Silicon Implementation of Pulse Coded Neural Networks,
Mona Zaghloul, Robert Newcomb, Jack L. Meador, (Eds),
Kluwer Academic, to appear 1993

A Digital Neural Network Architecture Using Random Pulse Trains

Gamze Erten
gamze@electra.caltech.edu

Rodney M. Goodman
rogo@caltech.edu

Department of Electrical Engineering, 116-81
California Institute of Technology
Pasadena, CA 91125

Abstract

A digital neural network architecture generating and processing random pulse trains, along with its unique advantages over existing comparable systems is described. In addition, test results from the VLSI implementation of its multiplication scheme are presented. These indicate that the implementation performs robustly and accurately.

1 Motivation and References to Previous Work

It has been shown that random pulse trains have interesting properties, which make them suitable for many kinds of computations [1][4]. Under certain restricted conditions, they can do quite complex computation with simple processing units, using probability and time to advantage. This is quite appropriate for VLSI implementations of neural networks where the high switching speeds of devices can be traded in for small processing units to implement massively parallel architectures. Thus, it is not surprising at all that random pulse trains have already been applied to hardware implementations of neural networks. Their properties have been exploited both in noise generation for Botzmann learning in Bellcore's stochastic learning microchip[5] and in computing the transfer function of neural processing units in Neural Semiconductor's Digital Neural Network Architecture (DNNA)[2].

In this paper, we propose a new digital architecture utilizing random pulse train properties, which combines several features of previous implementations: First, our architecture utilizes, as DNNA does, properties of uncorrelated pulse streams for computing the transfer function of neural units. Second, it also exploits pseudorandom bit streams generated by feedback shift registers for creating uncorrelated pulse streams, as does the Boltzman learning machine from Bellcore. However, our proposed architecture offers several features not present in these previous architectures. One significant enhancement is input amplification, which was not available in DNNA. Absence of amplifying ability could be a serious impediment to learning. This implementation affords amplification, while preserving the feature of gradual saturation of any activity at the level of a stuck-high signal, and thus preventing unbounded weight growth during training. Another additional feature is the elimination of correlation via uncorrelated sampling between layers of neural units, which we believe to be a more elegant and condensed hardware solution to de-correlation. De-correlation is essential because the crux of the principle which makes random pulse trains work is that no correlation exists both between the activity of computing elements and between the weights and the activities they selectively pass or block. We term this selection operation **sculpturing**. Maintaining the uncorrelation between layers is very difficult to accomplish while it is indeed a contradiction in terms. Since activity at each computing unit is obtained from a non-linear sum of common units from the previous layer, activities between units of the same layer are inevitably correlated at their outputs. Our architecture proposes to eliminate this correlation while filtering the output at the same time. This is achieved by using random bit streams in a shift chain to clock each sampled point, which essentially introduces uncorrelated phase shifts while sampling. We have also introduced a new weight and activity representation to provide a smoother curve of averaged activity. It is precisely this representation and its application to multiplication that we have tested in VLSI and will report results from, after briefly describing the proposed architecture. We envision the digital version of the implemented circuit to be embedded in the proposed architecture.

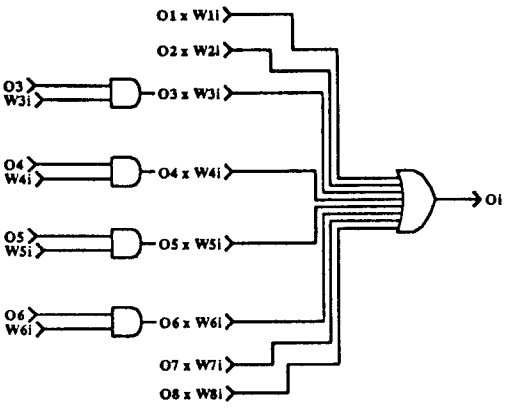

Figure 1: Neural computing unit using random pulse trains

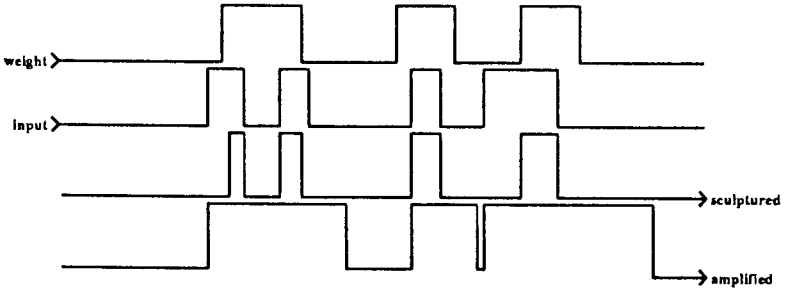

Figure 2: Sculpturing and amplification with weights

2 Description of Proposed Architecture

We consider a typical feed-forward neural network where the activation function of each computing unit o_i is derived from the outputs of the previous layer of computing units o_j, through weights w_{ji}. Each computing unit calculates the non-linear sum of $o_j w_{ji}$ product pairs to project to the following layer of computation. The product of two uncorrelated pulse streams can be obtained by a simple logical AND operation, and therefore a hardware synapse consists solely of an AND gate. The non-linear sum of product pairs, similar to the squashing transfer function of the linear sum, can be obtained from a logical OR of the outputs of such synapses [1] (Figure 1).

