Chapter 3

Very Large Scale Integration

Implementations of Artificial Neural Networks

3-1 Introduction

The new approach for information processing based on the implementation and execution of neural algorithms is called *neurocomputing*. Neurocomputing is the technological discipline concerned with parallel, distributed adaptive information-processing systems that develop information-processing capabilities for an information environment.

Conventional computers are based on the Von Neumann approach in which one instruction is executed at a time. The Von Neumann approach has been extended with the introduction of pipelining, array processing, and multiprocessing Single-Instruction-Multiple-Data (SIMD) and Multiple-Instruction-Multiple-Data (MIMD) architectures based Arithmetic Logic Units (ALU) to design fast parallel processors.

Parallel processors are also limited in their ability to solve real-time problems such as pattern recognition, speech recognition, optimisation...etc. Therefore, parallel network based on intelligent processors is proposed and thus a mimic of the human brain neural network (*Neurocomputer)* is in fact an ultimate alternative.

3-2 ANN in the Context of Computing Generation

There are some claims that ANNs represent the sixth generation in computing. The first to the fourth generations were characterised by the hardware advances (from the electromechanical relays to VLSI), and generation 5 was Artificial Intelligence AI. The assertion that ANNs represent a distinct computer generation is perhaps both arguable and premature. It is probably fair to say, however, that there is currently little consensus on how to exploit VLSI and Ultra Large-Scale Integration (ULSI) technology to achieve massively parallel ANN implementations.

We reiterate the remark that nature has already solved the scaling problem in implementing biological systems.

The entire field of ANN implementations, insofar as large-scale networks are concerned, is about 10 years old. At this time, many alternative implementation approaches are available, including optical, biological *wet-ware*, and electronic.

Specifically, efforts to achieve both practical simulator and dedicated hardware may be subdivided into three major objects:

1. Use of standard chips to design accelerator and coprocessor architectures ([1], [2]).
2. Use of supercomputers ([3], [4]).
3. Design of special-purpose *neurochips*, which may be analogue, digital or hybrid.

Neurocomputer building is typically very expensive, in terms of development time and required resources. Furthermore, the market for such devices is very unclear, especially in light of the continual evolution of new devices and neural architectures.

3-3 Artificial Neural Networks VLSI Design

The response and the characteristics of present models of artificial neural nets are primarily investigated by simulation on vector computers, workstations, special coprocessors or transputer arrays. The fundamental drawback of such simulators is that the spatio-temporal parallelism in the processing of information that is inherent to the neural net is lost entirely or partly. Therefore, the computing time of the simulated net especially for large associations of neurons grows to such orders of magnitude that a speedy acquisition of neural known-how is hindered or made impossible.

Figure (1) shows the performance obtainable with available commercially simulators [5] in term of implemented weights and executed (learned or processed) weights per second.

This must be confronted with the application needs. It becomes obvious that today's hardware capabilities are limiting the development of neural network research.

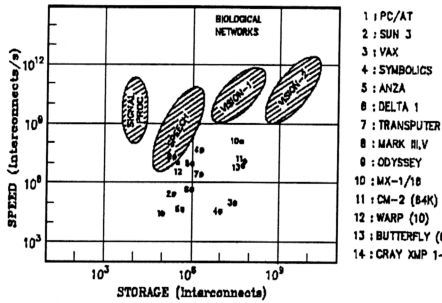


Figure (1). Synapse updating speedup of some commercial neuro-chips

An appreciable reduction in computing time and thus the handling of largish task or those that are to be executed in real-time become possible with specially designed neural hardware. Apart from the shortest possible computing time, neural hardware offers a very much smaller structural volume than can be implemented with simulators for the same task. This is especially important when neural hardware is to be incorporated in terminals for man-machine communication or mobile robotics.

3-4 VLSI Advantages

Depending on the application under consideration, the user would tell whether his problem is accessible by simulation on conventional computers or not. If yes, the application task under consideration can be well defined. Therefore, the user will specify

the kind of data format and the degree of weight resolution, the size and type of the network and the processing speed for the recall mode. If the real-time requirement cannot be satisfied by the software implementation, it makes sense to think about designing special hardware. Because the weights are computable in advance, there is no extra circuitry other than for programmable or hard-wired weights and discrimination. The designer task in this case is implementing the needed synapses, so that the pattern storage capacity increases with the number of implemented neurons and the computing time reduces linearly with the number of implemented synapses. Considering just one application area, namely signal processing, it has been demonstrated that the number of synapses required is of an order that can be implemented on a single chip with today's technology ([6], [7]). For small-scale [1] applications, there is thus the possibility of specific application neural chips with programmable or fixed weights (see Figure (2)).

