STUDY OF LOW POWER, LOW VOLTAGE ANALOG REALIZATION OF PROGRAMMABLE CELLULAR NEURAL NETWORKS

By

NASIR ALI KANT

Department of Electronics and Instrumentation Technology,
University of Kashmir,
Hazratbal, Srinagar,
190006

January 2014

A Dissertation submitted to the Faculty of Applied Sciences and Technology in partial fulfilment of the requirement for the Degree of Master of Philosophy in Electronics.

Certificate

This is to certify that ***Mr. Nasir Ali Kant*** has worked under my supervision for the dissertation entitled, "***Study of Low Power, Low Voltage Analog Realization of Programmable Cellular Neural Networks***" and the work is truly commendable of consideration for the award of the degree of Masters of Philosophy in Electronics.

It is further certified that

(i) The dissertation embodies the work of the candidate.

(ii) The candidate worked under my supervision for the period required under statues.

(iii) The candidate has put in the required attendance in the Department.

(iv) The conduct of the candidate remained very good during the period of the research.

(Prof. N. A. Shah)
Supervisor
Dept. of Electronics and
Instrumentation Technology
University of Kashmir
Srinagar-190006
Dated:

(Prof. G. M. Bhat)
Head of Department,
Department of Electronics and
Instrumentation Technology
University of Kashmir
Srinagar-190006
Dated:

Candidate's Declaration

I hereby declare that the work presented in this dissertation "***Study of Low Power, Low Voltage Analog Realization of Programmable Cellular Neural Networks***" in partial fulfilment of the requirements for the degree of ***Masters of Philosophy*** and submitted in the Department of Electronics and Instrumentation Technology, Faculty of Applied Sciences and Technology, University of Kashmir, Srinagar, has entirely been done by me under the supervision of ***Prof. N. A. Shah***.

I, further, declare that the work contained in the dissertation is the original research work conducted by me and has not been submitted in part or full, to any other University or Institute for the award of any degree.

(Nasir Ali Kant)

This is to certify that the above statement made by the candidate is correct to the best of my knowledge.

(Prof. N. A. Shah)
Supervisor
Department of Electronics and Instrumentation Technology
University of Kashmir
Srinagar-190006

The Viva-voce examination of Mr. Nasir Ali Kant, Research Scholar, has been held on…………………………………

Supervisor Head of the Department External Examiner

Acknowledgements

I am thankful to Almighty Allah for guiding me to the right path & for giving me the strength to undertake this work....

Words are often too less to reveal one's deep regards, I take the opportunity to express my deep sense of gratitude and respect to all those who helped me in this work.

This work would not have been possible without the encouragement and able guidance of my supervisor, Prof. N. A. Shah. His enthusiasm and optimism made this experience really enjoyable. I would like to express my deep sense of gratitude towards Prof. G. M. Bhat, Professor and Head, Department of Electronics and Instrumentation Technology, University of Kashmir for his support, guidance & encouragement and for providing necessary facilities in the Department. Further, I wish to express my appreciation to Dr. Farooq Ah. Khanday Asst. Professor Dept. of Electronics & I.T. University of Kashmir for his support & constant encouragement. Special thanks to faculty members of the Department of Electronics, University of Kashmir: Dr. M. Tariq Banday, Dr. Shabir Ahmad Parah, Dr. Javaid Ahmed Sheikh and others who directly or indirectly helped me in the work. Very special thanks to all the technical and non-teaching staff of the department for their valuable encouragement and support. I would take opportunity to thank the research scholars of the Department of Electronics, University of Kashmir: Mr. M. Rafiq Beigh, Mr. Imran Yaseen, Mr. Zubair Ahmad Bangi, Mr Javeed Iqbal and others for their assistance, encouragement and discussions. I would also like to thank all my friends and relatives for their valuable suggestions and support. I am highly indebted to my Family, my Sister and her family for their unconditional support and love.

Nasir Ali Kant

Abstract

"Neural network" is the term used to refer to a circuit of neurons that perform information processing. They are highly complex, nonlinear systems (with huge degree of freedom) which employ different principles of information processing than those of the conventional algorithmic computation of modern computer systems. The basic computational units of neural networks are neurons which are simulated mathematically on a computational substrate (e.g. a standard computer, an electronic circuit, etc.).

Cellular Neural Network (CNN) being the derivative of Artificial Neural Network (ANN) is also a parallel computing paradigm with the difference that communication is allowed between neighbouring units only. CNNs are characterized by simplicity of operation. The network consists of a large number of nonlinear processing units (cells) that are equally spread in the space. Each cell has a simple function that takes an element of a topographic map and then interacts with all cells within a specified sphere of interest through direct connections. CNNs have attracted the attention of a wide variety of scientists, due to their intrinsic parallel computing power. They find their use in different fields e.g., the fields of image and video processing, robotics.

A variety of problems can be solved with neural networks in the areas of Image Processing, Control Systems etc. and most of them have been demonstrated by employing their software based designs. While designing the software based solutions, many special features of neural networks does not persist as the employed machines perform many of their processes in a sequential manner. Therefore to fully utilize the application potential of ANNs, they need to be designed in hardware. Despite many years of studies involving neural networks, it is only due

to advancement in programmable hardware that actual implementation for study and their applications have become practical. There are many examples of neural network codes running on von Neumann computers, but as per the published reports, there is still unavailability of commercial Neural Networks (NNs) implemented in hardware.

In the hardware implementation of neural network, it is important to consider flexibility and power consumption in order to satisfy a wide range of applications. So designers focus on circuits which consume very little power per connection allowing for a high number of connections per neuron. The hardware implementation of the neural networks can be achieved through analog or digital means. However, analog realization of the neural networks provides a fast and power efficient realization compared to the digital realization as the later employ digital processors which often work in sequential way. In addition, the flexibility in analog hardware is added by employing electronic tunablity feature.

According to data that provides information about the near future of semiconductor technology, International Technology Roadmap for Semiconductors (ITRS), in 2013-14, the supply voltage of digital circuits in 32 nm technologies will be 0.5 V. Therefore, the trend for the implementation of analog integrated circuits is the usage of low-voltage building blocks that use a compatible single low-voltage power supply.

In order to achieve all the features of contemporary hardwired neural networks, companding (compression-expansion) technique is a very promising subclass of Low-Voltage Low-Power circuit design technique. The technique has three types: the Log-Domain (LD), the Square-Root Domain (SRD) and the Sinh-Domain (SD). The basic operation in all the three types is same i.e. the input current is first converted into a compressed voltage, the compressed voltage is then processed by the companding core and finally the output compressed voltage is converted into a linear current. The companding technique provides the features of: resistor less designs, electronic tunability and capability of operation low-voltage environment which are the primary requirements of the contemporary Very Large Scale Integration (VLSI) design. Among the three companding techniques, the inherent class-AB operation of SD technique offers the capability for handling signals greater than the bias current, leading to a further power saving.

Therefore, the present investigation was primarily concerned with the study and design of low-voltage and low-power companding ANNs. The work includes the study about: the building blocks required in implementing low-voltage and low-power Sinh-Domain Companding ANNs; the implementation of various analog activation functions for ANNs.

The low-voltage design of three Activation Functions (AFs), Tanh, Unipolar Sigmoidal, and Bipolar Sigmoidal, using SD technique were designed. The SD blocks have been implemented using MOS transistors in weak inversion which ensures the reduction of supply voltage operation of the circuits in addition to that provided by the companding technique itself. The AFs are subsequently used to design the neural network which have been trained in MATLAB environment to perform AND, OR, NOT, NAND, NOR, and XOR logic functions.

The investigations on these designs are based on the SPICE simulations using model parameters of the BSIM 0.35µm CMOS process MOS transistors. The performance of each circuit has been validated by comparing the characteristics obtained using simulation with the results present in the open literature.

The proposed designs could not be realized in silicon due to non-availability of foundry facility at the place of study.

Table of Contents

Chapter 4

Chapter 5

Chapter 6

Appendix

Table of Figures

Table of Tables

1.1 Introduction

In spite of the fact that different aspects of computational complexity have given rise to different complex computer architectures, the concept of scientific computing has not changed during the last six-seven decades. A computer is still built with stored programmability, i.e. with the algorithm as the underlying mechanism [1]. When Alan Turing introduced his abstract machine in 1936 it was meant to consist of a tape of symbols, a header to read/write the symbols, a state register and finally an action table that tells the machine what to do next. About ten years later, the foundation established by Turing was adopted in von Neumann's computer architecture. In general, a von Neumann machine stores both the program and the data in a memory that can be unified as in a Princeton architecture or separate as in Harvard architecture. A control unit features a program counter that keeps track of how instructions are executed on the arithmetic and logic unit. The program is executed sequentially in line with human thinking, which is the main reason for von Neumann machine to gain worldwide acceptance and to quickly become the fundament of future digital computing devices [2]. But being sequential, architectures based on von Neumann machine are characterized by low utilization of the computational components. As the execution of each instruction is divided into a number of stages, only the components belonging to the current stage are active while all other units in the architecture remain idle. Furthermore, we know nowadays engineering tasks are characterized by the high complexity of the underlying algorithms. Here, large amounts of information are handled in real-time hence conventional digital computation methods have run into a serious speed bottleneck due to their serial nature.

Also due to the Sensor revolution which means that cheap sensor and Micro-Electro-Mechanical System (MEMS) arrays are proliferating in almost all the conceivable forms. Thousands and millions of generically analog signals are produced waiting for processing. A new computing paradigm is needed. The cited technology assessment reads:

"The long-term consequence of the coming sensor revolution may be the emergence of a newer analog computing industry in which digital technology plays a mere supporting role, or in some instances plays no role at all [3].

For processing analog array signals, the revolutionary Analogic (generic terms for analog and logic) cellular computer paradigm is a major candidate. The core of this computer is a nonlinear cellular neural network (CNN), an array of analog dynamic processors or cells (called neurons). The key features of neural networks are asynchronous parallel processing, continuous-time dynamics, and global interaction of network elements [4]. At the same time, Analogic CNN computers mimic the anatomy and physiology of many sensory and processing organs with an additional capability of stored programmability.

CNNs were, originally proposed by Chua and Yang in 1988 [5], and since then they have been the object of a great deal of research work, concerning both theoretical studies and application-oriented circuit implementations. CNNs are well suited for high-speed image processing tasks; their reported applications cover a much wider range of activities, such as motion detection, classification and recognition of objects, associative memory, solution of partial differential equations, statistical and non-linear filtering, etc. [6]. Recent studies on optical and nano-scale implementations open up new horizons on the atomic and molecular levels. Unlike cellular automata, CNN host processors accepting and generating analog signals, the time is continuous, and the interaction values are also real values. Moreover, CNN becomes a rigorous framework for complex systems exhibiting emergent behaviour and the various forms of emergent computations. This allows not only modelling but also engineering of complex systems.

However, neural networks without learning are rather uninteresting. If the weights of a network were fixed from the beginning and were not to change, neural networks could be implemented using any programming language in conventional computers. But the main objective of building special hardware is to provide a platform for efficient adaptive systems, capable of updating their parameters in the course of time. Furthermore hardware neural networks are much faster compared to their software counterparts owing to the fact that the feature of full parallelism is captured by means of hardware realizations only. CNNs are particularly well suited for an analog VLSI implementation. The recent extension towards the definition of a programmable analogic array computer, the CNN Universal Machine, has opened many new application fields which can be handled through spatial and temporal task sequencing controlled by a stored program [6]. A key feature of CNNs is their potential for high operation speed in the processing of array signals.

Typical CNN chips may contain up to about 200 transistors per pixel (including sensory and processing devices) [7], [8]. On the other hand, practical applications require large enough grid sizes; around 100×100. Thus, CNN designers must confront a large level of complexity. To achieve an acceptable resolution with standard design procedures, a large cell-grid size is required, which results in considerable area occupation, low production yield and high-power consumption [9].

As more and more complex systems are being integrated on the same chip, area minimization is becoming of primary importance. As the size of batteries is now becoming the limiting factor, it is not sufficient to reduce the size of bulky components by integrating them, the reduction of the power dissipation is also very important. As a consequence, the key point is to develop, simultaneously, both low-voltage and low-power operating integrated circuits in order to reduce the battery size and chip area.

In this research program I have tried to achieve the Low-Voltage Low-Power analog CNN design and in this respect Low-Voltage Low-power implementation of various analog activation functions were designed in Sinh-Domain (SD) technique.

1.2 Outline of the dissertation

In this dissertation issues relating to the hardware implementation of an ANN in low power and low voltage design regimes are discussed. The aim is to present a potential solution to the problem of realizing artificial neurons in hardware since most work is currently conducted via software synthesis and modelling.

The outline of the dissertation structure is consequently presented below.

A general overview about neural networks is given in Chapter 2. The biological inspiration for designing ANNs is also presented. Besides CNNs are discussed in details and different types of CNNs is also presented in the chapter. Furthermore the importance of ANNs is also given at the end of the chapter.

The need for the hardware implementation of ANNs is discussed in Chapter 3. Besides different types of hardware implementation of ANNs and the complications in hardware implementation are also mentioned. Furthermore the special emphasis is lead on the hardware implementation of CNNs along with the different approaches of achieving its hardware.

Chapter 4 introduces to Low-Voltage Low-Power Analog Design techniques along with a brief discussion on companding techniques i.e. Log-Domain (LD), Square-Root-Domain (SRD) and Sinh-Domain (SD). Towards the end, the operators and building blocks required to design SD classification are discussed in detail.

Chapter 5 includes the designing of analog activation functions (unipolar, bipolar sigmoidal and Tanh) utilizing SD companding technique. Besides this the designed activation functions are used in designing single-layer and two-layer perceptron whose weights are trained in such a manner that the network implements various logic gate functions.

2.1 Neural Networks

"Neural network" is the term used to refer to a circuit of neurons that performs information processing. They are highly complex, nonlinear systems (with huge degree of freedom) which employ different principles of information processing than those of the conventional algorithmic computation of modern computer systems. The basic computational units of neural networks are neurons which are simulated mathematically on a computational substrate (e.g. a standard computer, an analogue circuit, etc.).

2.2 The Biological Inspiration for Artificial Neural Networks (ANNs)

The roots and inspiration for ANNs are drawn from biological nervous system. Such biological system or wetware consists of a multitude of simple processing elements which are connected together in a massively parallel architecture.

The human brain is indeed a triumph of nature as it consists of many neurons of different varieties but the general format remains almost the same and is illustrated in Figure 2.1. Neurons are cells that send and receive electro-chemical signals to and from the brain and nervous system. The neuron consists of a cell body (or soma) with branching dendrites (signal receivers) and a projection called an axon, which conduct the nerve signal. The terminating points of the divided axon form transmitting connections to the dendrites of other neurons or connect directly to the neurons via synaptic junctions or synapses. The axon terminals transmit the electro-chemical signal through synapse.

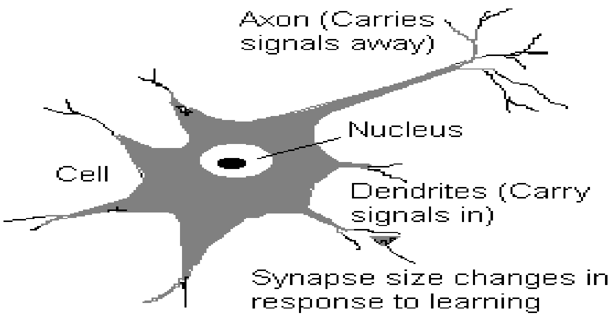

Axon (Carries signals away)
Nucleus
Cell
Dendrites (Carry signals in)
Synapse size changes in response to learning

Figure 2.1: A simple Neuron.

A complex chemical process is involved in signalling from one neuron to another with chemicals released from the sending side of the synapse. The effect of these chemical releases is to alter the electrical potential within the cell body. If the cell potential reaches certain given level, the neuron is activated releasing a fixed strength and duration signal along the axon to other neurons. A recovery period follows before the neuron is able to fire again once it is fired. Individual cells can achieve limited task, but the collective behaviour of these structures of biological formations performs a useful task in the embodying organism. Conservatively, it has been estimated that there are at least 10^{11} neurons in the human brain with 10^{14} interconnections.

Also an important point about neural networks (NNs) which needs to be mentioned here is that they are not a static i.e., the strengths of inter connections vary with time, new ones are formed and old ones may decay away. Due to the large quantity of parallelism there is redundancy built in to the system and a level of fault tolerance is available. Rather than being explicitly programmed, a NN evolves to perform an action by learning and adaption. Thus, given that the network changes through damage or the network has to increase its functionality, it is able to adapt to the new situation.

Given the above rudimentary description of a neuron's behaviour two main approaches can be adopted for the study and development of

ANNs. One approach is to study, model and possibly build analogous devices as accurately as possible. The second is to draw upon ideas from actual systems and develop simple processing element exemplar with in a massively parallel architecture. The former approach is normally adopted by biologists and psychologists in order to determine the functioning of the brain and nervous system. The latter approach is usually followed by engineers in pursuit of a system which will perform a computationally useful task.

But few important questions:

- Why study and develop Artificial Neural Networks at all?
- What task or tasks could they be used to perform?
- What benefit can they offer beyond a traditional von Neumann architecture machine?

By answering the latter two questions, hopefully a more complete reason for the study of ANNs will become apparent. Within a traditional computer, a failure in a processing section is catastrophic in terms of system performance; this is not necessarily the case with a NNs. Benefits of ANN are their potential robustness and only a gradual degradation in performance, but not the network stops working, if an area of the network becomes damaged. Furthermore, certainly for rapid exact algorithmic or mathematical operations a traditional computer is excellent but this is not the case for noisy, inexact information processing. Also as it has been stated that much of the interest and power of NNs is the ability they have to adapt and learn from the data presented to them. So the ANN designed will not need to be re-programmed once it is trained which is not the case with simple computer. The ANN has recently been applied in process control, identification, diagnostics, character recognition, sensory prediction, robot vision, and forecasting [10-16] etc.

2.3 The Biological Neural Networks

The networks formed by biological neurons which are connected to carry out the functionalities typical of the nervous system in biological life forms are referred as biological neural networks However, not all multicellular life forms on the earth have a nervous system. For example, sponges are very old life forms comprising colonies of cells which do not have a nervous system to allow electrical communication between the various parts of the body [17]. In particular the presence and the complexity of the nervous system have evolved during the various historical eras.

In the evolutionary scale, with regard to the nervous system, after the sponges it is possible to find the “Radiata” branch: life forms which have a radial symmetry. In these life forms it is possible to identify a “top” and a “bottom”, but not “left” and “right” sides; jellyfish are an example of this class. These life forms have a simple nerve net which allows reactions to external [18].

Life forms having the most complex neural system belong to the “Bilateria” branch. These are the life forms which are (approximately) symmetric with respect to a longitudinal axis, and for which it is possible to define a “left” and a “right” side in addition to a “top” and a “bottom”. This branch includes the human race which has the most complex nervous system known in nature [19], [20].

More generally, in the “Bilateria” branch, the “Vertebrate” subphylum occurs. In individuals of this subphylum, the nervous system can be divided into two interconnected halves [21]: the peripheral nervous system and the central nervous system. However, these two parts differ only from an anatomical point of view as depicted in Figure 2.2.

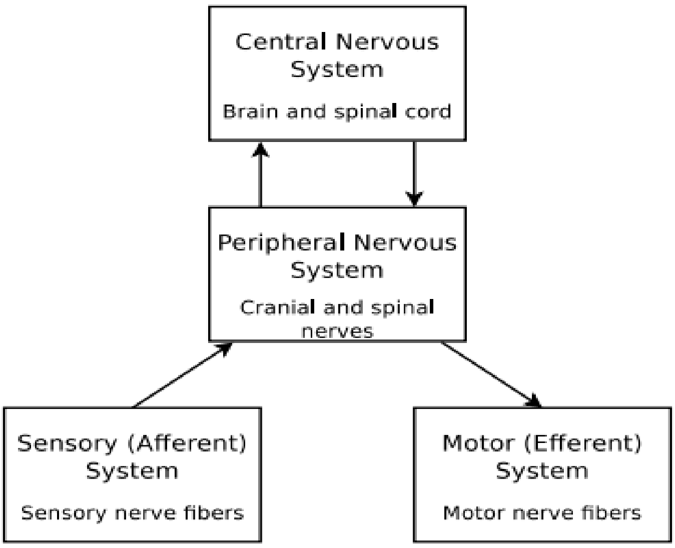


Figure 2.2: Simplified structure of the nervous system in vertebrate animals.

Considering the human nervous system, human brain consists of approximately 10^{11} computing elements called neurons. They communicate through a connection network of axons and synapses having a density of approximately 10^{4} synapses per neuron. The hypothesis regarding the modelling of the natural nervous system is that neurons communicate with each other by means of electrical impulses [22]. The neurons operate in a chemical environment that is even more important in terms of actual brain behaviour. The brain thus can be considered to be a densely connected electrical switching network conditioned largely by the biochemical processes. The vast NN has an elaborate structure with very complex interconnections. The input to the network is provided by sensory receptors. Receptors deliver stimuli both from within the body, as well as from sense organs when the stimuli originate in the external world. The stimuli are in the form of electrical impulses that convey the information into the network of neurons. As a result of information processing in the central nervous systems, the effectors are controlled and give human responses in the form of diverse actions. We thus have a three-stage system, consisting of receptors, neural network, and effectors, in control of the organism and its actions.

A lucid, although rather approximate idea, about the information links in the nervous system is shown in Figure 2.3. As we can see from the figure, the information is processed, evaluated, and compared with the stored information in the central nervous system. The necessary commands are generated there and transmitted to the motor organs. Notice that motor organs are monitored in the central nervous system by feedback links that verify their action. The implementation of commands is controlled with the help of both internal and external feedbacks. As can be seen, the overall nervous system structure has many of the characteristics of a closed-loop control system.

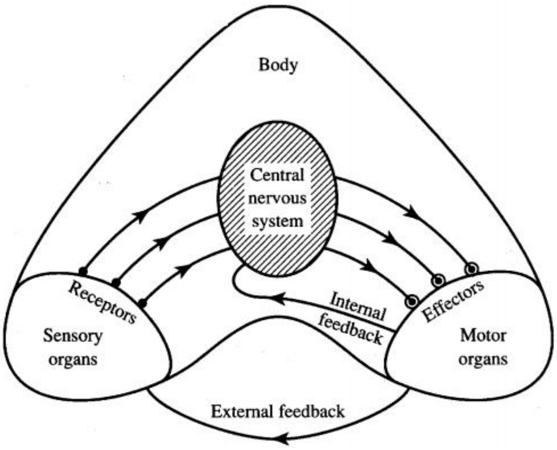


Figure 2.3: Information flow in nervous system.