Although the AND operation produces an accurate product, its range of multiplicands is limited. The 'product' is always less than or equal to the minimum of the multiplicands, thus the absolute value of the true multiplicand range is strictly between zero and one for both multiplicands. There is no single unit amplification in such a system. Amplification is possible only by the use of many units, which is quite

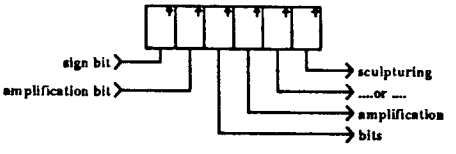

Figure 3: Digital weight representation

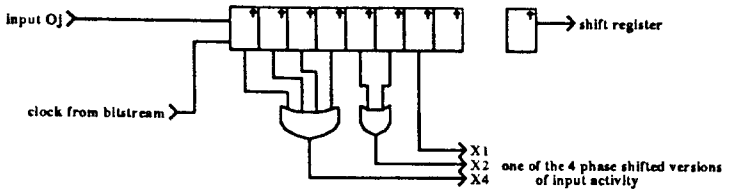

Figure 4: Output activity (o_j) sampling

wasteful. To alleviate this shortfall, in our architecture, weights are stored digitally in synapse registers in two modes, amplifying and sculpturing (Figure 2). The sculpturing weights only produce outputs less than the input. Amplifying weights; on the other hand, produce outputs greater than or equal to the input. Amplification is quite non-linear: Larger inputs are amplified less than smaller ones. Weights are continuous in the sense that a single increment of the maximum sculpturing weight corresponds to the minimum amplifying weight. One bit is reserved for sign (Figure 3).

In order to convert digital weights to random pulse trains additional weight representations are necessary. This representation was studied in hardware utilizing analog and digital techniques. The results are reported in the next section. The corresponding digital implementation is described in the following paragraph:

Discrete partial weights corresponding to the sculpturing bits are represented in a series of shift register chains. Each chain of registers contains ones and zeros, the numbers of which are proportional to the discrete weight being represented. For example, a chain corresponding to half of the maximum value allowed for sculpturing contains half ones and half zeros. The bits are recycled by a separate, usually slower clock than the one used for generating the random pulse streams. The length of the shift chain is exponentially related to the number of bits representing the sculpturing part of digital weights. The number of chains needed; on the other hand, increases linearly with each additional bit. For example, if four bits are reserved for sculpturing, which corresponds to six bit weights including sign, the length of the chain is sixteen and the number of chains needed is four. From each chain four outputs exactly 90° out of phase with one another are provided to each synapse. The reason for this is that random pulse trains with the desired average values are generated by selecting randomly between these four phase shifted versions. For amplifying weights, the same strategy is used. For each amplified version of input activity four phase shifted versions exactly 90° out of phase with one another are provided to all synapses using that input (Figure 4).

Input activity to the first layer of the network could be analog or digital, but needs to be converted to a pulse width modulated format. For analog inputs, a pulse width modulator circuit, similar to the one we used could be appropriate. This would provide continuous values of inputs to the network. For digital inputs, circuits described along with the DNNA architecture [1] could be utilized. In between layers; however, computation is communicated in pulses and there is no need for conversion. To eliminate undesirable correlation between layers, the output activity of each of the previous layer computing units is sampled at points uncorrelated with the rest. This also has a filtering effect, eliminating noise and preparing the input to the next layer of processing.

Each synapse multiplexes between the four phase shifted versions of its weight using two uncorrelated bit streams. Sculpturing is accomplished by AND'ing the incoming activity with the selected representation of the weight. Multiplexing between incoming weights is disabled if the amplification bit is turned on, and multiplexing in that case occurs between the four phase shifted versions of incoming activity at the selected amplification level. Amplification has no effect on zero activity, but enhances any non zero activity in proportion to the magnitude of the amplifying weight (Figure 5).