The learning algorithm, of an application that is accessible to the simulation, only has to be considered in hardware terms if there is a relevant real-time requirement. The latter is imaginable, for instance if the learnt information valid for a short time and new learning is repeatedly. Obviously, supporting the learning task will be possible at the expense of the number of synapses implemented, and it has to be checked whether single-chip integration will be possible at all. Wafer Scale integration may be turn out to provide the integration potential for computing and storing synaptic weights and intermediate results produced by the learning algorithm ([8], [9]). An alternative popular proposal is to distribute the neural net implementation plus the learning algorithm over several chips and cascade them.

[1] In terms of the number of synapses required.

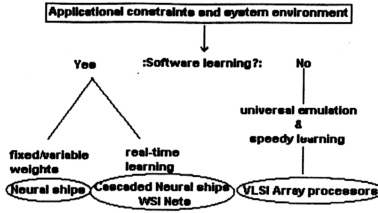


Figure (2). Specific neural chips with programmable or fixed weights

Neural nets for applications like vision or speech, on the other hand, overtax the single-chip integration potential of present technology as well as that of the future 0.3 μm technology by whole orders of magnitude (see Figure (1)). Particularly the weights will have to be stored off-chip. The size of such a net will not permit simulation (especially of the learning phase) within a reasonable period and therefore the weights cannot be determined by simulation. This means that little or no engineering expertise has been accumulated for these large applications. Consequently, the VLSI architecture must be designed for universal emulation of neural network structures and speedy learning. VLSI architectures of this type will obviously look different from those considered in small-scale applications.

3-5 Single-Chip Integration

Single-chip integration of neural net means on-chip storage of the weights (not more than few ten thousands of synapses today) [6]. Consequently, the amount of information processed in a neural chip will be limited in size. This means in turn that the application

must match the implementation potential of the technology. Fortunately, the learning for neural nets that can still be integrated on a chip can be performed on conventional computers with reasonable time expenditure. Therefore, the application task under consideration can be well defined and the hardware matched optimally to the task.

On one hand, on-chip storage of weights offers an easy way to achieve real-time action by neural networks, since there is no pad-bandwidth problem for the weights. On the other hand, VLSI technology faces the interconnection problem, irrespective of which design style (analogue or digital) or which technology is used ([10], [11]). If several thousands of weights were to be connected physically to a neuron and some thousands neurons to be implemented, the wiring area would grow drastically. It can even grow to such orders that the delay on the wires tends to exceed the latency time of the functional block representing a neuron.

In principle there are two ways to overcome the interconnection problem: firstly, by reducing the technological structure size, and secondly, by architectural means. The first way dominated the early era of neural hardware design was by H. P. Graph [12], C. A. Mead [13], and A. P. Thakkoor [14]. Nowadays, the second course is followed preferably, since architecture is the cheapest. This method has taken the most interest not only because of its expense only but also because of the availability of the array architectures such as the SIMD and MIMD on the dedicated parallel computer [15]. As a rule, a designer should check first for the required processing time (real-time conditions) of the application under consideration. Then, think out to what extent it is possible to form the ideal massive parallel networking of a neural net. Together with the decision of as to whether the analogue or the digital signal processing concepts is to be applied and

the selection of the type of technology to be implemented, will result an initial architecture draft. Instead of reviewing architectures, one can conclude that the technology and the signal processing concepts influence the design of the architecture whatever it may look like.

3-5-1 Analogue Design Implementations

This kind of implementation is considered be the closest implementation to the biological networks because of the signals continuity in both systems. Another property exploited in the implementation of analogue ANN is the temporal integration of the input signals. The biological cell, which deals mostly with the integration of the input, signals, i.e., its functionality is based on frequency (pulse stream) of incoming signals. An analogue implementation therefore, is the best candidate to mimic this behaviour ([16], [17], [18]), based on standard (Complementary Metal Oxide Semiconductor) CMOS technology. Other attempts were investigated by exploiting the relation between the current and the voltage in the CMOS device sub-threshold region ([19], [20]).