2.4 *Artificial Neural Networks*

The simplest definition of a neural network, more properly referred to as an 'artificial' neural network (ANN), is provided by the inventor of one of the first neuro-computers, *Dr. Robert Hecht-Nielsen* [23]. He defines a neural network as:

“...a computing system made up of a number of simple, highly interconnected processing elements, which process information by their dynamic state response to external inputs.”

There are a number of different answers possible to the question of how to define artificial neural networks. At one extreme, the answer could be that these are simply a class of mathematical algorithms, since a network can be regarded essentially as a graphic notation for a large class of algorithms. Such algorithms produce solutions to a number of specific problems. At the other end, the reply may be that these are synthetic networks that emulate the biological neural networks found in living organisms. In light of today's limited knowledge of biological neural networks and organisms, the more plausible answer seems to be closer to the algorithmic one. Thus we can say ANNs are the result of academic investigation that uses mathematical formulations to model nervous system operation. They are circuits made by the interconnection of artificial neurons, which mimic the behaviour of biological neurons. The circuits or simulators comprise hardware (analogue or digital) or software (digital) components which compute mathematical models of biological neurons and biological synapses.

Our knowledge about actual brain functions is still limited, and no model has been successful in duplicating the performance of the human brain. Despite ANNs have undoubtedly been biologically inspired, but the close correspondence between the biologic neural systems and ANN is still weak. Vast discrepancies exist between both the architectures and capabilities of natural and artificial neural networks. Therefore, the brain has been and still is only a metaphor for a wide variety of neural network configurations that have been developed [24].

Talking about the designing of ANNs, different approaches have been adopted resulting in different architectures like Perceptron, Madaline, and Hopfield Networks etc. One class of architectures that are designed taking the inspiration from the neural networks with the difference that communication is allowed between neighbouring units only is called as Cellular Neural Networks (CNNs). They are the class of neural networks whose concept was first introduced by Chua and Yang, in 1988, to efficiently perform large time-consuming tasks in real-time by using an array of simple, nonlinearly coupled dynamic circuits [5].

2.5 Cellular Neural Networks (CNNs)

Cellular Neural Network (CNN) is novel class of information-processing systems. The concept of CNN rests on two major sources of inspiration. The architecture possesses some of the key features of NNs [25], such as continuous time dynamics and global interaction of the network elements, which allows for real-time signal processing. On the other hand, it inherits the feature of local interconnectivity from the world of Cellular Automata [26], which makes it suitable for VLSI implementations.

Cellular Neural Networks are characterized by simplicity of operation. The network consists of a large number of nonlinear processing units; called cells; that are equally spread in the space. Each cell has a simple function that takes an element of a topographic map and then interacts with all cells within a specified sphere of interest through direct connections. Due to their intrinsic parallel computing power, CNNs have attracted the attention of a wide variety of scientists in, e.g., the fields of image and video processing, robotics and higher brain functions [27–29].

Since its inception, cellular network has been thoroughly investigated. Two-dimensional networks in which each cell is connected physically only to its eight nearest neighbours are studied in great detail in the literature, although cells in cellular network can be arranged in several spatial configurations. The radius of effect or the radius of neighbourhood r_N determines how a particular cell is affected by the neighbouring cells. For example, for $r_N = 1$ the neighbourhood includes the cell itself and its eight adjacent cells which directly affect it; for $r_N = 2$ the neighbourhood includes the cells mentioned for $r_N = 1$ plus16 additional cells which indirectly affect the central cell; for $r_N = 3$ the neighbourhood includes the cells mentioned for $r_N = 2$ plus 24 additional cells which again indirectly affect the central cell; and so on. In other words, $r_N = 1, 2,$ and 3 neighbourhoods can be respectively called as 3×3, 5×5 and 7×7 neighbourhoods. Each cell has its own input, state, and output. Depending on the nature of these variables (e.g. continuous vs. discrete) and the way in which the dynamics of each cell is defined, the cellular network receives different names in the scientific literature (continuous and discrete-time Chua–Yang CNN [5], full signal range CNN [30], time-derivative CNN [31]).

In upcoming section, the network structure is introduced, as it eases the understanding of CNNs basic equations, and is followed by a brief description of CNN models: Chua and Yang model that is also referred to as Continuous-Time CNN (CT-CNN) and the counterpart Discrete-Time CNN (DT-CNN), Full Signal Range CNN and Time-derivative CNN. The aim is to give an intuitive understanding of the concept, rather than discussing the theory in detail.

2.5.1 Standard CNN Architecture

A massive aggregate of regularly spaced processing units, called cells, forms the CNN architecture. Similar to Cellular Automata [26], any cell in the network is connected only to its neighbour cells, where direct interaction only occurs among adjacent cells. However, other cells are indirectly affected due the propagation effect of the continuous-time dynamics. Furthermore, a CNN of any dimension can be defined theoretically, as illustrated in Figure 2.4, which allows a CNN to handle spatial relations such as topographic maps. The discussion here will be however restricted to the 2-dimensional case.

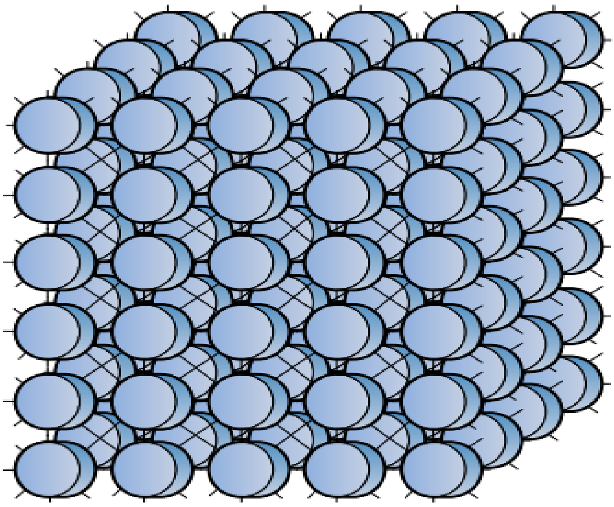


Figure 2.4: CNN, where the globes represent cells and the links represent direct coupling.

2.5.2 Sphere of Influence (Neighbourhood)

In the case of 2-dimensional finite-size CNN, cells are arranged in M rows and N columns, each cell $C(i, j)$ is indexed according to row *i* and column *j* and each cell communicates directly with its sphere of influence $S_r(i, j)$ of radius r, also called r-neighbourhood. Equation 2.1 gives the relation for such a neighbourhood and is defined as the set of cells within a certain distance r to $C(i, j)$, where r $\geq$ 0.

$$S_r(i,j) = \{C(k.l) | \max(|k-i|, |l-j|) \leq r; 1 \leq k \leq M, 1 \leq j \leq N\} \tag{2.1}$$

For instance, we have a 3×3 neighbourhood when r =1 as we obtain a 3×3 matrix in this case as can be seen in Figure 2.5. For r = 2, 5 × 5 neighbourhood and so on. In general, a neighbourhood of size $(2r + 1)^2$ is obtained for certain $r \geq 0$. Figure 2.5 shows different neighbourhood examples, with r = 1, 2 and 3. Observe that when r > N/2, and M = N, a fully connected CNN is obtained, i.e. $S_r(i,j)$ is the entire network.

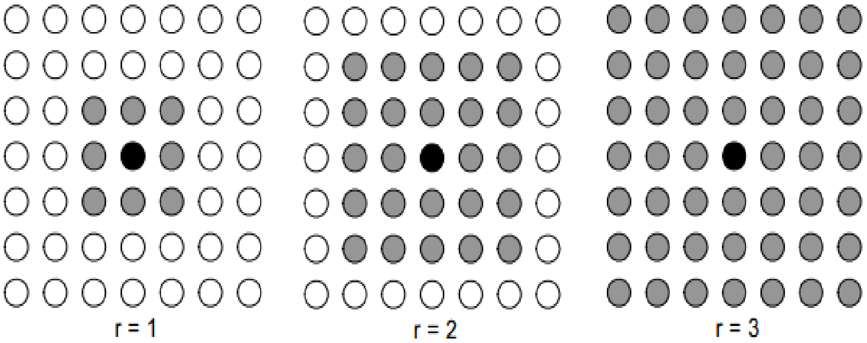


Figure 2.5: Different r-neighbourhoods for the centre cell (black circle). To avoid clutter all interconnections are dropped.

2.5.3 Standard CNN Equations

Considering a case of a cell with no coupling to any other cell in the grid, such a cell, called an isolated cell, is associated with four variables: input $u_{ij} \in R^u$, threshold $z_{ij} \in R^z$, state $x_{ij} \in R^x$, and output $y_{ij} \in R^y$, which are generally the functions of the continuous time t. The cell output value depends on the current state of the cell and also on the input value and the threshold. If the given input is $u_{ij}(t_0)$ at $t = t_0$, an initial state is $x_{ij}(t_0)$ and a threshold is $z_{ij}(t_0)$ the state $x_{ij}(t)$ evolves via the state equation given in Equation 2.2 where the "dot" denotes the time derivate and F is an ordinary nonlinear differential function.

$$\dot{x}_{ij} = F[x_{ij}(t), z_{ij}(t), u_{ij}(t)] \tag{2.2}$$

Now as we know that if the n$^{\text{th}}$ derivative of an unknown function $H:R\rightarrow R$ with respect to a variable h is a function of the lower-order derivatives, i.e. $F\left(h,H,H',H'',H''',.....H^{(n-1)}\right)=H^{(n)}$ then the function H is an ordinary differential. Furthermore, if the differential function is not dependent on the variable h, it is then considered autonomous. Equation 2.2, in this sense, is simply a non-autonomous system of ordinary differential equations [32]. In general, different non-linear functions can be used for different cells, but in almost all known applications the cells are identical and therefore employ the same function.

The operative description of a cell is concluded by the determination of the output $y_{ij}(t)$ by means of a nonlinear function. In most literature, this function is assumed to depend only on the state of the cell, as depicted in Equation 2.3 but it may depend on $x_{ij}(t)$, $z_{ij}(t)$ and $y_{ij}(t)$.

$$y_{ij}(t)=g\left(x_{ij}(t)\right) \tag{2.3}$$

For the quality of the obtained output and the speed achieved, the choice of function g is crucial. There are different functions that can be used but three different types of nonlinear functions are frequently used [33]: (a) threshold, (b) hyperbolic tangent and (c) piece-wise linear functions. The threshold function, commonly referred to as hard-limiter (or Heaviside) function, is binary-valued only and performs a binary decision. The hyperbolic tangent function is generally defined as a strictly increasing continuous s-shaped function and is a special case of the sigmoid function. Figure 2.6(a) shows the nature of this function and Equation 2.4 is used to describe it mathematically. By varying δ (the slope parameter), different sigmoid functions are obtained. As the slope parameter approaches infinity, in the case of Equation 2.4, the sigmoid function simply becomes a threshold function. Finally, the piecewise linear function can be defined as the one that is totally linear with positive slope within a certain interval [-a, a] and saturates outside this interval as illustrated in Figure 2.6(b). The function is mathematically described in Equation 2.5.

$$f(x) = \tanh(\delta x) \tag{2.4}$$

$$f(x) = \begin{cases} 1, & x \geq a \\ \delta x, & |x| < a \\ -1, & x \leq -a \end{cases} \tag{2.5}$$

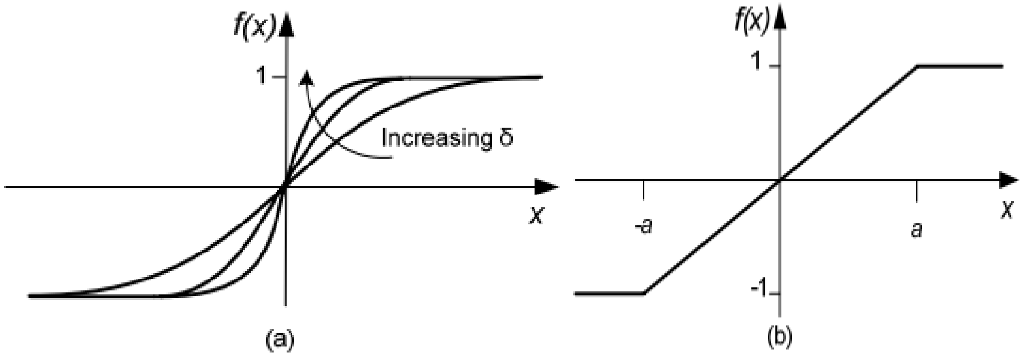


Figure 2.6: Sigmoid function (a) and piece-wise linear function (b).

The State equation standard isolated CNN cell is given in Equation 2.6, which is the most widely used in the literature. Coefficients a_{ij} and b_{ij} are the two weightings coefficients whose value define the contributions of state and input variables while as the threshold is simply assumed to be a scalar quantity of constant magnitude [32]. a_{ij} and b_{ij} are called feed-back and control coefficients respectively as a_{ij} mirrors the effect of the previous output value, while b_{ij} scales the current input value. A threshold z_{ij} is used to adjust the obtained state value into a desired range. The output is usually obtained by using the piece-wise linear function introduced in Equation 2.5 with the interval [-1, 1] and slope δ=1 resulting in Equation 2.7. Assuming all coefficients are linear, the dynamics of the isolated CNN cell are due to the nonlinear output function only.

$$\dot{x}_{ij} = -x_{ij} + a_{ij}y_{ij} + b_{ij}u_{ij} + z_{ij} \tag{2.6}$$

$$y_{ij} = f(x_{ij}) = \frac{1}{2}(|x_{ij}+1| - |x_{ij}-1| = \begin{cases} 1, & x_{ij} \geq 1 \\ x_{ij}, & |x_{ij}| < 1 \\ -1, & x_{ij} \leq -1 \end{cases} \tag{2.7}$$

Equation 2.6 is also commonly referred to as ‘cell dynamics’ as it explains how the state of the cell evolves over time. These dynamics depends upon two constraints: initial condition constraint where the state variable is assumed to be equal to a certain value upon start, and input constraint where input value $u \in [-1,+1]$.

In CNN architecture, generally, each cell is directly coupled to all other cells within the sphere of influence (S_r). The new output is produced by the consumption of both input u_{kl} and output y_{kl} of all available neighbouring cells. Inputs and outputs from cells belonging to S_r of the cell are weighted as b_{kl} and a_{kl} respectively, similar to that of an isolated cell. The state equation of a standard CNN cell can be obtained by simply summing the contributions of all cells in the sphere of influence, and can be written as in Equation 2.8. The output value is still obtained according to Equation 2.7.

$$\dot{x}_{ij} = -x_{ij} + \sum_{kl \in S_r(i,j)} a_{kl}y_{kl}(t) + \sum_{kl \in S_r(i,j)} b_{kl}u_{kl}(t) + z_{ij} \tag{2.8}$$

$$i = 1,2,3.......M; \quad j = 1,2,3........N$$

For solving ordinary differential equation systems, almost all theorems and numerical techniques are formulated in vector form [30].

Hence, it is desirable to express the state equation given in Equation 2.8 in vector form. The vector form of the state equation is given in Equation 2.9. The matrices A and B are $n \times n$ matrices whose nonzero entries are the feed-back coefficient a_{kl} and control coefficient b_{kl}.

$$\dot{x} = -x + Ay + Bu + z \tag{2.9}$$

One may conclude that in the mesh, each CNN cell is a dynamic system whose state evolves according to a prescribed state equation. The cell dynamics are coupled to the cells lying within the sphere of influence with the centre that is located at the position of the cell itself. However, the cell dynamics boundary cells have a great effect on the behaviour of the entire CNN. This will be discussed in boundary conditions. But first, a concise form of the state equation is introduced in the following section.

2.5.4 Cloning Template

Generally, all feedback and control coefficients of Equation 2.8 can be represented by time-dependent nonlinear operators of the coupled values. For simplification we consider them to be time-invariant and real-valued scalars. Furthermore, in the grid these coefficients are identical for all cells. The state Equation 2.8 is written in a more compact form by using the two-dimensional convolution operator defined in [5], in order to simplify the notation and is re-introduction below.

For any 3×3 matrix M, the convolution operator $\otimes$ is defined by Equation 2.10, where $M(m,n)$ denotes the entry in the mth row and the nth column of the matrix, and $m,n \in \{-1,0,+1\}$.

$$M \otimes v_{ij} = \sum_{kl \in S_r(i,j)} M(k-i, l-j) v_{kl} \tag{2.10}$$

$$\dot{x} = -x_{ij} + A \otimes y_{ij} + B \otimes u_{ij} + z \tag{2.11}$$

Equation 2.11 gives the compact form of the state equation which resulted as the weighting coefficients can be grouped in two square matrices: which holds all feedback coefficients and is accordingly called feedback template (A), and the control template (B). Together with the real-valued threshold (also known as bias), they constitutes a so-called cloning template Γ= (A, B, z). The latter term is commonly used to emphasize the property of space-invariance [34]. It is now obvious that in addition to given input and the initial conditions the cloning template Γ also participate in completely determining the dynamic behaviour of the cell.

The matrices in Equation 2.12 show the common notation of feedback and control templates respectively, for the case of 1-neighbourhood.

$$A = \begin{bmatrix} a_{-1,-1} & a_{-1,0} & a_{-1,1} \\ a_{0,-1} & a_{0,0} & a_{0,1} \\ a_{1,-1} & a_{1,0} & a_{1,1} \end{bmatrix} , \quad B = \begin{bmatrix} b_{-1,-1} & b_{-1,0} & b_{-1,1} \\ b_{0,-1} & b_{0,0} & b_{0,1} \\ b_{1,-1} & b_{1,0} & b_{1,1} \end{bmatrix} \tag{2.12}$$

The self-feedback entry i.e. the central entry of the feedback template is of significant importance for the stability of operation of a CNN. In this sense, in many cases, it is desired to decompose the A template in Equation 2.12 as shown in Equation 2.13. Matrices $\hat{A}$ and $\overline{A}$ are called centre and surround feedback template respectively [35].

$$A \;=\hat{A}+\overline{A} \;=\begin{bmatrix}0 & 0 & 0\\ 0 & a_{0,0} & 0\\ 0 & 0 & 0\end{bmatrix}+\begin{bmatrix}a_{-1,-1} & a_{-1,0} & a_{-1,1}\\ a_{0,-1} & 0 & a_{0,1}\\ a_{1,-1} & a_{1,0} & a_{1,1}\end{bmatrix} \tag{2.14}$$

The neighbourhood determines the number of the real-valued template coefficient. Three simple classes of special importance and are worth mentioning [17]. These classes are briefly introduced below.

- *Zero-feedback template*: in this case all the feedback coefficients are zero. Equation 2.15 describes the dynamics of each cell of a zero-feedback CNN.

$$\dot{x}=-x_{ij}+B\otimes u_{ij}+z \tag{2.15}$$

- *Zero-input template*: Zero-input CNNs are also termed as autonomous CNNs. In the case of Zero-input CNNs all control coefficients are zero. The dynamics of each cell of a zero-input CNN is described by Equation 2.16.

$$\dot{x}=-x_{ij}+A\otimes y_{ij}+z \tag{2.16}$$

- *Un-coupled template*: The dynamics of each cell of uncoupled CNN is described by a scalar nonlinear ordinary differential equation as shown in Equation 2.17. In the case of uncoupled CNN all surround control coefficients are zero, i.e. $A \;=\hat{A}$.

$$\dot{x}=-x_{ij}+a_{0,0}f(x_{ij})+B\otimes u_{ij}+z \tag{2.17}$$

2.5.5 Boundary Conditions

So far no restrictions have been imposed on the size of the CNN grid. Actually, the conceptual discussion carried out till now is valid for infinite CNN grids, but when CNNs of finite size are considered it suffers from a number of complications.

Equations 2.8 and 2.10 are not completely defined in the case of cells whose sphere of interest $S_r(i, j)$ extends outside of the boundary of the grid. Thus we can divide CNN cells into two different categories: regular cells and boundary cells. A regular cell can be defined as the one for which for a certain neighbourhood "r" it has $(2r+1)^2$ neighbour cells. All the other cells with less than $(2r+1)^2$ neighbours belong to the category of boundary cells. Also it is to be noted that all boundary cells are not edge cells, if r > 1 (Figure 2.7). Edge cells lie on the perimeter, i.e. they are the outermost boundary cells.

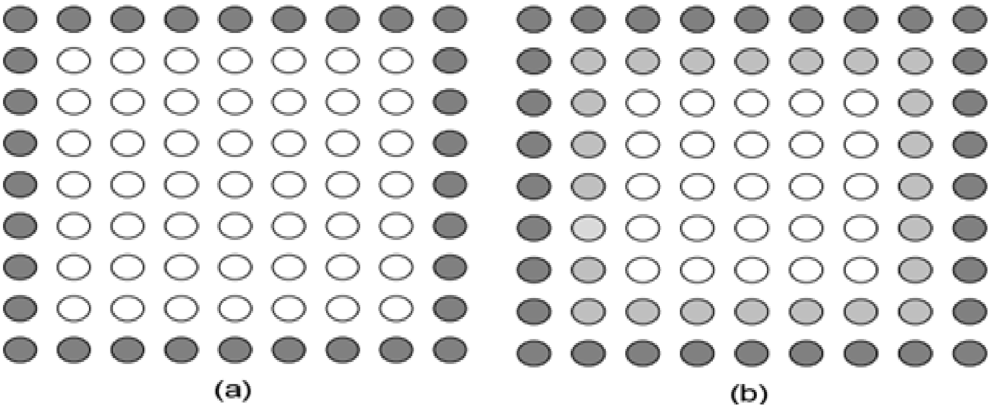


Figure 2.7: When r = 1, boundary cells coincide with edge cells (a) but for 1 boundary cells (light grey) are not located on the edges only (b).

Furthermore, due to the nature of indirect propagation, the absence of neighbouring cells doesn't affect the boundary cells only, but it has a great impact on the dynamic behaviour of the entire network. Thus a different interpretation of boundary cell is required. Traditionally, virtual CNN cells are introducing around the grid for remedying the boundary cell problem. The virtual cells complete the sphere of influence of all boundary cells and each virtual cell is associated with a virtual state, a virtual input, a virtual output and a virtual threshold [35]. These virtual parameters are specified via various boundary conditions. Three of the most commonly used boundary conditions for 1-neighborhood, as described in [34], and are rephrased as under:

- *Fixed (Dirichlet) boundary condition*: predefined constant values are assigned for virtual state and input of each virtual cell or in other words we can say that the boundaries of the network are tied to fixed values.
- *Zero-flux (Neumann) boundary condition*: usually, this condition applies to CNNs with no input, i.e. $u_{ij}=0$. In this case virtual cells are considered to have the same state and input values as their direct neighbouring boundary cells.
- *Periodic (Toroidal) boundary condition*: Here the first and last rows of the network are identical and the top the bottom column are identical as shown in Figure 2.8. Thus, the CNN behaves as if it is joined onto itself forming a torus.