Although it is not indicated on the circuit diagrams, positive and negative activity are separated from

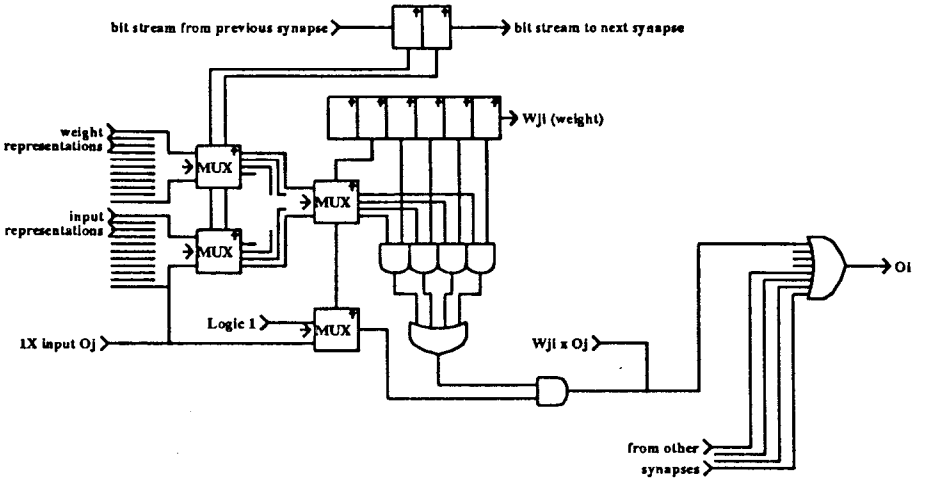

Figure 5: Synapse circuit in detail

each other and are processed independently.

It is obvious that computation layers can be stacked vertically with full interconnection between layers with this architecture. However, in that configuration the number of computing units within each layer would be restricted to what can be accomodated in a single IC. On the other hand, if the number of inputs to each computing unit were restricted, this architecture would also be horizontally cascadable.

3 Implementation of the Multiplication Unit in VLSI

We have implemented part of the above architecture in VLSI. Specifically, we have fabricated and tested the random pulse generation and the synaptic multiplication circuits to investigate how accurate and robust multiplication is using our scheme.

The circuit consists of a digital delay line to simulate the phase shifted weight and input activity level representations. The input to the delay line is provided from a pulse width modulator that is driven by an analog input value. This implementation mixing analog and digital techniques was chosen to test this multiplexing scheme over a continuous range of values. It is functionally identical the corresponding circuits described in the proposed digital architecture and varies only in implementation.

The pulse width modulator circuit is an extension of the current controlled neuron circuit which uses a positive feedback mechanism to generate pulses [6]. This circuit compares the input analog value to a triangular wave, the midpoint of which corresponds to the midpoint of the range of sculpturing weights. Thus, half of the maximum allowable analog value generates a pulse signal with a 50% duty cycle. This implementation proved to generate pulse width modulated signals quite accurately (Figure 7, reference signal PWM marked with circles).

The random bit streams are generated by a shift register sequence, based on irreducible polynomials, using specifically the polynomial $(x^{16} + x^{12} + x^3 + x + 1)$. This is done to obtain the pseudorandom bit sequence with the longest possible period. An important property of such bit sequences is that not only are an (almost) equal number of ones and zeros generated, but also their auto-correlation functions are very favorable, namely a single shift of the sequence leads to inconsequential correlation [3].