Particularly for analogue realisation, the emphasis of circuit design is on the exploitation of the functional properties inherent in the elements of the basic circuit [21]. Generally the architectural draft needed for the analogue implementation of a neural net concentrates on representing the synaptic weighting, the neural ignition response (discriminator function) and the controlling of data input and output. The important requirement is the compactness of the connection element, because the cell size mainly determines the overall area of the network.

Furthermore, the analogue implementations have the capability of processing more than 1 bit per transistor. If this benefit is to made use of, however, the following problems have to be mastered:

1. Non-volatile storage of analogue weights provides very high synaptic density, but may not be sufficiently often programmable[2].
2. The design of a synapse, the size of a neural network and the degree of analogue resolution are dependent on each other [22].
3. A major design problem with analogue circuits is the relation of accuracy to chip area. The more precisely one wishes to control the matching of the analogue components, the more chip area is needed. An analogue depth of not more than eight bit is recommendable. Crosstalk and susceptibility coupled in interference make special precautions necessary for analogue signal processing.
4. The minimal chip area is also influenced by many factors like noise and current consumption. Low current consumption calls for high-valued resistors in resistor-capacity circuitry. Low noise creates certain limits for minimal transistor Q surfaces and capacitors; this applies to switch-capacitor as well as to resistor-capacitor technique.
5. The temperature dependence, clock feed-through and process-parameter dependence can be reduced in analogue circuits so that they no longer interfere, but this is done at the expense of circuit complexity and has an effect on the chip area.

[2] *Programmable* here means the ability of changing the synaptic strength values known as *Online Learning*. For further reading see the book of U. Ramacher and R. Ruckert, VLSI Design of Neural Networks, Kluwer Academic Publishers 1991, 3rd chapter, Analog storage of adjustable synaptic weights, by E. Vittoz et al.

6. With future 0.3 μm transistor channel length, a lower supply voltage than 5 V must be expected, so questions of low-voltage design have to be considered. Accurate transistor modelling, innovative circuit techniques and design cleverness will be significant here like in the case above.
7. The limited precision in grading the weights (realised, for example, in the form of ohmic resistance or switch capacitors [23]) means, on one hand limited computing accuracy for an analogue implementation. Therefore, this influences the number and complexity of patterns that can be reliably processed with an analogue net. This applies equally on the selection of the discriminator: the computing accuracy of the entire analogue chip has to be considered.
8. The limited precision in processing information by the analogue neural net means, on the other hand, that the information must be encoded in a redundant or fuzzy fashion, i.e. only as sharply defined as is necessary for secure recognition.
9. If a learning algorithm is to be implemented in analogue circuitry, it is necessary to ensure a fortiori whether the intended application can at all be learnt.
10. Therefore, the implementation engineer has to characterise the tasks of pattern processing in which deviations of the actual weight values from the pre-computed ones can be tolerated and where it is possible to make do. The latter poses the least problems to the analogue designer.

In the analogue implementation of neural nets, it is consequently a matter of bringing together the application-oriented problem analysis and the circuit architecture; this is the only way to determine the application spectrum that can be implemented with analogue

hardware. Isolated definition of the effort for the analogue multiplication would be as inappropriate on the chip designer part as the weights determination on the user part.

3-5-2 Digital Design Implementation

A guiding principle in implementing neural algorithms is that if a particular network suggests a certain structure for its solution, an efficient computer implementation may be one that reflects that structure. For example, if the processing algorithm is based on the calculation of the local unit properties, a logical problem decomposition and associated architecture might be a parallel computer architecture in which each processing element independently processes neighbourhood unit data. For implementing a general network, one has to look for general techniques, which can support their inquiries. In the strategy of the digital implementation of ANN, one can divide the multitude of the currently implemented circuits under two major implementation strategies:

