This is to certify that the

dissertation entitled

CMOS VLSI IMPLEMENTATIONS OF A NEW
FEEDBACK NEURAL NETWORK ARCHITECTURE

presented by

YIWEN WANG

has been accepted towards fulfillment
of the requirements for

Ph.D. degree in Electrical Engineering

Major professor

Date Aug 8, 1991

DATE DUE	DATE DUE	DATE DUE
FEB 03 [illegible]		

CMOS VLSI IMPLEMENTATIONS OF
A NEW FEEDBACK NEURAL NETWORK ARCHITECTURE

By

Yiwen Wang

A DISSERTATION

Submitted to
Michigan State University
in partial fulfillment of the requirements
for the degree of

DOCTOR OF PHILOSOPHY

Department of Electrical Engineering

1991

ABSTRACT

CMOS VLSI IMPLEMENTATIONS OF A NEW FEEDBACK NEURAL NETWORK ARCHITECTURE

By

Yiwen Wang

This work develops CMOS VLSI implementations of a new architecture for feedback Artificial Neural Networks (ANNs). The new architecture lends itself directly to all-MOS implementations and it has been shown to exhibit qualitatively the same dynamic properties as gradient continuous-time feedback neural nets. Neural properties of a prototype of this new architecture had been verified via extensive SPICE simulations and discrete-component laboratory experiments.

A 6-neuron Tiny-chip of the new architecture was designed and fabricated as a prototype all-MOS chip of the new architecture for developing and assessing various potential off-chip learning algorithms. A new dynamic learning algorithm is described for general dynamic continuous-time models of ANNs. The learning algorithm is specialized to the new architecture of ANNs and is successfully tested on the 6-neuron Tiny-chip.

Subsequently a digital MOS realization of the learning algorithm is developed. A 50-neuron CMOS analog chip with on-chip digital learning scheme was designed and fabricated in $6.8mm \times 4.6mm$ chip size with 63,025 transistors and 1225 programmable synaptic (interconnect) weights via standard $2\mu m$ CMOS n-well technology.

Dedicated interface circuitries and software environments had been built to successfully demonstrate the use of the prototype chips of the new neural circuit. As an example of an application of the fabricated neural net chip, real-time experiments are described in which the chips are used as a coprocessor to a microcomputer. These experiments entail learning an arbitrary image which can be subsequently retrieved by images distorted by binary-noise in the order of 20 μsecs in real time.

To our knowledge, these are the first successful and effective (analog) neural chip experiments with guaranteed learning capability. The literature on the implementation of ANNs has dramatically grown, yet, to the best of our knowledge, none of the proposed neural implementations has been shown to execute and substantiate the claimed learning and retrieval capabilities in real time. The implementations and real-time experimental results described here have been shown to perform and substantiate the capabilities attributed to ANNs by pursuing an approach of analysis followed by direct electronic implementations.

To

my mother; my wife, Meei-Jyh; and my daughter, Ming.

ACKNOWLEDGMENTS

I would like to express my sincere gratitude to my major advisor, Dr. Fathi M. A. Salam, for his inspiration, guidance and support throughout the years of my graduate studies. Thanks to Dr. Lionel M. Ni, Dr. Timothy Grotjohn, and Dr. Paul Parker for their valuable comments and suggestions.

I wish to express my sincere thanks and appreciation to my mother and my wife's parents for their continued prayers and encouragement.

Form the bottom of my heart, I would like to thank my wife, Meei-Jyh, for her love, understanding, and support, my brother, Chen-Wan, and my sister, Yung-Shih, for their help and encouragement, and my lovely daughter, Ming.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

Artificial Neural Networks (ANNs) have recently become very popular due to their anticipated ability to solve problems such as vision, speech understanding, pattern recognition with distorted data, information retrieval with partial information, etc.. These networks attempt to mimic the functionality of the nervous system and the brain. The human nervous system and the brain have the capabilities of creativity, adaptive learning, error correction, and robustness. It is believed, for instance, that the massive parallelism and high interconnectivity in the biological neural networks account for some of their computational capabilities and intelligence. An ANN typically possesses parallel-like structure manifested as high connectivity.

Although it is perhaps proper that one includes a tutorial introduction to the field, this has not been possible for several reasons: one being the unbounded diversity of the field, and another is the publication of various recent textbooks covering numerous orientations and foci [26,48,81].

During the 1960's, artificial neurons were extensively studied. In many cases these were simulations of biological neurons which attempted to take into account all the many and varied nuances of real neurons. In other cases the neurons were oversimplified but the accompanying algorithms were not suitably developed. Recently, very simple neurons have been considered resulting in emphases have shifted away from the neuron itself toward the interconnections of neurons and the weighting of their signals. Many researchers have proposed electrical circuits that model the highly interconnected nervous system and the brain. These circuits are often referred to as artificial neural nets. Architectures for artificial neural nets have been reported in numerous works, e.g. [1-7] and [8-12]. In particular, [13-26] are concerned with some aspects of implementations.

However, there exist two major problems that must be attacked before the realizations of such artificial neural networks can be achieved. The first problem is how to implement those global and dense interconnections among many neuron-like elements, and the second is how to program such highly dense interconnect weights. One of the goals of the field is to produce hardware, containing millions of neurons, that mimics the signal processing capabilities of the brain where millions of neurons are present. The majority of approaches for implementation, however, fall into two classes: (i) those that are software-based and (ii) those that are hardware-based but with relevance toward hardware realization and design for real-time applications.

Software-based implementations usually employ an algorithm, which is based on the architecture and understanding of models of neural networks, mapped onto a conventional digital computer. Software-based implementations are flexible and easily modified. However, when large ANNs are represented by a set of differential equations, simulation on a digital computer may take a considerable amount of computing time. Consequently, it becomes apparent that the field will be driven by the development of devices that are simple and can readily be realized in hardware VLSI, optics, or electro-optics.

Neurons in the human brain are interconnected in 3-D space because it is the most natural and efficient way of interconnection, but VLSI-based interconnections are inherently 2-D in nature. The high interconnectivity of these networks makes an electronic implementation difficult. Several schemes of VLSI implementations of neural networks have been proposed [13-26]. Optical signals can flow through 3-D space to achieve the required interconnections between neuron-like logic elements. Psaltis and Farhat reported an optical implementation of the Hopfield neural network using an optical vector-matrix multiplier as a programmable interconnector and illustrated the potential feasibility of optical content addressable associative memory [27]. Salam and Wang developed a formulation for 2-D array neural net processing which is suitably mapped into various electro-optical implementations [28,29]. However, due to the immaturity of the present optical technologies, it is difficult to presently make optical devices that are compact and easily programmable (to implement large scale artificial neural networks).

Other goals of the field are to develop learning schemes [10,30-37] which effectively *choose* or "*program*" the interconnections so as to render a relative configuration of the neural network that corresponds to a pre-specified set of stored data. The Backward Error Propagation (BEP) [34] has been somewhat successful as a supervised learning rule for multilayered feedforward networks with hidden units. The multilayered feedforward network is a continuous valued mapping from the input to the output space. Its mapping function is adapted by repeatedly presenting the network with a set of input-output vector pairs and using an appropriate update law (typically gradient descent) to modify the mapping until a functional relationship is realized that approximately satisfies the set of input-output pairs. However, as originally introduced, BEP is limited to feedforward networks which gave marginal, albeit, and satisfactory success. Networks with feedback necessarily raise the problem of overall system stability [38]. There are several learning rules such as Hebb's rule [36] and the Pseudo-inverse rule [37] for feedback models, but none of the traditional rules has been shown to be effective in applications.

The overall objective of this research is threefold: *Firstly*, to establish simulation and experimental foundations for the operation of a certain, recently proposed, neural network architecture. *Secondly*, to design and implement the neural net architecture via CMOS digital/analog VLSI/LSI circuitry. *Thirdly*, to demonstrate the performance of the design using the fabricated VLSI chips. The focus for applications will be on the proper design and implementation of this architecture as a classification device.

CMOS VLSI implementations of a new architecture for feedback ANNs are described in this thesis. The new architecture has qualitatively the same dynamic properties as gradient continuous-time feedback neural nets [39,40]. Moreover, the architecture also has the following features:

i) It reduces the maximum number of connections to $n\ (n + 1)/2$, where n is the number of neuron processors in the network.

ii) It does not require the symmetry of interconnections (in order to ensure the convergence of all solutions to equilibria only).

iii) More importantly, it does not require the realization of linear resistive elements for the synaptic weights. Instead, the connections are realized via MOSFET conductance elements.

iv) It lends itself naturally to direct analog MOS VLSI silicon implementation.

A 3-*unit* prototype circuit is employed to illustrate and test the characteristics and the performance of the new architecture. Extensive SPICE simulations are conducted to obtain the transient evolution to the steady states (or stable equilibria). Discrete-component laboratory experiments are also performed. The results are qualitatively identical to the SPICE simulations.

A 6-neuron Tiny-chip of the all-MOS implementation of the new architecture was designed and fabricated as a prototype chip of the new architecture. Complete testing of the Tiny-chips has successfully revealed the proper operation of the neural circuit.

A new dynamic learning algorithm is described for general dynamic continuous-time models of ANNs [30] which was motivated from our efforts to develop a learning algorithm for the new architecture. The learning capabilities of the new algorithm were tested on the 6-neuron Tiny-chip for storing and retrieving arbitrary digitized images.

The new learning algorithm has subsequently been specialized to a digital learning scheme which is realizable in the all-MOS VLSI implementations of ANNs. A 50-neuron CMOS VLSI chip of the new architecture with on-chip *digital* hardware learning scheme was successfully designed and fabricated on $6800\mu m \times 4600\mu m$ chip size using a MOSIS 2-μm scalable CMOS technology and a 64-pin standard pad frame. There are 63,025 transistors, 1225 programmable synaptic weights, and 50 neurons on a single chip. Each neuron consists of two CMOS inverters in series with one feedback nMOS transistor between the input node and the output node. The gate voltage of the feedback nMOS transistors can be adjusted globally. Either the input node or the output node of the neuron can be connected to an external pin via a CMOS analog switch. Each synaptic weight can be set up via on-chip *digital* hardware learning circuitry or via direct assignment and it can be stored on an on-chip digital flip-flop.

Dedicated interface circuitries and software environments were designed and developed for a personal computer in order to facilitate testing and verifying the performance of the fabricated chips. Real-time experiments are conducted to successfully demonstrate the use of the 50-neuron chip with on-chip digital learning as a real-time pattern/character recognizer and associator.

The thesis is organized as follows. Chapter 2 covers some background material on neural networks which is pertinent to the content and emphasis of this thesis. It includes a biological overview of neural networks, some simplified neural net models, and an overview of the current literature on the circuit implementation of ANNs. Because standard CMOS technology is employed to implement ANNs in this thesis, Chapter 2 also briefly introduces the basic concepts of MOS transistors and the characteristics and limitations of VLSI/LSI design.

Chapter 3 starts out with an introduction of a new architecture of feedback ANNs which may be motivated from neuro-biology. The basic building block and the general n-neuron architecture with various topologies is also described. Then the basic theory and some characteristics of gradient models are presented as a theoretical foundation for the operation of the new architecture. Circuit implementations of the new architecture are also discussed in Chapter 3. It ends with extensive SPICE simulations and discrete-component experiments to verify the neural properties of a prototype of the new architecture. In Chapter 4, a new learning algorithm for dynamic continuous-time models of ANNs is described.

Chapter 5 describes a VLSI layout designed for a 6-*unit* neural circuit on a MOSIS Tiny-chip as a prototype chip of the new architecture. A dedicated interface circuitry is built to facilitate the programming of the chip. The new learning algorithm described in Chapter 4 is specialized for the 6-*unit* Tiny-chips. The learning algorithm is implemented in software on a PC; it interacts with the chip in real-time via the dedicated interface circuitry. Using the interface circuitry and the learning schemes, we successfully demonstrate the use of the Tiny-chips of the new neural circuits as a recognition and association device.

Chapter 6 describes the design of a 50-neuron CMOS analog chip with on-chip *digital* hardware learning scheme. SPICE simulations are conducted to verify the functionality of the neuron and synaptic weight circuits before the chip was fabricated. Extensive testing of the 50-neuron chip successfully substantiates the predicted operation of the neural circuit.

In Chapter 7, a real-time application is conducted using the 50-neuron chip with on-chip digital learning as a pattern/character associator. An interface circuitry and a software environment is designed and developed which provides a user friendly environment to easily control and operate the fabricated chips. Real-time experiments are conducted to demonstrate the learning capability of the 50-neuron chip for storing a single pattern/character or storing multiple patterns/characters via digital hardware learning circuitry. Finally, summary and conclusions are collected in Chapter 8.

CHAPTER 2

BACKGROUND

Interest in neural modeling and neural computing is not recent. It can be traced to the work of McCullough and Pitts [41] and Hebb [36] in the 1940's. By the early 1960's, active efforts in neural networks and learning were concentrated within relatively few research groups in this country and abroad. The two most active groups here were those of Prof. Frank Rosenblatt [42] at Cornell and Prof. Bernard Widrow [43] at Standford. More recent work by Kohonen [10,37], Grossberg [44-46], Hopfield [1-7], Rumelhart and McCleland [34,47], Arima [82,83], and others has led to new resurgence of the field. This interest is due to the development of new network topologies and algorithms, new analog VLSI implementation techniques, and some intriguing demonstrations as well as by a growing fascination with the function of the human brain.

2.1. Biological Overview of Neural Networks

The brain can perform many tasks that conventional digital computers still cannot. For this reason, it behooves us to examine various models of how a large number of simple devices can work together to perform useful computation.

A neuron is the fundamental unit of all nervous systems [48]. Figure 2.1.1 depicts a highly idealized description of a real neuron [35].

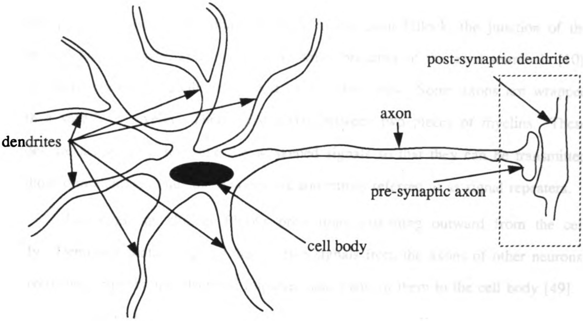


Figure 2.1.1 A highly idealized biological neuron.

Usually, all signals from other neurons come into the dendrite. A cell body integrates or sums all the information that comes in. Action potentials are initiated at the axon hillock when the cell fires. The firing rate depends on the aggregated signals received. The action potentials are a sequence of pulses whose frequency depends on the intensity and duration of signals that excited the cell [10]. Pulses are transmitted to the dendrites of other neurons through synapses. A synapse consists of the presynaptic cell and the postsynaptic cell, which are separated by a synaptic gap. Numerous chemical molecules are involved in transmitting signals through the synapse. If the action potentials arriving at the synapse exceed a certain threshold, the chemical molecules in the presynaptic cell are released and transmitted via the synaptic gap to the postsynaptic cell of another neuron. As the neuro-transmitter molecules combine with receptor cells on the postsynaptic cell membrane, the chemical signals transform into electrical pulses.

Axons are specialized for the conduction of an electrical impulse, called an action potential. Action potentials originate at the axon hillock, the junction of the axon and the cell body, and travel to the small branches of the axon terminals [10]. From there chemical signals are passed on to other cells. Some axons are wrapped with a sheath of myelin. There exist nodes between two pieces of myelins. These nodes are achieved to reinforce the weakened signals so that they can be transmitted without diminishing. Thus these nodes are sometimes referred to as signal repeaters.

Dendrites are thinner fibrous projections extending outward from the cell body. Dendrites contain regions that receive signals from the axons of other neurons, convert these signals into electrical impulses, and transmit them to the cell body [49].

A *cell body* receives signals independently as well. Electrical signals that are generated in the dendrites or cell body spread passively to the axon hillock. If the signal is great enough, an action potential - an electrical pulse - is generated and is actively conducted down the axon.

Synapses are specialized sites where neurons communicate with the cells. The axon terminal of the presynaptic cell contains vesicles filled with a particular neurotransmitter molecules. When the nerve impulse reaches the axon terminal, these vesicles are exocytosed, releasing their contents into the synaptic gap, the narrow space between the cells [50,84]. The transmitter diffuses across the synaptic gap and combines to receptors on the dendrite terminal of the postsynaptic cell. Upon combining, it induces a change in the ionic permeability of the postsynaptic membrane that results in a signal of the electrical potential at the point. With an excitatory synapse, the signal from the presynaptic cell will be more positive. With an inhibitory synapse, a nerve impulse in a presynaptic neuron will be more negative to prevent the generation of an action potential.

Usually, the synapses are connected from the axon terminal of the presynaptic cell to the dendrite terminal of the postsynaptic cell. However, synapses may also appear between presynaptic dendrites and postsynaptic dendrites, or even between presynaptic axons and postsynaptic axons [48]. For examples, the Horizontal and the Ganglion cells in the retina [26], their synaptic connections among neurons only occur via their input terminals only. The synaptic connections among neuron units which is established via dendrites only may be referred to as dendro-dendritic connections.

In a typical ***state of the art*** human brain, there are in the order of 10^{11} to 10^{12} neurons [51].

2.2. Basic Models of Neural Networks

Models of the biological neurons and their networks, if accurate, ought to capture at least some of the properties of biological networks. At least a model should possess an essence of the mechanisms of operations of biological neural nets. Models for neural nets can be classified into the following two categories according to their applications. The first kind of models is for the purpose of studying the functions of the brain. This category may be referred to as *reverse engineering*. These models are of primary interest to biologist, psychologists, physiologists, etc.. The other kind of models is for the purpose of advancing engineering technology. These are often referred to as *Neuro-engineering* or *Artificial Neural Networks* (ANNs). These models are inspired by the neurobiology and aim at improving the technological processing of data or information. We will primarily focus on the second category.

2.2.1. The Model of a Neuron

In a single neuron, a train of action potentials is propagating pulses of electro-chemical activity. If the neuron has a strong input, these pulses are generated at a high rate. If the input is weak or absent, they are generated at very low rate. The mean rate at which these action potentials are generated results in a smooth nonlinear function of the mean membrane potential, say $S_i(u_i)$ which is typically modeled as a monotone nondecreasing sigmoid function [1-5] or approximation thereof. Since the synapses are activated by arriving action potentials, $v_i = S_i(u_i)$ becomes an input and output relationship for a neuron, where u_i could be thought of as the mean soma potential of a neuron from the total effect of its excitatory and inhibitory inputs, and v_i could be viewed as the short-term average of the firing rate of the cell i. We use model neurons which lend themselves naturally to VLSI implementation that communicate by means of voltage levels. Think of a high voltage as representing a high level of activity, a low voltage as representing a low level of activity. The neurons (or the processing elements) are modeled as amplifiers having a monotone nondecreasing sigmoid input-output relation, as shown in Figure 2.2.1.1.

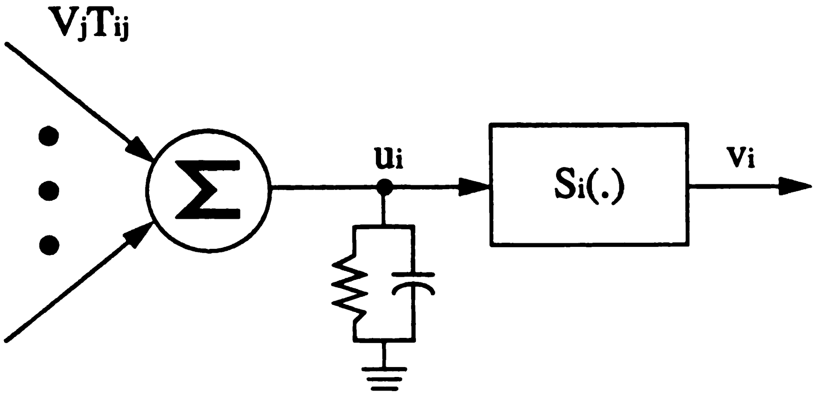


Figure 2.2.1.1 Neural model circuits.

2.2.2. Models of the Synapse

The strength of the synapse from a presynaptic neuron j to a postsynaptic neuron i can be modelled as a linear parameter W_{ij} so that the postsynaptic signal is given by $W_{ij}V_j$ or $W_{ij}S_j(u_j)$. If the synapse is excitatory, the output from neuron j will drive neuron i to produce more output, i.e. W_{ij} is positive. If it is inhibitory, less output will be produced and W_{ij} will be negative. The linear synaptic weights can be realized in VLSI implementation of an ANN via linear resistors [18, 21] or analog multipliers [17,20,52-57].