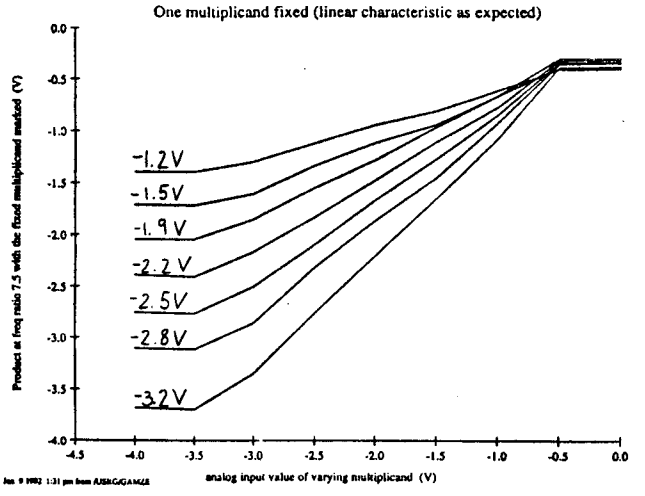

Figure 6: Time averaged product of pulses with one multiplicand fixed

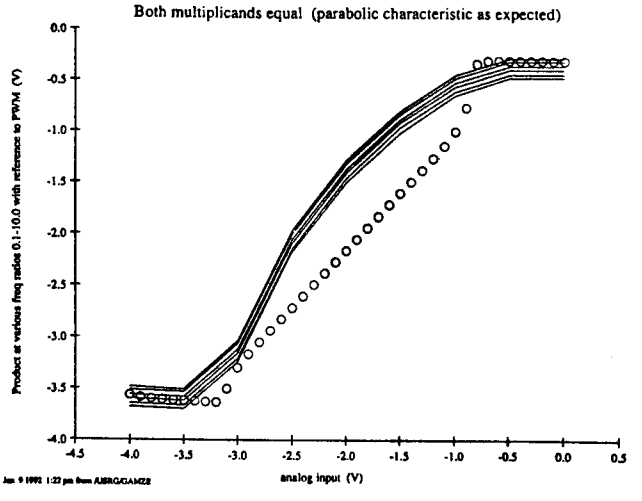

Figure 7: Time averaged product of pulses with both multiplicands equal

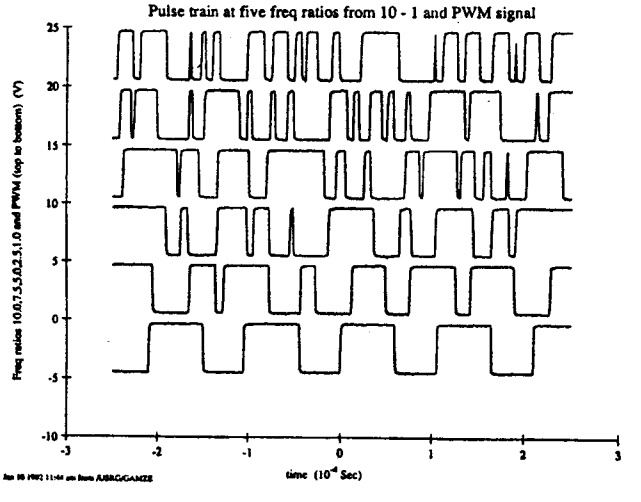

Figure 8: Pulse train for various frequency ratios

The bit streams are used to select between the four phase shifted versions of the sculpturing weight and the input. These signals are AND'ed with each other, and the result gets time averaged using a simple RC circuit.

AND multiplication has been tested both with one multiplicand fixed and both multiplicands equal (Figure 6) (Figure7). In both of these the chip has performed very robustly over frequency ratios ranging from 0.1 to 10. This has been demonstrated by time averaging the pulse trains through an RC filter with a fixed time constant (0.1 sec.). The frequency of activity levels are adjusted by changing the ratio of frequencies between the triangular wave generating the pulse-width-modulated representation and the clock generating the uncorrelated bit sequences which multiplex between them (Figure 8). In the proposed digital architecture, this would correspond to the frequency ratio between the clock recycling the partial weight representations and the clock generating the uncorrelated bit sequences.

4 Conclusions

We have described a digital neural network architecture and reported results from the VLSI implementation of the multiplication scheme embedded in it. As can be inferred from the test plots, the circuit is robust and quite accurate over a wide range of clock frequencies. We believe this robustness to be congenial to the exploration of novel training algorithms in such a system. One could, by reducing the time constant of the time averaging circuit, and thus averaging the pulse train over a smaller number of clock cycles, get significant variations in output activity for the same input. The use of the statistical properties of these variations among computing units could have novel implications in weight update decisions during training.

5 Acknowledgements

This work was supported in part by the Army Research Office under contract number DAAL03-89-K-0126, and in part by DARPA under contract number AFOSR-90-0199. G. Erten is supported in part by an NSF Graduate Fellowship.

References

[1] M.S. Tomlinson, M.A. Sivilotti. A High Performance, Scalable Digital Neural Network Architecture for VLSI. *Advanced Research in VLSI 1991* :UC Santa Cruz pp:262-273.

[2] M.S. Tomlinson, D.J. Walker, M.A. Sivilotti. A Digital Neural Network Architecture for VLSI. *IJCNN Proceedings 1990*, Vol II pp 545-550.

[3] J.G. Proakis. *Digital Communications.* McGraw Hill, 1989.

[4] A.F. Murray, D. Del Corso, L. Tarassenko. Pulse Stream VLSI Neural Networks Mixing Analog and Digital Techniques. *IEEE Transactions on Neural Networks*, Vol 2, No.2, March 1991, pp 193-204.

[5] J. Alspector, J.W. Gannett, S. Haber, M.B. Parker, R. Chu. A VLSI-Efficient Technique for Generating Multiple Uncorrelated Noise Sources and Its Application to Stochastic Neural Networks. *IEEE Transactions on Circuits and Systems*, Vol 38, No.1, January 1991.

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