1. The architectural-operational implementations.
2. The enhancing capabilities implementations.

The first category concerns the different ways of connecting the neurons in their operational processing methodology. The interconnections in the implemented circuits are not the same as for the ANN but they are an artificial way to imitate the later parallelism. Therefore, the meaning of the parallelism in the implementation of neural networks is to be clearly defined in all levels. The notion of parallelism has two major definitions: algorithmic parallelism and data parallelism. *Algorithmic Parallelism* (AP) involves decomposition of an operation or algorithm into component operations, which may be executed in parallel. S. Y. Kung [24] has shown that most neural models are parallelizable even with architectures such as the back-propagation. The major

consequence of paralleling the ANN based on a thorough understanding of explicit and inherent parallelism in neural models is the design-effective and real-time processing hardware. *Data parallelism* (DP) (or also physical architecture parallelism) involves decomposing input data into partitions over which the operations may be carried out independently in parallel fashion especially when these operations are repeated many times. Most neural algorithms involve primarily those operations that are repetitive and regular such as multiplication and summation. For these class or algorithms, an attractive and cost-effective architectural choice is the *array processors*, which use mostly local interconnection network. This paves the way for massively parallel processing that represents the most viable future solution to real-time neural information processing. Systolic arrays (see Figure (3)) are the most widely used array processor architecture beside their derivatives such as the toroidal and the ring architectures for the implementation of digital ANN.

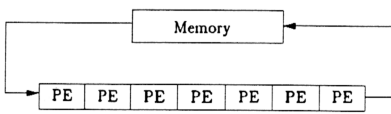


Figure (3). Basic configuration of systolic arrays

Systolic architectures ([25], [26]) are a class of pipelined array architectures. They are computing networks possessing the ability to parallel the computation in the most economical and fast way. According to Kung and Leiserson [25], "a systolic system is a network of processing elements (PE) which rhythmically compute and pass data through the system". For example, it can be shown that for some basic *inner product* PE can be

locally connected together to perform digital filtering, matrix multiplication, and other related operations. Therefore, the digital implementation of artificial neural networks using this technique consists in replacing the processing elements PE by digital neurons. The systolic array features the important properties of modularity, expandability (or extendibility), regularity, local interconnection, a high degree of pipelining, and highly synchronised multiprocessing. The data movements in systolic array architecture are often described in terms of the snapshots of the activities. A digital implementation of ANN using the linear systolic array adapted for both the backpropagation and the counter propagation architectures has been implemented by Marchesi et al [27]. Another but more exciting implementation of artificial neural network using the systolic array architecture is the work of S. Eun et al., [28]. They implemented systolic array architecture, and demonstrated that it can overcome the divergence of the parallel updating of the Hopfield network without loosing the speedup of the architecture. In this implementation, the neurons are not updated in parallel but they do the calculations by exploiting the power of this architecture in which they are serially updated avoiding the divergence of the system. It was proved that this architecture could achieve a linear speedup as the number of processors is increased. The two-dimensional systolic arrays, a kind processor array architectures, have been used to implement the Hopfield network [4]. Other systolic derivative architecture, which is more automatic and dynamic, is the ring systolic array also called the toroidal architecture. The toroidal implementation of artificial neural network was suggested by S. Jones et al., [30] to emulate a fully connected feedback neural network as well as a feed-forward neural network (see Figure (4)). The synaptic strengths are stored on shift register which presents successively the

corresponding synaptic strength to the input signal that is also stored on a shift register. The basis of operation revolves around the vertical axis, whereas the weights associated with each state revolve around the horizontal axis. Figure (5) presents a schematic diagram of the toroidal architecture of a feed-forward network composed of three layers using the toroidal architecture.

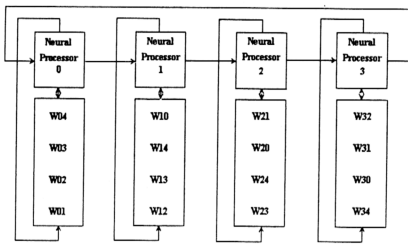


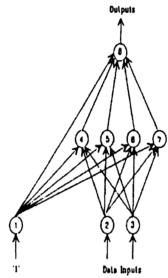
Figure (4). Toroidal architecture to emulate four fully interconnected neurons.

In brief, the array processor architectures deeply benefited neural networks digital implementation because of their important common property of *Parallelism*.

Another fact that raised the interest on the digital implementation of ANN is the great progress achieved by the integration technologies such as the VLSI and the ULSI (10^6 transistor on a chip). Therefore, exploiting the advanced power of the integration technologies and the advanced architectures of the array processor will certainly enable further digital implementation of more complex ANN.