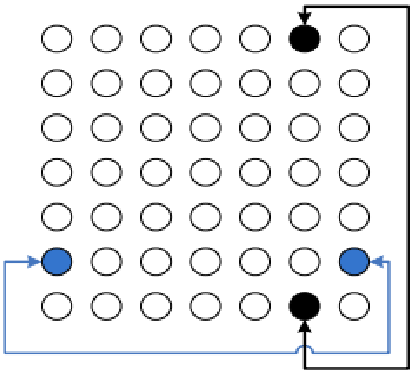


Figure 2.8: In periodic boundary condition the CNN is joined onto itself.

Figure 2.9 shows a block diagram of a CNN cell based on Chua-Yang's model. Each cell receives/generates a weighted contribution from/for each cell in its neighbourhood. Incoming contributions are added directly at the input of the cell and integrated. The output of the integrator represents the state variable x_{ij}, which is feedback to the input with a scaling factor -1 that represents a normalization factor for the rest of the weights. A nonlinear block is then used to generate the output y_{ij}, which is replicated and scaled to generate the out-coming contributions for the cells in the neighbourhood. The implementation of the control and the offset terms is not shown in the figure. Variables x_{ij} and y_{ij} in Figure 2.9 are identical while the cell is within its linear region. Otherwise y_{ij} , saturates at ±1 (after normalization), while x_{ij} goes beyond these limits. This means that the output signal range of the integrator must be wide enough as the circuit implementation must provide room for these larger variations of the state variable.

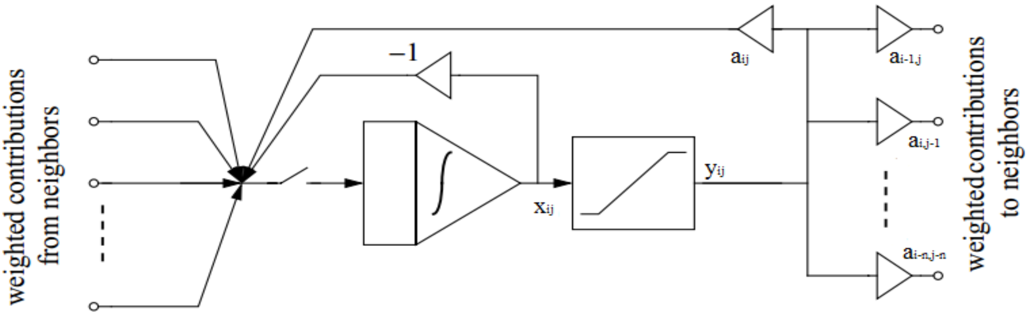

weighted contributions from neighbors
−1
a_{ij}
$a_{i-1,j}$
$a_{i,j-1}$
$a_{i-n,j-n}$
x_{ij}
y_{ij}
weighted contributions to neighbors

Figure 2.9: Block diagram of a CT CNN cell circuitry based on Chua-Yang's model. Control template and offset-term implementation are not shown.

2.6 Discrete-Time (DT) CNN

A DT-CNN poses a regular grid of locally connected cells that can theoretically, have any dimension, but again the focus here will be on the 2-dimensional case only for simplicity. The DT-CNN is a clocked system which is in contrary to CT-CNNs. The dynamics of DT-CNN are described by a set of discrete equations. This enforces the introduction of slightly different notations, so for that reason notations used in [1] are used. Furthermore, it is important to emphasize that the space invariance feature is assumed here as well and the size of the grid is assumed to be finite, unless it is explicitly stated otherwise.

The coordinate of the position of the cell in the grid, i.e. row c_i and column c_j helps in identifying the cell c. The cell communicates directly with all the neighbour cells belonging to the r-neighbourhood. A slight modification is made to the definition of r-neighbourhood given in Equation 2.1 to reflect the new notation of the cell but as depicted in Equation 2.18 the relation remains unchanged. The character d represents any cell belonging to the neighbourhood of cell c, including c itself.

$$N_r(c) = \{d \in Z^2 \,|\, \max(|d_i - c_i|, |d_j - c_j|) \leq r\} \tag{2.18}$$

The state x^c, of a cell c, mainly depends on the contribution of the time-independent input u^d and the time-variant output y^d. These dependencies at a discrete time k are depicted in Equation 2.19.

$$x^c(k) = \sum_{d \in N_r(c)} a_d^c y^d(k) + \sum_{d \in N_r(c)} b_d^c u^d + i^c \tag{2.19}$$

The real-valued coefficients, the feedback coefficients, the control coefficients and the threshold/bias values are represented by a_d^c, b_d^c and i^c respectively. While a_d^c reflect the contribution from the output of all cells in the neighbourhood, b_d^c describe the dependency on the inputs of the neighbours and i^c is added to adjust a cell's threshold. For DT-CNN, coefficients are commonly expressed in a compact form by means of matrices, as was also seen in the case of CT-CNN. The cloning template $\Gamma = \langle A, B, i \rangle$ that is often thought of as an elementary DT-CNN program or DT-CNN instruction [1] is thus used to specify the spatially invariant DT-CNNs.

A compact state equation is, in Equation 2.20, by substituting Equation 2.10 into Equation 2.19. Equation 2.20 is obviously equivalent to Equation 2.11. All cells in the DT-CNN have identical functionality, because of that cell subscripts can be omitted as shown in Equation 2.21.

$$x^c(k) = A \otimes y^d(k) + B \otimes u^d + i \tag{2.20}$$

$$x(k) = A \otimes y(k) + B \otimes u + i \tag{2.21}$$

An initial output $y^c(0)$ is of crucial importance for the dynamic behaviour of the network (compare with the initial condition constraint in section 2.5.3) in the case of non-zero feedback coefficients. On the other hand the output of the system remains constant after the first time step if all feedback coefficients are equal to zero.

The functionality of the system in accordance to CT-CNN is defined by the cloning template Γ.The dynamic behaviour of a DT-CNN is completely determined by cloning template together with the activation pattern u and the initial output $y^c(0)$. Figure 2.10 illustrates the functionality of a cell, explains schematically the influence of all involved parameters [1]. The data comes in over the u^d input and is modified through the control template B, while the interaction with the neighbouring cells is gathered through the y^d input and modified through the feedback template A. All modified input values are summed and discriminated after application of the bias i.

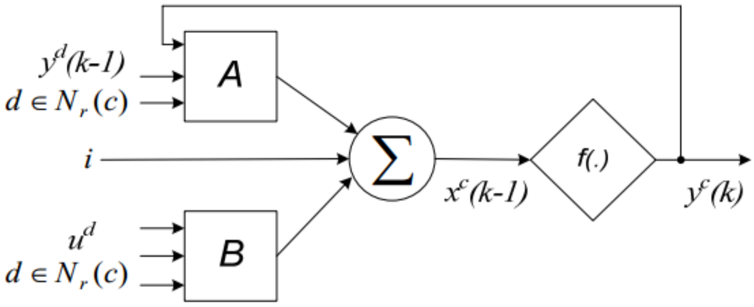
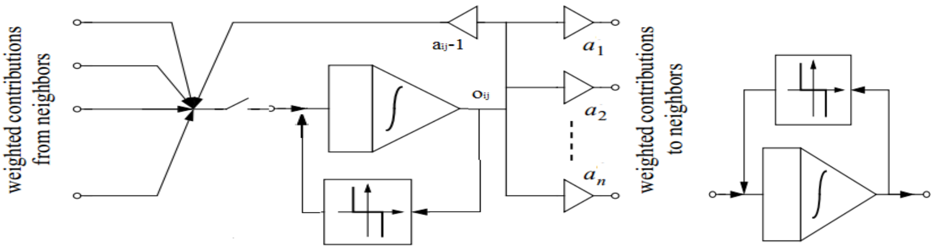


Figure 2.10: A schematic diagram illustrating a DT-CNN cell. [23]

2.7 Full Signal Range CNN

As we have seen in CT-CNN (of Figure 2.9) that the output signal range of the integrator must be wide enough which increases the circuit complexity. This reason motivates searching for alternative models, and resulted in Full Signal Range CNN (FSR-CNN). Again considering CT-CNN model (Figure 2.9) it appears from a functional point of view, that x_{ij} may be restricted to the unitary interval as well and that the state and output variables may be merged into a single one with no influence on the functionality. This is actually the idea behind the FSR model, whose block diagram is depicted in Figure 2.11(a). The main modification centres on the integrator, which in Figure 2.9 is designed with sufficient output signal range to avoid its saturation during the dynamic evolution of the cell, while in Figure 2.11(a), it is designed to have saturation limits equal to those of the output variable. Thus, this block responds to the usual integral law while its output o_{ij} (modified state variable) is within its saturation limits (after normalization). Whenever the state variable reaches +1 (alternatively−1), it remains clamped to this value as long as the input of the integrator is non-negative (alternatively non-positive), without modifying its value. As soon as the integrator input becomes negative (alternatively positive), the modified state variable will enter the linear region again. Figure 2.11(b) shows a conceptual block diagram to realize the integrator.


weighted contributions from neighbors
a_{ij}-1
o_{ij}
a_1
a_2
a_n
weighted contributions to neighbors

(a) (b)

Figure 2.11: (a) Block diagram of CNN cell circuitry using the FSR model, (b) Integrator with saturation. [30]

2.8 Time-Derivative CNN

Time-derivative CNN (TDCNN) [37] extends the original CNN description in [5] by adding derivative connections between cells. A time-derivative linear CNN is described by

$$\begin{aligned}\frac{dx(i,j,t)}{dt} = &\sum_{m,n=-r}^{r} A(m,n)x(i+m,j+n,t) + \sum_{m,n=-r}^{r} B(m,n)u(i+m,j+n,t) \\ &+ \sum_{q\in 1..D}\begin{bmatrix}\displaystyle\sum_{m,n=-r}^{r} A_q(m,n)\frac{dx^q(i+m,j+n,t)}{dt^q} \\ \displaystyle\sum_{m,n=-r}^{r} B_q(m,n)\frac{du^q(i+m,j+n,t)}{dt^q}\end{bmatrix}\end{aligned} \tag{2.22}$$

The first two terms on the right hand side of (2.22) are the same as in the case of original CNN equation, *A* and *B* are feedback and feed-forward cloning templates, *u* is input and *x* denotes the state and output of the linear network. A_q and B_q are defined as qth derivative of feedback and feed-forward templates, respectively, and r denotes the neighbourhood of the CNN. It has been shown that by adding first order derivatives of the outputs of the neighbouring cells to the original CNN equation, bandpass spatiotemporal filters can be realized [20, 37]. For these first derivative TDCNNs, Equation 2.22 becomes

$$\begin{aligned}\frac{dx(i,j,t)}{dt} = &\sum_{m,n=-r}^{r} A(m,n)x(i+m,j+n,t) + \sum_{m,n=-r}^{r} B(m,n)u(i+m,j+n,t) \\ &+ \sum_{m,n=-r}^{r} A_1(m,n)\frac{dx(i+m,j+n,t)}{dt}\end{aligned} \tag{2.23}$$

2.9 Application of ANNs

ANNs are powerful tools for modelling, especially when the underlying data relationship is unknown. Besides ANNs can identify and learn correlated patterns between input data sets and corresponding target values. They have seen an explosion of interest over the last few decades, as they perform “intelligent” tasks similar to those performed by the human brain, and are being successfully applied across an extraordinary range of problem domain, in the areas as diverse as medicine, engineering, biology, physics and finance. From a statistical perspective NNs are interesting because of their potential use in prediction and classification problems. ANNs acquires knowledge through learning and after training, ANNs can be used to predict the outcome of new independent input data.

ANNs imitate the learning process of the human brain and can process problems involving non-linear and complex data even if the data are imprecise and noisy. A very important feature of these networks is their adaptive nature, where “programing” is replaced by “learning by example” in solving problems. This feature makes such computational models very appealing in application domains where training data is readily available but one has little or incomplete understanding of the problem to be solved.

The applications of ANNs are limitless. They have been used in classification problems, such as identifying underwater sonar currents, recognizing speech, and predicting the secondary structure of globular proteins. In time-series applications, ANNs have been used in predicting stock market performance. Besides they have been applied within the medical domain for clinical diagnosis [38], image analysis and interpretation [39], signal analysis and interpretation, drug development [40] and to design and discover new drugs in pharmacology [41]. Furthermore, their application also covers activities such as motion detection, classification and recognition of objects, associative memory, and solution of partial differential equations, statistical and non-linear filtering and much more.

2.10 Conclusion

Neural networks are discussed in brief along with the biological inspiration for the implementation of ANNs. Besides this an introduction about CNNs is given in this chapter along with its various types. Few of the applications of ANNs are mentioned in the end of the chapter.

3.1 Introduction

Implementation of neural networks can be achieved either by software simulation in conventional computers or by a special hardware solution. Most of the Neural Network architectures and algorithms exist only as mathematical models or are implemented as a software solution upon a standard von Neumann style architecture machine, and despite the high speed of modern computer platforms for the simulation of ANNs, the platforms are not fast enough for very large networks or real-time applications. The hardwired neural network is capable of dramatically decreasing execution time as the feature of full parallelism is captured by means of hardware realizations only. Furthermore, the main objective of building special hardware is to provide a platform for efficient adaptive systems, capable of updating their parameters in the course of time as the main problem is the time required for the learning process, which can increase exponentially with the size of the network. Neural networks without learning, however, are rather uninteresting and if the weights of a network were fixed from the beginning and were not to change, neural networks could be implemented using any programming language in conventional computers.

The first modelling of neurons was devised by Pitts and McCulloch in 1943. In their view a neuron, which they called a formal neuron, is a logical gate with two possible internal states, active and silent. A neuron has a few entries provided by the outputs of other neurons. The entries are summed up and the state of the neuron is determined by the value of the resulting signal with respect to a certain threshold, if the signal is larger than the threshold the neuron is active otherwise it is silent.

3.2 Hardware implementation Artificial Neural Networks

Hardware implementation of neural networks can be achieved in different ways. They can be Analog, Digital or Hybrid depending upon the way signals is processed. A brief introduction of these is given in the next sections.

3.2.1 Analogue Artificial Neural Networks

In the analog implementation of neural networks, a coding method is used in which signals are represented by currents or voltages. This allows us to think of these systems as operating with real numbers during the neural network simulation. The basic operation of an ANN processing element can be summarized as

$$out = F\left[\sum(\Pi)\right] \qquad (3.1)$$

Therefore three operations of multiplication, summation $\sum$ and activation function are to be performed within analogue hardware. Graf and Jackal [42] and Foo et al [43] provide a general introduction into analogue implementations, while Mead [44] provides a greater depth and more specialized viewpoint for using analogue circuits.

3.2.2 Digital Artificial Neural Networks

In a digital implementation of an ANN it is obviously necessary to perform the same operations as with an analogue approach. A number of approaches can be taken: one is to form all the components of a neuron separately using digital technology, second is to generate digital architectures and processors tailored towards ANN implementation and application, i.e. to design neuro-computer devices/accelerator boards and the third is to make use of existing high performance parallel computers and devices to construct purpose built machines, for example using transputers, or parallel DSP devices [45, 46]. Atlas and Suziki in [47] have provided a general introduction to digital NN systems.

Yet another approach using digital circuits is to use pulse coded computation as exemplified by Murray et al [48], Tomlinson et al [49] and Leaver [50].

3.2.3 Hybrid Artificial Neural Networks

Analogue and digital techniques for the hardware implementation of ANNs could be combined to provide a hybrid solution. This could lead to the best of both disciplines being combined.

In a hybrid system actual computation could be performed using analogue processing circuits as this often provides the smaller, faster circuits while as weight storage and update can be performed digitally since this provides a more stable method than their analogue counterparts. Inter-element communication could be a mixture of digital and analogue. Analogue communication links could be used internally within an individual neural chip

Alternatively, pseudo analogue systems could be realized using digital signals by means of pulse encoding.

3.3 Hardware Implementations of CNN

Fabrication of Fully connected analogue neural networks is a very difficult task as they suffer because of number of connections and the distance that these connections need to propagate. CNNs in contrary to simple neural networks are characterized by local connectivity. The adoption of the concept of nearest neighbour interactions found in cellular automata [51] allows for the arrangement of the cells which are equidistant in regular grid. The routing and layout problems usually faced in traditional analogue circuits are then easily tackled in analogue CNN VLSI implementations. A cell is designed and replicated to form a regular network that is placed and routed rapidly. The first VLSI implementation of a CNN [52] has, naturally, been based on the analogue model of the standard cell as presented by Chua [5].

Fabrication usually has its issues as most of the fabricated circuits come with parasitic capacitances and resistances, which in many cases leads to undesired behaviour. In order to reduce the sensitivity of the cell to such fabrication deviations, the dynamics of the cell are dominated by large state capacitor. Furthermore, the state capacitors affect the initialization procedure. A single row of state capacitors (cells) is loaded at a time as all cells cannot be loaded with initial states simultaneously. Also while initialized, the state capacitors have to be disconnected from the remainder of the cells in order to prevent their voltages from dropping to such a level that it may affect the processing.

Another issue has to do with the degree of adaptability. High flexibility is difficult to achieve, as one can tell by the experience gained from neural network VLSI implementations. Hence, the CNN array is very difficult to be ‘programmed’, so the array is designed to perform one or a related set of processing functions using fixed coefficients. Complex tasks are proposed to be solved by cascading or paralleling multiple CNN VLSI devices. This is, apparently, not practical and removes most of the attraction of the CNN VLSI implementation as time and cost are then much higher in comparison to other established systems. Next, due to fabrication and the available VLSI technology issues, only small CNN chips (20 × 20) were realized. But with the advancement in hardware technology a wide range of concepts, models and architectures were proposed.

The first attempt toward a VLSI implementation of a CNN was presented by Yang and Chua [5]. Meanwhile, a hardware accelerator board (CNN-HAC), mainly for hardware simulation was developed by Roska et al. [53]. Chua and Roska in 1992 introduced first algorithmically programmable analogue array computer having real time and super-computer power on a single chip [54] called as the CNN Universal Machine (CNN-UM). A key feature in the CNN-UM is the "dual computing" paradigm. Logic operations that only involve the symbolic variables YES/NO are combined with analogue array processing; therefore denoted analogic computing. All signals and operators are either analogue or logic, which in principle removes the need for Analog to Digital and Digital to Analog conversions.

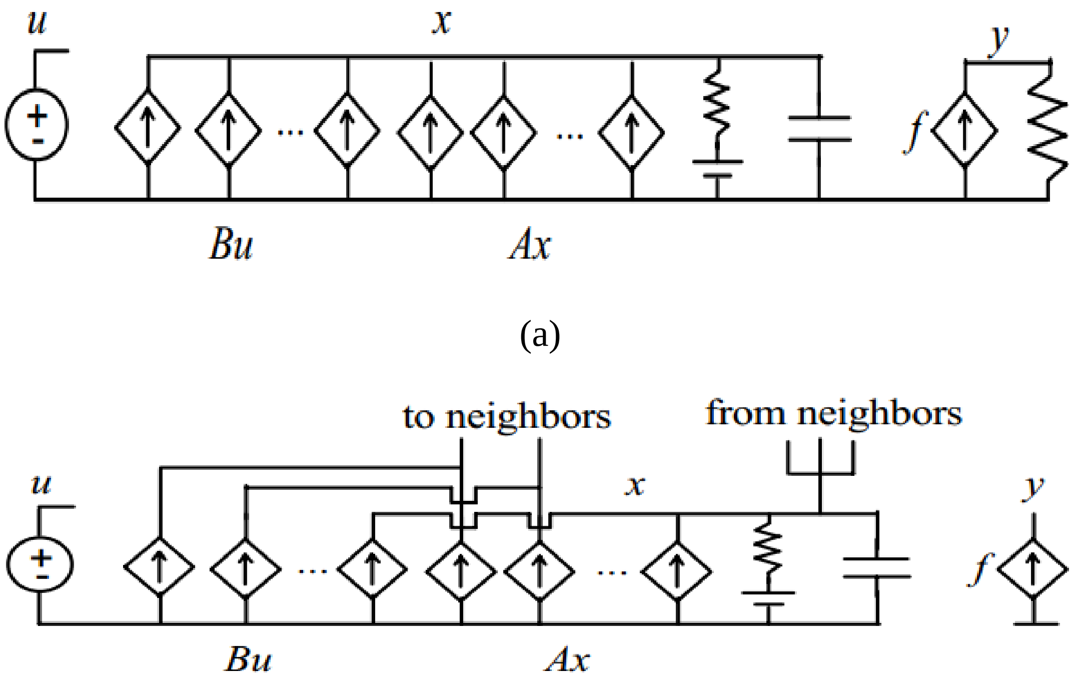
Now as we have already discussed in previous chapter that the Equation 3.2, re-written below, explains how the state of the cell evolves over time and is therefore commonly referred to as 'cell dynamics'. The state equation of a standard CNN cell can be obtained by simply summing the contributions of all cells in the sphere of influence, and can be written as in Equation 3.4. The output value is obtained according to the Equation 3.3. By interpreting cell state as voltage across an RC dipole realizing first-order dynamics, Equation 3.4 can be translated in a circuit scheme. Furthermore, so as the summations are performed by injecting currents into the same node, the connections are realized by voltage-controlled current sources (VCCS). This scheme is shown in Figure 3.1.

$$\dot{x}_{ij} = -x_{ij} + a_{ij} y_{ij} + b_{ij} u_{ij} + z_{ij} \qquad (3.2)$$

$$y_{ij} = f(x_{ij}) = \frac{1}{2}(|x_{ij}+1| - |x_{ij}-1| = \left\{ \begin{array}{ll} 1, & x_{ij} \geq 1 \\ x_{ij}, & |x_{ij}| < 1 \\ -1, & x_{ij} \leq -1 \end{array} \right\} \qquad (3.3)$$

$$\dot{x}_{ij} = -x_{ij} + \sum_{kl\in S_r(i,j)} a_{kl} y_{kl}(t) + \sum_{kl\in S_r(i,j)} b_{kl} u_{kl}(t) + z_{ij} \tag{3.4}$$

$$i = 1,2.......M; \quad j = 1,2........N$$



(a)

(b)

Figure 3.1: Alternative models of CNN cell: (a) Chua & Yang's model, with bias represented as voltage instead of current; (b) dual model with weighting at the output. [55]

Based on the scheme shown in Figure 3.1(a) and taking Operational Transconductance Amplifiers (OTA) as basic blocks, several implementations were proposed. Also to mention here is that the weighting of signals is not performed at the input but instead of that it is performed at the output of cells, together with nonlinear output function (Figure 3.1(b)).