The strength of the synapse in the biological neuron is a highly nonlinear function in order to describe the chemical substances transmitting signals through the synaptic gap. The dynamic neural net models supporting with good theoretical analysis, the synapse may also be modeled as a nonlinear function [39,40] so that the postsynaptic signal is given by $f_{ij}(S_j(u_j))$ where f_{ij} is a nonlinear function describing the strength of the synapse.

2.2.3. Feedback Models

Figure 2.2.3.1 depicts a usual feedback neural net model, where the outputs of a set of neurons return to become inputs with an interconnection weight matrix T_{ij}. This kind of neural net model has rich dynamics, but is hard to program.

In a biological system, u_i will lag behind the instantaneous outputs V_j of the other cells because of the input capacitance C_i of the cell membrane, the transmembrane resistance R_i, and the finite conductance T_{ij}. Thus the rate of change of u_i in feedback ANNs is determined by the following resistance-capacitance charging equation.

$$C_i \frac{du_i}{dt} = \sum_j T_{ij} V_j - \frac{u_i}{R_i} + I_i \qquad (2.2.3.1)$$

$$u_i = S_i^{-1}\left[V_i\right],$$

where $R_i^{-1} = \rho_i^{-1} + \sum\limits_j T_{ij}$,

which is the net input impedance, C_i is the total input capacitance of the neuron i, I_i is an external stimulation or excitation at the input of the neuron i, and $T_{ij}V_j$ is the postsynaptic current from neuron j to neuron i.

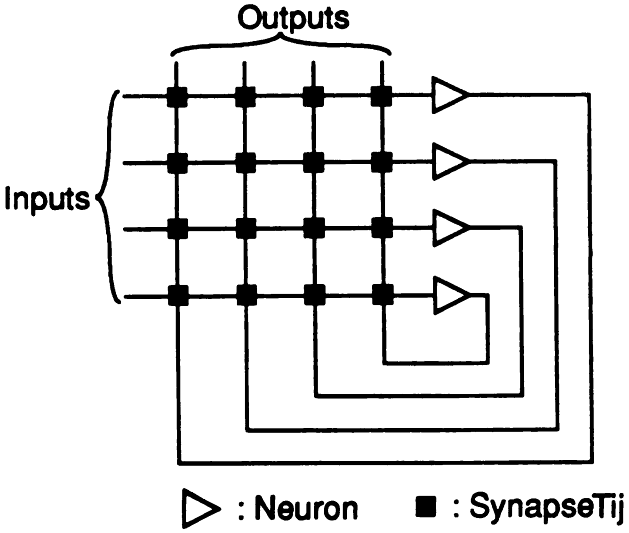


Figure 2.2.3.1 A typical feedback artificial neural network model.

We can now construct an electrical network model [1-7]. A cell body is implemented via an (operational) amplifier whose input-output relation is a sigmoid function. Axons and dendrites are each replaced by transmission wires. Synapses can be substituted by conductance devices.

The dynamic behavior of the feedback neural net model can be examined by considering the energy function of equation (2.2.3.1) [1-7,44,58]. There is symmetry requirement on the connections, $T_{ij} = T_{ji}$, for the existence of an energy function. The so-called potential energy function, which is a first integral of (2.2.3.1), can be derived as

$$E = -\frac{1}{2}\sum_i\sum_j T_{ij}V_iV_j + \sum_i R_i^{-1}\int_0^{V_i} S_i^{-1}(V)\,dV - \sum_i I_iV_i. \qquad (2.2.3.2)$$

Hence the dynamic equations of the neural network model can be rewritten as

$$C_i\frac{du_i}{dt} = -\frac{\partial E}{\partial V_i}. \qquad (2.2.3.3)$$

The time derivative of the energy function along trajectories is

$$\begin{aligned}\frac{dE}{dt} &= \sum_i \frac{\partial E}{\partial u_i}\frac{du_i}{dt} \qquad (2.2.3.4)\\ &= \sum_i \frac{\partial E}{\partial V_i}\frac{dV_i}{du_i}\frac{du_i}{dt}\\ &= -\sum_i \frac{1}{C_i}\frac{dV_i}{du_i}\left[\frac{\partial E}{\partial V_i}\right]^2\\ &= -\sum_i \frac{1}{C_i}\frac{dS_i(u_i)}{du_i}\left[\frac{\partial E}{\partial V_i}\right]^2\\ &\le 0,\end{aligned}$$

since $C_i \ge 0$ and $S_i(u_i)$ is a monotone nondecreasing function, thus $\frac{dS_i(u_i)}{du_i} \ge 0$. Therefore, this system is a gradient-like system. That is, this energy function decreases along trajectories and its time-derivative equals zero at $\frac{\partial E}{\partial V_i} = 0$, which is an equilibrium point of this system.

Some computational problems can be transformed into a "more-or-less" equivalent optimization problem by a so-called regularization procedure [59,60]. The energy function of feedback neural net model provides the link between this optimization problem and its solution in terms of neural network, since in the network the potential is automatically minimized (the characteristics of the gradient system) [1-7]. The computational dynamics of typical neural networks are characterized by the existence of several stable states. These steady states correspond to memories or possible answers of problems.

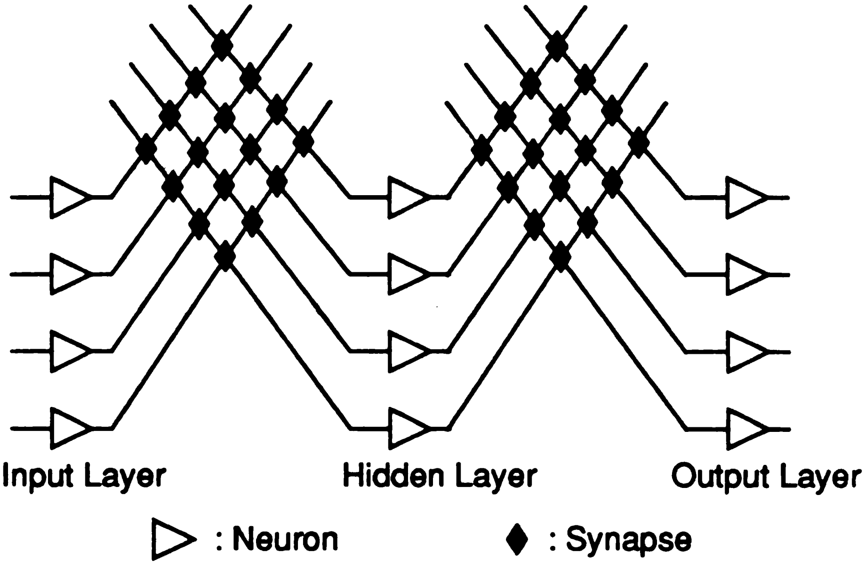


Figure 2.2.4.1 A typical two-layer feedforward artificial neural network model.

2.2.4. Feedforward Models

Figure 2.2.4.1 depicts a feedforward neural net model, where outputs of any layer are weighted and summed as an input to a neuron in the next layer. An external input is applied to the first layer, which is called the input layer and fed forward to the last layer, which is called the output layer. Any layer between the input layer and the output layer is called a hidden layer. The governing static equation for each neuron unit in any layer may be represented as

$$y_i = S_i(\sum_j w_{ij} x_j + \theta_i), \qquad (2.2.4.1)$$

where y_i is the output of the i-th neuron, $S_i(.)$ is a nonlinear monotone nondecreasing sigmoid function, w_{ij} is the connection weight from the j-th neuron output of the previous layer to the i-the neuron input, x_j is the j-th output of a neuron unit in the previous layer, and θ_i is the threshold bias at the input of the i-th neuron.

The feedforward neural nets are quite powerful from the standpoint of being able to program such a network through a learning scheme such as BEP [34] to do a useful task. BEP is a continuous valued mapping from input to output, where the mapping function is adapted by repeatedly presenting the network with a set of input-output vector pairs and using an appropriate feedback function (typically gradient descent) to update the mapping until a functional relationship is realized between the inputs and outputs.

The total squared error function is given by

$$E = \sum_p E_p \tag{2.2.4.2}$$

$$= \frac{1}{2} \sum_p \sum_i (t_{pi} - ty_{pi})^2,$$

where E_p is the squared error function for the desired target p, say $t_p^T = [t_{p1} \cdots t_{pn}]$ and y_{pi} is the actual output for the target p.

The BEP rule is governed by the following equations:

$$w_{ij}^k = w_{ij}^{k-1} + \sum_p \Delta_p w_{ij}^k \tag{2.2.4.3}$$

$$\Delta_p w_{ij} = -\eta \frac{\partial E_p}{\partial w_{ij}} \tag{2.2.4.4}$$

$$= \eta \delta_{pi} y_{pj},$$

where $\Delta_p w_{ij}$ is the change of the weight w_{ij} at the k-th iteration for the desired target p, η is the learning rate which is sufficiently small and positive, and y_{pj} is the output of the previous layer. If a unit j is in the output layer, δ_{pi} is given by

$$\delta_{pi} = \frac{dS_i}{du_i}[t_{pi} - y_{pi}]. \tag{2.2.4.5}$$

If a unit j is in the hidden layer, δ_{pi} is given by

$$\delta_{pi} = \frac{dS_i}{du_i}\sum_l \delta_{pl} w_{li}, \qquad (2.2.4.6)$$

where l is the index for the units in the next layer to which a unit i is connected.

A modified BEP learning rule was proposed [33] in order to realize on-chip learning circuits using standard CMOS technology. Using this modified BEP learning rule, sigmoid-derivative circuits are not necessarily required for the implementation of feedforward ANNs with learning.

Both feedback and feedforward models are interesting from the point of view of artificial neural computation. Technologically, they emphasize the same things, a set of sigmoid I/O neurons and a far larger set of interconnects that describe the relationships between the output of some neurons and input of others.

2.2.5. General Models

Besides the feedback and feedforward models, there are other ANN models such as the combination of the feedback and feedforward models [38], silicon retina [26,61] and cochlea model [26], self-organizing adaptation model [10]. The focus of the silicon retina is trying to mimic the biological functions of primate retina. Another mathematical model of ANNs is introduced by Hoppensteadt [62] using voltage controlled oscillators (VCOs).

2.3. Overview of the Present Literature

The interest in feedback neural networks has been revived by the recent publication [5] by Hopfield and Tank. In this publication, the authors were able to demonstrate the applicability of neural networks ideas to the design of A/D converters, signal decision circuits, and linear programming circuits.

Hopfield neural network is used for VLSI implementation due to the simple architecture and well-defined network behaviors. A major component of VLSI implementation of artificial neural networks is the array of connections between neurons, the synapses. The transfer function of the neuron itself is a monotonic sigmoid function which can be easily implemented in VLSI by an amplifier or two logical inverters in series.

Neural network hardware based on the silicon VLSI technology is actively being pursued by several research groups. Hubbard *et al.* [21] demonstrated a thin film synaptic array in submicron feature size fabricated by e-beam lithography. Selectively deposited amorphous silicon resistive elements at the nodes provide the resistive array synaptic connections which will be useful as an associative read-only memory. Sivilotti *et al.* [22] at California Institute of Technology had fabricated a programmable neural network chip with twenty-two neurons and +1, 0, and -1 synapses. This circuit is able to perform an on-chip learning by employing the truncated Hebb's rule.

Since digital circuits have proven themselves in VLSI, there is one chip [25] developed by T. K. Miller III *et al.*, which was implemented with fully digital circuits operating in a stochastic manner.

Researchers at AT&T Bell Laboratories [23] fabricated a 54-neuron CMOS chip with programmable +1, 0, and -1 synapses. Such an implementation uses static RAM cells as prespecified memories. The stored data are presented in the interconnect matrix and only one output is evaluated by the largest inner product value between the input vector and the stored vectors. Recently, Bell labs [16,85] fabricated a 32768 programmable binary connections neural-net chip which can be integrated with a digital signal processor and fast memory for image processing. They also fabricated a programmable neural network chip with over 130,000 connections and a $4.5 \times 7mm^2$ chip size using a $0.9\mu m$ CMOS technology [86]. A character recognition application was performed using the chip in conjunction with a digital signal processor.

Sage *et al.* [24] at MIT Lincoln Laboratory developed chips based on Metal-Nitride-Oxide-Semiconductor (MNOS) and Charge-Coupled Devices (CCDs) technology. Such an implementation would achieve analog synaptic weights via variable charge storage. Agranat *et al.* [15,87,88] at Caltech fabricated a neural network integrated circuit with 65536 analog programmable synapses and 256 fully interconnected neurons using CCD techniques.

Researchers at Jet Propulsion Laboratory [17] fabricated a 32x32 synapse chip using analog multiplier circuits and a 32-channel variable-gain neuron chip. Researchers at Naval Research Laboratory [55,90] fabricated both 32x32 and 128x64 programmable analog vector-matrix multiplier chips to implement multilevel artificial neural networks. A VLSI chip from Intel Corporation [20] has 64 neurons and 64x64 programmable synaptic weights which was implemented with fully analog circuitry using analog multiplier and floating gate device techniques.

Researchers at Lockheed Research Laboratory [18] built an analog neural network breadboard consisting of 256 neurons and 2048 programmable synaptic weights with 5-bit resolutions. Muller *et al.* [91,92] at the University of Pennsylvania developed a general purpose analog neural computer which is composed of interconnected modules containing arrays of neurons, modifiable synapses and switches.

In a different approach, Mead *et al.* [26,61] at Caltech integrated sensor arrays and processing elements to emulate some of the spatial and temporal properties of neural networks in the eye. Recently, they developed a working analog VLSI chip [19] that implements a model of early auditory processing in the brain.

As of the present time, there are no reported application results for most of these chips, except the reported AT&T results [85,86]. More importantly, there is no on-chip hardware learning scheme for any of these chips.

There are only a few silicon ANN chips with on-chip learning. There is a single neuron chip developed by Ricoh Corporation research group [89] with on-chip learning and a $8.39\times8.03mm^2$ chip size which was also implemented with fully digital circuits using a $1.5\mu m$ CMOS technology and operating in a stochastic manner. Schneider *et al.* [93] at University of Manitoba fabricated a 25-neuron chip with 600 on-chip *in situ* learning Hebbian synapses and a $5.5\times4.6mm^2$ chip size using a $3\mu m$ CMOS technology. However, there are no reported testing results for the functionality of these two chips and there are no reported application results.

Arima *et al.* [82,83] at Mitsubishi Electric Corporation, developed two on-chip self-learning neural network chips. One has 125 neuron units and 10K synapse units with a $13\times13mm^2$ chip size using a $1\mu m$ CMOS technology and the other has 336 neurons and 28K synapses with a $14.5\times14.5mm^2$ chip size. There are only two examples of association in the reported application results and both two examples can not successfully retrieve the complete desired patterns respectively after the network learned 10 desired patterns. We have reported our testing experiments of the 50-neuron chip with on-chip learning in [13], in June 1991. The 50-neuron chip can store a desired pattern and successfully retrieve it via all initial conditions with Hamming distances less than 9 from the desired pattern.

2.4. CMOS VLSI Circuits

It has been emphasized that a large number of neurons are needed in an ANN to produce emergent useful computations in the neural network sense. Since then, many workers have sought to implement the feedback (and the feedforward) ANNs using VLSI/LSI silicon electronics.

Over the past few years, Complementary Metal Oxide Silicon (CMOS) technology has played an increasingly important role in the world integrated circuit industry. An MOS structure is created by superimposing several layers of conducting, insulating, and transistor forming materials. After a series of processing steps, a typical structure might consist of levels called diffusion, polysilicon, and metal that are separated by insulating layers. CMOS technology provides two types of transistors, an n-type transistor (nMOS) and a p-type transistor (PMOS). Typical physical structures for the two types of MOS transistors are shown in Figure 2.4.1. For the nMOS transistor, the structure consists of a section of p-type silicon separating two diffused areas of n-type silicon. The area separating the n region is capped with a sandwich consisting of an insulator and a conducting electrode called the *gate*. The transistors have two n-type diffused areas, which are designated the *drain* and the *source*.

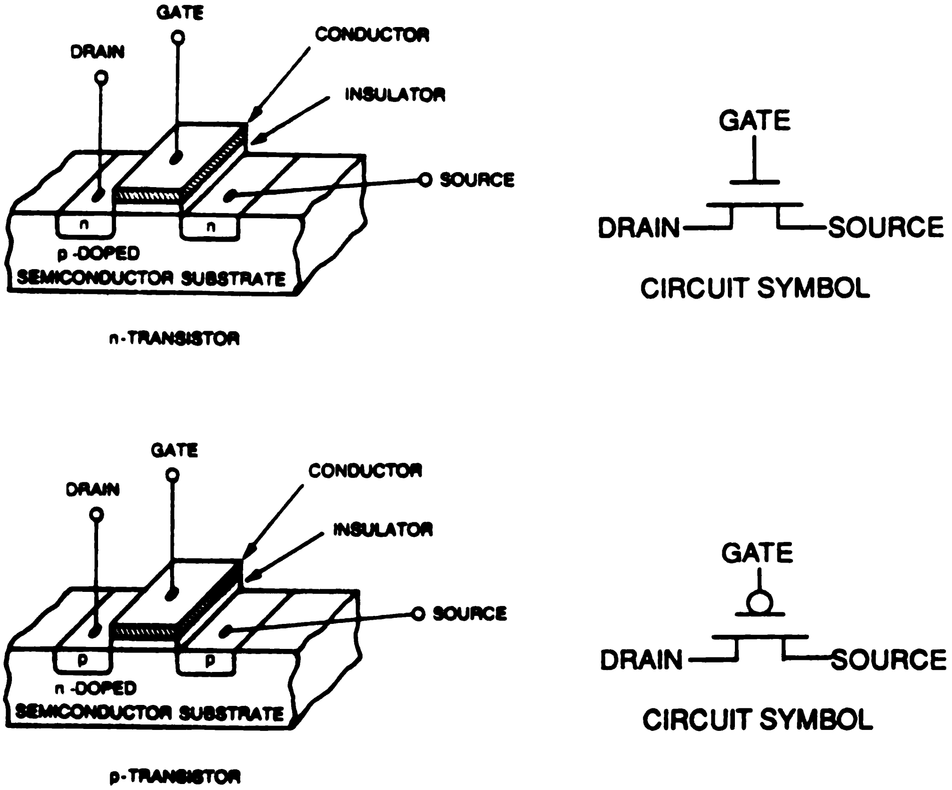


Figure 2.4.1 MOS transistor physical structures.

2.4.1. MOS Transistors

Recall that a MOS transistor current-voltage characteristic function, say I_{ds}, denotes the current flowing from the drain to the source. Denote the source, gate, drain, and threshold voltages respectively as v_s, v_g, v_d, and v_t. Then, according to the square-law theory, the current characteristic function I_{ds} is given as

Cutoff: if $(v_g - v_s - v_t) \leq 0$

$$I_{ds}(v_d, v_s, v_g) = 0 \qquad (2.4.1.a)$$

Triode: if $(v_g - v_s - v_t) \geq (v_d - v_s)$

$$I_{ds}(v_d, v_s, v_g) = \frac{u}{2} C_{ox}(W/L)\left[2(v_g - v_s - v_t)(v_d - v_s) - (v_d - v_s)^2\right] \qquad (2.4.1.b)$$

Saturation: if $(v_g - v_s - v_t) \leq (v_d - v_s)$

$$I_{ds}(v_d, v_s, v_g) = \frac{u}{2} C_{ox}(W/L)\left[v_g - v_s - v_t\right]^2, \qquad (2.4.1.c)$$

where W/L is the ratio of gate width to length, C_{ox} is the oxide capacitance per unit area, and u is the mobility of carriers.

2.4.2. Design Considerations

In order to design CMOS VLSI layout, *Magic* design tools has been employed. *Magic* is the backbone of the Berkeley integrated circuit Computer Aided Design (CAD) software system. With *Magic*, designers can paint geometry using a mouse and a graphic display system. The layers painted are not the actual mask layers used in fabrication. The actual CIF layers are generated by *Magic* from the abstract layers.

Both the number of pins and the chip area of the neural network circuitry increase as the number of neurons increases. If we want to put more neurons on a single chip with the same chip area, then there is a trade-off between the pin count and the circuit area. The more pins we put on the same chip, the more area is used to locate these pins and correspondingly we have less area to locate the network circuitry.