The above-mentioned properties such as the reconfigurability and the expandability are cornerstones of almost all of the architectural implementations for constructing a

readaptable architecture of any computational algorithm. In this way, Raggad and Jin [31] implementation is worthy to be mentioned. They proposed a Basic Neural Unit (BNU) to be the foundation of reconfigurable architecture for constructing any ANN.



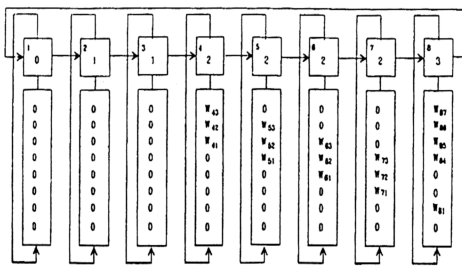


Figure (5). The toroidal architecture emulating three layer feed forward NN
Layer '0' = Bias Node. Layer '1' = Input Buffer Layer.
Layer '2' = Hidden Layer. Layer '3' = Output Layer.

Beside the architectural criterion of the implementation of ANN, the internal structure of the basic neuron adapted for its different phases (learning and processing) is the other

strategy characterising any digital implementation attempt. The goal looked through this strategy is the hardware and timing cost saving when implementing the different neural operations. Particularly, the multiplication between the inputs and their corresponding weights is considered the most hardware consuming and slow operational part of the neuron circuitry. This problem was considered by many authors replacing the classical digital multiplication by other techniques. An alternative method used in the digital filter design consists of reducing multiplications cost, using integer weights whose values are power of 2 or sums of power-of-two. This digital technique was originally proposed by Y. C. Lim and B. Liu [32] and exploited by B. A. White et al [33] in implementing the multiplication operations. This method consists of substituting the multiplications with simple shifts or shift-and-add operations, which are faster and require less hardware. The previous work of Y. C. Lim and B. Liu were also exploited by M. Marchesi et al [34] to implement a digital feed-forward neural network. Consequently, the implemented chip presented the two major goal of the implementation of the ANN, saving the chip area and computation time in agreement with a similarly previous work [35]. Despite these results, this implementation has some restriction because of the limited range of values used in processing and because of the sensitivity of the learning rule (backpropagation). These disadvantages restrict this implementation for digital signal processing of wide-band signals, non-linear channel equalisation and others.

The scalability and the power dissipation are important parameters influencing the performances of the digital ANN implementation. Furthermore, because of the fragmentation of the real problems into many processors, the need of large-scale implementation of processing neurons is unavoidable. The implementation containing

huge number of neurons distributed in a parallel architecture to achieve best timing performance has to reduce power dissipation. This increase in the number of processing elements affects certainly the power dissipation of the system (chip) and probably the processing time too. Therefore, solutions were investigated to overcome this problem. Firstly, the internal structure of the neuron is to be changed to reduce the internal hardware cost of the neuron. This is achieved by two ways. The first is to reduce the hardware cost of the neuronal arithmetic operations mainly the multiplication. The other way is to use neuronal general architecture reducing hardware cost and emulating most of the ANN functionalities. Second alternative is the use of more economical technologies, namely the CMOS technology. Under this implementation philosophy comes the work of K. Uchimura et al [36] trying to implement high-speed digital neural network chip in which they take profit from the two mentioned ways of implementation visions. In their attempts, they implement the Polyhedric Discrimination Neuron (PDN), which is a vector distance type based on the Radial-Basis Function (RBF) with a slight difference concerning the distance measurement. The schematic of the PDN model basic unit and its architecture are shown in Figure (6) and Figure (7). It is verified [37] that the RBF can perform similar tasks like the feed-forward network such as classification and generalisation.

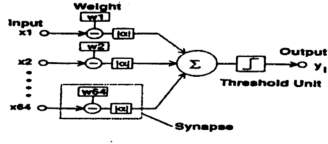


Figure (6). Schematic of single PDN neuron.

However, the vector-distance neuron features such as fewer interconnections or additional learning have most important characteristics for large-scale neural networks.

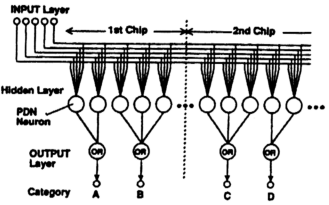


Figure (7). Architecture Schematic of the PDN network model.