OTAs are used both in their linear operation range and in saturation, exploiting their transfer function that approximates a Piecewise linear (PWL) sigmoid very well. Utilizing the above mentioned concept, a cell composed of three OTAs and two capacitors was designed by Cruz and Chua [56] as depicted in Figure 3.2. In this circuit, Cu is loaded with cell input while as Cx represents the state capacitor, OTA 'A' implements all current sources controlled by state voltage, and PWL output function. OTAs 'B' and 'R' work in their linear range; while the latter works as state resistor (with voltage bias), the first implements all current sources controlled by input voltage. A 6×6 prototype was realized in 2μm CMOS technology. Each cell contains about 50 transistors and occupies an area of 31,000μm^2 (32cells/mm^2). Settling time of the circuit is of order 1μs.

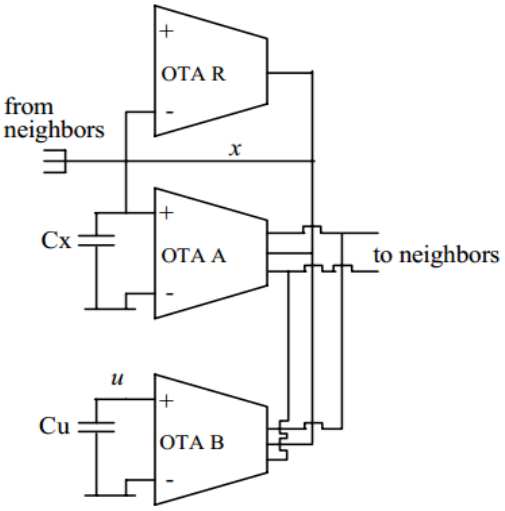


Figure 3.2: Cruz & Chua's CNN cell. [55]

In a similar way a CNN containing local digital memory and logic was realized by Halonen and et al. [57]. A 4×4 prototype was realized in 2μm CMOS technology. In this prototype each cell contained about 500 transistors and occupied 1mm^2. Settling time was about 3μs. Moreover, the weights were programmable in a discrete set and biased continuously.

Also a fully-programmable scheme based on Operational Amplifiers (Op-Amp) and variable conductance blocks was proposed by Nossek et al [58] as shown in Figure 3.3. The inner cell circuit in their proposed design was first transformed by adding an op amp, so that currents could be drawn from virtual ground node instead of inner node, thereby stabilizing its voltage against loading effects.

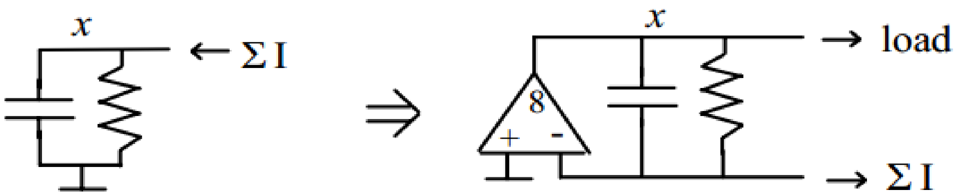


Figure 3.3: Equivalent inner circuit structures.

The circuit is then transformed to a balanced structure as shown in Figure 3.4 that employs variable conductance blocks that can be realized with four transistors, as in Figure 3.5.

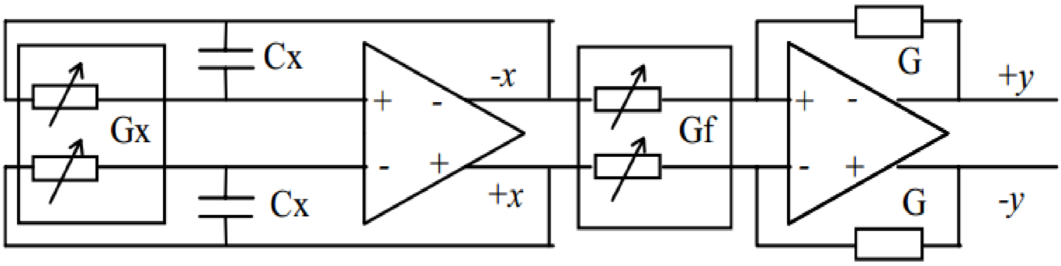


Figure 3.4: Balanced inner circuit structure.

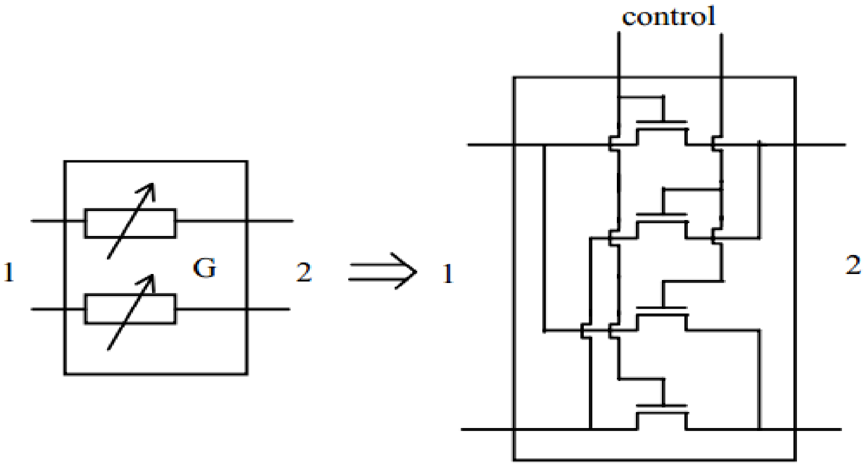


Figure 3.5: Balanced variable conductance block.

Almost the same structure was used by Harrer and et al [59] to design a DT-CNN. A 4×4 prototype on hexagonal grid was fabricated in 1.5µm CMOS technology, with 12cells/mm^2. The circuit was checked to operate correctly at 1MHz clock frequency.

Rodriguez-Vazquez and et al. took a different approach [60]. They associated all variables to currents. The reason to abandon voltage variables is that combination of voltage and current variables complicates design for the necessity of scaling signals to compensate nonlinearities, and requires high impedance internal nodes, that cause large time constants to appear. Besides, an efficient way of realizing input is by means of photo sensors that give current output.

The design given by Rodriguez-Vazquez and et al. was based on the FR-CNN model, obtaining simplified design, good speed/power ratio and low cell complexity. Basic building blocks for this realization were current mirrors, which were used for weighted state replication for connections, and to realize lossy integration and delay operators required to generate continuous- or discrete time dynamics Figure 3.6.

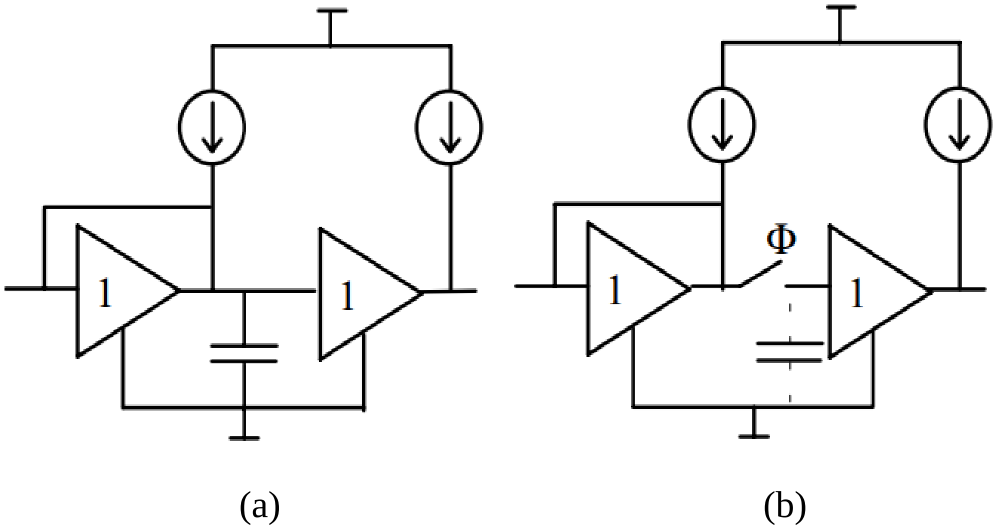


Figure 3.6: (a) lossy integrator; (b) half-clock delay in current mode.

Realization and testing of several CT and DT prototypes in 1.6μm CMOS technology are reported in [60]. 9×9 and 1×16 CT prototypes have less than 20 transistors per cell and 60 to 160 cells/mm^2 density is achieved. These circuits settle in about 0.25 to 1.5μs. A DT-CNN programmable 9×9 network with local logic was successfully tested at 5 MHz clock frequency.

In practical realizations the important issues that need to be confronted are those of control and initialization. Generally, accessing all cells by means of electronic signals at once is impossible because of the excessive number of lines required. Multiplexed accessing (e.g. by rows) is therefore necessary, together with analog storage, that may be done by capacitors connected to voltage followers. Besides, the weight programming is one of the most difficult task.

The easiest case is when cloning templates are used, and programming is only allowed in discrete values, that can be selected by a few global lines and some local logic and memory. Using on-chip photo sensors as input devices is a promising alternative, especially for image processing purposes.

An attempt to realize CNNs was also attempted by use of optical devices [58]. Main advantages are speed-of-light computing in the forward path along with the possibility of large neighbourhood. However, bottlenecks occur in electronic addressing of cells for input (but optical addressing might also be implemented) and in electronic feedback of intermediate results.

Liquid crystal devices (Spatial Light Modulators - SLM), and lenses are the main building blocks for the optical realization. SLMs are used to perform analog multiplication between images, by applying variable attenuation to an image transmitted through the panel. Their nonlinear sigmoid-like transparency vs. voltage characteristic is also used to implement output nonlinearity. Lenses are used to realize cross-correlation with cloning template. In fact, they realize a Fourier transform, which can be followed by SLM multiplication by a hologram representing cloning template. Inverse transform is obtained by another lens, after observing that a second direct transform yields a mirror image of the desired inverse transform. Complete optical CNN block scheme is depicted in Figure 3.7 [57].

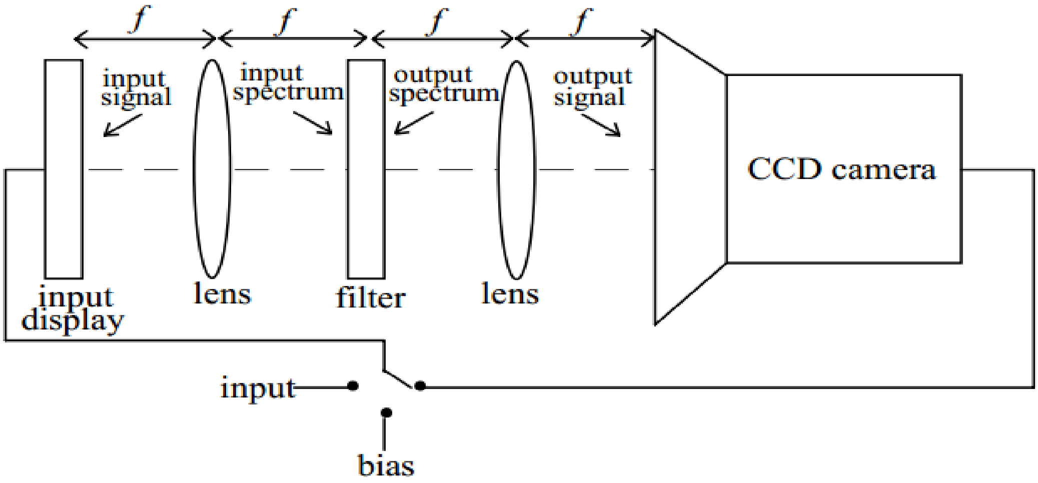


Figure 3.7: Optical realization of a DT-CNN.

In order to minimize speed limitations of optical CNNs, design should concentrate on minimizing the number of time steps necessary to obtain desired task, by maximizing derivatives of state components and enlarging neighbourhood size [61].

3.4 Conclusion

Among the software simulation in conventional computers and a special hardware solution for the implementation of neural networks, the lateral one is the most preferred way. The hardware implementation of neural networks gives a lot of advantages over the software implementation. Besides the speed that the hardware implementation provides owing to its parallel nature it is also cost efficient. However one of the most difficult part in the implementation of neural network is its training.

4.1 Introduction

Besides the need of hardware implementation of ANNs, as discussed in the previous chapter, another matter of great concern is the power and voltage requirements of the ANN electronic circuits. It is important that the designed circuits consume less power and also should be able to operate al low voltage levels.

The advancements in the VLSI technologies have led to the dramatic increase in the achievable integration densities that has made possible to achieve the rapid improvement of circuit functionality. But unfortunately the battery technologies do not evolve as fast as applications demand, so the combination of battery supply and miniaturization often turns into a low-voltage and/or low-current circuit design problem. In particular, these restrictions affect more drastically the analog part of the whole mixed system-on-chip. As a result, specific analog circuit techniques are needed to cope with such power supply limitations. In [62] Wang et al have suggested that weak inversion mode is one of the remedy suitable for the design of continuous time ultra-low power systems.

In our work we have tried to use the companding technique (Sinh-Domain (SD)) to achieve the goal of low-power low-voltage design of ANNs. The implementation of different analog activation functions for ANNs in SD companding technique is done besides the implementation of SD complex TD-CNN cell, which are presented in the chapters to follow. But firstly, a short description of the specific circuit approaches for low-voltage operation is listed below:

- ***Rail-to-Rail:*** this includes all the strategies oriented to extend the signal voltage range up to the available room between supply rails. Most of the techniques under this category are mainly based on the redesign of the input and output stages in order to increase their linear range [63–67].
- ***Multistage:*** instead of single cascaded structures multiple but simple cascaded stages are used in this technique. Efforts are then focused on their frequency stabilization with nested compensating loops [68, 69].
- ***Bulk-Driven:*** strategies make use of the MOSFET local substrate as an active signal terminal to obtain lower equivalent threshold voltages [70, 71].
- ***Supply Multipliers:*** a step-up conversion of supply voltage through charge pumps [72–78], typically from 1.5V to 3V, to bypass the low-voltage restriction.

The said low-voltage techniques have the following drawbacks:

- The said low-voltage strategies are only the partial solutions since they are addressed mainly to the design of operational amplifiers only. This is not the case only in case of strategies which employ supply multipliers.
- The bulk-driven option is also in opposition to general anti-latch-up rules of any standard CMOS process.
- Although the most used solution for very low-voltage operation is using supply multipliers but they require large capacitive components, which take an important Si area overhead and exhibit high extra current consumption. Thus making them unsuitable for small package and low current applications.

In a similar way, the main circuit techniques for low-current consumption applications are enumerated as follows:

- ***Adaptive Biasing***: to optimize consumption according to signal demands a non-static current bias is used. Bias dynamics are defined either by local positive feedback [79, 80] or by feed-forward [81, 82] controls.
- ***Subthreshold Biasing***: MOS transistors operating in the weak inversion region at very low-current levels are utilized in these topologies [83].

In addition to the techniques mentioned above, LP LV designs have been achieved by substituting traditional voltage-mode techniques by the current-mode techniques. Current-mode approaches deserve particular mention [84] since they provide a large dynamic range for the currents, considered now as processing variables, while maintaining reduced voltage swings, being accompanied by an increase in circuit bandwidth. Therefore, many current-mode techniques came into existence, Companding-mode design being one such technique for AICs. Companding describes the linearization mechanism in which the signals are first compressed to an intermediate integration node and then subsequently expanded after being processed. The distinct characteristic from the classical techniques is that it is the large-signal transfer function of the circuit that is linearized and not the individual trans-conductance or active resistive elements. Thus the Companding systems perform an externally Linear and Time-Invariant (LTI) operation on the signal, even though internally this is not the case; these systems can thus be considered as a particular case of Externally Linear and Internally Nonlinear (ELIN) systems [85].

4.2 Companding Techniques

In companding, the input and output amplifiers can have characteristics of the form y= gx with a variable gain g. The gain of these amplifiers is made to depend on the input signal. We have two special companding techniques that are described depending upon how gain is dependent on the signal.

- ***Instantaneous companding***: The input amplifier includes nonlinearity whose slope (equivalently, the small signal gain) decreases as the input increases. The output amplifier should have the opposite behaviour. This case, where the output of the amplifier is a nonlinear function of the instantaneous value of the input is termed instantaneous companding.
- ***Syllabic companding:*** In this case, the output of the amplifier is a nonlinear function of an average measure of the input signal strength (e.g. The envelope or the root-mean-square value).

4.3 Types of Companding

Low-voltage operation is the current requirement for realizing analog designs and an endeavour of achieving these goals thus gained a significant research effort. Companding technique for designing the analog circuits is an interesting technique with potential for low-voltage operation. In companding the linear input current is initially converted to a non-linear compressed voltage and then processed by a companding core. The resulted compressed output voltage is then expanded and simultaneously converted into a linear current. The three of ways in which input current compression could be performed are given as:

- ***Log-Domain (LD) companding:*** compression through the logarithmic V–I relationship of bipolar transistor in active region or MOS transistor in weak inversion [86–90].
- ***Square-Root Domain (SRD) technique:*** compression through the square-root V–I relationship of MOS transistor in strong inversion [91–94].
- ***Sinh-Domain (SD) companding:*** compression through the inverse of the hyperbolic sine function realized by translinear loops formed by bipolar transistors in active region or MOS transistors in weak inversion [95– 97].

The concept of LD companding was first introduced in the 1970's [98]. LD belongs to instantaneous companding systems that use Logarithmic (LOG) and Exponential (EXP) functions for compression and expansion [99] respectively. This makes it possible for LD circuits to operate with very low supply voltage without sacrificing the dynamic range. Also, these circuits contain low impedance nodes along the signal path, which can be exploited to achieve greater bandwidths. But LD circuits are limited by the fact that they have to be operated in class A, i.e., a DC component has to be introduced at the input in order to guarantee their proper operation [100]. This constraint increases the quiescent power consumption and severely limits the dynamic range that is potentially offered by the companding action. Besides, they are also sensitive to threshold voltage matching, and the bandwidth becomes restricted due to its limited operation that is merely within the kHz range.

The problems encountered in the above method were solved by using MOS transistors operated in the strong inversion; the resulted circuits are known as SRD circuits, due to the quadrature I-V characteristic of MOS transistor in strong inversion region [101–106]. But SRD has its limitations too. They like LD do not have inherent class-AB nature. The most used technique, now days, is SD companding.

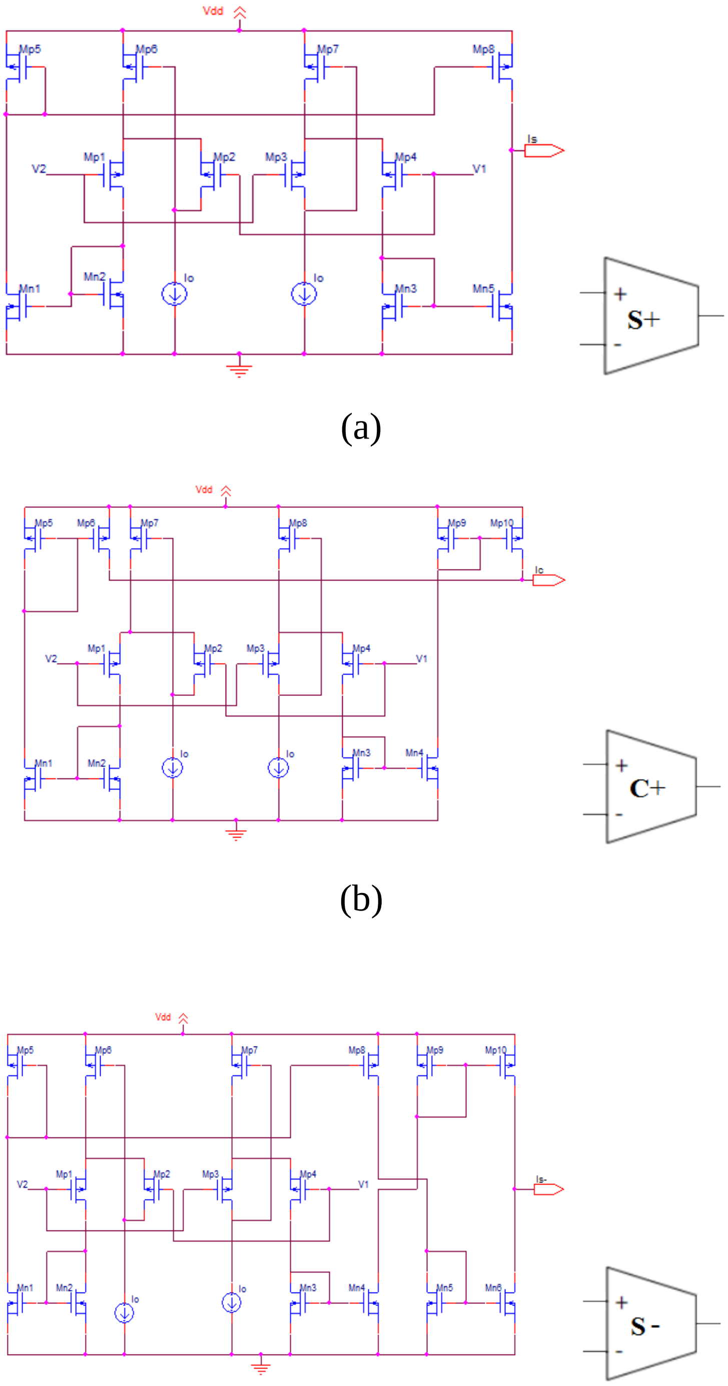
4.4 Sinh-Domain (SD) Companding

Sinh-Domain (SD) companding is an important technique for realizing analog circuits with inherent class-AB nature. In contrast to LD where a pseudo class-AB operation is realized by establishing two identical class-AB signal paths and employing a current splitter at the input, the required current splitting is simultaneously realized with the compression of the linear input current and its conversion into a non-liner voltage. The produced intermediate output currents are then subtracted in order to derive the final output. Besides, the aforementioned feature, SD technique also offers the capability for electronic adjustment of their frequency characteristics because the realized time constants are controlled by a dc current. Because of the companding nature, SD circuits also allow the capability of operation under a low-voltage environment. SD compared to their corresponding counterparts LD and SRD, offer more power efficient realizations but the price paid may be an increased circuit complexity [95, 96, 107, and 108].

The basic building blocks of the Sinh Domain technique are Sinh and Sinh^{-1} operators, Lossy and Lossless Integrators, and, algebraic summation/subtraction blocks. Therefore, SD design of the said blocks will be discussed in the following section.