The minimal acceptable pad geometry by MOS Implementation System (MOSIS) [63] is an 88 × 88 micron glass cut box over a 100 × 100 micron metal box. The pad geometry is not scalable while the advanced technology is used. In addition, bonding pads should be placed along the edges of the project, with at least 200 micron center-to-center spacing. Thus, the network circuit area dramatically decreases as the number of pins is increased.

The VLSI design project can be packaged in a 28, 40 or 64 pin DIP (dual Inline Package) or an 84, 108, or 132 PGA (Pin Grid Array) package. Table 2.4.2.1 tabulates the relation among pin count, package type, and cavity size. The MOSIS Service offers Standard Frames that specify bonding pad locations and their minimum sizes. Table 2.4.2.2 shows the available Standard Frames for different project sizes.

Table 2.4.2.1 MOSIS package types.

MOSIS PACKAGE TYPES		
PIN COUNT	PKG TYPE	CAVITY SIZE
28	0.6" DIP	.310 x .310"
40	0.6" DIP	.310 x .310"
64	0.9" DIP	.400 x .400"
84	1.1" PGA	.350 x .350" .470 x .470"
108	1.2" PGA	.350 x .350" .450 x .450"
132	1.4" PGA	.350 x .350" .450 x .450"

Table 2.4.2.2 MOSIS Standard Frames.

Project Size	Frame Name (by package pin count)					
sq. mm	28	40	64	84	108	132
7.9x9.2	- - -	- - -	64P79x92	84P79x92	- - -	- - -
6.9x6.8	- - -	40P69x68	64P69x68	84P69x68	- - -	- - -
4.6x6.8	- - -	40P46x68	- - -	- - -	- - -	- - -
4.6x3.4	28P46x34	40P46x34	- - -	- - -	- - -	- - -
2.3x3.4	28PC23x34	- - -	- - -	- - -	- - -	- - -
2.22x2.25	- - -	40PC22x22	- - -	- - -	- - -	- - -

CHAPTER 3

A NEW NEURAL NETWORK ARCHITECTURE

A new architecture of feedback artificial neural nets was proposed in [39,40] which has been shown to exhibit qualitatively the same dynamic properties as gradient continuous-time feedback neural nets. This model (i) has a maximum number of connections equal to $n(n+1)/2$, where n is the number of neurons, i.e., we reduce the routing overhead for VLSI design. (ii) The synaptic weights are naturally "symmetric" since there is a single element connecting *unit* i to *unit* j. Then, it becomes possible to view gradient models as valuable and implementable in hardware. In addition to these two features, this model utilizes simple nMOS transistors as its synaptic connection, where its conductance is controlled via the gate voltage. It doesn't require the realization of linear resistive elements for the synaptic weights. This new architecture lends itself naturally to analog all-MOS VLSI implementation.

3.1. Motivation from Biology

This new neural network architecture is motivated from biological neural nets where neurons have dendro-dendritic connections, i.e., connections among neurons which occur via dendrites only. Consider neural units which communicate via their dendrites, i.e. the synaptic connection between neuron units is established via the dendrites. Such synaptic connections are referred to as dendro-dendritic connections.

Indeed, the Horizontal and the Ganglion cells in the retina are examples of networks with dendro-dendritic connections [26].

3.2. One Neuron with Self-feedback

Each *neuron* is a processing device with input u_i and output v_i related by the usual sigmoid function S_i. It is modeled as an operational amplifier (op-amp) with a capacitive element C_i and a resistive element R_i at the input node. The neuron may be realized via two CMOS inverters in series instead of an op-amp. For VLSI implementation, the capacitive element C_i and the resistive element R_i at its input node are eliminated, since the parasitics compensate for their roles.

There is a self-feedback MOSFET with gate voltage Vfb_i between input u_i and output v_i. We shall refer to the neuron with self-feedback as a neuron unit. It represents the building block for the new neural network architecture. The CMOS circuit model of this neuron unit is depicted in Figure 3.2.1. This neuron unit is capable of processing two stable equilibria. It is, in fact, a model for a flip-flop.

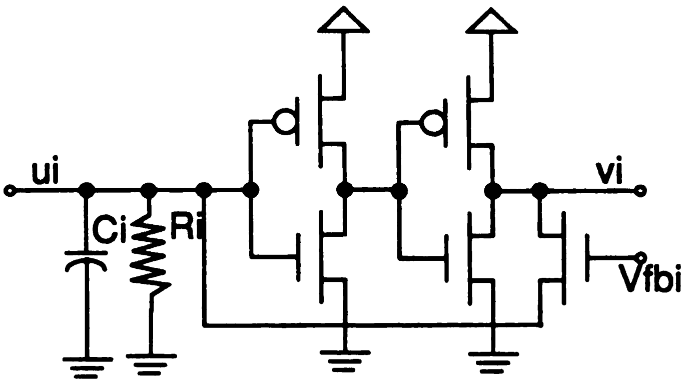


Figure 3.2.1 One neuron with self-feedback.

3.3. General n-Neuron Architecture

The input of *unit i* is connected to the input of *unit j* via a single MOSFET with gate voltage VG_{ij}. External input I_i may be injected at node i. The architecture simply connects neuron *unit* via their input nodes. The CMOS circuit architecture, for a 3-*unit* prototype circuit, is depicted in Figure 3.3.1. One of the important advantages of this architecture is that between any two neuron *units* there is a single physical connection. It preserves the *symmetry* in the connection required in the theory of gradient dynamical systems. Moreover, the architecture reduces the maximum number of connections to $n\ (n+1)/2$, where n is the number of neuron processors in the network and it does not require the realization of linear resistive elements for the synaptic weights.

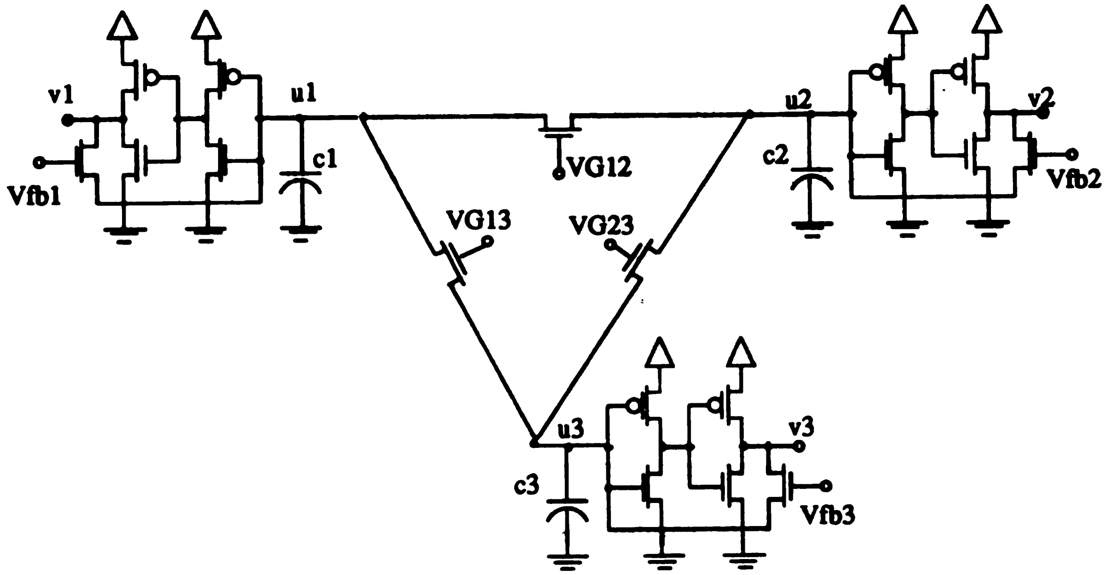


Figure 3.3.1 A 3-*unit* prototype circuit of the new architecture.

We can also connect the output of *unit i* to the output of *unit j* via a single MOSFET with gate voltage VG_{ij} [64]. This output-connected topology has the qualitatively same dynamic properties as the input-connected topology.

In planar VLSI implementation, it is difficult to make a global interconnection among all *units* in a network, especially when the number of *units* is very large. Generally these networks are implemented in an array structure by connections to nearest neighbors. Retinal structures have a high degree of symmetry and connectivity, and thus they are attractive for VLSI implementation. This topology has been used to design the silicon retina in [26]. In order to form a retinal structure network, our *units* are tiled in a hexagonal array with six connections converging into the input node of each *unit* [64].

3.4. Mathematical Analysis

From Kirchhoff's Current Law (KCL), applied at every input node, one obtains the mathematical model for the new architecture as follows:

$$C_i \frac{du_i}{dt} = \sum_{j}^{n} I_{ds}(u_j, u_i, VG_{ij}) + I_{ds}(v_i, u_i, Vfb_i) - \frac{u_i}{R_i} + I_i \qquad (3.4.1a)$$

$$v_i = S_i(u_i), \qquad (3.4.1b)$$

where S_i is a sigmoid function which represent the input and output relation of the double inverter, I_{ds} is the MOSFET current-voltage characteristic function which is described in equation (2.4.1), C_i and R_i are the parasitic capacitor and resistor, and I_i is an external bias.

With appropriate assumptions and approximations [40], we may consider the MOSFET characteristic function I_{ds} to be an odd function of $(v_d - v_s)$. Moreover, we note that in this architecture, the gate voltages VG_{ij} and VG_{ji} are identically the same, i.e. $VG_{ij} = VG_{ji}$.

Hence the dynamic equations of the new neural circuit model can be rewritten as

$$C_i \frac{du_i}{dt} = \sum_j^n I_{ds}(u_j - u_i, VG_{ij}) + I_{ds}(v_i - u_i, Vfb_i) - \frac{u_i}{R_i} + I_i \tag{3.4.2a}$$

$$v_i = S_i(u_i). \tag{3.4.2b}$$

Define $u_{ij} = u_i - u_j$ and $u_{ji} = u_j - u_i$. Because $VG_{ij} = VG_{ji}$, $I_{ds}(u_{ij}, VG_{ij}) = -I_{ds}(u_{ji}, VG_{ji})$. One can obtain a first integral or energy function [40] of equation (3.4.2) as

$$E = -\frac{1}{2}\sum_i \sum_{j \neq i} \int_0^{u_{ji}} I_{ds}(\, x_i,\, VG_{ij}\,)\, dx_i - \sum_i \int_0^{u_i} I_{ds}(\, S_i(y_i) - y_i,\, Vfb_i\,)\, dy_i \tag{3.4.3}$$

$$+ \frac{1}{2}\sum_i \frac{u_i^2}{R_i} - \sum_i I_i u_i .$$

Hence the dynamic system (3.4.2) can be rewritten as

$$C_i \frac{du_i}{dt} = -\frac{\partial E}{\partial u_i}. \tag{3.4.4}$$

The time derivative of the energy function along trajectories is

$$\frac{dE}{dt} = \sum_i \frac{\partial E}{\partial u_i} \frac{du_i}{dt} \tag{3.4.5}$$

$$= -\sum_i \frac{1}{C_i} \left[\frac{\partial E}{\partial V_i} \right]^2$$

$$\leq 0,$$

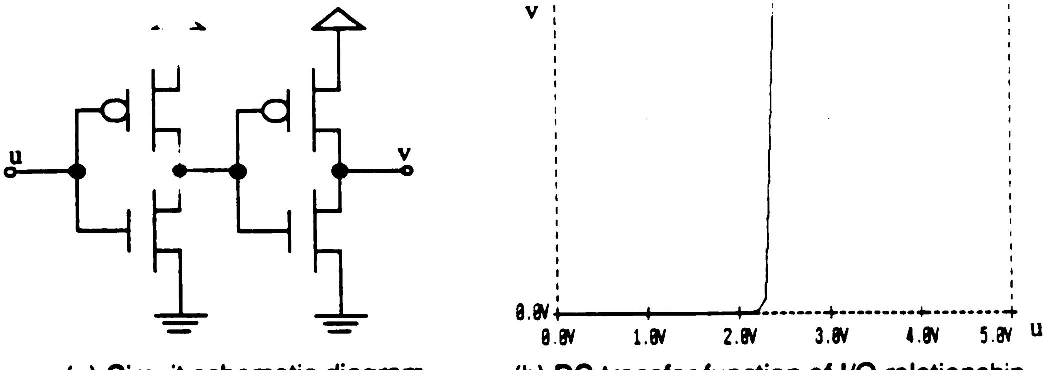
Therefore, this system is a gradient-like system endowed with the properties attributed to gradient dynamic systems [40, 65]. A theorem [58,65-67] confirms that all solutions converge to equilibria only. No oscillations or any other complicated behavior is permitted in this system as consequence of being a gradient system. Thus the overall dynamic system is structurally stable.

3.5. Circuit Implementation of the New Architecture

If artificial neural nets are to be used in large scale, then the medium of their implementation becomes critical in enhancing the quality of their performance. Indeed, implementation in large scale is the vehicle through which artificial neural networks would reveal their computational powers. Hardware implementation is natural, convenient, and more powerful where the mechanism of the proposed neural networks would be directly realized, and suitably accommodated, into VLSI hardware.

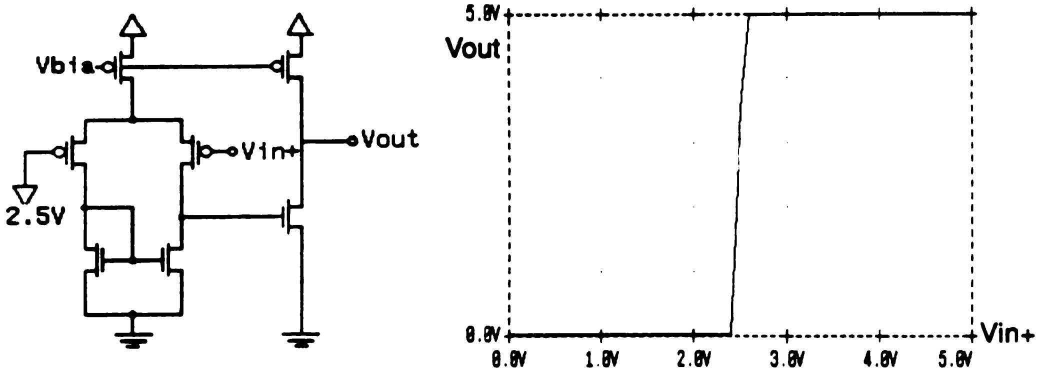
3.5.1. A Neuron

The transfer function of a neuron itself is a monotonic nondecreasing sigmoid function which can be easily implemented in CMOS VLSI via an op-amp or two logical inverters in series. The circuit schematic diagram and DC transfer function of a neuron which is implemented via two CMOS logical inverters in series are depicted in Figure 3.5.1.1. An advantage of the implementation using two CMOS logical inverters in series is its simplicity because a neuron consists of only four MOS transistors. The circuit schematic diagram and DC transfer function of a neuron which is implemented via a CMOS operational amplifier are depicted in Figure 3.5.1.2. An advantage for an op-amp is that the threshold value of the sigmoid function can be adjusted after fabrication.



(a) Circuit schematic diagram (b) DC transfer function of I/O relationship

Figure 3.5.1.1 A implementation of one neuron via two CMOS inverters in series.



(a) Circuit schematic diagram (b) DC transfer function of I/O relationship

Figure 3.5.1.2 A implementation of one neuron via an amplifier.

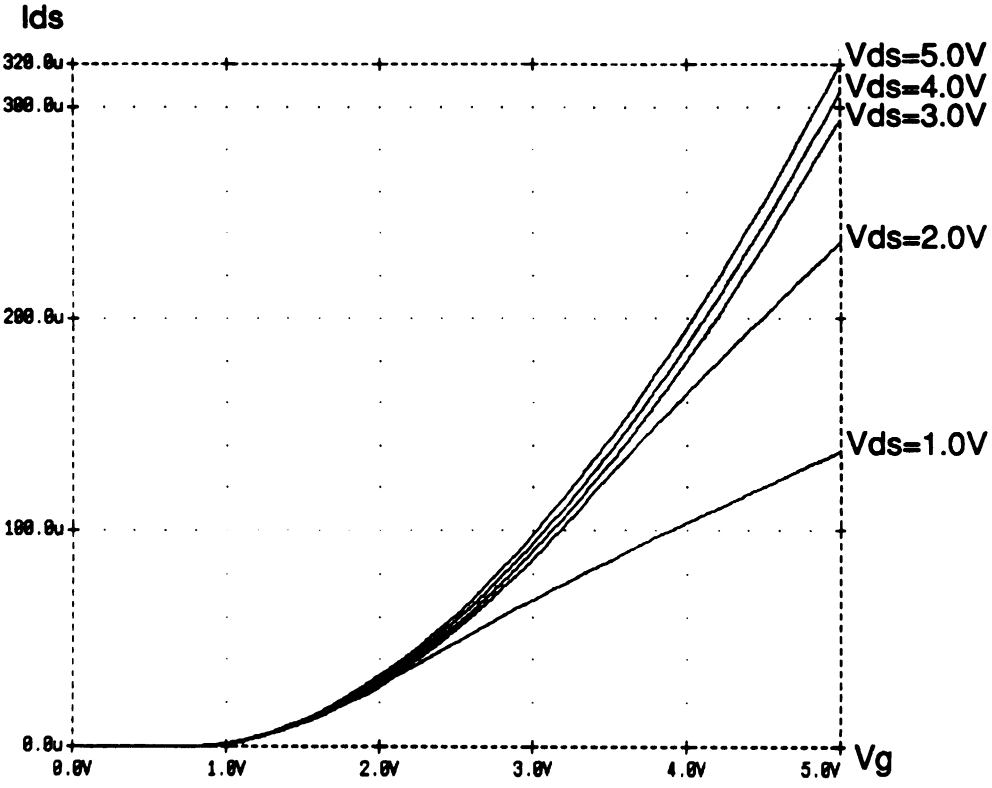


Figure 3.5.2.1 Transfer characteristics of a nMOSFET.

3.5.2. A Programmable Synapse

It is necessary that synaptic weights can be programmed so as to render a relative configuration of the neural network which corresponds to various set of stored memories. Linear resistive elements in electronic neural network can be realized via analog multipliers which have been introduced [17,20,52-57] to implement the programmable synaptic weights using a standard CMOS technology. The new circuit architecture does not require the realization of linear resistive elements for the synaptic weights. Thus a programmable synapse can be easily realized via a single nMOS transistor [64,68]. The conductance of an nMOS transistor is capable of being adjusted via controlling the gate voltage of the nMOS transistor. For a fixed drain to source voltage, the characteristic of nMOS conductance is depicted in Figure 3.5.2.1. Although the programmable synapse of the new architecture is modeled as a nonlinear function because the conductance of nMOS transistor is a nonlinear function of the gate voltage, the dynamic behavior of the new architecture is supported with good theoretical analysis.

3.5.3. Storage of Synaptic Weights

The configuration of a neural network is characterized by the synaptic weights. Therefore, it is necessary to store synaptic weight values in order to preserve the desired configurations. There are several approaches to implement the storage of synaptic weight values, such as on-chip digital memory [23], floating-gate device [69-73], CCDs [24,74], capacitors [75], and external digital memory with A/D and D/A converters [17,76].

On-chip digital memory restricts its programmability, but it is useful when the network has a suitable digital learning scheme and is used in a static mode such as associative memory and pattern classifier applications. The floating-gate devices are attracting considerable attention as a nonvolatile analog memory. However, its control circuitry and scheme are still complicated. Furthermore, the accuracy and repeatability is difficult to control. To maintain the analog weight, a twin capacitor storage [74] cell had been proposed utilizing the CCD technique but its control scheme is still complicated. External digital memory with A/D and D/A converters combines two popular methods: Storing weights as digital words that are converted to analog values, and storing each weight as a charge on an on-chip capacitor. Rather than implementing a D/A converter for each weight, only one off-chip, high-precision converter is time multiplexed to serially refresh all the capacitor charges on one chip. A disadvantage of this approach, that the analog weight values are quantized by digital word length and precision of the digital-to-analog converter, is more than compensated by the ease with which the weights can be manipulated by the digital host computer.

In this work, on-chip digital memory or floating gate devices were used as the storage of synaptic weights.

3.6. SPICE Simulations and Laboratory Experiments of the New Architecture

A 3-*unit* prototype circuit in Figure 3.2.1 has been employed to illustrate and to test the performance of the proposed circuit. SPICE was used to simulate the transient evolution to the steady states (or stable equilibria). SPICE parameters for 2-micron technology are provided by MOSIS.

The voltages (across the three capacitors) at the input nodes of the 3 *unit*s are simultaneously initialized. Then the circuit evolves through its transients before settling to the steady state values. The simulation has exhaustively been repeated using various initial conditions and verified the convergence of the resulting solutions to equilibria.