Furthermore, the simplicity of the PDN model and future VLSI technologies will make possible large-scale neural network chip. The power dissipation of the above mentioned implementation is increasing steadily with increasing neuron density. Figure (8) shows how much it has increased with the miniaturisation of conventional digital chips.

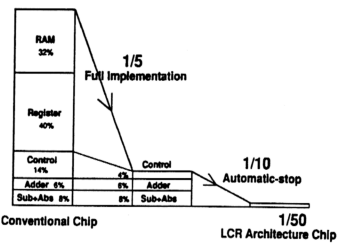


Figure (8). Chip power dissipation versus CMOS process technology.

The current parallel processing chips are fast approaching the upper limit of allowable power dissipation. Therefore, low-power technology is needed to implement large-scale implementation such as that for ANN. In order to make the first steps in the low-power technology the authors of the current implementation proposed the use of architecture implementation for this purpose. However, they developed the Low-power Chain-Reaction (LCR) architecture for the PDN network. The LCR architecture consists of three circuit techniques. The first is an automatic-stop technique for arithmetic circuits. In the PDN model, the summing operation (see Figure (9)) is stopped at the end of a transient region that is in the transfer function curve of the neuron. The second technique is a fully implemented digital synapse (see Figure (10)).

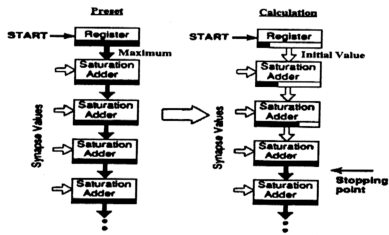


Figure (9). Primitive LCR Operating Diagram.

All synapse units have a calculating circuit and embedded weight memories to cut down the power dissipation in memory access, bus drivers, and register operation. The third technique is a self-controlled operation without internal clocks. Automatic-stop operation is controlled by the carry signals of the summing adders using control gates on

signal paths. With this method, there is no problem with fast internal clock generation and synchronisation. The authors concluded that the use of the PDN model and the LCR architecture could make possible large-scale neural networks of 10,000 neurons or more, because a 10,000 neurons system dissipates only 54 *W* while maintaining high-speed performance.

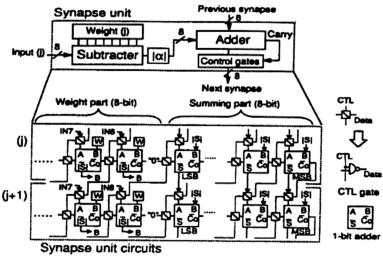


Figure (10). Circuit schematic of the synapse units.

Figure (11) shows how the LCR architecture based on the PDN neuron can reduce the power dissipation of the chip comparing with conventional chips.

Finally, another simplification of the hardware is mentioned in the same sense of minimising the implementation coast, which consists of reducing the size of the summation of the product circuit.

This technique consists of using the Fast Fourier Transform for the precedent operation but restricted to the backpropagation algorithm. This method combined with the distributed arithmetic technique have been found to be a promising technique tools [38] not only to minimise the hardware coast but also as a solution for many real-time pattern

recognition problems.

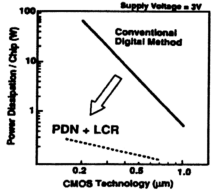


Figure (11). Reduction ratio of the chip power dissipation by the LCR architecture.

Architectural strategy in the algorithmic sense [3] combined with the parallel pipelined array processors in the hardware sense will represent the essential techniques for future ANN implementations. The ultimate objective is the spread of the ANN circuits as powerful intelligent real-time processors by reducing the power dissipation of the implementations and improving their timing, and hence the CMOS technology will be unavoidable. An interesting implementation in the same sense as the above philosophy and goals worthy to be mentioned was realised in the work of T. Watanaba and his group [39]. Finally, the most important observation that may and deserve to be the reason of the power of the biological neural networks is the importance of the architecture. This major factor is in fact a twice-important factor of the ANN hardware implementation because of its influence on the hardware-power dissipation and timing. Any improvement in this factor will certainly play the major role in the future implemented ANN circuits.

[3] In the sense of processing distribution progress