4.4.1 Sinh-Cosh (SC) transconductor cells

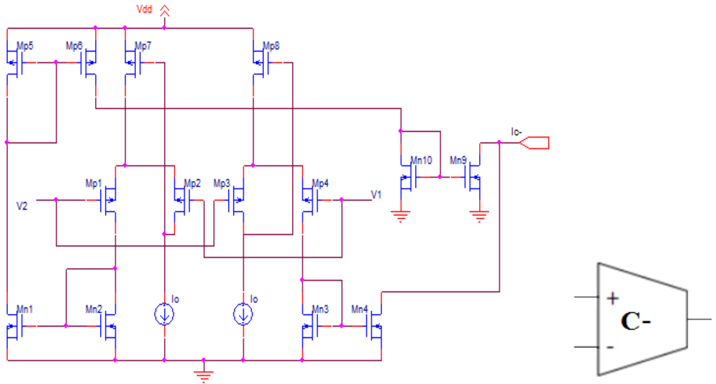
The weak inversion MOSFET (WIMOSFET) based Sinh, Cosh, negative Sinh and negative Cosh cells along with their corresponding symbols are given in Figures 4.1(a)–(d). In case of Figure 4.1(a) and 4.1(b) utilizing the translinear principle (109) and performing a routine algebraic analysis, it can be easily obtained that the output currents are given by the following equations



(a)

(b)

(c)



(d)

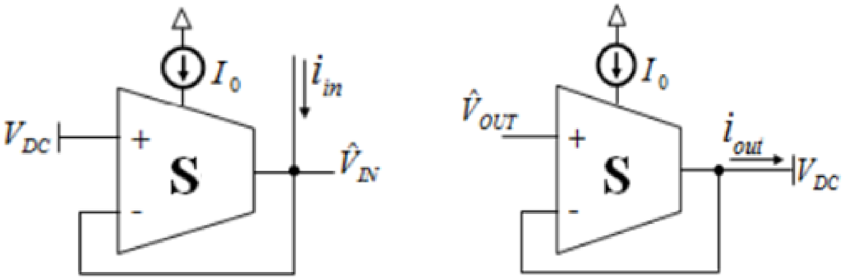
Figure 4.1: (a) Positive Sinh, (b) Positive Cosh,

(c) Negative Sinh and (d) Negative Cosh transconductor cell.

$$I_S = 2I_0 Sinh\left(\frac{\hat{v}_{IN}}{U_T}\right) \tag{4.1}$$

$$I_C = 2I_0 Cosh\left(\frac{\hat{v}_{IN}}{U_T}\right) \tag{4.2}$$

where, $U_T = nV_T$, n is the sub-threshold slope factor of a WIMOSFET and V_T is the well-known thermal voltage. The corresponding S cell with an inverted output is shown in Figure 4.2(b). Using Equation 4.1 and inspecting the topology in Figure 4.2(a), it is readily obtained that the voltage $(\hat{v}_{IN})$ at its non-grounded terminal is given by Equation 4.3. That is, a linear input current is converted into a compressed voltage. In addition, from the configuration in Figure 4.2(b) and the employment of Equation 4.1, it is derived that the expression in Equation 4.4 is realized. In other words, the topology in Figure 4.5(b) performs an expansion of a compressed voltage and simultaneously a conversion of it into a linear current. Consequently, the topologies in Figure 4.5 perform two complementary operations as those described by SINH^{-1} and SINH operators introduced in Equations 4.3 and 4.4, respectively.



(a) (b)

Figure 4.2: Realization of the SD operators: (a) SINH^{-1} and (b) SINH.

$$\hat{v}_{IN} = SINH^{-1}(i_{in}) = U_T Sinh^{-1}\left(\frac{i_{in}}{2I_0}\right) \tag{4.3}$$

$$i_{out} = SINH(\hat{v}_{OUT}) = 2I_0 Sinh\left(\frac{v_{OUT}}{U_T}\right) \tag{4.4}$$

Expressions for COSH and COSH^{-1} can be derived in a similar manner and are given as

$$\hat{v}_{IN} = COSH^{-1}(i_{in}) = U_T Cosh^{-1}\left(\frac{i_{in}}{2I_0}\right) \tag{4.5}$$

$$i_{out} = COSH(\hat{v}_{OUT}) = 2I_0 Cosh\left(\frac{v_{OUT}}{U_T}\right) \tag{4.6}$$

For the simplification of the circuit complexities we can make use of a multiple output transconductor cell where we get all the functions from the same circuit as depicted in Figure 4.3. In a similar way with slight change in the topology of circuit of Figure 4.3 we can achieve the circuit that will give multiple Sinh outputs, multiple Cosh outputs and so on.

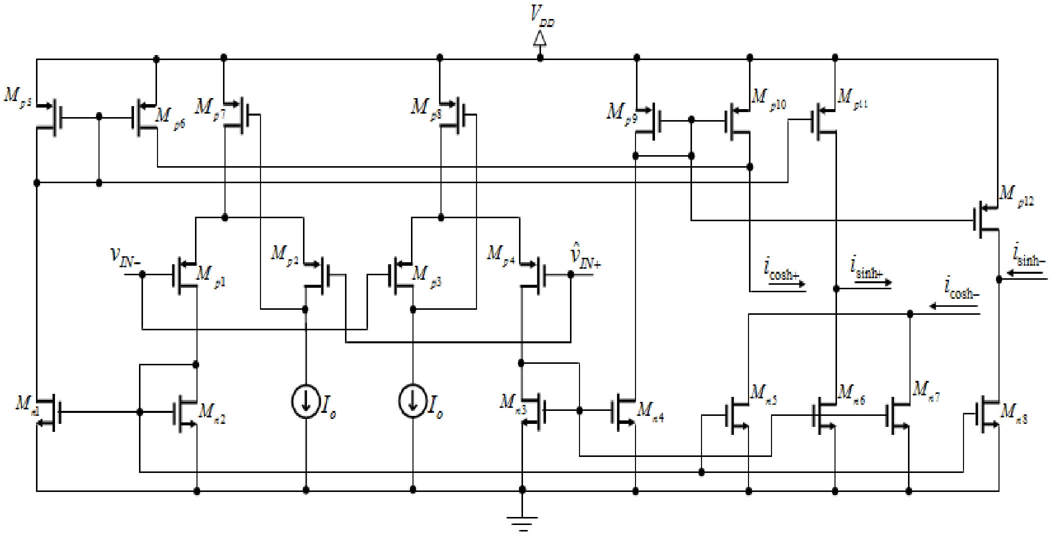


Figure 4.3: Multiple output non-linear transconductor cell.

4.4.2 Low-Voltage Two-Quadrant SD Divider

Another important block is a two-quadrant multiplier/divider block. Following a similar concept to that introduced in [59] the derivation of this class-AB divider will be performed. The topology and the corresponding symbol are depicted in Figure 4.4, where the label 2Q depicts the two-quadrant operation capability of the cell. This originates from the fact that both currents I_0 and i_2 are dc bias currents and, consequently, they must be strictly positive. In addition, due to the employment of the proposed S cell, the benefit for low-voltage operation capability is preserved.

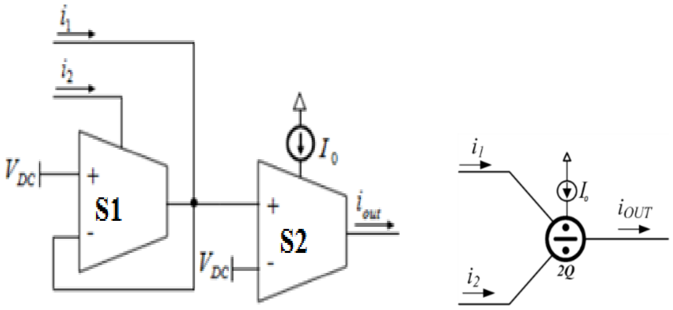
According to Equation 4.4 and the fact that the cell S_1 is biased at a current i_2, the intermediate voltage $(\hat{v})$ is given by Equation 4.5 as

$$\hat{v} = V_{DC} + V_T \cdot Sinh^{-1}\left(\frac{i_1}{2i_2}\right) \tag{4.5}$$

where, V_{DC} is a dc voltage.

The configuration of the cell S_2 establishes that its output current could be written, using Equations 4.4 and 4.5, as in Equation 4.6

$$i_{out} = I_o \cdot \frac{i_1}{i_2} \tag{4.6}$$



(a) (b)

Figure 4.4: (a) Two-quadrant multiplier/divider and (b) associated symbol.

4.4.3 SD Summation/Subtraction Block

The realization of SD algebraic summation block with a weighted input is that given in Figure 4.5. Applying the KCL at the output node, it is derived that

$$2I_0 Sinh\left(\frac{\hat{v}_{OUT} - V_{DC}}{U_T}\right) = 2I_0 Sinh\left(\frac{\hat{v}_{IN1} - V_{DC}}{U_T}\right) - a \cdot 2I_0 Sinh\left(\frac{\hat{v}_{IN2} - V_{DC}}{U_T}\right) \tag{4.7}$$

Using Equations 4.4 and 4.7 can be written as

$$SINH \quad (\hat{v}_{OUT}) = SINH \quad (\hat{v}_{IN1}) - a \cdot SINH \quad (\hat{v}_{IN2}) \tag{4.8}$$

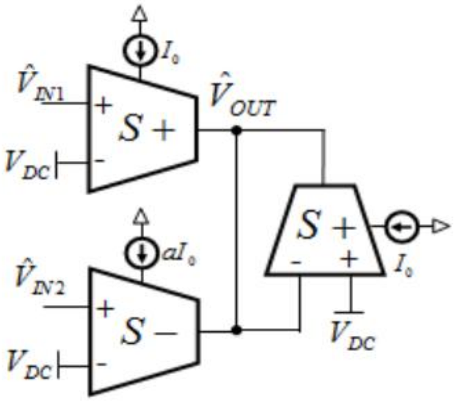


Figure 4.5: SD algebraic summation/subtraction block with weighted input.

4.4.4 SD Integrators

A typical configuration of SD two-input lossless integrator, constructed from blocks mentioned is demonstrated in Figure 4.6(a). The current that flows through the capacitor $\hat{C}$ is given by

$$i_c = \hat{C}\frac{d\hat{v}_{out}}{dt} = \frac{2I_o \cdot Sinh\left(\dfrac{\hat{v}_{IP} - V_{DC}}{U_T}\right) - 2I_o \cdot Sinh\left(\dfrac{\hat{v}_{IN} - V_{DC}}{U_T}\right)}{2I_o \cdot Cosh\left(\dfrac{\hat{v}_{OUT} - V_{DC}}{U_T}\right)} \tag{4.9}$$

After some algebraic manipulations, we have

$$\frac{\hat{C}U_T}{2I_o} \cdot \frac{d\left[2I_o \cdot Sinh\left(\dfrac{\hat{v}_{OUT} - V_{DC}}{U_T}\right)\right]}{dt} = 2I_o \cdot Sinh\left(\frac{\hat{v}_{IP} - V_{DC}}{U_T}\right) - 2I_o \cdot Sinh\left(\frac{\hat{v}_{IN} - V_{DC}}{U_T}\right) \tag{4.10}$$

Defining the pair of inverse SINH^{-1} and SINH mappings as given in Equations 4.3 and 4.4, we can rewrite Equation 4.10 as

$$\hat{\tau} \cdot \frac{d}{dt} SINH(\hat{v}_{OUT}) = SINH(\hat{v}_{IP}) - SINH(\hat{v}_{IN}) \tag{4.11}$$

where, $\hat{\tau} = CU_T/2I_o$, is the time-constant in SD. As the value of $\hat{\tau}$ is dependent on I_o, which can be changed externally, the SD circuits possess the inherent property of tunability.

The SD two-input damped (lossy) integrator is shown in Figure 4.6(b). Following the same procedure, the input-output relationship of the damped integrator can be given as:

$$SINH(\hat{v}_{OUT}) = \frac{2I_o}{\hat{C}U_T} \cdot \int \{SINH(\hat{v}_{IP}) - SINH(\hat{v}_{IN}) - SINH(\hat{v}_{OUT})\} dt \tag{4.12}$$

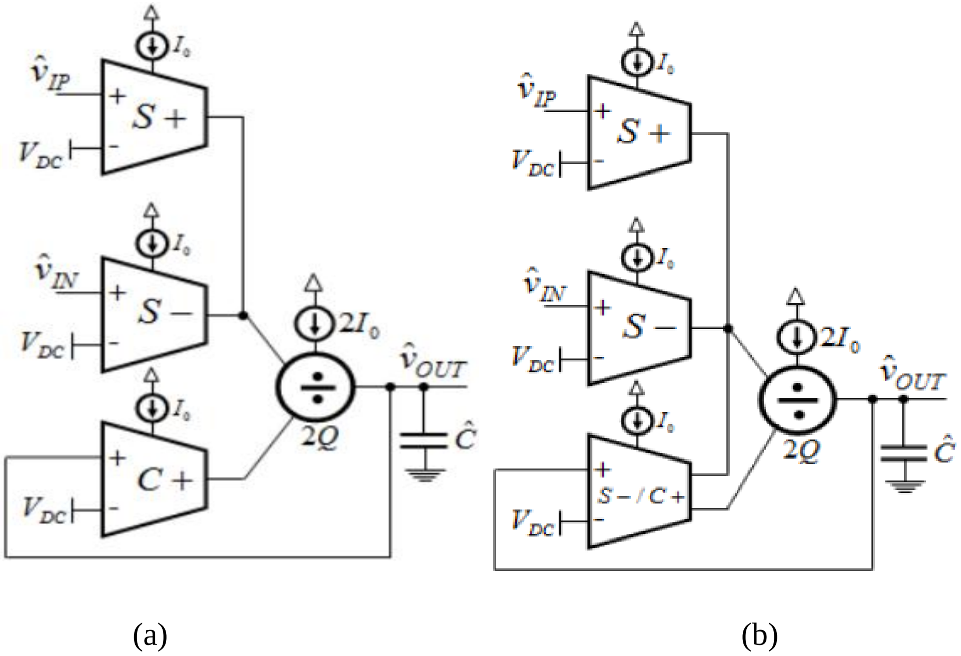


Figure 4.6: SD Integrators. (a) Two-input SD lossless integrator. (b)Two input SD lossy integrator.

In Laplace domain, the input-output relationships of the SD two-input lossless and lossy integrators can be respectively given by Equations 4.13 and 4.14.

$$\frac{i_{out}}{i_{ip}-i_{in}}=\frac{SINH(\hat{v}_{OUT})}{SINH(\hat{v}_{IP})-SINH(\hat{v}_{IN})}=\frac{g_m/\hat{C}}{S}=\frac{1}{S\tau} \tag{4.13}$$

$$\frac{i_{out}}{i_{ip}-i_{in}}=\frac{SINH(\hat{v}_{OUT})}{SINH(\hat{v}_{IP})-SINH(\hat{v}_{IN})}=\frac{g_m/\hat{C}}{S+\left(g_m/\hat{C}\right)}=\frac{1}{S\tau+1} \tag{4.14}$$

where, $g_m = {}^{2I_0}\!/_{U_T}$, is the trans-conductance of the SC cell and ($g_m/\hat{C}$) is the reciprocal of the integrator’s time-constant.

4.5 Conclusion

A brief introduction about the need of Low-power Low-voltage circuit implementation along with the different ways of achieving it is presented in this chapter. Among the various techniques used for Low-power Low-voltage circuit designing Companding techniques is elaborated with special emphasis being laid on the Sinh-Domain companding technique. Various building Blocks required in SD companding circuit design are discussed besides their characteristic equations are also derived in this chapter.

Chapter 5: Sinh-Domain Implementation of Various Activation Functions

5.1 Introduction

The main building block of neuron is the activation function. Among the activation functions, the sigmoid and hyperbolic tangent functions are most often used in the design of ANNs [110]. These activation functions have a wide range of applications in Sigma-pi and Hopfield neural networks in addition to being employed in multilayer perceptron neural network.

Straightforward implementation of these functions in hardware is not practical because it will require exponentiation and division, both of which are expensive operations. Several different approaches exist for the hardware implementation of the activation functions, including piecewise linear approximation, piecewise non-linear approximation, and lookup tables [111–114]. As per author’s best knowledge, only few attempts have been made in the open literature to obtain direct realizations of activation functions [115–122]. Most of these implementations are either using high voltage circuits or floating gate MOSFETs which is a costly technique in itself.

In this chapter, the SD technique has been employed to design circuits to achieve three different activation functions (Tanh, Unipolar Sigmoidal and Bipolar Sigmoidal). Attractive offered benefits are the capabilities for achieving resistor less realizations, electronic adjustment, and operating in a low-voltage and low-power environment essential for hardware NN design. In fact the electronic adjustment feature can be explored for programming the network in the form of weights to obtain the various logic functions. In addition, the inherent class-AB operation of SD technique offers the capability for handling signals greater than the bias current , leading to a power saving [123–131]. It is worth to mention here that added advantage of the SD technique is that the transistors do not have to jump from one region to the other thereby avoiding the noise spikes to the power line [115].

The SD design of activation functions are used to design single neuron network and multilayer perceptron whose weights are trained in such a manner that the network implements various logic gate functions.

The chapter is organized as follows: the multi-layer perceptron and the activation functions are introduced in Section 5.2. The SD implementation of activation functions, single neuron network, multi-layer perceptron and the SD building blocks are presented in Section 5.3 while as the derived simulation results are presented in Section 5.4.

5.2 Neural Network Design

The multi-layer perceptron design of neural network is shown in Figure 5.1, where the dash line enclosed blocks represent the neuron circuit.

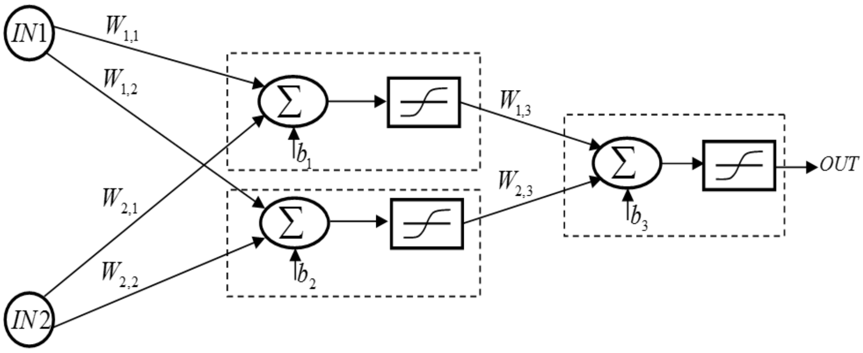


Figure 5.1: A Multilayer Perceptron.

It is clear from the diagram that it contains two neurons in the hidden layer and one neuron in the output layer. The multi-layer perceptron can be mathematically represented by Equation 5.1

$$OUT = AF\left[\sum_{j=1}^{2} W_{j,3}.AF(\sum_{i=1}^{2} W_{i,j} IN_i + b_j) + b_3\right] \tag{5.1}$$

Where $W_{i,j}$ are weights, b_k is the bias, and AF is the activation function.

An important component of artificial neural network is the nonlinear activation function, which is used at the output of every neuron. Therefore, the special focus is laid on its implementation during the hardware implementation of neural networks.

Several different activation functions are available today including the sigmoid, hyperbolic tangent (Tanh) and step functions [132]. The Tanh and sigmoid functions both produce a curve with an "S" shape. Mathematically, these functions are defined by Equations 5.2 − 5.4 as

$$\text{Tanh} \qquad \left(\frac{e^x - e^{-x}}{e^x + e^{-x}}\right) \tag{5.2}$$

$$\mathit{Unipolar\quad Sigmoidal} \qquad \left(\frac{1}{1+e^{-x}}\right) \tag{5.3}$$

$$\mathit{Bipolar\quad Sigmoidal} \qquad \left(\frac{1-e^{-x}}{1+e^{-x}}\right) \tag{5.4}$$

The proposed idea of their SD implementations is expressed in Equations 5.5 − 5.7.

$$\text{Tanh} \qquad \mathrm{K}_1\left(\frac{\sinh(x)}{\cosh(x)}\right) \tag{5.5}$$

$$\textit{Unipolar}\quad \textit{Sigmoidal} \qquad K_2\left(\frac{1}{1+\cosh(x)-\sinh(x)}\right) \tag{5.6}$$

$$\textit{Bipolar}\quad \textit{Sigmoidal} \qquad K_3\left(\frac{1-\cosh(x)+\sinh(x)}{1+\cosh(x)-\sinh(x)}\right) \tag{5.7}$$

From Equations 5.5 − 5.7, it is clear that the required operations for implementing the activation functions are Sinh, Cosh, summation/subtraction, and multiplication; these could be readily realized through the SD circuit design technique as discussed in chapter 4. Further, the included constants K$_1$, K$_2$ and K$_3$ will get electronically adjusted in the associated weights. The implementation of these activation functions in SD is given in the next Section.

5.3 SD design of activation function and multi-layer perceptron

5.3.1 SD Building Blocks

As we have already seen in chapter 4 the non-linear transconductor cell [127, 128] depicted in Figure 5.2 realizes the expression given by Equation 5.8 for a hyperbolic sine output

$$i = 2I_o \sinh\left(\frac{v_{IN+} - v_{IN-}}{nV_T}\right) \tag{5.8}$$

by Equation 5.9 for an hyperbolic cosine output

$$i = 2I_o \cosh\left(\frac{v_{IN+} - v_{IN-}}{nV_T}\right) \tag{5.9}$$

by Equation5.10 for an inverted hyperbolic sine output

$$i = -2I_o \sinh\left(\frac{v_{IN+} - v_{IN-}}{nV_T}\right) \tag{5.10}$$

by Equation5.11 for an inverted hyperbolic cosine output

$$i = -2I_o \cosh\left(\frac{v_{IN+} - v_{IN-}}{nV_T}\right) \tag{5.11}$$

In Equations 5.8 − 5.11, I_o is a dc current, V_T is the thermal voltage (≅26mV @ 27°C), n is the sub-threshold slope factor (1<n<2), and are the voltages at the non-inverting and inverting inputs, respectively.

Also, the two-quadrant divider employed in the design of activation functions and multilayer perceptron is constructed from appropriately configured S cells as shown in Figure 5.3, in order to realize the relationship given by Equation 5.12.

$$i_{OUT} = I_o \frac{i_1}{i_2} \tag{5.12}$$

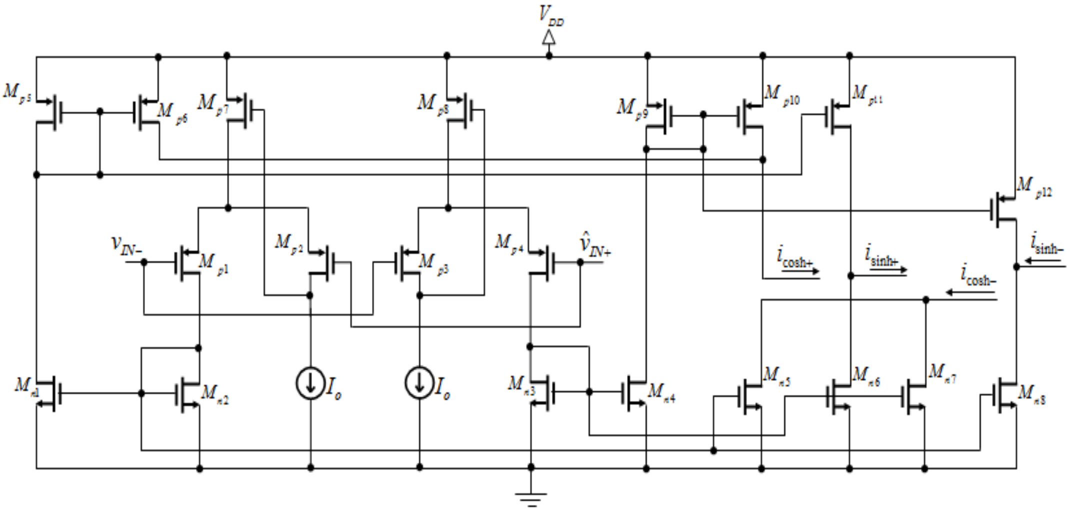


Figure 5.2: Multiple output non-linear transconductor cell.