Once the interconnect configuration is set, i.e., the gate voltages of interconnect MOSFETs and the gate voltages of the self-feedback MOSFETs are set, the network possesses a specific set of stable equilibrium states as its only limit set. As one alters the interconnect configuration, the network will alter the set of equilibria accordingly. In fact, one can adjust the interconnect configuration to obtain any different number of stable equilibrium states which in turn correspond to different number of memory data.

By only adjusting the gate voltages VG_{ij} of interconnect MOSFETs, the 3-*unit* prototype circuit can exhibit all possible number of equilibrium states; for a fixed interconnect configuration, the number of possible equilibria ranged from two distinct stable equilibria to eight distinct stable equilibria. Simulation results are depicted in Table 3.6.1.

When the gate voltages Vfb_i of the MOSFETs across the double inverters are adjusted, the circuit can then posses a single stable equilibrium state.

In order to verify the SPICE simulation, a discrete-component realization of the 3-*unit* prototype circuit has been built and its performance tested in the laboratory.

Every CMOS inverter is realized by one inverter of the 74C04N which is a single stage and unbuffered CMOS inverter. Every n-channel MOSFET conductance element is realized via a 2N4351. Table 3.6.2 depicts the experimental results of a 3-*unit* prototype circuit.

In the discrete-component realization, the component variations such as threshold voltages and transistor size, will make the experimental results different from the SPICE simulations. However, the experimental results are qualitatively identical to the SPICE simulations.

Table 3.6.1 SPICE simulation results of a 3-*uint* prototype circuit.

$Vfb_i = 5.0\text{V}, i = 1, \cdots, 3$		
Double Inverter	nMOS pMOS	W/L = 4.0/2.0 W/L = 12.0/2.0
Interconnect Transistor	nMOS	W/L = 4.0/2.0
Feedback Transistor	nMOS	W/L = 8.0/2.0

$(VG_{12}, VG_{23}, VG_{31})$	# of Stable Equilibria	Stable Equilibria (v_1, v_2, v_3)
(5.0, 5.0, 5.0)	2	(0, 0, 0), (1, 1, 1)
(2.6, 2.7, 5.0)	3	(0, 0, 0), (1, 1, 1), (1, 0, 1)
(0.5, 0.5, 5.0)	4	(0, 0, 0), (1, 1, 1), (1, 0, 1) (0, 1, 0)
(2.8, 2.8, 2.8)	5	(0, 0, 0), (1, 1, 1), (1, 0, 1) (1, 1, 0), (0, 1, 1)
(2.7, 3.3, 2.2)	6	(0, 0, 0), (1, 1, 1), (1, 0, 1) (1, 1, 0), (0, 1, 1), (1, 0, 0)
(2.6, 2.6, 0.0)	7	(0, 0, 0), (1, 1, 1), (1, 0, 1) (1, 1, 0), (0, 1, 1), (0, 0, 1) (1, 0, 0)
(0.0, 0.0, 0.0)	8	(0, 0, 0), (1, 1, 1), (1, 0, 1) (1, 1, 0), (0, 1, 1), (0, 0, 1) (1, 0, 0), (0, 1, 0)

Table 3.6.2 Experimental results of a 3-*uint* prototype circuit.

Vfb_i = 5.0V, i = 1, $\cdots$, 3		
Double Inverter	the MM74C04N	
Interconnect Transistor	nMOS	the 2N4351
Feedback Transistor	nMOS	the 2N4351
(VG_{12}, VG_{23}, VG_{31})	# of Stable Equilibria	Stable Equilibria (v_1, v_2, v_3)
(5.0, 5.0, 5.0)	2	(0, 0, 0), (1, 1, 1)
(1.4, 5.0, 1.75)	3	(0, 0, 0), (1, 1, 1), (1, 0, 1)
(0.0, 0.0, 5.0)	4	(0, 0, 0), (1, 1, 1), (1, 0, 1) (0, 1, 0)
(1.35, 1.9, 1.75)	5	(0, 0, 0), (1, 1, 1), (1, 0, 1) (1, 1, 0), (0, 1, 1)
(1.4, 1.85, 1.75)	6	(0, 0, 0), (1, 1, 1), (1, 0, 1) (1, 1, 0), (0, 1, 1), (1, 0, 0)
(1.4, 1.9, 0.0)	7	(0, 0, 0), (1, 1, 1), (1, 0, 1) (1, 1, 0), (0, 1, 1), (0, 0, 1) (1, 0, 0)
(0.0, 0.0, 0.0)	8	(0, 0, 0), (1, 1, 1), (1, 0, 1) (1, 1, 0), (0, 1, 1), (0, 0, 1) (1, 0, 0), (0, 1, 0)

CHAPTER 4

LEARNING
OR THE DYNAMIC UP-DATE OF "WEIGHTS"

Learning is a key feature of artificial neural nets that critically affects hardware designs. Learning schemes are usually used to determine the weights from a set of training examples [30,33,34]. In some cases, neural networks have been shown to be capable of learning the input-output relation from a finite set of examples, defined as the training set, simply by minimizing a given measure of the error from the training examples over the network's parameter-space. This minimization is called the learning process. Because the learning algorithms generally takes orders of magnitude longer than the operation of '*reading*' the network, most applications requiring fast adaptability cannot yet be implemented in hardware. Consequently, non-adaptive applications requiring weights to be set only once (not in real time) should be developed first [21]; in that case learning algorithm efficiency is not an important issue.

Different types of learning definitions may be formulated [31,32]. We limit attention here to the so-called supervised learning [31], and in particular for dynamic feedback models of ANNs. In the context of ANNs, supervised learning loosely means that the network acquires a desired set of data (vectors) as stable equilibria in the case of dynamic feedback ANNs.

Even in theory, learning for neural nets, particularly feedback neural nets, remains a very difficult problem. The main reason is that the models of ANNs are *nonlinear*. When the model of the ANN is dynamic, as oppose to static, its dynamics will interact with the dynamics of a learning algorithm (or an update law). This type of problem has plagued the development of the so-called robust adaptive algorithms for linear systems in the area of *adaptive control* [77]. The situation is perhaps worse in ANNs because the model of an ANN is *nonlinear*.

Learning for static models, such as the case for feedforward ANNs [34], is conceptually simpler because the model does not have dynamics -- it is only a static map. Thus the problem of coupling between the model and a dynamic learning algorithm does not exist. This, to some extent, explains the relative success of the gradient learning algorithm popularized as the error back-propagation [34].

However, in electronic implementations, the implementation of a static model (or a map) results in a dynamic model, nonetheless. This is caused by physical reality and by the presence of parasitic capacitors, resistors, transistors, etc.. Consequently, one is faced with the possibility of interaction or coupling between the dynamics of the ANN and its dynamic learning algorithm.

In this work, a dynamic learning algorithm is introduced for general dynamic continuous-time models of ANNs.

The learning algorithm

Consider the general dynamic model of ANNs. (In fact, it may be any general dynamic model.)

$$\dot{x}_i = \frac{dx_i}{dt} = f_i(x, p), \tag{4.1a}$$

$$y_i = S_i(x_i), \tag{4.1b}$$

where $x = [x_1,...,x_n]^T$ is the $n\text{-}d$ state vector, $p = [p_1,...,p_l]^T$ is an $l\text{-}d$ parameter vector which needs to be updated (dynamically). (Supervised) learning means that a given vector, say x^* (or a set of m vectors $x^{1*},...,x^{m*}$) becomes a stable equilibrium point. It is assumed, of course, that solutions of equation (4.1a) exist and are unique for each initial condition $x(0)$, as it is often assumed for differential equations.

Suppose that we desire the learning algorithm to "learn" a given vector x^*. First, we define the energy function for the learning scheme as follows:

$$E = \frac{1}{2}\sum_{i=1}^{n} (\dot{x}_i\ )^2 = \frac{1}{2}\sum_{i=1}^{n} f_i(x,p)^2. \tag{4.2}$$

The learning scheme updates each parameter according to the dynamic equation:

$$\frac{dp_j}{dt} = -\frac{\partial E}{\partial p_j} = -\sum_{i=1}^{n}\dot{x}_i\frac{\partial f_i(x,p)}{\partial p_j} = -\sum_{i=1}^{n} f_i(x,p)\frac{\partial f_i(x,p)}{\partial p_j} \tag{4.3a}$$

That is,

$$\frac{dp_j}{dt} := -f(x,p) * Df(x,p). \tag{4.3b}$$

where $f(x,p) = [f_1(x,p),...,f_n(x,p)]^T$ is the vector field of equation (4.1a) and $Df(x,p)$ denotes the vector composed of the partial derivatives on the right-hand side of equation (4.3a). Note that in the case that p is a matrix, each element of p will be specified by two indices.

The algorithm proceeds as follows. Clamp, i.e. fix x in equation (4.3) at the desired value, say $x = x^*$. Let the dynamics of equation (4.3) evolve until they converge to a value for the parameter, say p^*. For $x = x^*$, the system of equation (4.3) is a gradient dynamic system and hence p^* must be an equilibrium point. In computer simulations or in implemented circuit hardware, the dynamics realistically converge to only stable equilibria. Hence, in practice, p^* is a stable equilibrium point. The vector p^* is also a minimum point of the energy function in equation (4.2), where $x = x^*$. When equation (4.3b) converges to p^*, we have

$0 = f(x^*, p^*) \, . \, Df(x^*, p^*).$

This means that (at least) one of the following is true:

Case (1): $f(x^*, p^*) = 0.$

Case (2): $Df(x^*, p^*) = 0$

Case (3): the vectors

$f(x^*, p^*)$ and $Df(x^*, p^*).$

are perpendicular to one another.

If case (1) is true, then the point x^* is an equilibrium point of (1i) for $p = p^*$. If the "physical" system or circuit does not permit case (2), then case (1) is the only possibility. In addition, if the model of equation (4.1a) is flexible enough, then it can support the (augmented) equilibrium (x^*, p^*) as stable equilibrium. Idealy if we plug p^* in equation (4.2), $E = \frac{1}{2}\sum_{i=1}^{n} (\dot{x}_i)^2 = \frac{1}{2}\sum_{i=1}^{n} f_i(x, p^*)^2 = 0$. Hence inserting p^* in equation (4.1) renders the desired vector x^* a *stable* equilibrium of equation (4.1) The desired vector can be retrieved by initial conditions sufficiently close to x^*.

CHAPTER 5

A PROTOTYPE 6-NEURON CMOS TINY-CHIP

A 6-neuron Tiny-chip of the all-MOS implementation of the new architecture has been designed and fabricated. Complete testing of the Tiny-chip [64,78] has successfully substantiated the predicted operation of the neural circuit. A dedicated interface circuitry has been built to facilitate the programming of the chip via learning algorithms developed onto a software environment with graphics display on a Personal Computer (PC). Using the interface circuitry and the learning schemes, we demonstrate the use of the Tiny-chips of the new neural circuits as a classification device.

5.1. Layout design

A VLSI layout was designed for a 6-*unit* neural circuit on MOSIS Tiny-chip as a prototype chip of the new architecture. This layout was fabricated by MOSIS using an n-well 2 μm CMOS process, with $2.2 \times 2.2\ mm^2$ chip area in a 40-pin package. The schematic diagram of this 6-*unit* neural circuit is depicted on Figure 5.1.1. Every node in the neural circuit is connected to an external pin to facilitate testing. This includes the input and output of every neuron and the gates of every connecting MOSFET. For the 6-neuron chip the number is $2\times(6) + \frac{(6)\times((6)+1)}{2} = 33$. Analog bonding pads are used in this design, i.e., there are no buffers between the I/O pins and our designed circuit. Thus, we can easily verify the characteristics of this neural

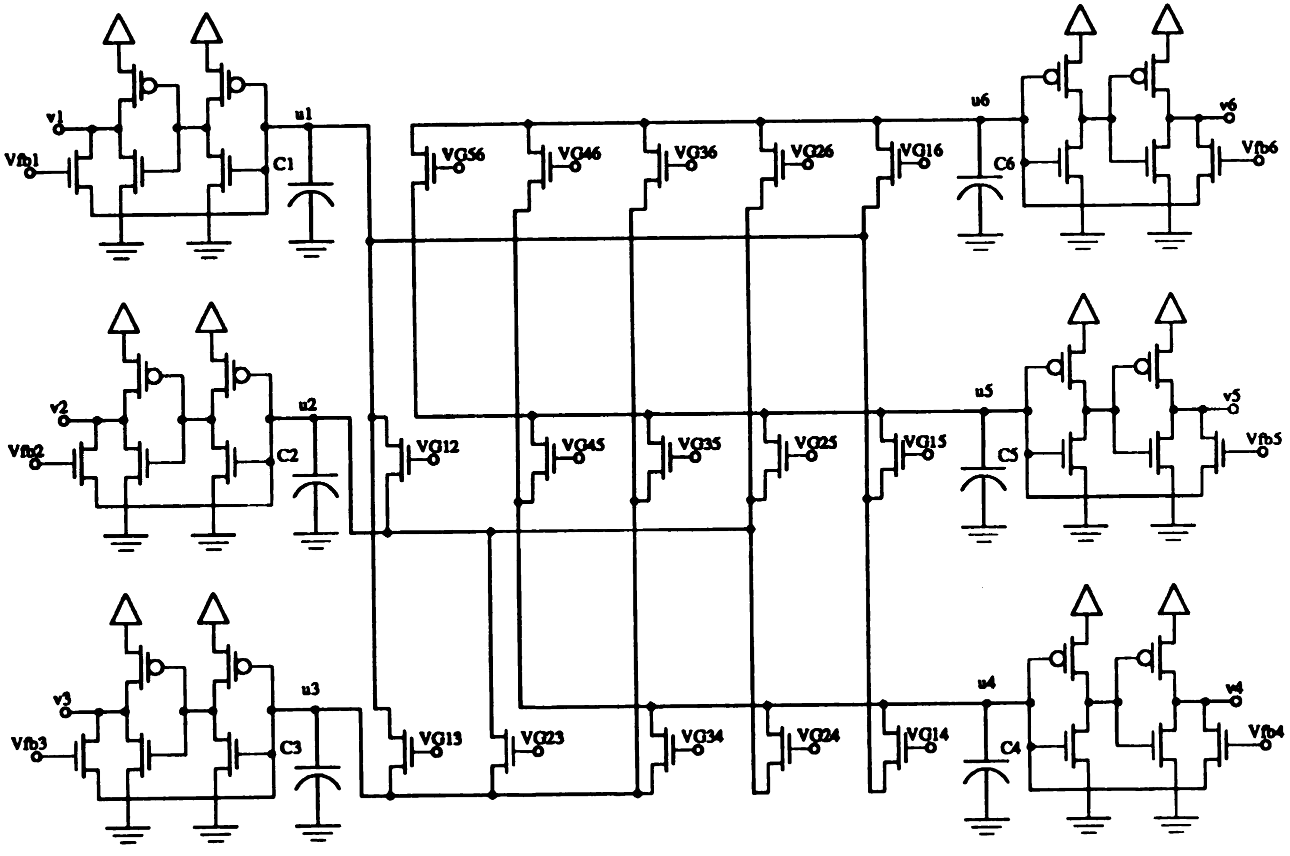


Figure 5.1.1 The schematic diagram of a fully connected 6-*unit* neural circuit.

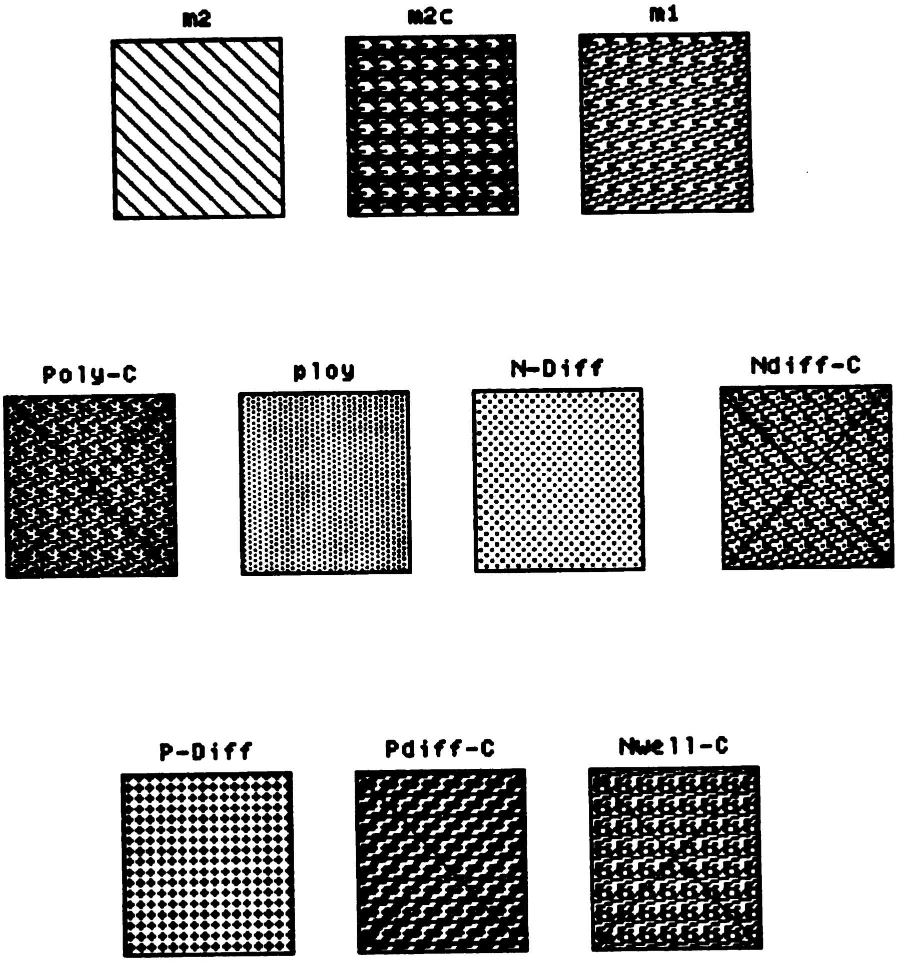


Figure 5.1.2 the layer names and their corresponding symbols.

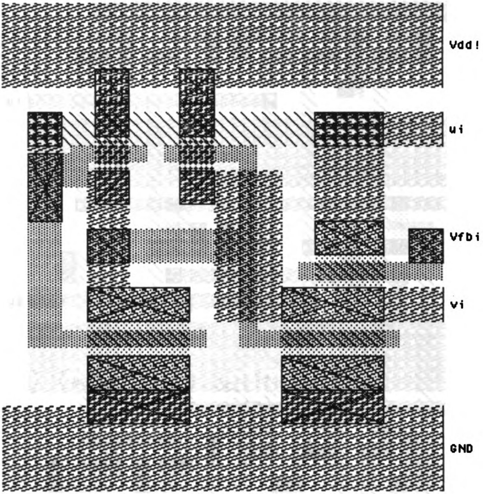


Figure 5.1.3 A VLSI layout of one neuron with self-feedback.

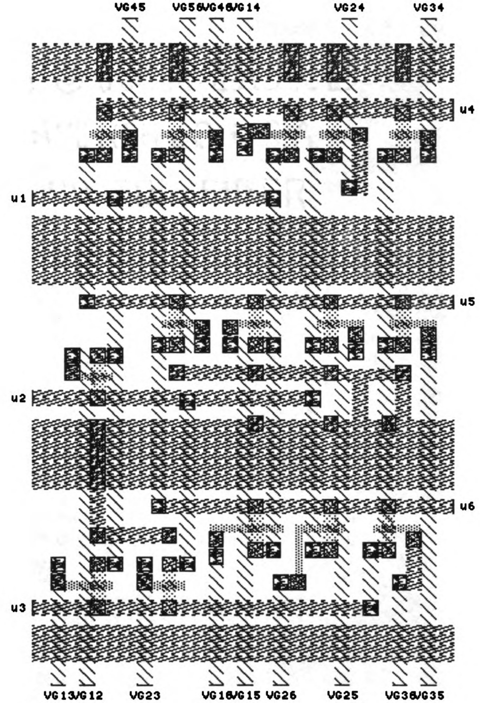


Figure 5.1.4 A VLSI layout of the interconnect network for the 6-*unit* neural circuit.

