. Syst. 3, 159–174.
Sun, X., Chow, M., Leung, F., Xu, D., Wang, Y., Lee, Y.-S., 2002. Analogue implementation of a neural network controller for UPS inverter applications. IEEE Trans. Power Electron. 17 (3), 305–313.
Szabo, T., Antoni, L., Horvath, G., Feher, B., 2000. A full-parallel digital implementation for pre-trained NNs. In: IEEE International Joint Conference on Neural Networks (IJCNN00), vol. II, pp. 49–54.
Szmidt, E., Kacprzyk, J., 2001. Intuitionistic fuzzy sets in some medical applications. In: 5th International Conference on IFSs, vol. 4, pp. 58–64.
Takagi, H., Hayashi, I., 1991. NN driven fuzzy reasoning. Int. J. Approx. Reason. 5 (3), 191–212.
Tarighat, Y., Hubin, M., 1998. FPGA implementation of a multilayer perceptron neural network using VHDL. In: 4th International Conference on Signal Processing (ICSP98), vols. I–II, pp. 1311–1314.
Tensilca. URL: ⟨http://www.tensilica.com/uploads/pdf/xtensa_arch_white_paper.pdf⟩.
Thareja, V., Montcalm, M., Bolic, M., 2009. Configurable fuzzy logic coprocessor for small scale food preparation. In: 3rd International Workshop on Soft Computing Applications (SOFA09), pp. 229–234.
Togai, M., Watanabe, H., 1986a. A vlsi implementation of a fuzzy-inference engine: toward an expert system on a chip. Inf. Sci. 38 (2), 147–163.
Togai, M., Watanabe, H., 1986b. Expert system on a chip: an engine for real-time approximate reasoning. IEEE Expert 1 (3), 55–62.
Torres-Huitzil, C., Girau, B., Gauffriau, A., 2007. Hardware/software codesign for embedded implementation of neural networks. In: ARC 2007. Springer-Verlag, pp. 167–178.
Tsoukalas, L., Uhrig, R., 1997. Fuzzy an Neural Approaches in Engineering. John Wiley & Sons, Inc., Wiley-Interscience
Tsukano, K., Inoue, T., 1995. Synthesis of operational transconductance amplifier-based analog fuzzy functional blocks and its applications. IEEE Trans. Fuzzy Syst. 3 (1), 61–68.
Tzafestas, S., Deliparaschos, K., Moustris, G., 2010. Fuzzy logic path tracking control for autonomous non-holonomic mobile robots: design of system on a chip. Robot. Auton. Syst. 58, 1017–1027.
Uddin, M., Wen, H., 2007. Development of a self-tuned neuro–fuzzy controller for induction motor drives. IEEE Trans. Ind. Appl. 43 (4), 1108–1116.
Ungering, A., Bauer, H., Goser, K., 1994. Architecture of a fuzzy-processor based on an 8-bit microprocessor. In: IEEE Conference on Fuzzy Systems, vol. 1, pp. 297–301.
Varrientos, J., Sánchez-Sinencio, E., Ramirez-Angulo, J., 1993. A current-mode cellular neural network implementation. IEEE Trans. Circuits Syst. 48 (3), 147–155.
Venayagamoorthy, G., Harley, R., Wunsch, D., 2003. Implementation of adaptive critic-based neurocontrollers for turbogenerators in a multimachine power system. IEEE Trans. Neural Netw. 14 (5), 1047–1064.
Vidal-Verdu, F., Delgado-Restituto, M., Navas, R., Rodriguez-Vazquez, A., 1999. A design approach for analog neuro/fuzzy systems in CMOS digital technologies. Comput. Electr. Eng. 25, 309–337.
Wakami, N., Arakis, S., Nomura, H., 1993. Recent applications of fuzzy logic to home appliances. In: IEEE Conference on Emerging Technologies, and Factory Automation, vol. 1. IEEE Computer Society Press, pp. 155–160.
Wang, L.-X., 1992. Fuzzy systems are universal approximators. In: IEEE International Conference on Fuzzy Systems, pp. 1163–1170.
Wang, W., Jin, D., 2004. CMOS design of analog neuro–fuzzy system with improved circuits. In: 7th International Conference on Solid-State and Integrated Circuits Technology.
Wang, W.-Z., Jin, D.-M., 2006. Neuro–fuzzy system with high-speed low-power analog blocks. Fuzzy Sets Syst. 157, 2974–2982.
Wang, L., Langari, R., 1994. Complex systems modeling via fuzzy logic. In: Proceedings of the 3rd Conference on Decision and Control. IEEE, pp. 4136–4141.
Wang, L.-X., Wei, C., 2000. Approximation accuracy of some neuro–fuzzy approaches. IEEE Trans. Fuzzy Syst. 8 (4), 470–478.