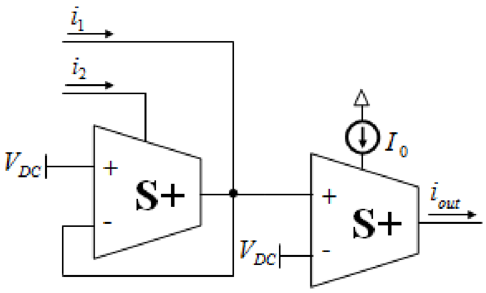
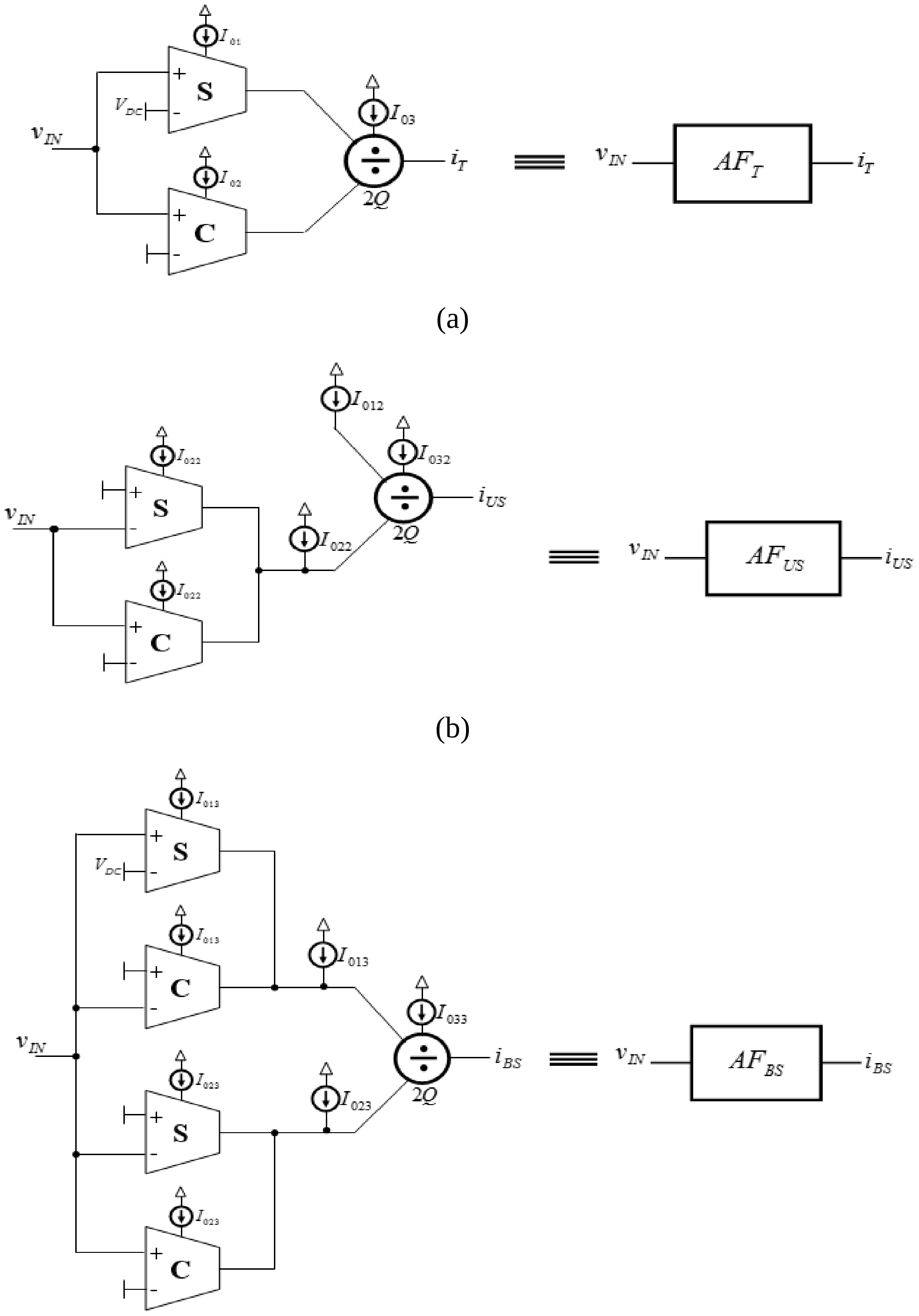


Figure 5.3: Two-quadrant multiplier/divider. [130]

5.3.2 SD Realization of Activation Functions

The SD realizations of Tanh, Unipolar Sigmoidal and Bipolar Sigmoidal activation functions are respectively given in Figures 5.4(a) − (c), respectively.



(a)

(b)

(c)

Figure 5.4: SD realization of activation functions (a) Tanh, (b) Unipolar Sigmoidal, and (c) Bipolar Sigmoidal.

After routine algebraic manipulations the equations for Tanh, Unipolar Sigmoidal and Bipolar Sigmoidal activation functions can be obtained as given by Equations 5.13, 5.14, and 5.15 respectively

$$I_{out} = K_1\left(\tanh\left(\frac{v_{IN+} - v_{IN-}}{nV_T}\right)\right) \tag{5.13}$$

$$I_{out} = K_2\left(\frac{1}{1+e^{-\left(\frac{v_{IN+}-v_{IN-}}{nV_T}\right)}}\right) \tag{5.14}$$

$$I_{out} = K_3\left(\frac{1-e^{-\left(\frac{v_{IN+}-v_{IN-}}{nV_T}\right)}}{1+e^{-\left(\frac{v_{IN+}-v_{IN-}}{nV_T}\right)}}\right) \tag{5.15}$$

where, K_1, K_2 and K_3 are respectively given by Equations 5.16, 5.17, and 5.18 as

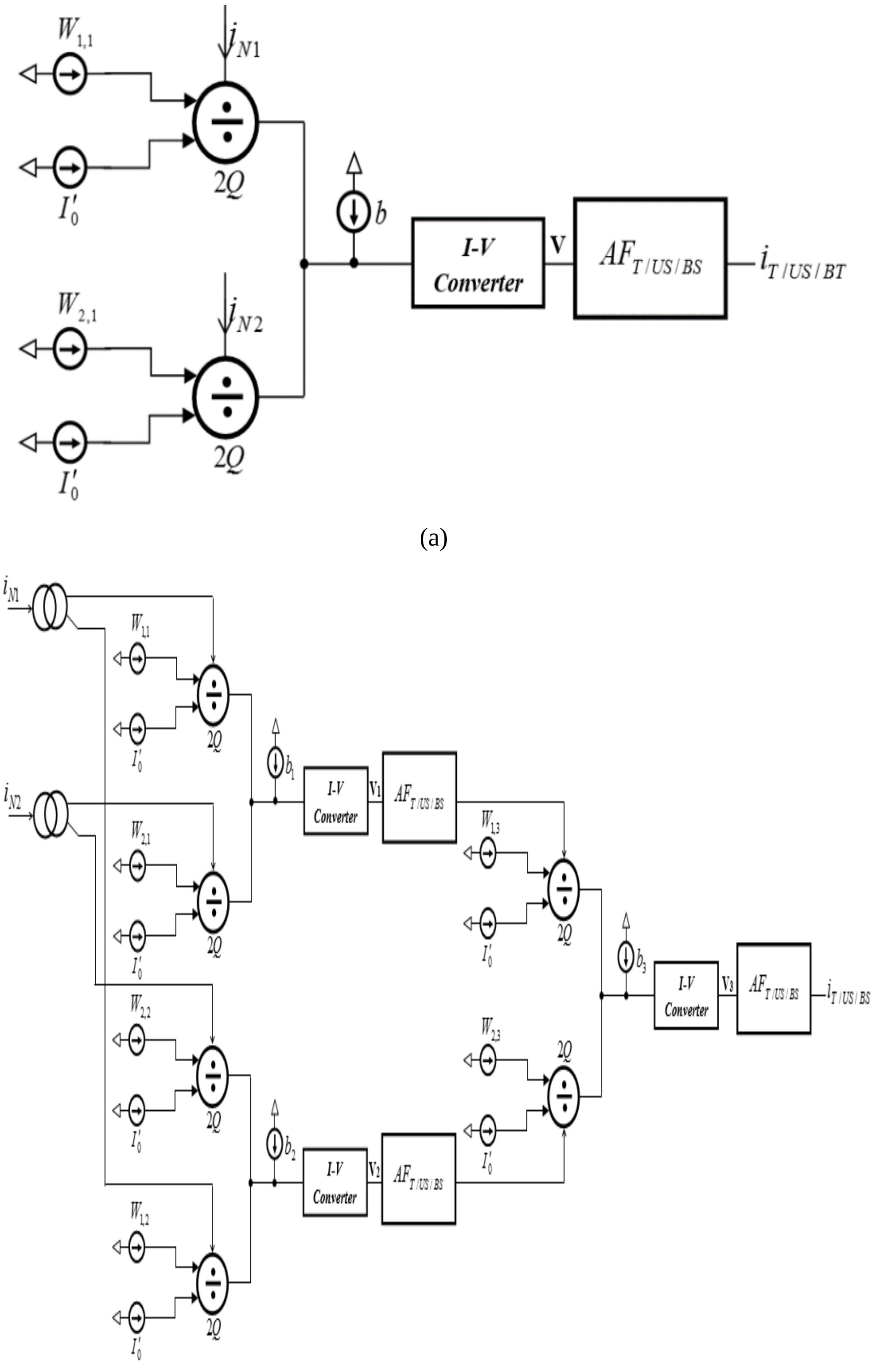
$$K_1 = I_{o31}\left(\frac{I_{o11}}{I_{o21}}\right) \tag{5.16}$$

$$K_2 = I_{o32}\left(\frac{I_{o12}}{I_{o22}}\right) \tag{5.17}$$

$$K_3 = I_{o33}\left(\frac{I_{o13}}{I_{o23}}\right) \tag{5.18}$$

5.3.3 SD Realization of Single Neuron Network and Multi-Layer Perceptron

The SD realizations of the single neuron network and multi-layer perceptron using the three activation functions given in Figure 5.4 are respectively given in Figures 5.5(a) and 5.5(b).



(a)

(b)

Figure 5.5: SD realization of: (a) Single neuron Network (b) Multi-layer perceptron.

The single layer and multi-layer perceptron can be mathematically represented in SD by Equations 519 and 5.20 respectively.

$$i_{T/US/BS} = AF_{T/US/BS}\left\{\left(\sum_{i=1}^{2}\frac{i_{Ni}.W_{i,1}}{I'_o}+b\right) \quad \Rightarrow \quad V\right\} \tag{5.19}$$

$$i_{T/US/BS} = AF_{T/US/BS}\left(\sum_{j=1}^{2}W_{j,3}.AF_{T/US/BS}\left\{\left(\sum_{i=1}^{2}\frac{i_{Ni}.W_{i,j}}{I'_o}+b_j\right) \quad \Rightarrow \quad V_j\right\}+b_3 \quad \Rightarrow \quad V_3\right) \tag{5.20}$$

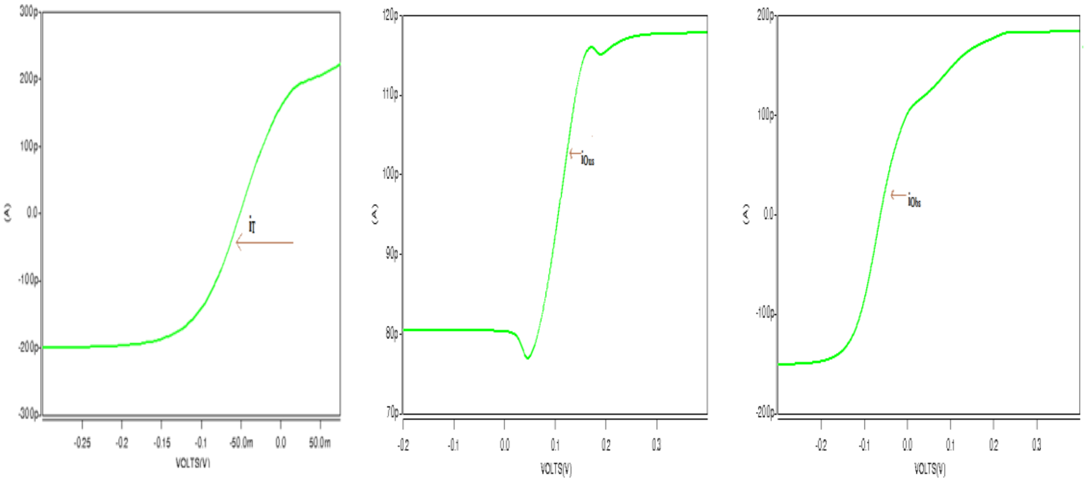
It is worth to mention here that the I-V converter circuit given in reference [133] has been employed in Figure 5.5.

5.4 Simulation Results

The SD circuits were simulated using the SPICE software and the model parameters of 0.35µm AMS CMOS process were utilized for the MOS transistor. The chosen bias scheme is given in Table 5.1. The aspect ratio of MOS transistors employed in the non-linear transconductor cell in Figure 5.2 are given in Table 5.2. The bias scheme was realized by employing current sources and current-mirrors in order to distribute the required dc bias currents. For this purpose PMOS transistors with aspect ratio 55µm/1.5µm have been employed. With the above mentioned values the input output characteristics of the three activation functions are given in Figure 5.6.

The activation functions almost show symmetry in the saturation levels, or, if not, nonzero offset. It is desirable to have a function with programmable saturation limits, so that these can be adjusted to control the dynamics [116]. For programmable saturation limits together with the monolithic integration feature, electronic tuning of the circuits is a desired parameter. Out of the reported implementations, only reference [116] includes the tunability of the saturation limits. In order to demonstrate the electronic tunability of the proposed circuits, currents I_{012} and I_{022} are tuned to change the saturation limits of the unipolar sigmoidal activation function and the results are given in Figure 5.7.

The SD single layer and multi-layer perceptron given in Figure 5.5 have been trained using the back propagation algorithm in the MATLAB environment to obtain the outputs of the various logic functions. For this purpose, Figure 5.5(a) has been trained for NOT (single input and single weight), AND, OR, NAND and NOR, and, the employed values of the weights and biases for different logic functions are given in Table 5.3(a) for NOT and 5.3(b) for AND, OR, NAND and NOR. Secondly, the multi-layer perceptron given in Figure 5.5(b) has been trained for typical XOR function and the employed values of the weights for different activation functions are given in Table 5.3(c). The obtained results of the NOT, AND, OR, NAND, NOR and XOR functions, for different activation functions are respectively given from Figure 5.8 to Figure 5.13.



(a) (b) (c)

Figure 5.6: Input/output characteristics of activation functions (a) Tanh (b) Unipolar Sigmoidal (c) Bipolar Sigmoidal.

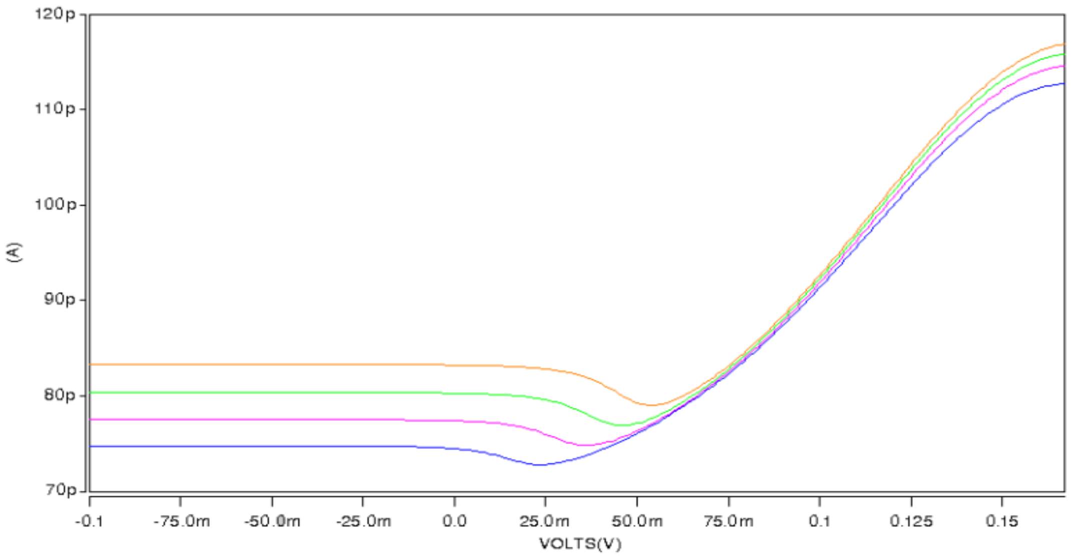
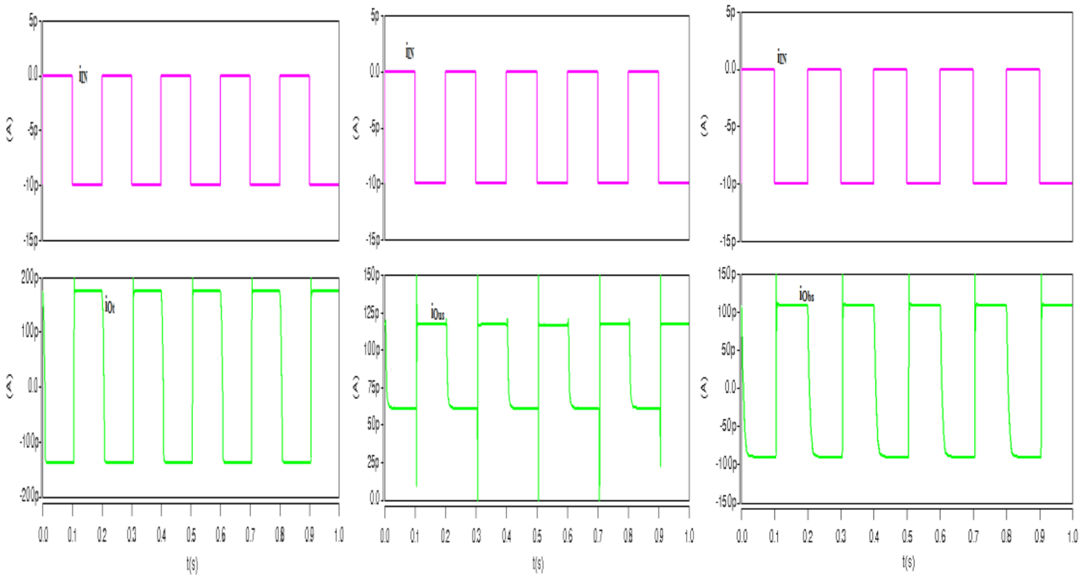
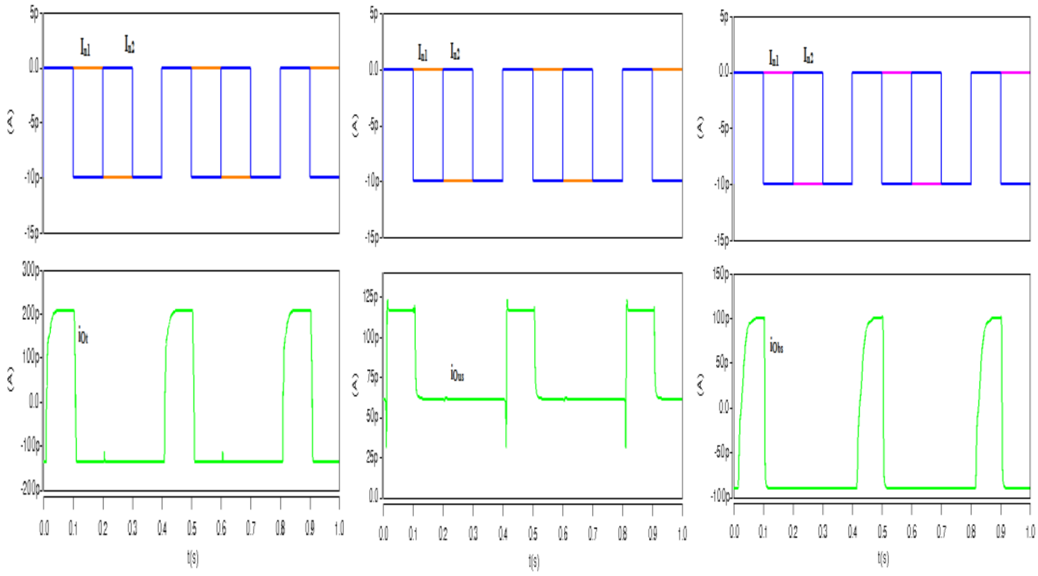


Figure 5.7: Demonstration of electronic tunability of saturation limits for Unipolar Sigmoidal Activation Function.



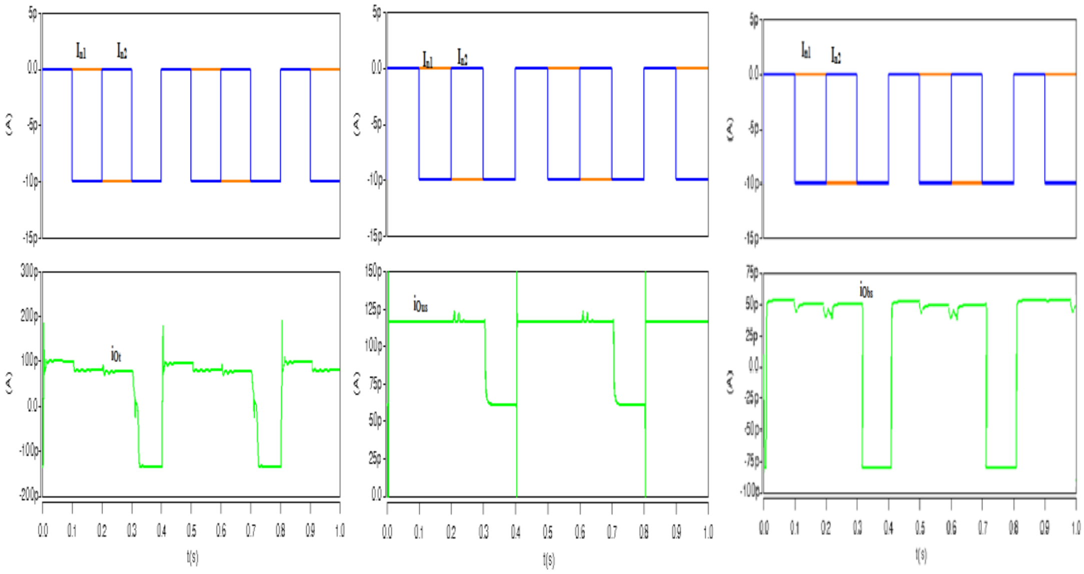
(a) (b) (c)

Figure 5.8: Simulation results of trained single neuron network for NOT function (a) Tanh function (b) Sigmoidal function (c) Bipolar Sigmoidal function.