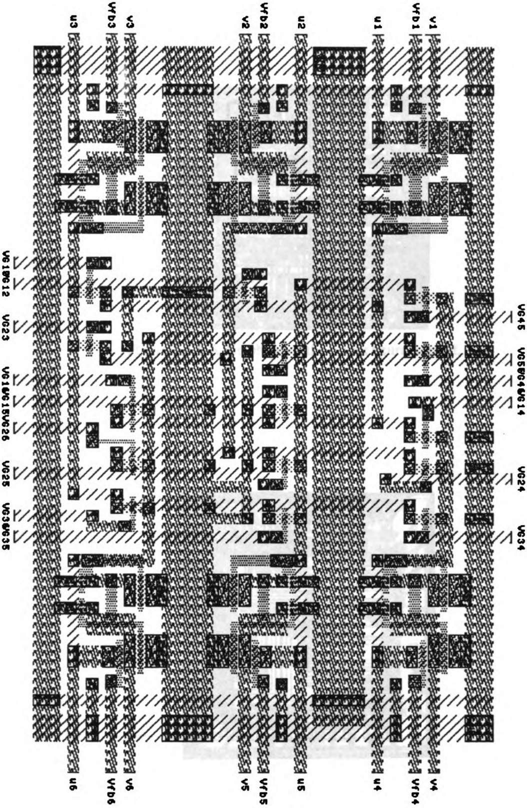


Figure 5.1.5 A VLSI layout of the *6-unit* neural circuit.

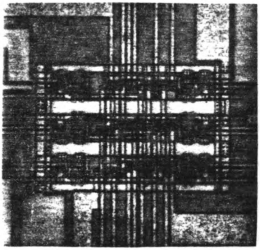


Figure 5.1.6 A die photo of the 6-*unit* neural circuit.

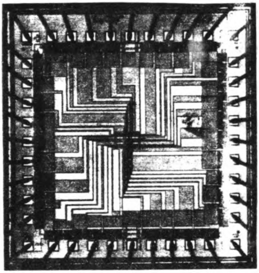


Figure 5.1.7 A die photo of the 6-*unit* neural circuit in MOSIS Tiny-chip.

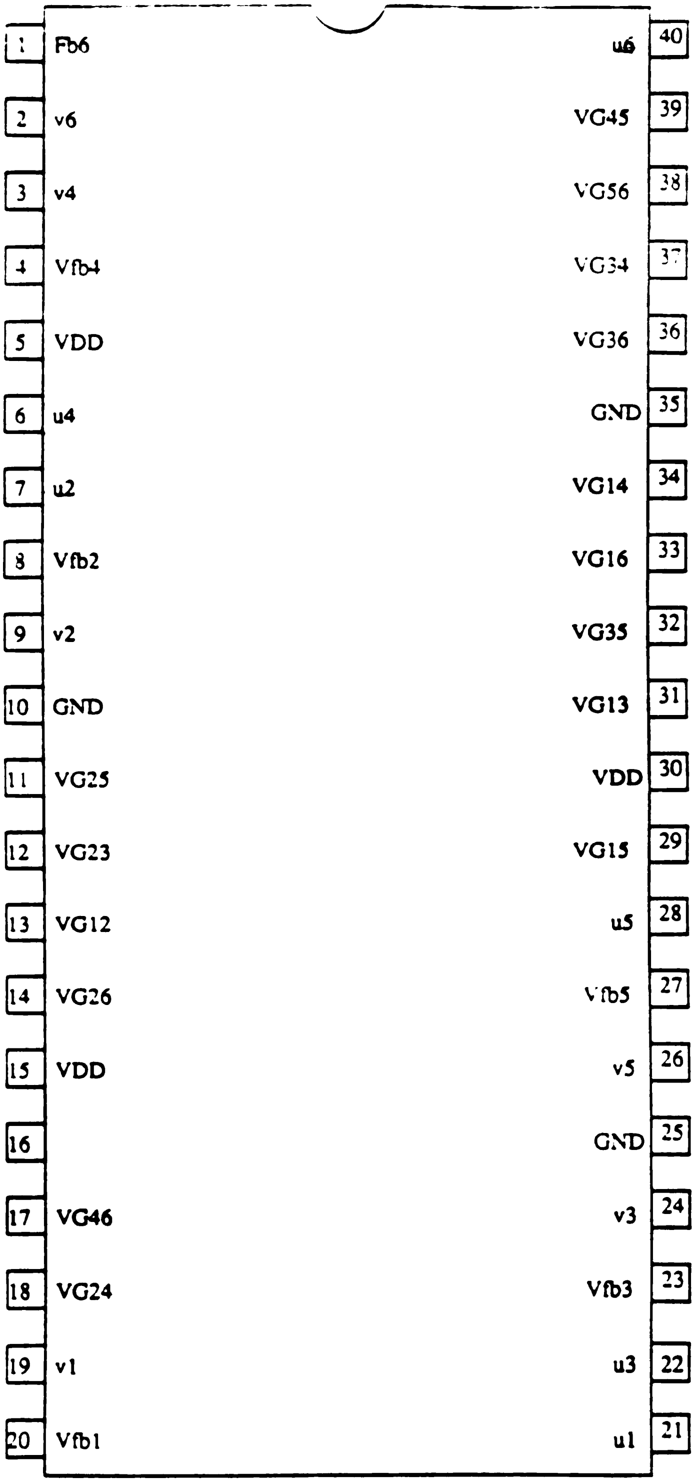


Figure 5.1.8 A pin configuration of the 6-neuron chip.

circuit and examine the performance of this chip design. Figure 5.1.2 depicts the layer names and their corresponding symbols. Figure 5.1.3 depicts the VLSI layout of one neuron with self-feedback. Figure 5.1.4 depicts the interconnect network with 15 nMOS transistors for the 6-*unit* neural circuit. Figure 5.1.5 depicts the VLSI layout of the 6-*unit* neural circuit corresponding to Figure 5.1.1. Figure 5.1.6 shows the die photo of the 6-*unit* neural circuit corresponding to Figure 5.1.5. Figure 5.1.7 shows the die photo of a 6-*unit* neural circuit in a MOSIS Tiny-chip with a $2.2 \times 2.2\ mm^2$ chip area and a 40-pin package. Figure 5.1.8 depicts the pin configuration of the 6-neuron chip.

In the testing process, we set the gate voltage of each feedback MOSFET to zero, then we measured the I/O characteristics of each double inverter. The resulting I/O function is indeed a sigmoid curve with a switch-point around 1.57 volts. Using the new SPICE parameters obtained by MOSIS from measurements of the MOSIS test structures on the selected wafers of this specific fabrication lot, the SPICE simulations have almost identical result. We used a 3-*unit* subcircuit for more testing to compare the results to the SPICE simulations.

We can adjust the gate voltage of the interconnected MOSFET conductance elements among the 3-*unit* subcircuit to obtain a different number of distinct steady states or stable equilibrium points. The experimental results are depicted in Table 5.1.1. The testing result is highly repeatable. We tested all four chips obtained from MOSIS for the same circuit design. The results for all four chips are similar to one another.

Complete testing of the Tiny-chips has successfully revealed the proper operation of the neural circuit [64,78].

Table 5.1.1 Experimental results of a 3-*unit* subcircuit of the 6-neuron chip.

Vfb_i = 5.0 volts for i=1,2,3.									
$(VG_{12}, VG_{23}, VG_{31})$	Number of Stable Equilibria	Initial States							
		000	001	010	011	100	101	110	111
		Steady States							
(5.0,5.0,5.0)	2	000	000	000	000	000	000	000	111
(3.2,3.5,5.0)	3	000	000	000	000	000	101	000	111
(0.0,0.0,3.7)	4	000	000	010	010	000	101	010	111
(3.1,3.1,3.1)	5	000	000	000	011	000	101	110	111
(3.2,3.5,0.0)	6	000	001	000	011	000	101	110	111
(2.7,2.7,0.0)	7	000	001	000	011	100	101	110	111
(0.0,0.0,0.0)	8	000	001	010	011	100	101	110	111

5.2. Interfacing with a Personal Computer

To ensure testing of the implementation and also to provide an environment for developing and assessing various potential learning algorithms, we planned at the design stage for interfacing the Tiny-chips with a PC. An interface [78-80] was developed such that the gate voltages of the 15 MOSFETs, which are the interconnect transistors among the input nodes of the 6-*unit* neural circuit, can be flexibly read or assigned analog values between 0 and 5 volts.

Only a single 8-bit D/A converter (DAC0800) is employed to convert the digital data to the corresponding analog voltages. The analog outputs of DAC0800 are multiplexed to the gate voltages of the 15 MOSFETs via analog multiplexers (CD4051). Fifteen Sample and Hold Amplifiers (LF398N) are required for holding the analog voltages at the gates of the 15 interconnect MOSFETs. To keep the precision of each gate voltage, the Sample and Hold Amplifiers should be refreshed within a certain time constant which depends on the size of the storage capacitor.

Two 8-bit 8-channel A/D converters (ADC0808) were employed to simultaneously read out the steady states at both input node and output node of a single neuron. The output of the ADC0808 is stored in an octal latch and then transferred to the PC.

The exact initial values are dependent on the outputs of the octal latch (specifically, 74LS373). The analog voltage of the octal latch output is about 3.5 volts when the corresponding input is high and is about 0.22 volts when the corresponding input is low.

5.3. A Learning Scheme

The learning algorithm described in Chapter 4 was specialized for the 6-*unit* Tiny-chips. The learning algorithm was implemented in software on a PC [30]; it interacts with the chip in real-time via a special purpose interface described in the previous section.

5.3.1. Dynamic Learning Model

Assumption: In equation (3.4.1), assume that $I_i = 0$, and the local resistance, R_i, is quite large. This is reasonable since R_i represents an input to a double inverter [1]. Thus, equation (3.4.1) can be rewritten as

$$C_i \frac{du_i}{dt} = \sum_k^n I_{ds}(u_k, u_i, VG_{ki}) + I_{ds}(v_i, u_i, Vfb_i). \tag{5.3.1.1}$$

We define the energy function for the learning scheme as the following:

$$E = \frac{1}{2}\sum_{i=1}^{n} (C_i \dot{u}_i)^2. \tag{5.3.1.2}$$

Therefore the learning scheme is governed by the following dynamic equations:

$$\begin{aligned} \frac{dVG_{ji}}{dt} &= -\frac{\partial E}{\partial VG_{ji}} \\ &= -C_i \dot{u}_i \left[\frac{\partial C_i \dot{u}_i}{\partial VG_{ji}}\right] - C_j \dot{u}_j \left[\frac{\partial C_j \dot{u}_j}{\partial VG_{ji}}\right] \\ &= -C_i \dot{u}_i \left[\frac{\partial I_{ds}(u_j, u_i, VG_{ji})}{\partial VG_{ji}}\right] - C_j \dot{u}_j \left[\frac{\partial I_{ds}(u_i, u_j, VG_{ji})}{\partial VG_{ji}}\right] \\ &= -C_i \dot{u}_i \left[\frac{\partial I_{ds}(u_j, u_i, VG_{ji})}{\partial VG_{ji}}\right] - C_j \dot{u}_j \left[\frac{-\partial I_{ds}(u_j, u_i, VG_{ji})}{\partial VG_{ji}}\right] \\ &= \left[C_j \dot{u}_j - C_i \dot{u}_i\right]\left[\frac{\partial I_{ds}(u_j, u_i, VG_{ji})}{\partial VG_{ji}}\right] \\ &= Gm_{ji}(u_j, u_i, VG_{ji})\left[C_j \dot{u}_j - C_i \dot{u}_i\right] \end{aligned} \tag{5.3.1.3}$$

where $C_j \dot{u}_j$ and $C_i \dot{u}_i$ may be obtained from equation (5.3.1.1) and $Gm_{ji}(u_j, u_i, VG_{ji})$ is the transconductance of a MOSFET which can be expressed as follows:

Cutoff: if $(VG_{ji} - X - v_t) \le 0$

$$Gm_{ji}(u_j, u_{i,} VG_{ji}) = 0 \tag{5.3.1.3a}$$

Triode: if $(VG_{ji} - X - v_t) \geq |(u_j - u_i)|$

$$Gm_{ji}(u_j, u_{i,} VG_{ji}) = K(u_j - u_i) \tag{5.3.1.3b}$$

Saturation: if $(VG_{ji} - X - v_t) \leq |(u_j - u_i)|$

$$Gm_{ji}(u_j, u_{i,} VG_{ji}) = K(VG_{ji} - X - v_t) \tag{5.3.1.3c}$$

where v_t is the threshold voltage, X is the minimum of u_j and u_i, and $K = uC_{ox}(W/L)$.

5.3.2. Software Implementation of the Dynamic Learning

The learning scheme is accomplished via software implementation using the Runge-Kutta fourth order integration routine running on an AT personal computer. A hardware interface was built so that the gate voltages of the fifteen MOSFETs, which are the connection transistors among the input nodes of the 6-unit neural circuit, can be controlled, i.e., analog voltage values read and written onto them. The Tiny-chip neural circuit can be initialized by the interface at the input nodes; the initial values are dependent on the outputs of the one Octal latch (74LS373). Each output bit of the Octal latch controls one input node of the neural circuit. The analog voltage of the Octal latch output is about 3.5 volts when the corresponding input is high; otherwise the output is about 0.22 volts. The square law MOS transistor model which was described by equation (2.4.1) is employed to simulate the interconnect and feedback MOSFETs. The parameters of the transistor model are obtained from MOSIS after the fabrication of the chip. The learning scheme communicates with the actual 6-unit neural circuit via the computer interface.

The learning procedure is summarized in the following:

Step 1:

The desired training pattern is represented in binary format and is set by the computer interface.

At every input node of the 6-unit neural circuit, "0" means 0.22 volts and "1" means 3.4 volts as set by the interface.

The initial values of the gate voltages of the interconnect transistors are also set by the computer interface. In this procedure, all gate voltages are initialized at 5 volts (or the high level).

Step 2:

The actual analog voltages of the input and the output nodes, obtained from the interface are used to compute the new gate voltages of the interconnect MOSFETs according to dynamics of the learning scheme in equation (5.3.1.3).

Step 3:

The new gate voltages of the interconnect MOSFETs are used in the chip by the interface. Then the process is repeat. Go to Step 2 until the change of the gate voltages reaches a chosen stoppage criterion.

A computer graphic environment displays the interconnected architecture as well as the updating of the analog values for each interconnect gate voltage. After the stoppage criterion is satisfied, the personal computer will automatically test all possible initial conditions (all combinations of high and low) and display them next to their corresponding steady states at the input nodes of the units. The whole testing procedure is sequentially displayed in the computer graphics. Using this procedure, one can verify that a given pattern is indeed stored as a stable equilibrium point. Moreover, one can test all the binary initial conditions and determine the steady states to which they converge.

5.4. Chip Experimental Results with Learning

A desired pattern (i.e., a desired stable equilibrium point) is represented as a binary number $(X_6X_5X_4X_3X_2X_1)$, where X_i $(i = 1, \cdots, 6)$ represents the logical value of the desired steady state at the input node (or at the output node). We set the gate-voltages of the feedback MOSFETs to logical high (i.e. about 5 volts). When all the gate-voltages of the 15 interconnect MOSFETs are high, the neural circuit would have only two stable equilibria, namely (000000) and (111111) [13,14]. All initial conditions applied at the input nodes of the neural circuit will converge to either one of these two states. These two states shall be referred to as the no-information states. We note that these no-information states will remain regardless of the values of the gate-voltages of the 15 interconnect MOSFETs [13,14]. The initial conditions applied at the gates of the 15 interconnect MOSFETs are set at 5 volts. Now, we apply the outlined learning procedure in section 5.3.2.

Storing a single pattern

The test results show that the learning scheme can successfully store any desired pattern. The complement of the desired pattern, however, will also be stored. That is, the neural circuit would now store the desired pattern, its complement and the always-present no-information states. If appropriate bias or restriction on the values of the gate-voltages of the interconnect MOSFETs are applied, the complement of the desired pattern can be eliminated.

Storing multiple patterns

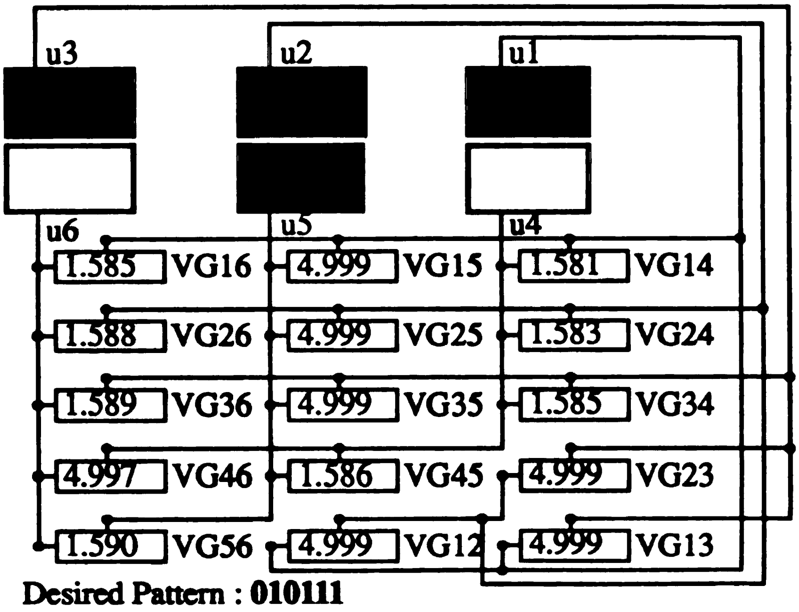
Multiple patterns can be applied sequentially. After the first pattern is applied, the learning scheme would converge to a set of gate-voltages for the 15 interconnect MOSFETs. As the next pattern is applied, the last values of the gate-voltages of the 15 interconnect MOSFETs are used as initial conditions for the learning procedure. The test result shows that multiple desired patterns can successfully be learned sequentially. We found that each time a new pattern is learned, other patterns of intersection images among the desired patterns and their complements will also be stored.

Experimental examples

Figure 5.4.1 depicts a graphic display after each of three patterns was learned in a sequential learning experiment. Figure 5.4.1.a displays the first desired pattern, namely (010111), which looks like the capital letter "T". The display also shows the steady state analog values of the "converged" gate-voltages of the 15 interconnect MOSFETs. This set of gate-voltages enables the network to store (as stable equilibria) the pattern (010111), its complement (101000), in addition to the two no-information states (000000) and (111111). The right-hand side of the display depicts the roster of the distinct steady states and the corresponding count of initial conditions converging to each steady state. In the roster, 11 (distinct initial conditions) converged to the desired pattern, 11 converged to its complement, 5 converged to the all-one state, and 37 converged to the all-zero state.

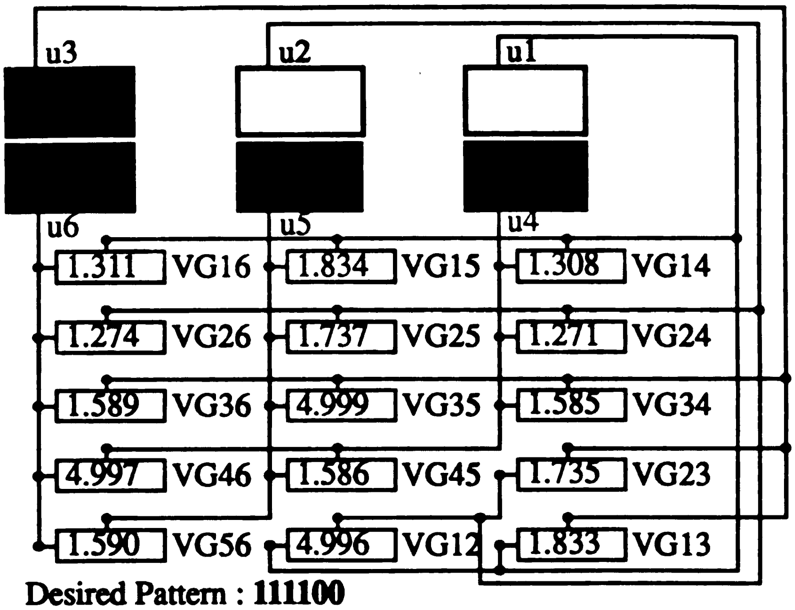
Figure 5.4.1.b shows the second desired pattern (111100), which looks like the capital letter "L", and its corresponding (converged) gate-voltages of the 15 interconnect MOSFETs after the network had learned the second desired pattern. There are 8 distinct (stable) equilibria stored in the network, the first desired pattern (010111) and its complement (101000), the second desired pattern (111100) and its complement (000011), the intersection pattern of the first desired pattern and the second desired pattern (010100) as well as its complement (101011), and the two no-information states. The roster shows the number of initial conditions that converged to each steady state.

