Regular Brief Papers

An Hierarchical VLSI Neural Network Architecture

R. Mason, W. Robertson, and D. Pincock

Abstract—**As neural network systems are scaled up in size it will become extremely difficult, if not impossible, to maintain full connectivity. This paper describes a digital architecture which exhibits hierarchical connectivity similar to that observed in many biological neural networks. At the lowest level, clusters of fully connected neurons correspond to subnetworks. These subnetworks are then sparsely connected to form the complete neural network system. The architecture exploits the inherent density and large bandwidth of on-chip RAM and can use either a large number of bit-serial processors or a reduced number of bit-parallel processors. A prototype chip which implements a complete subnetwork has been fabricated in 3-μm CMOS and is fully functional.**

I. Introduction

NEURAL NETWORKS are undoubtedly one of the fastest growing areas of research today. The understanding and modeling of neural networks and the application of these concepts is both challenging and exciting. One of the most promising technologies for implementing artificial neural networks (ANN's) is VLSI. The requirement for multichip (wafer) networks presents a problem in that there is a large discontinuity in the levels of connectivity on chip and off chip. As we scale up the size of our networks we are no longer able to support high connectivity without paying a severe performance penalty for multiplexing. The result is a desire for hierarchical networks in which we have highly connected subnetworks on chip, with reduced connectivity between these subnetworks.

There is an obvious need to develop new hierarchical models, whether they be based on older models or incorporate totally new principles. In a previous publication the authors addressed the problem of building hierarchical models [1]. In the following discussion they will present a hardware architecture that will serve as a substrate for hierarchical models.

In Section II the authors describe a 3-μm CMOS implementation of the architecture using bit-serial processing and communications. Section III summarizes the fabrication results, and in the final section concluding remarks are presented along with future work.

II. Neural Network Architecture

The majority of electronic ANN implementations can be placed in one of three categories: single processor systems that rely on a complex high-speed processor to achieve high performance [2], [7], multiprocessor systems with a small number of complex processors per chip [3], [4], [7], and multiprocessor systems with a larger number of simpler processors per chip [5]–[7]. It is with this latter category of systems that we will concern ourselves.

Given the present and near-term state of VLSI technology, the number of processors that can be implemented on a chip or wafer dictates the use of multiple chips for large networks. However, many of the VLSI implementations to date take advantage of the high on-chip connectivity while ignoring the off-chip connectivity constraints [5]–[7].

The architecture we have developed contains subnetworks of highly connected neurons which correspond to chips. Although there are a reduced number of connections between chips, the judicious use of multiplexing and feedthrough structures maximizes their effectiveness.

A. Subnetwork

Our architecture supports the use of many subnetworks with high internal connectivity and relatively sparse external connectivity. Fig. 1 shows a block diagram of a single prototype subnetwork that was implemented in 3-μm CMOS and contained 12 neurons.

One of the key attributes of the architecture is the use of 256 × 5 on-chip RAM for the WEIGHT MEMORY. By placing this memory on-chip we eliminate off-chip delays and can take advantage of the larger (80 b) internal memory bus width.

The PROCESSOR section generates a weighted sum of 12 on-chip inputs and four off-chip inputs for each neuron one at a time. It is composed of 16 serial/parallel multipliers which receive 5-b parallel weight values (1 b for sign) and 8-b serial inputs. The outputs of the multipliers are then accumulated using a tree of 15 one-bit adders. The final 16-b weighted sum is then passed to the TRANSFER MEMORY.

The transfer memory is a 64K × 8 off-chip (dashed box) table look-up memory which applies an arbitrary transfer function to the weighted sum of the inputs, thereby generating an 8-b neuronal output.

The output of the transfer memory is fed into one buffer of a 2 × 8 × 12 DOUBLE BUFFERED MEMORY. The other buffer contains the previous output states which are used by the processor to generate the new outputs. Once a complete set of outputs has been generated, the buffers switch.

Manuscript received September 12, 1989; revised July 16, 1991.
The authors are with the Technical University of Nova Scotia, Halifax, Nova Scotia B3J 2X4.
IEEE Log Number 9103862.