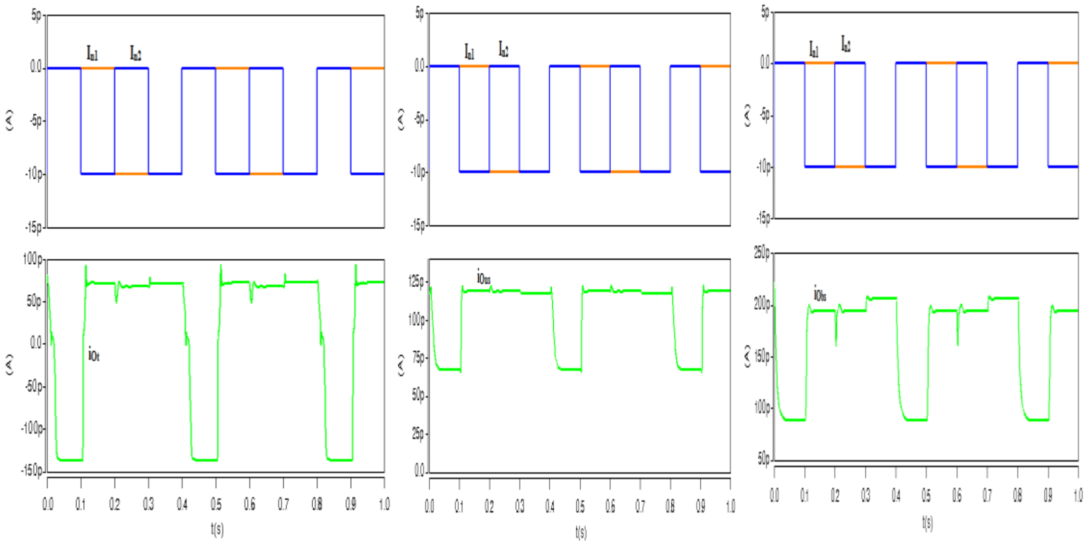
(a) (b) (c)

Figure 5.9: Simulation results of trained single neuron network for AND function, (a) Tanh function (b) Sigmoidal function (c) Bipolar Sigmoidal function.



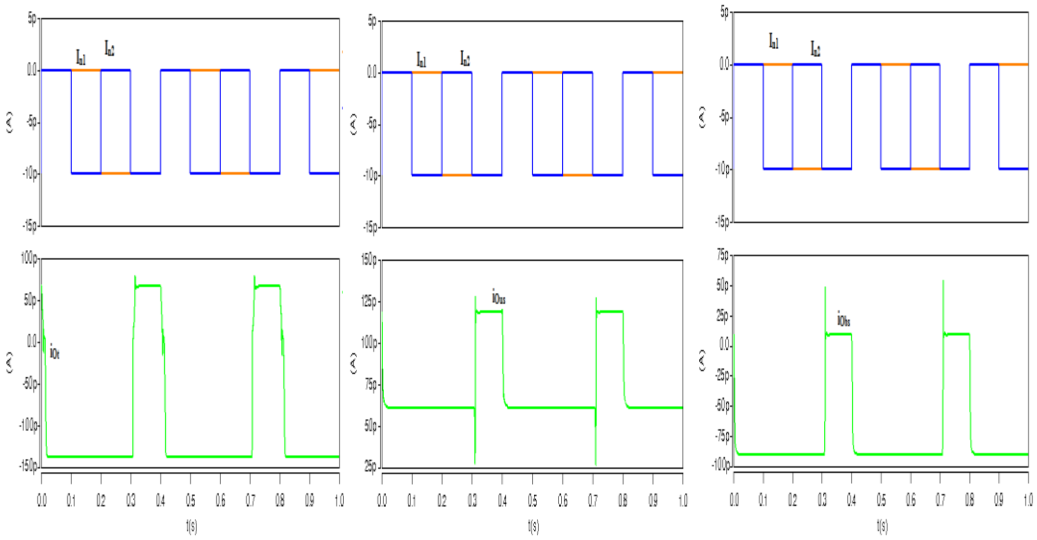
(a) (b) (c)

Figure 5.10: Simulation results of trained single neuron network for OR function, (a) Tanh function (b) Sigmoidal function (c) Bipolar Sigmoidal function.



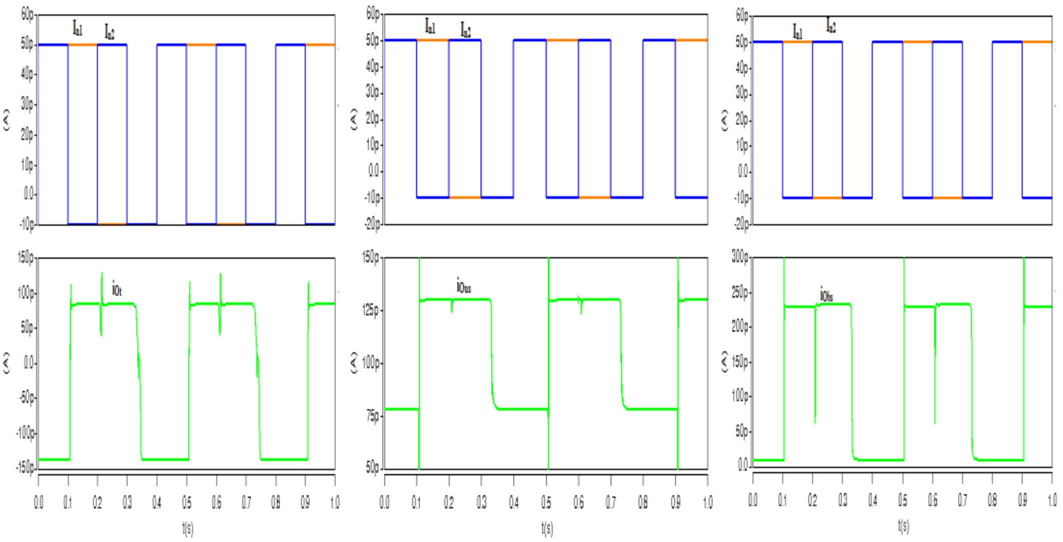
(a) (b) (c)

Figure 5.11: Simulation results of trained single neuron network for NAND function, (a) Tanh function (b) Sigmoidal function (c) Bipolar Sigmoidal function.



(a) (b) (c)

Figure 5.12: Simulation results of trained single neuron network for NOR function, (a) Tanh function (b) Sigmoidal function (c) Bipolar Sigmoidal function.



(a) (b) (c)

Figure 5.13: Simulation results of trained single neuron network for XOR function, (a) Tanh function (b) Sigmoidal function (c) Bipolar Sigmoidal function.

It is worth to mention here that the easy training of the various logic functions has been possible due to the inherit flexibility of the circuits which is due to the fact that the weights are represented by the current sources. The current sources can be adjusted externally which leads to the complete integrability of the proposed circuits.

The total power dissipation of various circuits is given in Table 5.4. From Table 5.4, it is clear that the proposed circuits consume smaller power than the only value (134nW) reported in Reference [122]. Further, the proposed SD uses lower supply voltage than the reported implementations and is therefore compatible with the contemporary IC design.

Parameter	Value	Parameter	Value
V_{DD}	0.5V	I_{012}	25pA
V_{DC}	0.1V	I_{022}	25pA
I'_0	1pA	I_{032}	100pA
I_{01}	30pA	I_{013}	15pA
I_{02}	30pA	I_{023}	15pA
I_{03}	100pA	I_{033}	50pA

Table 5.1: Bias scheme for the circuits.

Transistor	Aspect Ratio
M_{p1}-M_{p4}	35um/0.55um
M_{p5}-M_{p12}	58um/0.6um
M_{n1}-M_{n8}	21um/1um

Table 5.2: The aspect ratio of MOS transistors employed of the non-linear transconductor.

Weights & bias	Tanh function	Unipolar Sigmoidal	Bi-polar Sigmoidal
$W_{1,1}$	-17.73pA	-38.93pA	-18.73pAdc
B	45.87pA	83.87pA	35.87pA

(a)

Activation Function	Weight and bias	Logic Function			
		AND	OR	NAND	NOR
Tanh	$W_{1,1}$	6.98pA	9.25pA	-7.99pA	-9.25pA
	$W_{2,1}$	6.98pA	11.25pA	-7.480pA	-11.25pA
	B	-21.87pA	-111.07pA	38.87pA	111.07pA
Unipolar Sigmoidal	$W_{1,1}$	5.83pA	10.80pA	-5.80pA	-10.80pA
	$W_{2,1}$	5.98pA	12.88pA	-5.88pA	-12.88pA
	B	-17.97pA	-110.87pA	18.87pA	110.87pA
Bi-polar Sigmoidal	$W_{1,1}$	10.88pA	6.80pA	-6.98pA	-11.80pA
	$W_{2,1}$	10.88pA	6.88pA	-6.98pA	-12.88pA
	B	-15.98pA	-92.87pA	15.87pA	110.87pA

(b)

Weights & bias	Tanh function	Unipolar Sigmoidal	Bi-polar Sigmoidal
$W_{1,1}$	-30.83pA	-28.99pA	-30.83pA
$W_{2,1}$	11.98pA	11.64pA	11.98pA
b_1	-7.97pA	-7.97pA	-7.97pA
$W_{1,2}$	28.21pA	27.11pA	28.21pA
$W_{2,2}$	-42.96pA	-40.44pA	-42.96pA
b_2	-9.70pA	-9.70pA	-9.70pA
$W_{1,3}$	-9.99 pA	-5.821pA	-3.5740pA
$W_{2,3}$	-8.984pA	-5.58pA	-3.56pA
b3	101.89pA	191.89pA	284.89pA

(c)

Table 5.3: Weights and bias for different activation functions for: (a) NOT (b) AND, OR, NAND and NOR (c) XOR

	NOT	AND	OR	NAND	NOR	XOR
Tanh function	2.82nW	5.24nW	17.18nW	2.82nW	2.85nW	7.42nW
Unipolar Sigmoidal	1.34nW	4.36nW	13.90nW	1.36nW	2.55nW	8.46nW
Bi-polar Sigmoidal	3.58nW	2.62nW	16.81nW	4.10nW	3.64nW	8.25nW

Table 5.4: Total power dissipation of the SD neural network trained for various logic functions employing the three activation functions.

5.5 Conclusion

Low-power low-voltage realizations of three activation functions using SD technique have been introduced. The SD realizations of the activation functions have been subsequently applied to realize the artificial neural network. Finally, the neural network has been trained to obtain the all the basic, universal and typical XOR logic functions. The performed simulation results verify the theoretical predictions. The simulation results indicate that the realized low-voltage NN can be well applied for computational purposes with contemporary technology requirements.

6.1 Conclusion

An attempt to summarize the work done during the course of this study is made in this chapter. The physical realization of circuits presented in this dissertation was not done due to the lack of adequate resources at the place of research. The theoretical behaviour of the circuits has only been supported by SPICE simulation results.

The present investigation is primarily concerned with the study and design of low-voltage low-power neural networks (NNs). The motivation for this study emanated because NNs are finding a great deal of applications. They are being applied successfully across an extraordinary range of problem domain, in the areas as diverse as medicine, engineering, physics, finance, and biology. Furthermore design of circuits using low-voltage low-power techniques is the demand of the day. The amalgamation of low-voltage low-power technique and ANNs will prove to be very fruitful. Though work on the low power implementation of ANNs is already present in the literature but no work is present in the field of ANN implementation using companding techniques. In this work the implementation of various analog activation functions, which are one of the main component in the design of ANNs, and single cell architecture of complex TD-CNN is done in Sinh-Domain companding technique. The circuits have been tested using simulation results and verified for theoretical predictions. Furthermore, the idea about implementation of CNN network, which will later be trained for real time image processing, is also drawn at the end of this chapter.

Chapter 1gives a brief idea about the motivation for the study and objectives achieved in the dissertation.

Chapter 2 gives the brief introduction of the NNs. The inspiration of designing ANNs drawn from the biological neuron is given in this chapter. Besides CNNs are discussed in details and different types of CNNs is also presented in the chapter.

Chapter 3 discusses the need for the hardware implementation of ANNs. Besides, different types of hardware implementation of ANNs and the difficulties in achieving hardware implementation are also included. Furthermore the special emphasis is lead on the hardware implementation of CNNs along with the different approaches of achieving its hardware.

Chapter 4 presents a brief introduction of different techniques for low-power low-voltage implementation given in the literature. Companding techniques for designing low-power low-voltage circuits are discussed with the special emphasis on the SD companding technique. Besides various building blocks for the implementation of low-power low-voltage circuits in the SD technique are discussed in this chapter.

In chapter 5, analog activation functions (unipolar sigmoidal, bipolar sigmoidal and Tanh) implemented utilizing SD companding technique is presented. Besides, the designed activation functions are used in designing single-layer and two-layer perceptron whose weights are trained in such a manner that the network implements various logic gate functions.

6.2 Suggestions for Future Work

Despite much work has been done on low-voltage low-power designing of ANNs and in spite of the fact that wealth of the literature is available on the subject, there is still a lot of scope to further this knowledge.

The simulated results may not be exactly same as the results obtained from the hardware implementations despite the fact that SPICE simulators now have the capability to model the devices as close to the actual parameters. Thus, the first and foremost proposal for further work is to implement the proposed circuits in silicon. The proposed designs then can be applied by the real time signal obtained from the live source. This will give the scope for improving the circuit performances.

The designed activation functions are tested for their implementation for digital functions only but for further consideration they can be utilized for training and implementation of pure analog NNs besides their performances tested for different analog ANNs.

Also for the future works, the new nano-electronic devices (Multi-Gate FETs) can be used for designing of companding circuits. As the devices are compatible with the present planer CMOS technology and helps in designing of low-voltage circuits, their assimilation with companding techniques will help in further reduction of voltage and power requirements. Along with this, the newly found, the fourth basic circuit element ‘Memristor’ which has the property of remembrance besides the property of changing its value with the applied supply value, they are thought to be well suited for the implementation of synaptic connections of NNs. So the utilization of Memristors in the designing of ANNs can be tested and then those NNs can be used for achieving different applications.

A: List of Publications

- N. A. Kant, F. A. Khanday, C. Psychalinos and N. A. Shah, “0.5V Sinh-Domain Realization of Activation Functions and There Employment in Neuron Network Design for Logic Gate Implementations”, Communicated to Circuits, Systems & Signal Processing.
- N. A. Shah, F. A. Khanday, N. A. Kant, “Neural Networks in Hardware”, 8th JK Science congress, Srinagar, 2012.
- N. A. Kant, Imran Yaseen, F. A. Khanday, N. A. Shah, “Investigation of Memristor in Neural Network Design”, 9th JK Science congress, Srinagar, 2013.

B: Bibliography

[1] M. H. terBrugge, “Morphological Design of Discrete-Time Cellular Neural Networks”, Ph.D. thesis, Rijksuniversiteit Groningen, Groningen, The Netherlands, 2005.

[2] C. Bobda, “Introduction to Reconfigurable Computing: Architectures, Algorithms, and Applications”, Springer, ISBN 978-1-4020-6100-4, 2007.

[3] J. J. Hopfield, “Neural networks and physical systems with emergent computational abilities”, Proc. Natl. Acad. Sci. USA., volume 79, pp. 2554–2558, 1982.

[4] L. O. Chua and T. Roska, “Cellular neural networks and visual computing”, Cambridge University Press, page no. 1, 2004.

[5] L. O. Chua and L. Yang, “Cellular neural networks: Theory and applications”, IEEE Trans. Circuits Syst. I, Fundam. Theory Appl., volume 35, pp. 1257–1290, 1988.

[6] L. O. Chua and T. Roska, “The CNN paradigm”, IEEE Trans. Circuits Systems I: Fund. Theory Appl., volume 40, pp. 147–156, 1993.

[7] S. Espejo, R. Carmona, R. Dominguez-Castro and A. Rodriguez-Vazquez, “A CNN universal chip in CMOS technology”, Int. Circuit Theory Appl., volume 24, issue 1, pp. 93–110, 1996.

[8] P. Kinget and M. Steyaert, “An analog parallel array processor for real-time sensor signal processing”, Proc. IEEE Int. Solid-State Circuits, Conf. San Francisco, CA, U.S.A, pp. 92–93, 1996.

[9] A. R. Vazquez, S. Espejo, R. Dominguez-Castro, R. Carmona and E. Roca, “Mixed-signal CNN array chips for image processing”, Proceedings of SPIE, volume A2950, pp. 218–229, 1996.

[10] D. B. Kadam, A. B. Patil and K. P. Paradeshi, “Neural Network Based Intelligent Process Control System”, International Conference on

Recent Trends in Information, Telecommunication and Computing (ITC), Kochi, Kerala, pp. 356–358, 2010.

[11] Mohammed R. Sunnya and R. K. Kapania, “Artificial-Neural-Network-Based Identification of a Modified Dynamic Preisach Model”, International Journal for Computational Methods in Engineering Science and Mechanics, Special Issue: Composites for 21st Century: Current & Future Trends—Special Issue of CMESM, volume 15, issue 1, pp. 45–53, 2014.

[12] O. TOgaji Stephen and R. Singh, “Advanced engine diagnostics using artificial neural networks”, ScienceDirect, Applied Soft Computing, volume 3, issue 3, pp. 259–271, 2003.

[13] V. Frinken, A. Fischer, R. Manmatha and H. Bunke, “A Novel Word Spotting Method Based on Recurrent Neural Networks”, IEEE Transactions on Pattern Analysis and Machine Intelligence, volume 34, issue: 2, pp. 211– 224, 2012.

[14] E. Golfinopoulos, J. A. Tourville and F. H. Guenther, “The integration of large-scale neural network modeling and functional brain imaging in speech motor control”, Elsevier, NeuroImage, Computational Models of the Brain, volume 52, issue 3, pp. 862–874, 2010.

[15] Servet Soyguder, “Intelligent control based on wavelet decomposition and neural network for predicting of human trajectories with a novel vision-based robotic”, Elsevier, Expert Systems with Applications, volume 38, issue 11, pp. 13994–14000, 2011.

[16] Mehdi Khashei and Mehdi Bijari, “A novel hybridization of artificial neural networks and ARIMA models for time series forecasting”, Elesvier, Applied Soft Computing, The Impact of Soft Computing for the Progress of Artificial Intelligence, volume 11, issue 2, pp. 2664–2675, 2011.

[17] D. H. Sanes, T. A. Reh and W. A. Harris, “Development of the nervous system”, Academic press, Ch. 1., 2000.

[18] A. Garm, M. O’Connor, L. Parkefelt and D. E. Nilsson, “Visually guided obstacle avoidance in the box jellyfish tripedaliacystophora and chiropsellabronzie”, Journal of Experimental Biology, volume 210, issue (20, pp. 3616–3623, Oct. 2007.

[19] R. W. Williams, K. Herrup, “The control of neuron number”, Annual review of neuroscience, volume 11, issue 1, pp. 423–453, 1988.

[20] B. Pakkenberg, D. Pelvig, L. Marner, M. J. Bundgaard, H. J. J. Gundersen, J. R. Nyengaard and L. Regeur, “Aging and the human neocortex”, Experimental gerontology 38 (1-2), pp. 95–99, 2003.

[21] E. R. Kandel, J. H. Schwartz and T. M. Jessell, “Principles of Neural Science”, McGraw-Hill Medical, 4th Edition. Ch. 17, 2000.

[22] M. A. Arbib, “Brains, Machines and Mathematics”, New York: Springer Verlag, 2nd edition, 1987.

[23] Maureen Caudill, “Neural Network Primer: Part I”, AI Expert, volume 2, issue 12, pp. 46–52, 1987.

[24] R. Durbin, “On the Correspondence between Network Models and the Nervous System”, Addison-Wesley Publishing Co., Computing Neuron (Book), ISBN 0-201-18348-X, pp. 1–10, 1989.

[25] S. Haykin, “Neural Networks: a comprehensive foundation”, Prentice Hall, 2nd edition, 1999.

[26] J. Kari, “Theory of cellular automata: A survey”, Elsevier, Theoretical Computer Science, volume 334, pp. 3–33, 2005.

[27] Huaqing Li, Xiaofeng Liao, Chuandong Li, Hongyu Huang and Chaojie Li, “Edge detection of noisy images based on cellular neural networks”, Elsevier, Communications in Nonlinear Science and Numerical Simulation, volume 16, issue 9, pp. 3746–3759, September 2011.

[28] B. Zineddin, Wang Zidong and Liu Xiaohui, “Cellular Neural Networks, the Navier–Stokes Equation, and Microarray Image Reconstruction”, IEEE Transactions on Image Processing, volume 20, issue 11, pp. 3296–3301, Nov. 2011.

[29] F. Jeanson and A. White, “Dynamic memory for robot control via delay neural networks”, AMC Digital Library, Proceeding of the fifteenth annual conference companion on Genetic and evolutionary computation conference companion, New York, pp. 29–30, 2013.

[30] S. Espejo, R. Carmona, R. Dominguez-Castro and A. Rodriguez-Vazquez, “A VLSI-oriented continuous time CNN model”, International Journal Circuit Theory Appl. Volume 24, pp. 341–356, 1996.

[31] M. D. I. Henry, E. M. Drakakis and A. A. Bharath, “Towards analog VLSI arrays for non-separable 3-D spatiotemporal filtering”, in Proc. IEEE International Workshop Cellular Neural Networks and their Applications (CNNA’06), Istanbul, Turkey, pp. 1–6, 2006.

[32] L.O. Chua, “CNN: A Paradigm for Complexity”, World Scientific Series on Nonlinear Science, Series A, volume 31, ISBN 981-02-3483-X, 1998.

[33] H. Harrer, “Discrete-Time Cellular Neural Networks”, Ph.D. thesis, Verlag Shaker, ISBN 3-86111-286-8, Aachen, 1992.