Figure 5.4.1.c shows the third desired pattern, (101111) which looks like the upper half of the capital letter "O", and its corresponding gate-voltages of the 15 interconnect MOSFETs after the network had learned the third pattern. There are 15 distinct (stable) equilibria stored in the network. We observe that the intersection pattern of (010111), (111100), and (101111), is not stored in the network, but its complement is.



S. S.	#
000000	37
010111	11
101000	11
111111	5

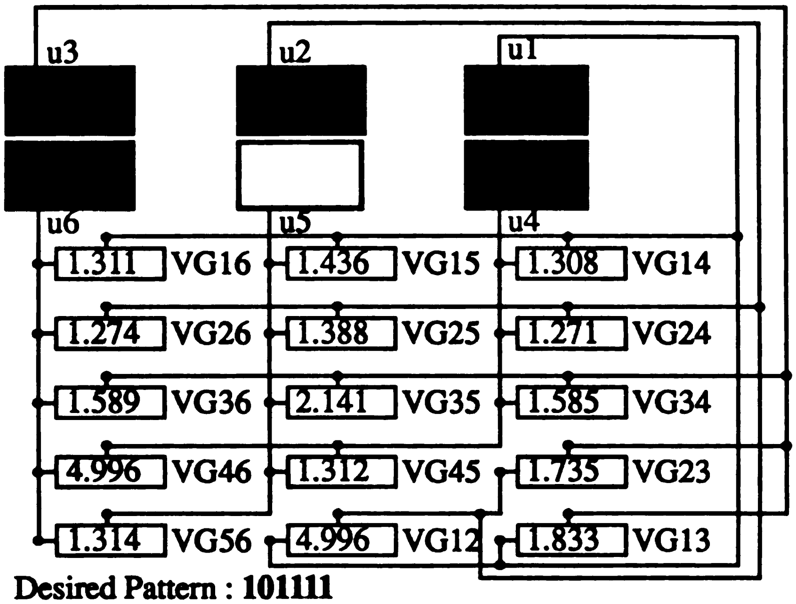
Figure 5.4.1.a A graphic display after the network had learned the first desired pattern.



Desired Pattern : **111100**

S. S.	#
000000	27
000011	9
010100	9
010111	3
101000	9
101011	3
111100	3
111111	1

Figure 5.4.1.b A graphic display after the network had learned the second desired pattern.



Desired Pattern : **101111**

S. S.	#
000000	18
000011	3
000111	3
010000	9
010011	3
010100	9
010111	3
101000	3
101011	1
101100	3
101111	1
111000	3
111011	1
111100	3
111111	1

Figure 5.4.1.c A graphic display after the network had learned the third desired pattern.

CHAPTER 6

A 50-NEURON CHIP WITH DIGITAL HARDWARE LEARNING SCHEME

A digital realization of the learning algorithms was developed for the new analog feedback ANN. A VLSI layout of a 50-neuron CMOS analog chip with on-chip *digital* hardware learning scheme was designed and fabricated with a 6.8*mm* × 4.6*mm* chip size using a MOSIS 2-*μm* scalable CMOS technology and a 64-pin standard pad frame [14]. Extensive testing of the 50-neuron chip has successfully substantiated the predicted operation of the neural circuit.

6.1. A Digital Learning Scheme

Subsequently, the new algorithm described in Chapter 4 is specialized to a digital learning scheme which is realizable and valuable in the all-MOS VLSI implementations The digital learning scheme is also tested on the 6-neuron Tiny-chip for successfully storing and retrieving arbitrary digitized images.

6.2. Hardware Overview of the 50-neuron Chip

The digital learning scheme can easily be implemented via simple logic circuitry. The layout design of the 50-neuron chip had been fabricated by MOSIS via a

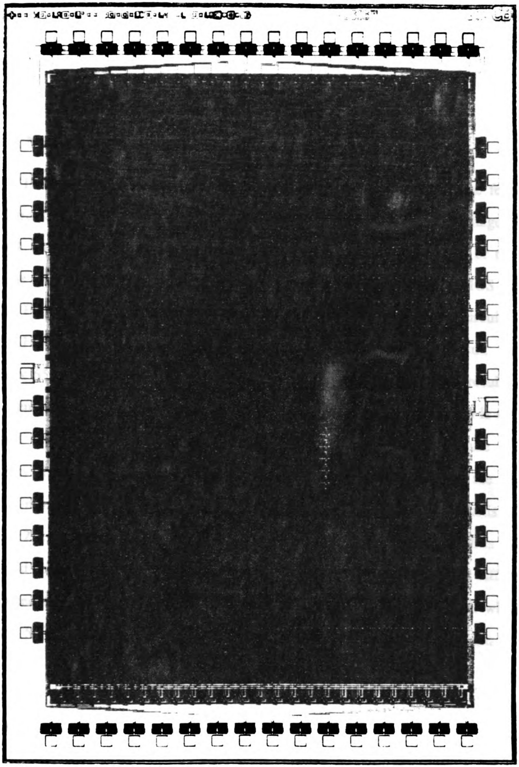


Figure 6.2.1 A die photo of the 50-neuron chip.

standard CMOS 2-μm n-well technology. Figure 6.2.1 shows the die photo of the 50-neuron chip. From a conservative estimation, a 124-neuron chip can be designed and fabricated via a MOSIS 1.2μm CMOS technology on the MOSIS large chip of a 7900μm × 9200μm chip size.

6.2.1. A Neuron

Each neuron is represented two CMOS inverters in series with one feedback nMOS transistor between the input node and the output node. The gate voltage of the feedback nMOS transistors can be adjusted globally. Either the input node or the output node of the neuron will be connected to an external pin via a CMOS analog switch. Figure 6.2.1.1 depicts the schematic diagram of a single neuron circuit. Figure 6.2.1.2 depicts the magic layout design of a single neuron circuit with a 154μm × 65μm chip size. Figure 6.2.1.3 depicts the SPICE simulation results of a single neuron while the gate voltage of the feedback nMOS transistor (Vfb_i) is set at 0 volt and the I/O selector is set at 0 volt. The horizontal axis of Figure 6.2.1.3 represents the signals applied at the external pin. The vertical axis of the Figure 6.2.1.3.a represents the signals at the input node of a neuron. The vertical axis of the Figure 6.2.1.3.b represents the signals at the output node of the neuron which is indeed a sigmoid function. The vertical axis of the Figure 6.2.1.3.c represents the difference of between the signals at the input node of the neuron and the external pin which is less than 500nV. Figure 6.2.1.4 shows the die photo of a single neuron circuit.

6.2.2. A Programmable Synapse

In a single 50-neuron chip, there are 1225 synaptic weights which are programmable and can be set up via on-chip *digital* hardware learning circuitry or

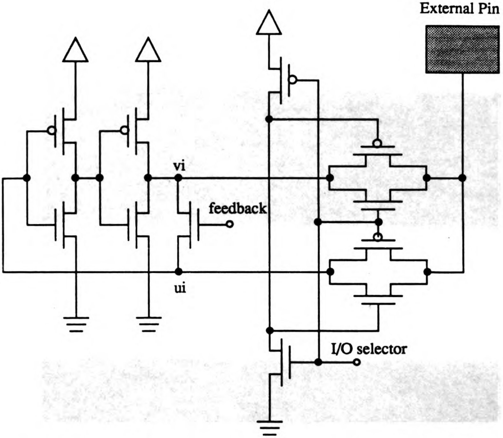


Figure 6.2.1.1 A schematic diagram of a single neuron with self-feedback and I/O selector.

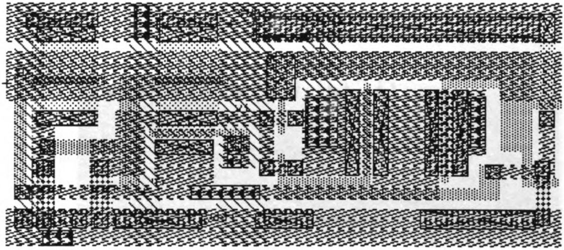
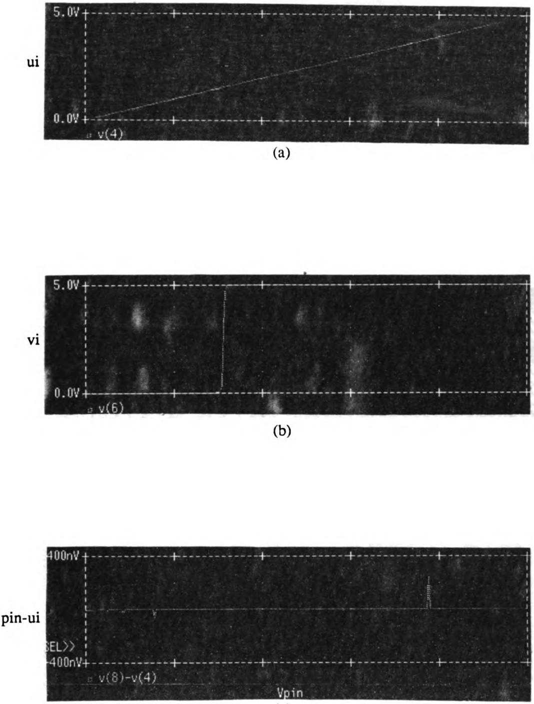


Figure 6.2.1.2 A VLSI layout of a single neuron circuit.



(a)

(b)

(c)

Figure 6.2.1.3 SPICE simulation results of a single neuron circuit.

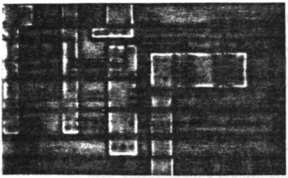


Figure 6.2.1.4 A die photo of a single neuron circuit.

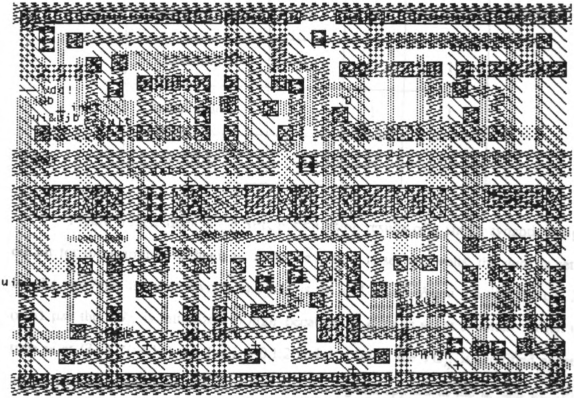


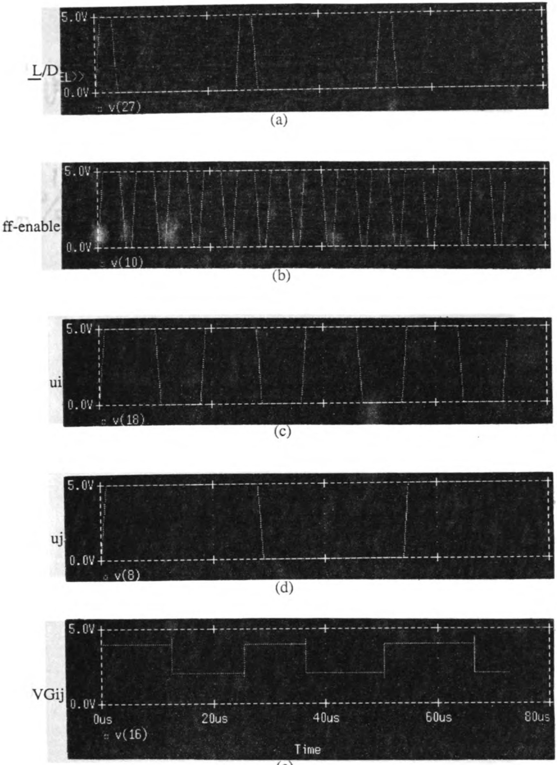
Figure 6.2.2.1 A VLSI layout of a single synaptic weight circuit.

via direct assignment. Each synaptic weight, i.e., each gate voltage of an interconnect nMOS transistor, exhibits either logical high or logical low and it can be stored on an on-chip digital flip-flop. The logical high and the logical low can globally be set at any analog values. Each synaptic weight can be set at any logical value individually via direct assignment logic circuitry. Figure 6.2.2.1 depicts the magic layout design of a synaptic weight circuit in 154μm × 112μm chip size. Table 6.2.2.1 depicts the different operating functions and their corresponding control signals.

Table 6.2.2.1 Control signals for the operating functions.

	assign	L/D	ff-enable
Direct assignment	0	1	X
Learning	1	0	1
Initialization	1	1	1
Running	1	X	0

Figure 6.2.2.2 depicts the SPICE simulation results of a single synaptic weight circuit for the learning mechanism, while the $\underline{assign}$ is set at 5 volts, the *data* is set at 5 volts, the ***high*** is set at 4 volts, and *low* is set at 2 volts. When the time is at 3, 27, or 51 μs, the VG_{ij} is initialized to ***high*** because the *data* is set at 5 volts. When the time is at $(3+30*i+6*j)$ μs, for i = 0,1,2 and j = 0,1,2, the VG_{ij} is modified according to the the digital learning circuitry and the signals applied at the u_i and u_j. Figure 6.2.2.3 depicts the SPICE simulation results of a single synaptic weight circuit for the direct assignment mechanism, while the $\underline{assign}$ is set at 0 volt, the ***high*** is set at 4 volts, and *low* is set at 2 volts. When the time is at $3+6*i$ μs, for $i = 0, \cdots, 6$,



(a)

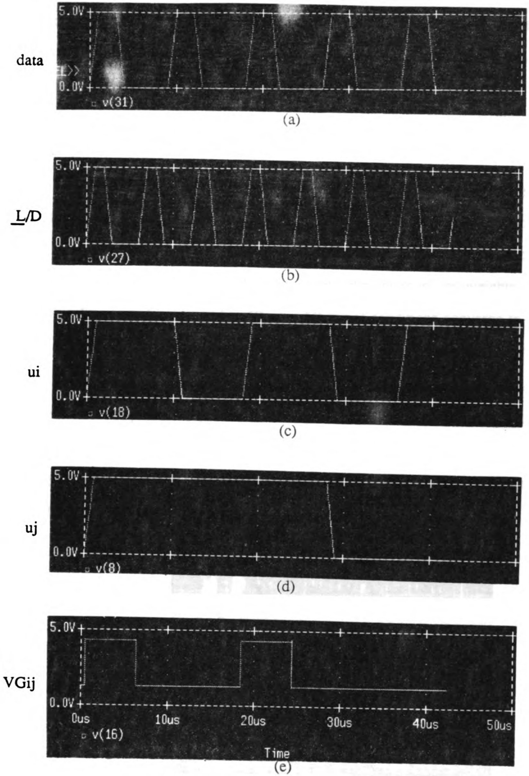
(b)

(c)

(d)

(e)

Figure 6.2.2.2 SPICE simulation results of a single synaptic weight circuit for the learning.



(a)

(b)

(c)

(d)

(e)

Figure 6.2.2.3 SPICE simulation results of a single synaptic weight circuit for the direct assignment.

VG_{ij} will be set at the *high* or the *low* according to the *data*, if both u_i and u_j are simultaneously at logical high. From Figure 6.2.2.2 and Figure 6.2.2.3, the functionality of a single synaptic weight was verified, before we have the design fabricated. Figure 6.2.2.4 shows the die photo of a single neuron circuit.

6.3. Chip Experimental Results

The fabricated chips were received on October 19, 1990. Extensive testing of the 50-neuron chips has successfully substantiated the predicted operation of both the neural circuit and the on-chip digital learning circuit. A desired equilibrium point can successfully be stored into the chip by setting the interconnect weights via digital hardware learning circuitry within 50 *ns*. All initial conditions converge to the steady states in the order of 20 μs.

From experimental measurement using a digital meter, the power consumption of the neural chip in steady state is less than 1 mWatt with power supplies of 0/5 volts. The neural circuit safely retains the same dynamic properties when the power supply high level is decreased down to 3 volts to dramatically reduce the power consumptions.

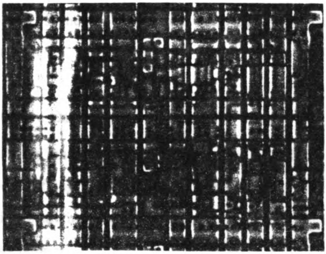


Figure 6.2.2.6 A die photo of a single synaptic weight circuit.

CHAPTER 7

A REAL-TIME APPLICATION USING THE 50-NEURON CHIP AS A PATTERN/CHARACTER ASSOCIATOR

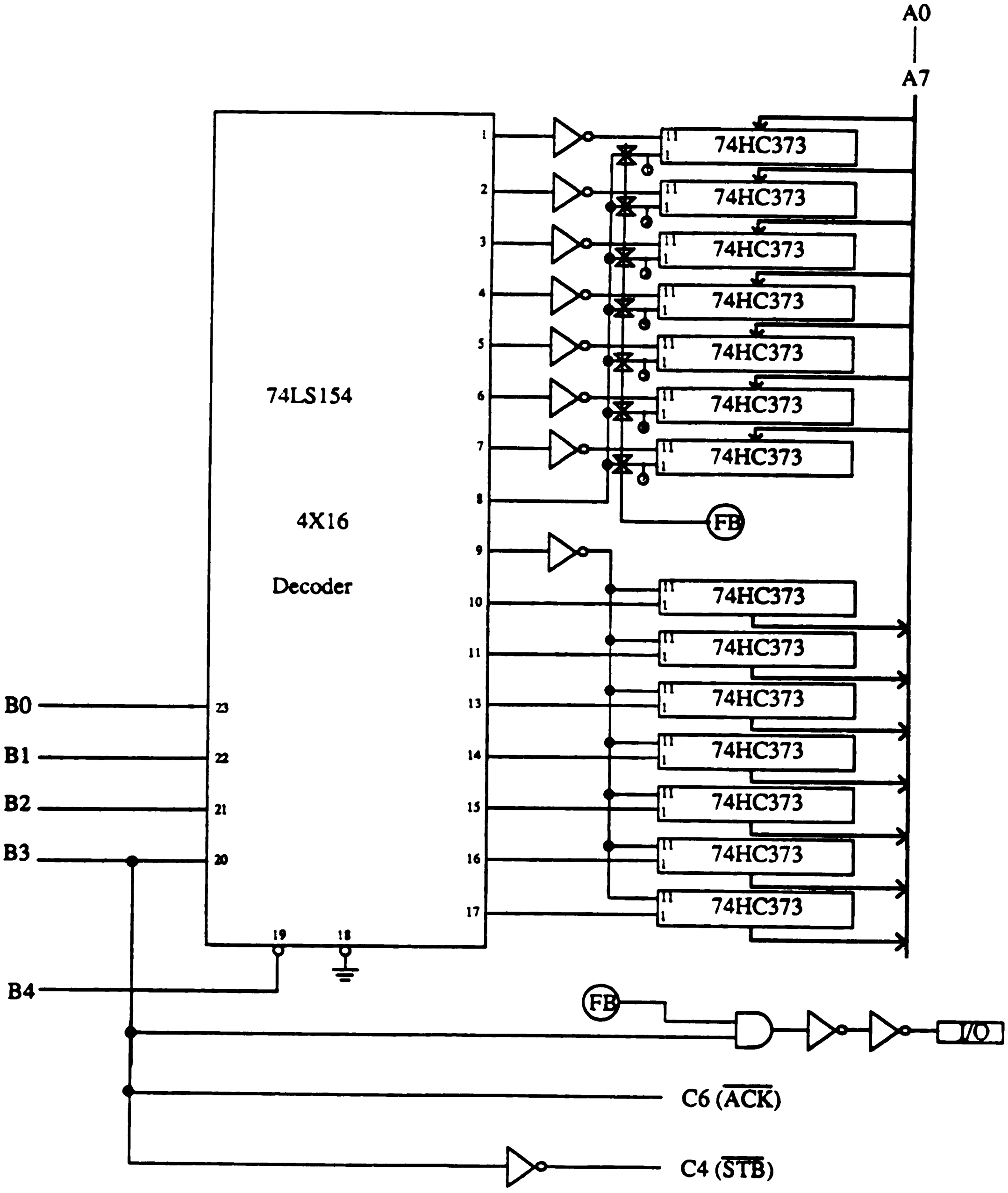
Forty nine neurons of the fifty neurons in the 50-neuron chip are employed to form a 7×7 pixel array to process 2-D images. Thus, each neuron represents one pixel. A pattern/character is represented as a 7×7 resolution image. If the output voltage of a neuron is higher (lower) than some threshold value, the corresponding pixel is considered to be one (respectively, zero).