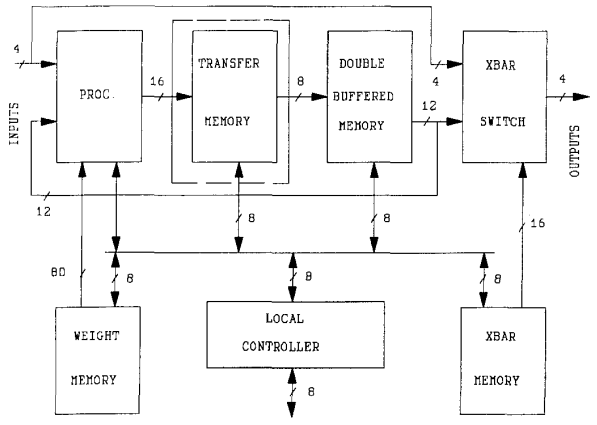

Fig. 1. Block diagram of prototype subnetwork implemented in 3-μm CMOS.

In addition to being fed to the processor section, the previous outputs pass serially through a 16 × 4 crossbar (XBAR) SWITCH (each of the four outputs can be any one of the 16 inputs). The crossbar switch, under the control of the 12 × 16 crossbar (XBAR) MEMORY, selects on a cycle-by-cycle basis which outputs are to be passed off chip.

This multiplexing of outputs maximizes connectivity as each neuron can now have a different set of off-chip inputs in addition to inputs from any of the neurons within the subnetwork. The crossbar switch outputs can also be independently tri-stated which provides additional flexibility for busing structures between subnetworks. Another feature is four feedthrough connections that pass directly from the inputs through the crossbar switch to the outputs. This allows us to make a limited number of global connections by passing neuron outputs through one or more subnetworks.

The local controller provides overall control of the subnetwork and also provides an interface to the global controller for functions such as downloading weights, downloading crossbar memory, and reading neural outputs.

B. Network

At the highest level, subnetworks may be connected in various topologies that would be tailored to the specific application. A single-bus architecture (see Fig. 2(a)) would provide direct connectivity between all the subnetworks. However, as the number of subnetworks grows there would be undue competition for the available connections. A mesh architecture (see Fig. 2(b)) would provide high connectivity to nearest neighbors with connectivity to nonneighbors dependent on available feedthrough paths.

The global controller is responsible for host interfacing. Through the control bus it supervises all the local (subnetwork) controllers. External inputs to the network can be fed directly to the inputs of subnetworks or can be allocated positions in the previous output buffer and downloaded through the global and local controllers.

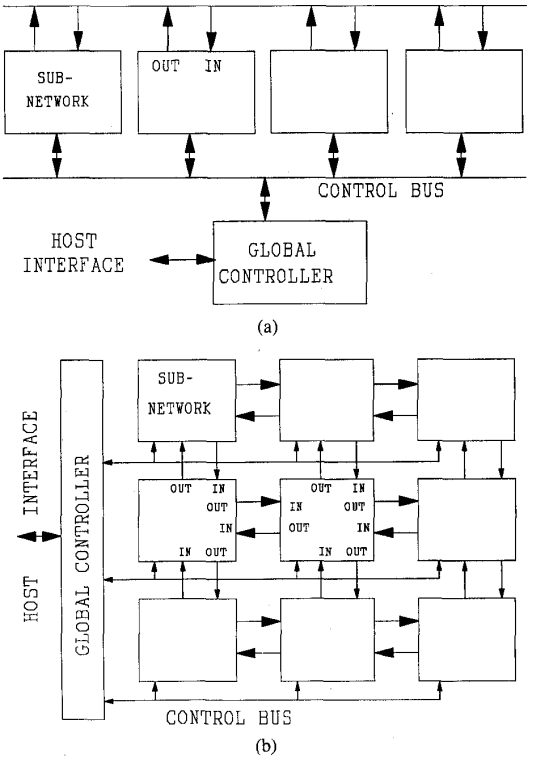

Fig. 2. Block diagram showing possible high-level network topologies: (a) conventional single-bus architecture, and (b) mesh architecture.

C. Learning

In most learning procedures a large part of the calculations involves multiply and accumulate operations. For the above implementation the intent is to use the parallel hardware as a learning accelerator.

The procedure would consist of the host downloading weights and inputs, the subnetworks performing the multiply–accumulate operations, and the host then uploading the results before performing the remainder of the learning algorithms. In this type of configuration the control bus becomes the obvious bottleneck, however, multiple control buses are possible as seen in Fig. 2(b).

Applications that require maximum learning speed dictate the use of more complex on-chip processors. Given the wide variety of learning procedures, design of the processing sections involves numerous trade-offs, especially when a number of different models are to be run on the same type of hardware. This is presently one of our key areas of activity.