[34] G. Manganaro, P. Arena and L. Fortuna, “Cellular Neural Networks, Chaos, Complexity and VLSI Processing”, Springer, Verlag, Berlin Heidelberg, ISBN 978-3-642-60044-9, 1999.

[35] L. O. Chua and T. Roska, “Cellular neural networks and visual computing – Foundation and applications”, Cambridge University Press, ISBN 13 978-0-521-01863-0, 2002.

[36] Suleyman Malki, “On Hardware Implementation of Discrete-Time Cellular Neural Networks”, Ph.D. thesis, Department of Electrical and

Information Technology, Faculty of Engineering, Lund University, 2008.

[37] H. D. I. Henry, E. M. Drakakis and A. A. Bharath, “On analog networks and mixed-domain spatio-temporal frequency response”, IEEE Trans. Circuits Systems I, volume 55, pp. 284–297, 2008.

[38] W. G. Baxt, “Application of Artificial Neural Networks to Clinical Medicine”, The Lancet, volume 346, issue 8983, pp. 1135–1138. 1995.

[39] A. Miller, B. Blott and T. Hames, “Review of Neural Network Applications in Medical Imaging and Signal Processing”, Medical and Biological Engineering and Computing, volume 30, issue 5, pp. 449–464, 1992.

[40] K. K. Weinstein, M. Grever and et al., “Neural Computing in Cancer Drug Development: Predicting Mechanism of Action”, Science, volume 258, pp. 447–451, 1992.

[41] J. N. Weinstein, T. Myers, J. Buolamwini, K. Raghavan and et al., “Predictive statistics and artificial intelligence in the U.S. National Cancer Institute’s Drug Discovery Program for Cancer and AIDS”, Stem Cells, volume 12, issue 1, pp. 13–22, 1994.

[42] H. P. Grafand and L. D. Jackal, “Analog electronic neural network circuits”, IEEE Circuits and Devices Magazine, volume 5, issue 4, pp. 44–49, 1989.

[43] S. Y. Foo, L. R. Anderson and Y. Takefufi, “Analog components for the VLSI of neural Networks”, IEEE Circuits and Devices Magazine, volume 6, issue 4, pp. 18–26, 1990.

[44] C. Mead, “Analog VLSI and Neural Systems”, Addison-Wesley Longman Publishing Co., Inc. Boston, MA, USA, ISBN:0-201-05992-4, 1989.

[45] C. Chao-Ming, L. Chih-Min, C. Ching-Tsan and D. S. Yeung, “Hardware implementation of CMAC neural network using FPGA

approach", IEEE International Conference on Machine Learning and Cybernetics, Hong Kong, volume 4, pp. 2005–2011, 2007.

[46] J. Seul and K. Sung, "Hardware implementation of a real-time neural network controller with a DSP and an FPGA for nonlinear systems", IEEE Trans. Industrial Electronics, volume 54, 265–271, 2007.

[47] L. E. Atlas and Y. Suziki, "Digital systems for artificial neural networks", IEEE Circuits and Devices Magazine, volume 5, issue 6, pp. 20–24, 1989.

[48] A. F. Murray and L. Tarassenko, "Analogue Neural VLSI: A pulse stream approach", Chapman & Hall, Ltd. London, UK, ISBN 0-412-45060-7, 1994.

[49] M. S. Tomlinson, D. J. Walker and M. A. Sivilotti, "A digital neural network architecture for VLSI", in Proc. IEEE IJCNN, San Diego, CA, volume 2, pp. 545–550, 1990.

[50] R. A. Leaver, "Stochastic Arrays and Learning Networks", Ph.D. thesis, University of Durham, 1988.

[51] J. Kari, "Theory of cellular automata: A survey", Elsevier, Theoretical Computer Science, volume 334, pp. 3–33, 2005.

[52] L. Yang, L. O. Chua and K. R. Krieg, "VLSI Implementation of Cellular Neural Networks", in Proc. IEEE International Symposium on Circuits and Systems (ISCAS-90), volume 3, pp. 2425–2427, 1990.

[53] T. Roska, G. Bartfai, P. Szolgay and et al. "A hardware accelerator board for Cellular Neural Networks: CNN-HAC", Proceedings IEEE International Workshop on Cellular Neural Networks and their Applications, Budapest, pp. 160–168, 1990.

[54] L. O. Chua and T. Roska, "The CNN universal machine – Part I: The architecture", Proceedings 2nd IEEE International Workshop on

Cellular Neural Networks and their Applications, Munich, pp. 1–10, 1992.

[55] V. Cimagalli and M. Balsi, “Cellular Neural Networks: A Review”, Proc. of 6th Italian Workshop on Parallel Architectures and Neural Networks, World Scientific (E. Caianiello, ed.), Italy, pp. 1–31, 1993.

[56] J. M. Cruz and L. O. Chua, “Design of High-Speed, High-Density CNNs in CMOS Technology”, International journal Circuit Theory and Applications, volume 20, issue 5, pp. 555–572, 1992.

[57] K. Halonen, V. Porra, T. Roska and L.O. Chua, “Programmable Analog VLSI Chip with Local Digital Logic”, International journal Circuit Theory and Applications, volume 20, issue 5, pp. 573–582, 1992.

[58] J. A. Nossek, G. Seiler, T. Roska and L.O. Chua, “Cellular Neural Networks: Theory and Circuit Design”, International Journal of Circuit Theory and Applications, volume 20, issue 5, pp. 533–553, 1992.

[59] H. Harrer, J.A. Nossek and R. Stelzl, “An Analog Implementation of Discrete-Time Cellular Neural Networks”, I EEE Transactions on Neural Networks, volume 3, issue 3, pp. 466 – 476, 1992.

[60] A. Rodriguez-Vazquez, S. Espejo, R. Dominguez-Castro, J. L. Huertas, E. S. Sinencio, “Current-Mode Techniques for the Implementation of Continuous and Discrete-Time Cellular Neural Networks”, IEEE Transactions on Circuits and Systems CAS-II, volume 40, issue 3, pp. 132–146, 1993.

[61] K. Slot, “Optically Realized Feedback Cellular Neural Networks”, Proceedings Second International Workshop on Cellular Neural Networks and their Applications, Munich, pp.175–180, 1992.

[62] A. Wang, B. H. Calhoun and A. P. Chandrakasan, “Sub-threshold Design for Ultra Low-Power Systems”, Series: Series on Integrated Circuits and Systems, Springer, USA, ISBN 978-0-387-34501-7, 2007.

[63] A. A. El-Adawy and A. M. Soliman, “A Low-Voltage Single Input Class AB Transconductor with Rail-to-Rail Input Range”, IEEE Transactions on Circuits and Systems-II, volume 47, issue 2, pp. 236–242, 2000.

[64] J. F. Duque-Carrilloet et al., “1-V Rail-to-Rail Operational Amplifiers in Standard CMOS Technology”, IEEE Journal of Solid State Circuits, volume 35, issue 1, pp. 33–43, 2000.

[65] C. Lin and M. Ismail, “Robust Design of LV/LP Low-Distortion CMOS Rail-to-Rail Input Stages”, Journal of Analog Integrated Circuits and Signal Processing, volume 21, pp. 153–161, 1999.

[66] Klaas-Jan deLangen and J. H. Huijsing, “Compact Low-Voltage Power-Efficient Operational Amplifier Cells for VLSI”, IEEE Journal of Solid State Circuits, volume 33, issue 10, pp. 1482–1496, 1998.

[67] G. Ferri, W. Sansen and V. Peluso, “A Low-Voltage Fully-Differential Constant-Gm Rail-to-Rail CMOS Operational Amplifier”, Journal of Analog Integrated Circuits and Signal Processing, volume 16, pp. 5–15, 1998.

[68] G. Xu et al., “A Low Voltage Fully Differential Nested Capacitance Amplifier: Analysis and Design”, Proceedings of the IEEE International Symposium on Circuits and Systems, volume 2, pp. 606–609, 1999.

[69] S. Setty and C. Toumazou, “N-Folded Cascode Technique for High Frequency Operation of Low Voltage Op-amps”, Electronics Letters, volume 32, issue 11, pp. 955–956, 1996.

[70] R. Fried and C. Enz, “Nano-Amp, Active Bulk, Weak-Inversion Analog Circuits”, Proceedings of the IEEE Custom Integrated Circuits Conference, pp. 31–34, 1998.

[71] B. J. Blalock, P. E. Allen and G. A. Rincon-Mora, “Designing 1-V Op- Amps Using Standard Digital CMOS Technology”, IEEE

Transactions on Circuits and Systems-II, volume 45, issue 7, pp. 769–779, 1998.

[72] J. Shin et al., “A New Charge Pump Without Degradation in Threshold Voltage Due to Body Effect”, IEEE Journal of Solid State Circuits, volume 35, issue 8, pp. 1227–1230, 2000.

[73] J. Wu and K. Chang, “MOS Charge Pumps for Low-Voltage Operation”, IEEE Journal of Solid State Circuits, volume 33, issue 4, pp. 592–597, 1998.

[74] T. A. F. Duisters and E. C. Dijkmans, “A -90dB THD Rail-to-Rail Input Opamp Using a New Local Charge Pump in CMOS”, IEEE Journal of Solid State Circuits, volume 33, issue 7, pp. 947–955, 1998.

[75] H. Lin, K. Chang and S. Wong, “Novel High Positive and Negative Pumping”, Proceedings of the IEEE International Symposium on Circuits and Systems, volume 1, pp. 238–241, 1999.

[76] R. St. Pierre, “Low-Power BiCMOS Op-Amp with Integrated Current Mode Charge Pump”, Proceedings of the IEEE European Solid-State Circuits Conference, pp. 70–73, 1999.

[77] J. F. Dickson, “On-Chip High-Voltage Generation MNOS Integrated Circuits Using an Improved Voltage Multiplier Technique”, IEEE Journal of Solid State Circuits, volume 11, issue 3, pp. 374–378, 1976.

[78] P. Favrat, P. Deval and M. J. Declercq, “An Improved Voltage Doubler in a Standard CMOS Technology”, Proceedings of the IEEE International Symposium on Circuits and Systems, volume 1, pp. 249–252, 1997.

[79] J. Sim et al., “Adaptive Biasing Folded Cascode CMOS OP Amp with Continuous-Time Push-Pull CMFB Scheme”, IEICE Transactions on Electronics, volume E80-C, issue 9, pp. 1203–1210, 1997.

[80] M. G. Degraune et al., “Adaptive Biasing CMOS Amplifiers”, IEEE Journal of Solid State Circuits, volume 17, issue 3, pp. 522–528, 1982.

[81] R. Klinke, B. J. Hosticka and H. J. Pfleiderer, “A Very-High-Slew-Rate CMOS Operational Amplifier”, IEEE Journal of Solid State Circuits, volume 24, issue 3, pp. 744–746, 1989.

[82] K. Nagaraj, “Slew Rate Enhancement Technique for CMOS Output Buffers”, Electronics Letters, volume 25, issue 19, pp. 1304–1305, 1989.

[83] E. Vittoz and J. Fellroth, “CMOS Analog Integrated Circuits Based on Weak Inversion Operation”, IEEE Journal of Solid State Circuits, volume 12, issue 3, pp. 224–231, 1977.

[84] C. Toumazou, F. J. Lidgey and D. G. Haigh, “Analogue IC design: The current mode approach”, Stevenge, U. K.: Peregrinus Ltd, 1990.

[85] Y. Tsividis, “Externally linear time-invariant systems and their applications to companding signal processors”, IEEE Transaction on Circuits and Systems-II, volume 44, issue 2, pp. 65–85, Feb. 1997.

[86] D. R. Frey, “Log-domain filtering: an approach to current mode filtering”, IEE Proc. G 1993, volume 140, issue 6, pp. 406–416, Dec. 1993.

[87] N. Fragoulis, C. Psychalinos and I. Haritantis, “Modular Log-Domain Filters Realized Using Wave Port Terminators”, IEEE Transaction on Circuits and Systems-I, volume 51, issue 11, pp. 2235–2244, 1995.

[88] C. Psychalinos, “On the transposition of Gm–C filters to DC stabilized log-domain filters”, International Journal of Circuit Theory and Applications, volume 34, issue 2, pp. 217–236, 2006.

[89] X. Zhang, W. Mekaw and E. El-Masry, “Leap-frog-based synthesis of CMOS micropower nth-order log-domain filters”, Proceeding IEEE International Symposium on Circuit and Systems (ISCAS’2004), pp. 305–308, 2004.

[90] N. Kontogiannopoulos and C. Psychalinos, “Log-domain filtering by simulating the topology of passive prototypes”, IEEE Transaction on Circuits and Systems-I, volume 52, pp. 2043–2054, 2005.

[91] C. Psychalinos and S. Vlassis, “Asystematic design procedure for square-root domain circuits based on the signal flow graph approach”, IEEE Transaction on Circuits and Systems-I, volume 49, pp. 1702–1712, 2002.

[92] C. Psychalinos, “Design of square-root domain filters by substituting the passive elements of the prototype filter by their equivalents”, International Journal of Circuit Theory and Applications, volume 36, pp. 185–204, 2008.

[93] E. Stoumpou and C. Psychalinos, “Square root domain linear transformation filters”, International Journal of Circuit Theory and Applications, volume 39, pp. 719–731, 2011.

[94] A. J. Lopez-Martin and A. Carlosena, “Systematic design of companding systems by component substitution”, Analog Integrated Circuits and Signal Processing, volume 28, pp. 91–106, 2001.

[95] W. Serdijn et al., “Design of High Dynamic Range Fully Integrable Translinear Filters”, Analog Integrated Circuits and Signal Processing, volume 19, pp. 223–239, 1999.

[96] A. Katsiamis, K. Glaros and E. Drakakis, “Insights and Advances on the Design of CMOS Sinh Companding Filters”, IEEE Transactions on Circuits and Systems-I, volume 55, pp. 2539–2550, 2008.

[97] D. Frey, “Exponential state-space filters: a generic current mode design strategy”, IEEE Transaction on Circuits and Systems-I, volume 43, pp. 34–42, 1996.

[98] R. W. Adams, “Filtering in the log domain”, Preprint #1470 presented at the 63rd AES Conference, New York, NY, May 1979.

[99] B. Gilbert, “Translinear circuits: a proposed classification”, Electronics Letters, volume 11, issue 1, pp. 14–16, 1975.

[100] D. Frey, “Log-domain filtering: an approach to current- mode filtering”, IEE Proc. Pt. G, volume 140, pp. 406–416, 1993.

[101] M. Eskiyerli, A. J. Payne and C. Toumazou, “State-space synthesis of integrators based on the MOSFET square law”, Electronics Letters, volume 32, pp. 505–506, 1996.

[102] M. Eskiyerli and A. Payne, “Square-root domain filter design and performance”, Analog Integrated Circuits and Signal Processing, volume 22, pp. 231–243, 2000.

[103] J. Mulder et al., “Current-mode companding $\sqrt{x}$ -domain integrator”, Electronics Letters, volume 32, pp. 198–199, 1996.

[104] A. J. Lopez-Martin and A. Carlosena, “Geometric-mean based current-mode multiplier/divider”, Proceedings of IEEE International Symposium on Circuits Systems (ISCAS’99), pp. 342–345, 1999.

[105] A. J. Lopez-Martin and A. Carlosena, “A 3.3 V tunable current-mode square-root domain biquad”, Proceedings of IEEE International symposium on circuits and systems (ISCAS), Geneva, Switzerland, pp. 5–8, 2000.

[106] R. Wiegerink, “Analysis and synthesis of MOS translinear circuits”, Dordrecht, The Netherlands: Kluwer Academic Publishers Group; pp. 176,246, 1993.

[107] P. Poort et al., “A 1-V class AB translinear integrator for filters applications”, Analog Integrated Circuits and Signal Processing, volume 21, pp. 79–90, 1999.

[108] C. Sawigun and W. Serdijn, “Ultra-low power, class-AB, CMOS four quadrant current multiplier”, Electronics Letters, volume 45, pp. 483–484, 2009.

[109] F. A. Khanday, “Realization of Integrable Low-Voltage Companding Filters for Portable System Applications”, Ph.D. thesis, University of Kashmir, Srinagar, 2013.

[110] S. Haykin, “Neural networks: a comprehensive foundation”, Prentice Hall, USA, ISBN 0132733501 1998.

[111] K. Basterretxea, J. M. Tarela and I. Del Campo, “Approximation of sigmoid function and the derivative for hardware implementation of artificial neurons”, IEE Proceedings Circuits, Devices and Systems, volume 151, pp. 18–24, 2004.

[112] K. Basterretxea and J. M. Tarela, “Approximation of sigmoid function and the derivative for artificial neurons”, Advance in neural networks and applications, WSES Press, Athens, pp. 397–401, 2001.

[113] H. K. Kwan, “Simple sigmoid-like activation function suitable for digital hardware implementation”, Electronic Letters, volume 28, pp. 1379–1380, 1992.

[114] F. Piazza, A. Uncini and M. Zenobi, “Neural networks with digital LUT activation functions”, Proceedings of 1993 International Joint Conference on Neural Networks, IJCNN, pp. 1401–1404, October 1993.

[115] G. Khodabandehloo, M. Mirhassani and M. Ahmadi, “Analog Implementation of a Novel Resistive-Type Sigmoidal Neuron”, IEEE Transactions Very Large Scale Integration (VLSI) Systems, volume 20, pp. 750–754, 2012.

[116] S. Tabarce, V. G. Tavares and P. Guedes de Oliveira, “Programmable analogue VLSI implementation for asymmetric sigmoid neural activation function and its derivative”, Electron Letter, volume 41, pp. 1–2, 2005.

[117] N. Nedjah and R. M. DaSilva, “Analog hardware implementations of artificial neural networks”, Journal of Circuits Systems Comp., volume 20, pp. 349–373, 2011.

[118] A. I. Khuder and S. H. Husain, “Hardware Realization of Artificial Neural Networks Using Analogue Devices”, Al-Rafidain Eng. 21, pp. 77–90, 2013.

[119] G. Bogason, “Generation of a neuron transfer function and its derivatives”, Electron Letter, volume 29, pp. 1867–1869, 1993.

[120] C. Lu and B. Shi, “Circuit design of an adjustable neuron activation function and its Derivative”, Electron Letter, volume 36, pp. 553–555, 2000.

[121] F. Keles, M. Avci and L. Yıldırım, “Floating gate MOS transistor based low voltage neurondesign and XOR problem implementation”, Proceedings of the 3rd International Conference on Electrical and Electronics Engineering (ELECO’2003), Bursa, Turkey, pp. 41–45 (2003).

[122] F. Keles and L. Yıldırım, “Low voltage low power neuron circuit design based on sub-threshold FGMOS transistors and XOR implementation”, Proceedings of the XIth International Workshop on Symbolic and Numerical Methods, Modeling and Applications to Circuit Design (SM2ACD), Tunisia, pp. 1–5, 2010.

[123] A. Katsiamis, K. Glaros and E. M.Drakakis, “Insights and Advances on the Design of CMOS Sinh Companding Filters”, IEEE Transactions Circuits Systems-I, volume 55, pp. 2539–2550, 2008.

[124] C. Kasimis and C. Psychalinos, “0.65 V class-AB current-mode four-quadrant multiplier with reduced power dissipation”, International Journal Electron Comm., AEU, volume 65, pp. 673–677, 2011.

[125] C. Kasimis and C. Psychalinos, “Design of Sinh-Domain Filters Using Complementary Operators”, International Journal Circuit Theory Appl., volume 40, pp. 1019–1039, 2012.

[126] F. A. Khanday and N. A. Shah, “A Low Voltage and Low Power Sinh-Domain Universal Biquadratic Filter for low frequency Applications”, Turkish Journal Elect. Eng. Comp. Sci., DOI: 10.3906/elk-1203-128.

[127] C. Kasimis and C. Psychalinos, “1.2V Bi-CMOS Sinh-Domain Filters”, Circuit Systems Signal Proc., volume 31, pp. 1257–1277, 2012.

[128] F. A. Khanday, E. Pilavaki and C. Psychalinos, “Ultra Low-Voltage, Ultra Low-Power Sinh-Domain Wavelet filter for ECG analysis”, ASP J. Low Power Electron., volume 9, pp. 1–7, 2013.

[129] M. Panagopoulou, C. Psychalinos, F. A. Khanday and N. A. Shah, “Sinh-Domain Multiphase Sinusoidal Oscillator”, Microelectronics Journal, http://dx.doi. org/10.1016/j.mejo.2013.06.017.

[130] K. Roumelioti, C. Psychalinos, F. A. Khanday and N. A. Shah, “1.2V Sinh-Domain Allpass Filter”, International Journal Circuit Theory Appl., DOI: 10.1002/cta.1922.

[131] F. A. Khanday, C. Kasimis, C. Psychalinos and N. A. Shah, “Sinh-Domain Linear Transformation Filters”, International Journal Electron., http://dx.doi.org/10.1080/00207217.2013.780265.

[132] D. E. Rumelhart, J. L. McClelland, the PDP Research Group, “Parallel Distributed Processing”, The MIT Press. 1: Foundations 1987.

[133] H. Barthelemy, S. Meillere and E. Kussener, “CMOS sinusoidal oscillator based on current controlled current conveyors”, Electron Lett., volume 38, pp. 1254–1256, 2002.

[134] P. B. L. Meijer, “Neural network applications in device and subcircuit modeling for circuit simulation”, PhD thesis, Eindhoven University, 1996.

C: Nomenclature

Activation Functions (AFs)

Artificial Neural Network (ANN)

Bipolar Complementary Metal Oxide Semiconductor (Bi-CMOS)

Boltzmann's Constant (k)

Cellular Neural Network (CNN)

Continuous -Time Cellular Neural Network (CT-CNN)

Discrete-Time (DT)

Discrete-Time Cellular Neural Network (DT-CNN)

Externally Linear and Internally Nonlinear (ELIN)

Full Signal Range (FSR)

Hardware Accelerator Board (HAC)

International Technology Roadmap for Semiconductors (ITRS)

Linear and Time-Invariant (LTI)

Log-Domain (LD)

Low-Power Low-Voltage (LP LV)

Micro-Electro-Mechanical System (MEMS)

Micro-Meter (um)

Milli-Volt (mV)

Nano-Watt (nW)

Operational Amplifiers (Op-Amp)

Operational Transconductance Amplifiers (OTA)

Pico-Ampere (pA)

Pico-Farad (pF)

Piecewise linear (PWL)

Resistance Capacitance (RC)

Sinh-Domain (SD)

Spatial Light Modulators (SLM)

Square-Root Domain (SRD)

Very Large Scale Integration (VLSI)

Voltage-Controlled Current Sources (VCCS)

Weak Inversion MOSFET (WIMOSFET)