A dedicated interface circuitry and a software environment had been built to successfully demonstrate the use of the 50-neuron chips. A real-time application has been performed using the 50-neuron chip with on-chip digital learning as a pattern/character associator.

7.1. Interfacing and Software Environment

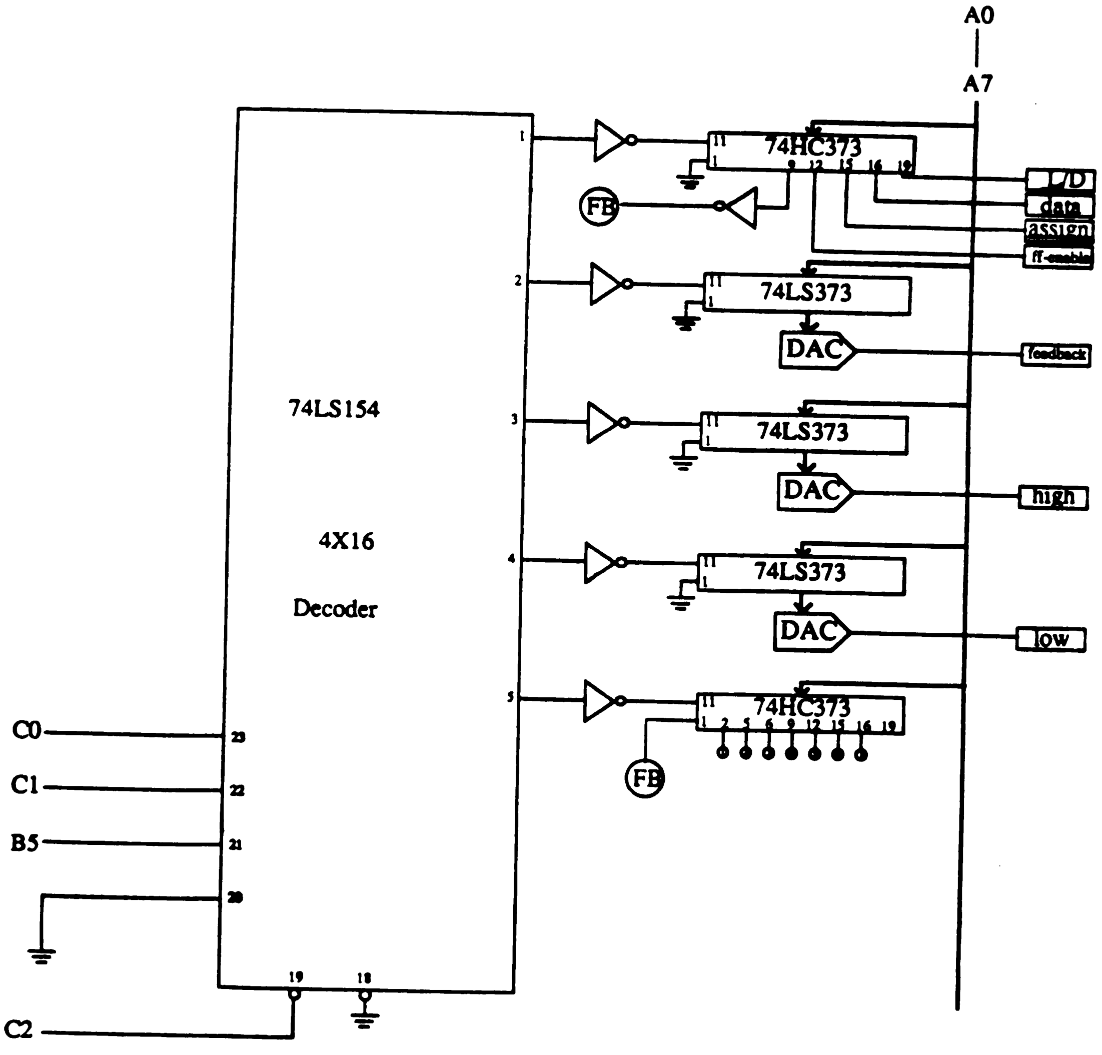
An interface circuitry and a software environment have been designed and developed for testing and verifying the functionality and the performance of the fabricated chips as well as demonstrating a real-time application using the 50-neuron chip as a pattern/character associator.

The interface is used to control each neuron's input and output, the weight of each interconnect, and a few logical controls of the chip. An Intel 8255 is used as a



(a)

Figure 7.1.1 A block diagram of the interface circuitry for the 50-neuron chip.



(b)

Figure 7.1.1 A block diagram of the interface circuitry for the 50-neuron chip.

digital I/O port to communicate between a PC AT and the interface. Three 8-bit D/A converters (DAC0800) are used to globally set up the gate voltages of the feedback nMOS transistors and the gate voltages of the interconnect nMOS transistors. Several octal latches (74HC373) were employed to build 50 digital output channels and 50 digital input channels. Figure 7.1.1 depicts the block diagram of the interface circuitry.

Using the new architecture as a pattern/character associator, a software program is developed using the C language on a PC AT which provides a user-friendly environment to easily control and operate the fabricated chip. A desired pattern/character can directly be specified from keyboard interactively or from an external file and then the program will generate the proper signals via interface circuitry to have the 50-neuron chip learn and store this specifically desired pattern/character in the chip. The current network configuration, the logical values of the gate voltages of the interconnect MOSFETs, can graphically be displayed on a screen. Initial conditions with various Hamming distances from the desired pattern/character can be tested via this program. An initial input and a steady state output can be transferred to and from the 50-neuron chip via this software environment. All experimental data and the graphics are stored in files for later use.

7.2. Storing a Single Pattern/Character

Initially, as the power is turned on, the interface sets the gate voltages of the feedback and the interconnect nMOS transistors to high (i.e., 5 V). In this configuration, the chip has only two steady states stored in it, namely, the all-high state (all outputs are near 5 V) and the all-low state (all the outputs are near 0 V); all initial conditions of the chip now converge to either the all-high of the all-low states. The network retains these two states for all the subsequent learning configurations. We label these two states the no-information states.

A pattern/character can successfully be stored into the chip by setting the interconnect weights via hardware learning circuitry within 50 ns. After a desired character is stored, the complement of this desired character will also be stored in the network. All initial condition images with 18.4% binary distortions from the desired character can retrieve the desired character in the order of 20 μs.

In the particular experiments reported here, we supply the network with the image of the capital-letter **F**. The network stores the image of the character successfully. We then test some initial conditions with Hamming distances less than 9 from the character **F**. All such initial conditions will retrieve the stored character **F**. Figure 7.2.1 tabulates some of the experiments which were executed. Beginning from the top of the tabulation in Figure 7.2.1, the first (top) row denotes initial images which were supplied as initial conditions to the network. Each initial image converged to the image immediately beneath it in Figure 7.2.1. Similarly, the third row of initial images correspondingly converged to the images on the row beneath it, and so on. It is clear that all images converge to the image of the desired character **F**. Figure 7.2.1, therefore, depicts a set of noisy variations of the character **F** which the network tolerates in retrieving the character **F**. The network, therefore, performs association in addition to the character recognition.

In this particular experiment, initial-condition images with more Hamming distance than 9 do not all converge to the desired character **F**. Figure 7.2.2 tabulates some of the results where the initial images have Hamming distances 10 from the character **F**. In Figure 7.2.2, however, an initial image which loses more than 9 (on) pixels compared with the image of the character **F** converges to the all-low state. It is interesting to note that the initial images that converged to the all-low state may not be distinguishable by a human observer.

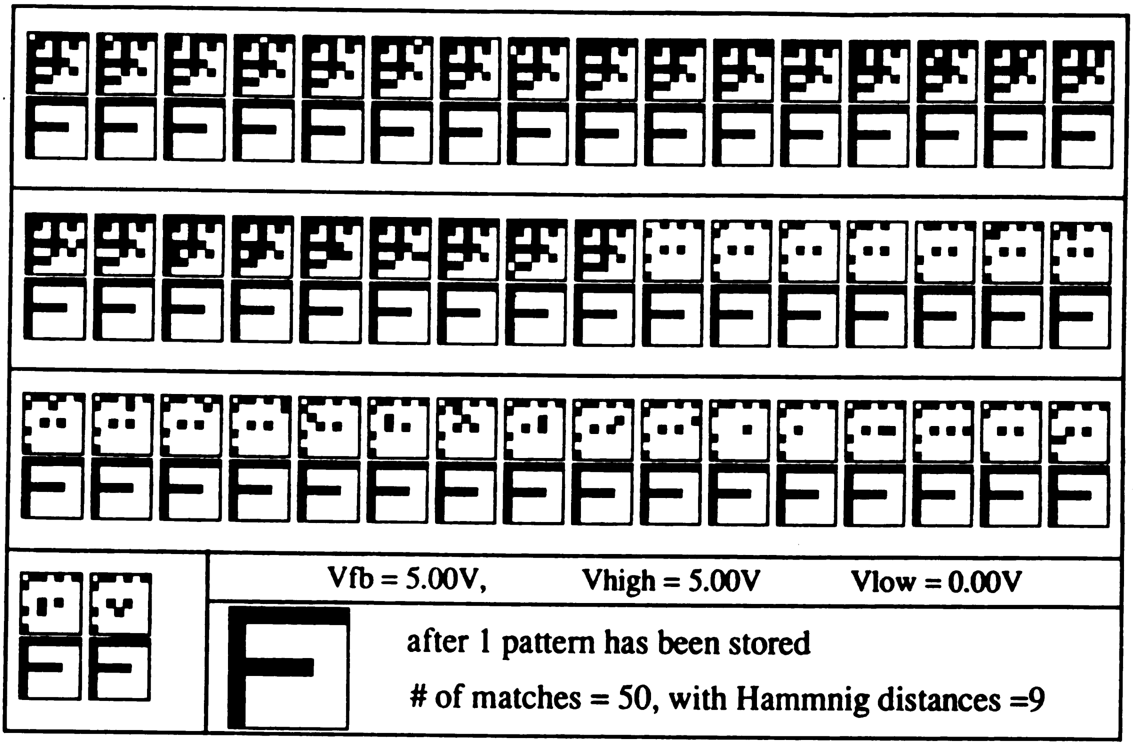

Vfb = 5.00V,
Vhigh = 5.00V
Vlow = 0.00V
after 1 pattern has been stored
# of matches = 50, with Hammnig distances =9

Figure 7.2.1 Experimental results of the 50-neuron chip;
all initial conditions with Hamming distances 9 from the character F.

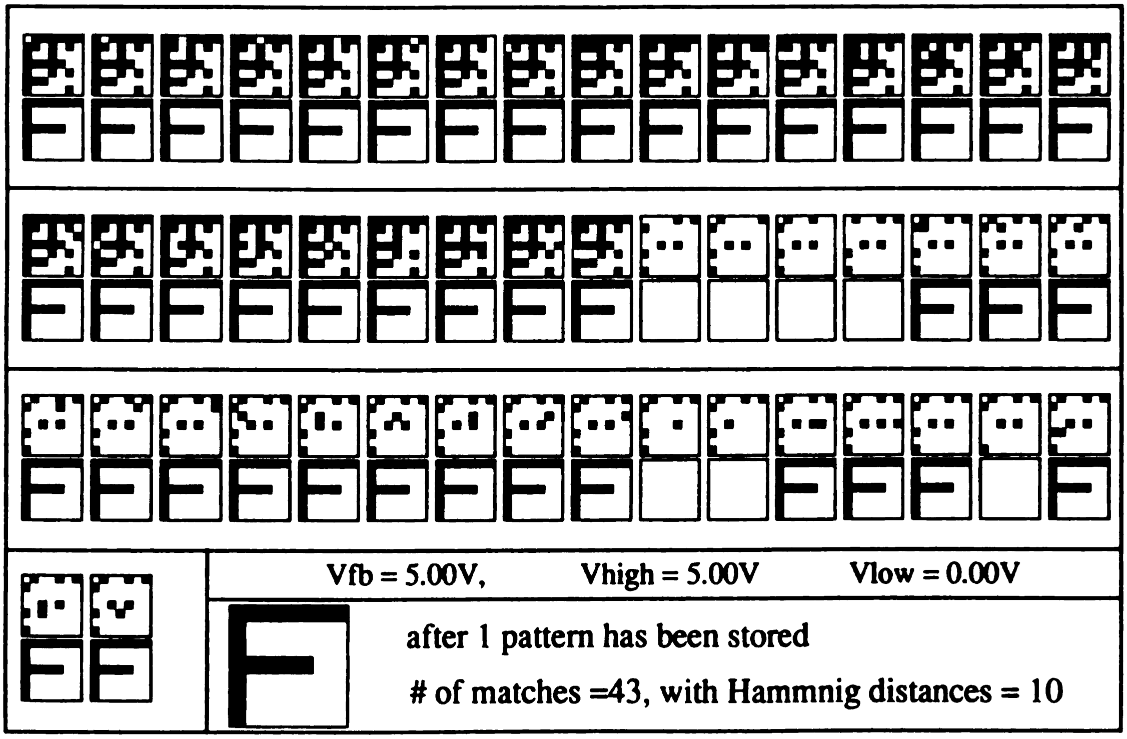

Vfb = 5.00V,
Vhigh = 5.00V
Vlow = 0.00V
after 1 pattern has been stored
# of matches =43, with Hammnig distances = 10

Figure 7.2.2 Experimental results of the 50-neuron chip;
all initial conditions with Hamming distances 10 from the character F.

7.3. Storing multiple patterns/characters

The chips also have the capability of learning multiple patterns sequentially. Figure 7.3.1 tabulates some experimental results after the network stored two images of the character **F** and the character **T**. In this particular experiment, all initial images with Hamming distance 3 from the the character **F** (respectively, **T**) converged to the image of the character **F** (respectively, **T**). We note that the network stored not only the desired characters but also the intersection images among the desired characters and their complements. Under this specific digital hardware learning scheme, the ***maximum*** number of the intersection images increases exponentially as the number of the desired characters increases.

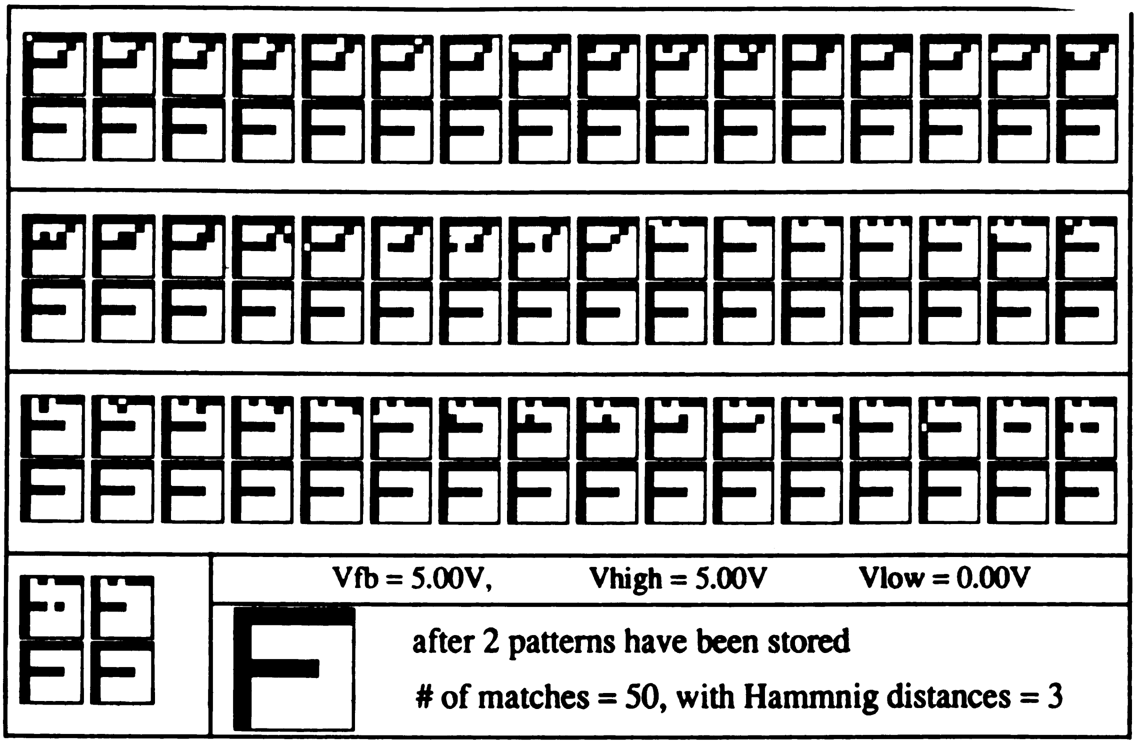

Vfb = 5.00V,
Vhigh = 5.00V
Vlow = 0.00V
after 2 patterns have been stored
# of matches = 50, with Hammnig distances = 3

Figure 7.3.1.a Experimental results of the 50-neuron chip;
all initial conditions with Hamming distances 3 from the character F.

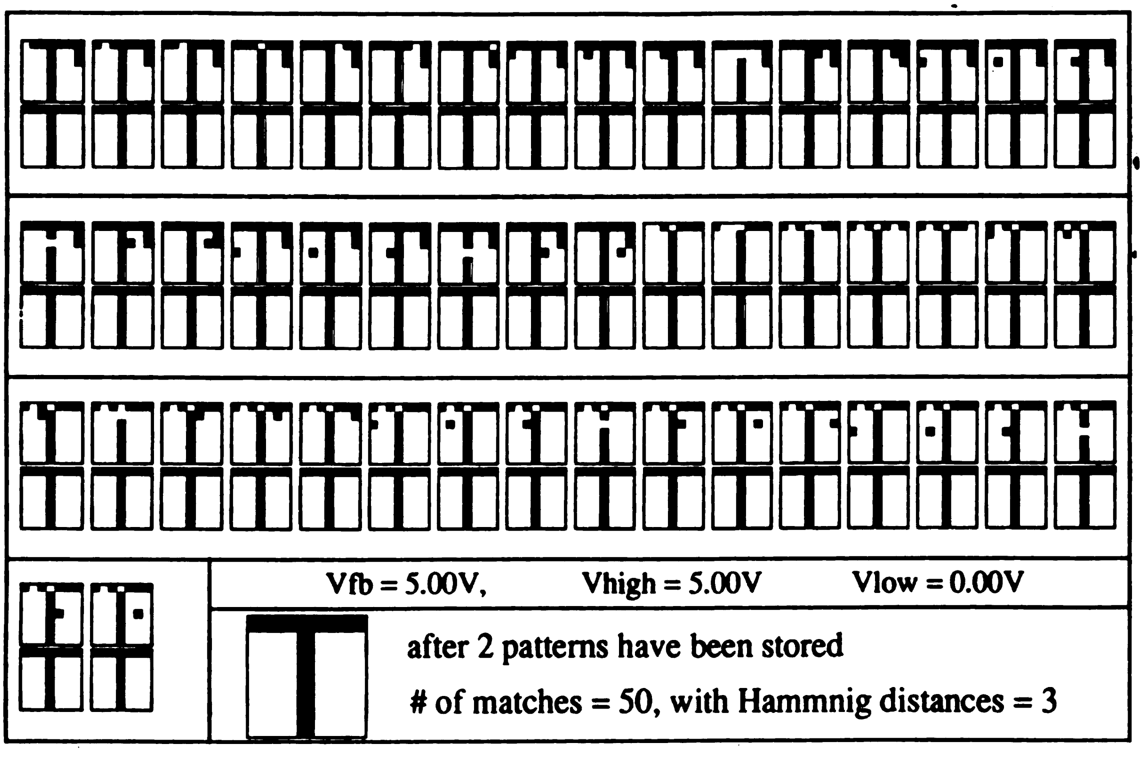

Vfb = 5.00V,
Vhigh = 5.00V
Vlow = 0.00V
after 2 patterns have been stored
# of matches = 50, with Hammnig distances = 3

Figure 7.3.1.b Experimental results of the 50-neuron chip;
all initial conditions with Hamming distances 3 from the character T.

CHAPTER 8

SUMMARY AND CONCLUSIONS

To make neural computing more powerful in terms of speed, the design and implementation of compact and electrically programmable artificial neural networks in hardware is becoming a necessity. With the massively parallel processing capabilities, VLSI neural circuits play an important role in the success of future high-performance, low cost computing machines.

8.1. Summary

In this research, theory and practical design examples for a new feedback neural network architecture with digital hardware on-chip learning scheme have been described.