Watanabe, H., Chen, D., 1993. Evaluation of fuzzy instructions in a risc processor. In: IEEE International Conference on Fuzzy Systems, pp. 521–526.
Watanabe, H., Dettloff, W., 1991. VLSI fuzzy chip and inference accelerator board systems. In: Proceedings of the 21st International Symposium on Multiple-Valued Logic. IEEE, pp. 120–127.
Watanabe, H., Chen, D., Konuri, S., 1996. Evaluation of min/max instructions for fuzzy information processing. IEEE Trans. Fuzzy Syst. 4, 369–374.
Wegmann, H., 1994. Fuzzy control and neural networks industrial applications in the world of PLCs. In: Conference on Control Applications, vol. II. IEEE, Piscataway, pp. 1245–1249.
Werbos, P., 1992. Approximate dynamic programming for real-time control and neural modeling. In: Neural, Fuzzy, and Adaptive Approaches, Handbook of Intelligent Control. Van Nostrand Reinhold, NY, pp. 493–525.
Wilamowski, B., Jaeger, R., Kaynak, M., 1999. Neuro–fuzzy architecture for CMOS implementation. IEEE Trans. Ind. Electron. 46 (6), 1132–1136.
Wongsuwarn, H., Laowattana, D., 2006. Neuro–fuzzy algorithm for a biped robotic system. World Acad. Sci. Eng. Technol. 15, 138–144.
Wu, W., Li, L., Yang, J., Liu, Y., 2010. A modified gradient-based neuro–fuzzy learning algorithm and its convergence. Inf. Sci. 180, 1630–1642.
Xilinx. URL: ⟨www.xilinx.com⟩.
Xilinx, January 2007. Power PC Processor Reference Guide (v1.2). Technical Report. Xilinx.
Xilinx, 2008. Microblaze Processor Reference Guide (v9.2). Technical Report. Xilinx.
Yamakawa, T., 1988. High-speed fuzzy controllers hardware system: the mega-fips machine. Inf. Sci. 45, 113–128.
Yamakawa, T., 1993. A fuzzy inference engine in nonlinear analog mode and its applications to a fuzzy control. IEEE Trans. Neural Netw. 4 (3), 496–522.
Yamasaki, T., Shibata, T., 2003. Analog soft-pattern-matching classifier using floating-gate MOS technology. IEEE Trans. Neural Netw. 14 (5), 1257–1265.
Yosefi, G., Khoei, A., Hadidi, K., 2007. Design of a new CMOS controllable mixed-signal current mode fuzzy logic controller (FLC) chip. In: Proceedings of the IEEE International Conference on Electronics, Circuits and Systems, pp. 951–954.
Yubazaki, N., Otani, M., Muto, A., Ashida, T., Yi, J., Hirota, K., Shi, Y., 1998. Fuzzy inference chip FZP-0401a based on interpolation algorithm. Fuzzy Sets Syst. 98, 299–310.
Yun, S., Kim, Y., Dong, S., Lee, C., 2002. Hardware implementation of neural network with expansible and reconfigurable architecture. In: Wang, L., Rajapakse, J., Fukushima, K., Lee, S.-Y., Yao, X. (Eds.), 9th International Conference on Neural Information Processing, vol. 2, pp. 970–975.
Zamanlooy, B., Mirhassani, M., 2014. Efficient VLSI implementation of neural networks with hyperbolic tangent activation function. In: IEEE Transactions on Very Large Scale Integration (VLSI) vol. 22 (1), pp. 39–48.
Zeng, X.-J., Singh, M., 1996. Approximation accuracy analysis of fuzzy systems as function approximators. IEEE Trans. Fuzzy Syst. 4 (1), 44–63.
Zengqi, S., Zhidong, D., 1996. A fuzzy neural network and its application to controls. Artif. Intell. Eng. 10, 311–315.
Zhai, J., Zhou, J., Zhang, L., Zhao, J., Hong, W., 2008. Anfis implementation in FPGA for power amplifier linearization with digital predistortion. In: International Conference on Microwave and Millimeter Wave Technology (ICMMT 2008), vol. 3, pp. 1474–1476.
Zhu, J., Milne, G., Gunther, B., 1999. Towards an FPGA based reconfigurable computing environment for neural network implementations. In: 9th International Conference on Artificial Neural Networks (ICANN99), vols. 1–2. IEE Conference Publications, pp. 661–666.
Zou, A.-M., Hou, Z.-G., Fu, S.-Y., Tan, M., 2006. Neural Networks for Mobile Robot Navigation: A Survey. Lecture Notes in Computer Science, vol. 3972. Springer

Please cite this article as: Bosque, G., et al., Fuzzy systems, neural networks and neuro-fuzzy systems: A vision on their hardware implementation and platforms over two decades. Eng. Appl. Artif. Intel. (2014), http://dx.doi.org/10.1016/j.engappai.2014.02.008