III. Fabrication Results

The final design has over 27K transistors and occupies a 195-mil × 180-mil die. The chip was fabricated through the Canadian Microelectronics Corporation implementation service and Northern Telecom Canada (see Fig. 3).

Subsequent testing on a batch of five test chips revealed that two were fully functional. Due to overloading on two critical nets, the maximum operating frequency was lim-

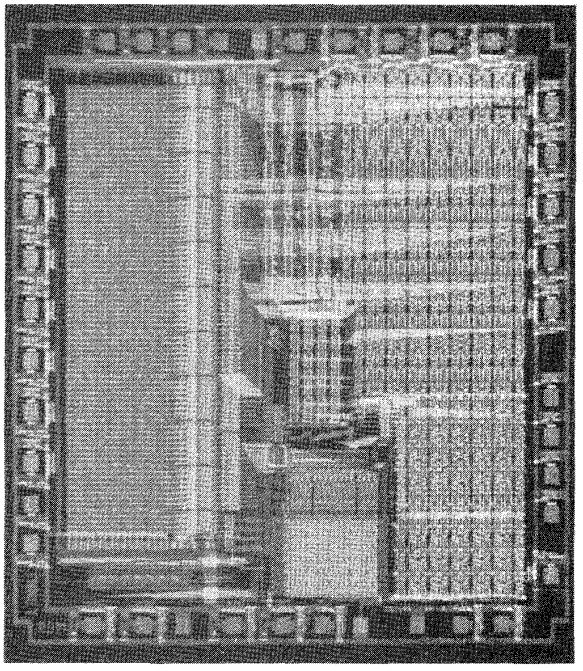

Fig. 3. Microphotograph of prototype subnetwork.

ited to 5.3 MHz as compared to the maximum 10-MHz simulation frequency (room temperature). This gave a overall nonlearning performance of 7.1-million connections per second (CPS) for this small prototype chip. We are presently evaluating the learning performance for a number of common models.

IV. Conclusions and Future Work

Although our first implementation of the architecture is modest in scale, it has given us a practical demonstration of the architecture and insights into future implementation details. It has also laid all the groundwork for control structures and interfaces for larger systems. Our estimates indicate that a 1K-neuron subnetwork with 100 inputs and outputs and a nonlearning performance of over 2.3-billion CPS could be incorporated on a single chip (assuming 64-Mb DRAM technology, 8-b weights, and a 25-MHz clock).

This compares favorably to the 1.6 GCPS quoted for the CNAPS chip from Adaptive Solutions Inc. [8]. There are, off course, major differences in the architectures in terms of processing functionality, number of on-chip weights, etc.

We are presently constructing a board-level network that will incorporate 16 subnetworks, as well as the global controller and host interfaces. In addition, we are evaluating a number of learning procedures and models to determine optimum processing structures and subnetwork topologies. This will be driven initially by our own applications with an attempt to maintain some generality.

References

[1] R. D. Mason, Ph.D. dissertation (submitted), Tech. Univ. Nova Scotia, Halifax, Canada, 1991.

[2] G. Works, "The creation of delta: A new concept in ANS processing," in *Proc. IEEE Int. Conf. Neural Networks*, Mar. 1988, pp. 41–49.

[3] W. B. Feild and J. K. Navlakha, "Transputer implementation of Hopfield neural network," presented at the IEEE Int. Conf. Neural Networks (poster session), 1988.

[4] D. A. Pomerleau, G. L. Gusciora, D. S. Touretzky, and H. T. Kung, "Neural network simulation at warp speed: How we got 17 million connections per second," in *Proc. IEEE Int. Conf. Neural Networks*, 1988, pp. II-143–II-150.

[5] A. F. Murray and A. V. W. Smith, "Asynchronous VLSI neural networks using pulse-stream arithmetic," *IEEE J. Solid-State Circuits*, vol. 23, no. 3, pp. 688–697, June 1988.

[6] J. Cleary, "A simple VLSI connectionist architecture," presented at the IEEE Int. Conf. Neural Networks, San Diego, CA, 1987.

[7] *Darpa Neural Network Study*. Fairfax, VA: AFCEA International, 1988.

[8] H. McCartor, "Back-propagation implementations on the adaptive solutions neurocomputer chip," in *Advances in Neural Information Processing System II*. San Mateo, CA: Morgan Kaufman, 1990.