This new neural network architecture is equally motivated from biological neural nets where neurons have dendro-dendritic connections, i.e., connections among neurons occurring via dendrites only. The new architecture is realizable via electronic circuits using a single MOSFET transistor for each dendro-dendritic synapse. The maximum number of connections is reduced to approximately one half of the maximum connections in the architecture of the Hopfield circuit. The dynamic behavior and characteristics of the new neural network architecture is supported by mathematical analysis. The new architecture preserves its gradient dynamic properties, including symmetry of the synaptic weights when they are implemented in the physical world of hardware.

Neural properties of a prototype of this new architecture circuits have been verified via extensive SPICE simulations and discrete-component experiments. The simulation and experimental results show that by only adjusting the gate voltages of the interconnect nMOS transistors, the network can possess all possible number of stable equilibrium states which in turn correspond to different number of memory or code data.

A 6-neuron Tiny-chip of the new architecture was successfully designed and fabricated using 2 μm CMOS n-well technology with a $2.2 \times 2.2\ mm^2$ chip size as a prototype chip of the new architecture for developing and assessing various potential off-chip learning algorithms. A specialized dynamic learning algorithm was successfully developed onto software environment with graphics display on a personal computer and tested on the 6-neuron Tiny-chip.

Eventually a 50-neuron CMOS analog chip with on-chip digital learning scheme was successfully designed and fabricated with a $6.8mm \times 4.6mm$ chip size having 63,025 transistors via standard $2\mu m$ CMOS n-well technology. A desired equilibrium point can successfully be stored into the chip by setting the interconnect weights via hardware learning circuitry within 50 ns. In addition, all initial conditions converge to the steady states in the order of 20 μs. From experimental measurement using a digital meter, the power consumption of the neural chip in steady state is less than 1 mWatt with power supplies of 0/5 volts. The neural circuit safely retains the same dynamic properties when the power supply high level is decreased down to 3 volts to dramatically reduce the power dissipation of the analog CMOS VLSI chip.

Dedicated interface circuitries and software environments had been built to successfully demonstrate the use of the prototype chips of the new neural circuit. A real-time application has been performed using the 50-neuron chip with on-chip digital learning as a coprocessor of a personal computer to demonstrate the applicability of the new architecture as a pattern/character associator.

On-chip learning is very important to compensate nonideal characteristics of VLSI neural chips and to reduce total system learning time. To the best of our knowledge, this is the only successful and effective (analog) neural chip experiment with guaranteed learning capability. These implementations and experimental results have been shown to perform and substantiate the capabilities attributed to artificial neural nets by pursuing an approach of analysis followed by direct implementations.

8.2. Conclusions

As a whole, this work lays a solid groundwork for the implementation of ANNs in both the theoretical and practical aspects. Further studies to improve the performance and to reduce the area of synapse cells and neuron cells are necessary to integrate a larger neural network into a single VLSI chip. Future research should emphasize advance development toward system integration for large and practical application problems such as pattern recognition and image processing.

BIBLIOGRAPHY

BIBLIOGRAPHY

[1] J. J. Hopfield, "Neural Networks and Physical Systems with Emergent Collective Computational Abilities," Proceeding of National Academy of Science U.S.A., vol. 79, pp. 2554-2558, April 1982.

[2] J. J. Hopfield, "Neurons with Graded Response Have Collective Computational Properties like those of Two-state Neurons," Proceeding of National Academy of Science U.S.A., vol. 81, pp. 3088- 3092, May 1984.

[3] J. J. Hopfield and D. W. Tank, "Neural Computation of Decisions Optimization Problems," Biological Cybernetics, vol. 52, pp. 141-152, 1985.

[4] J. J. Hopfield and D. W. Tank, "collective computation with continuous variables," in Disordered Systems and Biological Organization, E. Bienenstock, F. Fogelman, and G. Weisbush, eds., Springer-Verlag, 1985.

[5] J. J. Hopfield and D. W. Tank, "Simple Neural Optimization Networks: An A/D Converter, Signal Decision Circuit, and a Linear Programming Circuit," IEEE Transactions on Circuits and Systems, vol. CAS-33, no.5, pp. 533-541, May 1986.

[6] D. W. Tank and J. J. Hopfield, " Collective Computation in Neuronlike Circuits," Scientific American, pp. 104-114, December 1987.

[7] J. J. Hopfield, "Artificial Neural Networks," IEEE Circuits and Devices Magazine, pp. 3-10, September 1988.

[8] R. Rosenblatt, *Principle of Neurodynamics,* New York, Spartan Books (1959).

[9] R. O. Duda and P. E. Hart, *Pattern Classification and Scene Analysis,* John Wiley & Sons, New York.

[10] T. Kohonen, *Self-Organization and Associative Memory,* Springer-Verlag, 2nd ed. 1988.

[11] L. O. Chua and L. Yang, "Cellular Neural Networks: Theory," IEEE Transactions on Circuits and Systems, vol. CAS-35, no. 10, pp. 1257-1272, October 1988.

[12] L. O. Chua and L. Yang, "Cellular Neural Networks: Applications," IEEE Transactions on Circuits and Systems, vol. CAS-35, no. 10, pp. 1273-1290, October 1988.

[13] Y. Wang and F. M. A. Salam, "Experiments Using CMOS Neural Network Chips As Pattern/Character Recognizers," 1991 IEEE International Symposium on Circuits and Systems (ISCAS), Singapore, June 1991.

[14] F. M. A. Salam and Y. Wang, "A Real-Time Experiment Using a 50-Neuron CMOS analog Silicon Chip with On-Chip Digital Learning," IEEE Transactions on Neural Networks, Vol. 2, no. 4, pp. 461-464, July 1991.

[15] A. Agranat, C. Neugebauer, and A. Yariv, "A CCD Based Neural Network Integrated Circuit with 64K Analog programmable Synapses," the proceedings of IJCNN, 1990, pp. II-551-555.

[16] L. Jakel, *et al.*, "Hardware Requirements for Neural-net Optical Character Recognition," the proceedings of IJCNN, 1990, pp. II-855-861.

[17] S. Eberhardt, T. Duong, and A. Thakoor, "Design of parallel hardware neural network systems from custom analog VLSI 'building block' chips", the proceedings of IJCNN, 1989, pp. 183-190.

[18] W. Fisher, R. Fujimoto, and R. Smithson, "A programmable Analog Neural Network Processor," IEEE Trans. on Neural Networks, vol. 2, No. 2, pp. 222-229, March 1991.

[19] C. Mead, X. Arreguit, and J. Lazzaro, "Analog VLSI Model of Binaural Hearing," IEEE Trans. on Neural Networks, vol. 2, No. 2, pp. 230-236, March 1991.

[20] M. Holler, S. Tam, H. Castro, and Ronald Benson, " An Electrically Trainable Artificial Neural Network (ETANN) with 10240 Floating Gate Synapses," the proceedings of IJCNN, 1990, pp. II-191-196.

[21] W. Hubbard et. al., "Electronic Neural Networks," AIP Neural Networks for computing, pp. 227-234, 1986.,

[22] M. A. Sivilotti et. al., "VLSI Architectures for Implementation of Neural Networks," AIP Conf. Proc. 151, 408, 1986.

[23] H.P. Graf and P. Vegvar, "A CMOS Implementation of a Neural Network Model," Proc. Stanford Conf. Advanced Research in VLSI, MIT Press, pp. 351-367, 1987.

[24] J. P. Sage, et. al., "An Artificial Neural Network Integrated Circuit Base on MNOS/CCD Principle" AIP Neural Networks for computing, pp. 381-385, 1986.

[25] D. E. Van Den Bout and T. K. Miller III, "A digital architecture employing stochasticism for the simulation of Hopfield neural nets," IEE Trans. on Circuit and Systems, vol. 36, no. 5, pp. 732-738, May 1989.

[26] C. Mead, *Analog VLSI and Neural Systems,* Addison Wesley, 1989.

[27] D. Psaltis and N. Farhat, "Optical information processing based on an associative-memory model of neural nets with thresholding and feedback," Opt. Let. 10, 98 (1985).

[28] F. M. A. Salam and Y. Wang, "A new architecture for 2-D array neural nets with an electro-optical implementation", SPIE Vol. 977, Real Time Signal Processing XI (1988).

[29] F. M. A. Salam and Y. Wang, "A formulation for 2-D Array Neural nets for Electro-Optical Implementation", 1989 IEEE International Symposium on Circuits and Systems (ISCAS), Portland, Oregon, CA, May 9-11, 1989, pp. 478-481.

[30] F. M. A. Salam and Y. Wang, "A Learning Algorithm for Feedback Neural Network Chips," 1991 IEEE International Symposium on Circuits and Systems (ISCAS), Singapore, June 1991.

[31] F. M. A. Salam and S. Bai, "A New Feedback Neural Network with Supervised Learning," IEEE Transactions on Neural Networks, January 1991.

[32] Fathi M. A. Salam, S. Bai, and J. Hou, "Dynamics of Feedback Neural Nets with Unsupervised Learning," IEEE International Joint Conference on Neural Networks, San Diego, California, vol. II, June 17-21, 1990.

[33] F. M. A. Salam and M. R. Choi, "An All-MOS Analog Feedforward Neural Circuit With Learning", 1990 IEEE International Symposium on Circuits and Systems (ISCAS), May 1990, pp. 2508-2511.

[34] D. Rumelhart, G. Hinton, and G. Williams, "Learning Internal Representations by Error Propagation," in Parallel Distributed Processing, vol. 1, eds. D. Rumelhart and J. McCleland, MIT Press.

[35] J. S. Denker, "Neural Network Models of Learning and Adaptation," Physica 22D, 1986, pp. 216-232.

[36] D. O. Hebb, *The Organization of Behavior,* Wiley, New York, 1949.

[37] T. Kohonen and M. Ruohonen, IEEE Transactions on Computer C-22 pp. 701, 1973.

[38] R. P. Lippman, "An Introduction to Computing with Neural Nets," IEEE ASSP, pp. 4-22, April 1987.

[39] F. M. A. Salam, "A Model of Neural circuits for Programmable VLSI Implementation of the Synaptic Weights for Feedback Neural Nets," 1989 IEEE International Symposium on Circuits and Systems (ISCAS), Portland, Oregon, May 1989, pp. 849-851.

[40] F. M. A. Salam, "New Artificial Neural Models: Basic Theory and Characteristics" 1990 IEEE International Symposium on Circuits and Systems (ISCAS), New Orleans, Louisiana, May 1990, pp. 200-203.

[41] W. S. McCulloch and Pitts, "A Logical Calculus of the Ideas Imminent in Nervous Activity," Bulletin of Mathematical Biophysics, 5, 115-133, 1943.

[42] R. Rosenblatt, *Principle of Neurodynamics,* New York, Spartan Books 1959.

[43] B. Widrow and M. E. Hoff, "Adaptive Switching Circuit," 1960 IRE WESCON Conv. Record, Part 4, 96-104, August 1960.

[44] M. A. Cohen and S. Grossberg, "Absolute Stability of Global Pattern Formation and Parallel Memory Storage by Competitive Neural Networks," IEEE Transaction on Systems Man, and Cybernetics, Vol. SMC-13, No. 5, September/October 1983, pp. 815-826.

[45] S. Grossberg, *The Adaptive Brain, II: Vision, Speech, Language, and Motor Control,* North-Holland, Amsterdam, 1987.

[46] S. Grossberg, Eds., *Neural Networks and Natural Intelligence,* MIT Press,

[47] D. E. Rumelhart, J. L. McClelland, and the PDP Research Group Eds., "Parallel distributed processing-Explorations in the microstructure of cognition," vol. 1, Foundations. Cambridge, MA: MIT Press, 1986.

[48] T. H. Bullock, R. Orkand and A. Grinnel, *Introduction to Nervous Systems,* W.H. Freeman, San Francisco, 1977.

[49] L. Stryer, *Biochemistry,* W. H. Freeman and Company, 2nd ed., 1981.

[50] G. D. Pappas and D. P. Purpura, eds., *Structure and function of Synapses,* Raven Press, NY, 1972.

[51] S. W. Kuffler, J. G. Nichols and A. R. Martin, *From Neuron to Brain,* 2nd. ed. 1984.

[52] L. W. Massengill, "A Dynamic CMOS Multiplier for Analog VLSI Based on Exponential Pulse-Decay Modulation," IEEE Journal of Solid-State Circuits, vol. 26, no. 3, pp. 268-276, March 1991.

[53] F. M. A. Salam, N. Khachab, M. Ismail, and Y. Wang, "An Analog MOS Implementation of the Synaptic Weights for Artificial Neural Nets," Analog Integrated Circuits And Signal Processing, an international journal, Kluwer Academic Publishers, Vol. 2, 1991.

[54] P. W. Hollis and J. J. Paulos, "Artificial Neural Networks Using MOS Analog Multipliers," IEEE Journal of Solid-State Circuits, vol. 25, no. 3, pp. 849-855, June 1990.

[55] F. Kub, K. Moon, I. Mack, and F. M. Long, "Programmable Analog Vector-Matrix Multipliers," IEEE Journal of Solid-State Circuits, vol. 25, no. 1, pp. 207-214, February 1990.

[56] F. M. A. Salam, N. Khachab, M. Ismail, and Y. Wang, " An Analog MOS Implementation of the Synaptic Weights for Feedback Neural Nets," 1989 IEEE International Symposium on Circuits and Systems (ISCAS), Portland, Oregon, CA, May 9-11, 1989.

[57] Y. Wang and F. M. A. Salam, "Design of Neural Network Systems from Custom Analog VLSI Chips," 1990 IEEE International Symposium on Circuits and Systems (ISCAS), New Orleans, Louisiana, May 1990, pp. 240-243.

[58] F. M. A. Salam, Y. Wang, and R. Y. Choi, "On The Analysis and Design of Neural Nets," IEEE Transactions on Circuits and Systems, vol. CAS-38, no. 2, pp. 196-201, February 1991.

[59] C. Chiu, C.-Y. Maa, and M. A. Shanblatt, "Energy Function Analysis of Dynamic Programming Neural Networks," IEEE Transactions on Neural Networks, Vol. 2, no. 4, pp. 418-426, July 1991.

[60] M. P. Kennedy and L. O. Chua, "Neural Networks for Nonlinear Programming," IEEE Transactions on Circuits and Systems, vol. CAS-35, no. 5, pp. 554-562, May 1988.

[61] M. A. Mahowald and C. Mead, "The Silicon Retina," Scientific American, pp. 76-82, May 1991.

[62] F. C. Hoppensteadt, *an introduction to the mathematics of neurons,* Cambridge University Press, 1986.

[63] C. Tomovich, *MOSIS User Manual,* Release 3.1.

[64] F. M. A. Salam and Y. Wang, "Neural Circuits for Programmable Analog MOS VLSI Implementation", Proc. of 32nd Midwest Symposium on Circuits and Systems, Champaign, Illinois, August, 1989.

[65] F. M. A. Salam and Y. Wang, "Some Properties of Dynamic Feedback Neural Nets," in the session on Neural Networks and Control Systems, the 27th IEEE Conference on Decision and Control, December 1988, pp. 337-342.

[66] F. M. A. Salam, "Global convergence properties of a new feedback model for neuro-engineering," Memorandum No. MSU/EE/S 89/01, Department of Electrical Engineering, Michigan State University, East Lansing, MI 48824-1226, 25 January 1989.

[67] F. M. A. Salam, R. Y. Choi, Y. Wang, and B. Javidi, "On The Design of Neural Nets," 25th Annual Allerton Conference on Communication, Control,

and Computing, Sep. 30 - Oct. 2, 1987.

[68] F. M. A. Salam, R. Y. Choi, Y. Wang, "An Analog MOS Implementation of the Synaptic Weights for Feedback/Feedforward Neural Nets," Proc. of 32nd Midwest Symposium on Circuits and Systems, Champaign, Illinois, August, 1989.

[69] B. W. Lee, B. J. Sheu, and H. Yang, "Analog Floating-Gate Synapses for General-Purpose VLSI Neural Computation," IEEE Transactions on Circuits and Systems, vol. CAS-38, no. 6, pp. 654-658, June 1991.

[70] R. Shimabukkuro et. al., "Dual-Polarity Nonvolatile MOS Analog Memory Cell for Neural -type Circuitry," Electronic Lett., Vol. 24, No. 19, pp. 1231-1232, 1988.

[71] T. Borgstrom et. al., "A Neural Network Integrated Circuit Utilizing Programmable Threshold Voltage Devices," IEEE Proc. of ISCAS'89, pp. 1227-12300, 1989.

[72] R. Shimabukkuro et. al., "Circuitry for Artificial Neural Networks with Non-Volatile Analog Memory," IEEE Proc. of ISCAS'89, pp. 1217-1220, 1989.

[73] M. White et. al., "Electrical Modifiable Nonvolatile Synapses for Neural Network," IEEE Proc. of ISCAS'89, pp. 1213-1216, 1989.

[74] D. Schwartz et. al., "A Programmable Analog Neural Network Chip," IEEE J. of SSC, Vol. 24, No. 2, pp. 313-319, Feb. 1989.

[75] Y. Tsividis et. al., "Analog Circuits for Variable-Synapse Electronic Neural Networks," Electron. Lett., Vol. 23, No. 24, pp. 1313-1314, Nov. 1987.

[76] B. Lee, J. Lee, and B. Sheu, "VLSI Image Processors Using Analog Programmable Synapses and Neurons," The proceedings of International Joint Conference on Neural Networks, San Diego, California, June 1990, pp. II 575-580.

[77] F.M.A. Salam and S. Bai, "Complicated Dynamics of a Prototype Continuous-Time Adaptive Control System," IEEE Trans. on Circuits and Systems, Vol. CAS-35, July 1, 1988.

[78] Y. Wang and F. M. A. Salam, "VLSI-design and Testing of an Analog Programmable feedback neural circuit," Memorandum No. MSU/EE/S 89/09, Department of Electrical Engineering, Michigan State University, East Lansing, MI 48824-1226, 29 November 1989.

[79] F. M. A. Salam and Y. Wang, "Simulation , experiment, and VLSI-design of an analog programmable feedback neural circuit," Memorandum No. MSU/EE/S 89/02, Department of Electrical Engineering, Michigan State University, East Lansing, MI 48824-1226, 2 February 1989.

[80] Atul Dhablania, "Interfacing and Computer Simulations for the 6-*neuron* Tiny Chip," Project Report of EE801-2, Department of Electrical Engineering, Michigan State University, East Lansing, MI 48824-1226, December 1989.

[81] R. Beale and T. Jackson, *Neural Computing*, Adam Hilger 1990.

[82] Y. Arima et. al., " A Self-Learning Neural Network Chip with 125 Neurons and 10K Self-Organization Synapses," IEEE Journal of Solid-State Circuits, Vol. 26, No. 4, April, 1991.

[83] K. Mashiko et. al., "Silicon Implementation of Self-Learning Neural Networks," 1991 IEEE International Symposium on Circuits and Systems (ISCAS), Singapore, June 1991.

[84] G. Shepherd, *The Synaptic Organization of the Brain*, 2nd ed., New York: Oxford University Press, 1979.

[85] H. Graf, *et al.*, "A Neural-Net Board System for Machine Vision Applications," the proceedings of IJCNN, 1991, pp. I-481-486.

[86] B. Boser, *et al.*, "An Analog Neural Network Processor and its Application to High-Speed Character Recognition," the proceedings of IJCNN, 1991, pp. I-415-420.

[87] A. Agranat, C. Neugebauer, R. Nelson, and A. Yariv, "The CCD Neural Processor: A Neural Integrated Circuit with 65,536 Programmable Analog Synapses," IEEE Transactions on Circuits and Systems, Vol. 37, pp. 1073-1075, 1990.

[88] C. Neugebauer, and A. Yariv, "A Parallel Analog CCD/CMOS Neural Network IC," the proceedings of IJCNN, 1991, pp. I-447-451.

[89] H. Eguchi, *et al.*, "Neural Network LSI Chip with On-Chip Learning," the proceedings of IJCNN, 1991, pp. I-453-456.

[90] F. Kub, K. Moon, and J. Modolo, "Analog Programmable Chips for Implementing ANNs Using Capacitive Weight Storage," the proceedings of IJCNN, 1991, pp. I-487-492.

[91] P. Muller, *et al.*, "A General Purpose Analog Neural Computer," the proceedings of IJCNN, 1989, pp. II-177-182.

[92] P. Muller, *et al.*, "Design and Performance of A Prototype General Purpose Analog Neural Computer," the proceedings of IJCNN, 1991, pp. I-463-468.

[93] C. Schneider and H. Card, "CMOS Implementation of Analog Hebbian Synaptic Learning Circuit," the proceedings of IJCNN, 1991, pp. I-437-442.