1988 Symposium on VLSI circuits, Tokyo (Japan), 22-24 August 1988, pp.27-28

III-2

# A LARGE VLSI FULLY INTERCONNECTED NEURAL NETWORK

M. VERLEYSEN (*), B. SIRLETTI (*), P. JESPERS

UNIVERSITE CATHOLIQUE DE LOUVAIN, LABORATOIRE DE MICROELECTRONIQUE

3, PLACE DU LEVANT, 1348 LOUVAIN-LA-NEUVE, BELGIUM

Neural networks don't have any more to prove their utility and their power in the field of pattern recognition, multi-parameters optimization and np-complete problems resolution. They offer indeed a new kind of solution to all these problems, with a resolution speed which can be 100 or 1000 times faster than the one obtained with the classical artificial intelligence approach.

Nevertheless, practical solutions cannot be realized without considering the implementation of very large arrays with hundreds of neurons; large networks are indeed necessary to preserve the intrinsic parallelism and speed properties of such architectures.

A neural network consists in an array of elementary processors called neurons and a coupling network formed by resistive elements called synapses ([1]). Through this network, each neuron can be connected to each other. The connections can be positive or negative. A positive one is called "excitatory connection", a negative one is called "inhibitory". Each synapse can either source or sink current to the input line of the connected neuron; the direction of the current is determined by a combination between two values: the connected neuron output value (which is supposed to be boolean in our model) and a connection weight programmed into the synapse. We recommend here to use a XOR function between these two values. Simulation showed indeed that the performances are increased when we consider that a neuron in a low state connected by a negative synapse has the same effect as a neuron in a high state with a positive synapse. This is not true when we use an AND function like the one presented by Hopfield ([2]). We will suppose in the following that the weight memorized into each synapse can take three different values: -1, 0 or +1. The architecture proposed here can be generalized if we need more than three different states.

In the theoretical models, connections consist in coupling resistors between the input of one neuron and the output of another one. The first realized networks imitated these theoretical models by implementing connections with resistors whose values determine the connection strengths ([3]). Since we need excitatory and inhibitory connections, i.e. positive and negative weights, the resistors can be connected either to the non-inverting or to the inverting output of the neuron.

In more recent applications, coupling is made by means of active elements (typically, programmable current sources which inject or sink current to the input line of each neuron as illustrated in figure 1) ([4]).

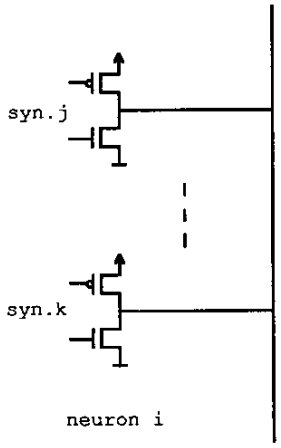


fig. 1

A problem arises immediately: the current injected by the P-type transistors can be different from the one sunk by the N-type transistors because of the unrealized matching between the two types of transistors (a good matching is very difficult to obtain because of the mobility difference between N-type and P-type carriers). The function of each neuron is to detect the sign of the sum of the other neurons values, weighed by the synapses strengths. The possible mismatching between the sourced and sunk currents are summed when increasing the number of synapses. Since we want the neurons to be able to discriminate the sign of their input even when the difference between excitatory and inhibitory currents equals a single synaptic current, the latter must be greater than n times the difference between the P- and N-type current sources; this limits considerably the dimension of the network.

We propose here a new architecture for neural associative memories. In our solution, the sourced and sunk currents are summed separately on two different lines. Each synapse is a current source (figure 2) programmed by *mem1*; if *mem1=0*, neurons *j* and *i* are not connected together.

(*) sponsored by IRSIA

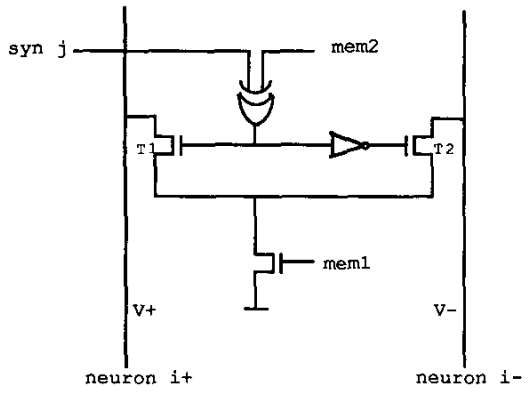


fig.2

*Mem 2* determines the sign of the connection, i.e. if the current must be sourced or sunk. In the first case *T1* derives current from the line *i+*, in the second one *T2* derives it from the line *i-*. All we have to do now is to compare the currents on lines *i+* and *i-*; this is done by means of the current reflector described in figure 3. The synapses have to be connected to the *neuron i+* and *neuron i-* inputs. The two currents on these lines are converted into voltages by transistors *T3* and *T4*; these voltages are themselves compared in the reflector formed by transistors *T5* to *T9*. Because of the two-stage comparator in the neuron, its gain is very important and the output *(out)* is always saturated, either to 5V if the current in *neuron i-* is greater than the one in *neuron i+*, or to OV in the opposite case.

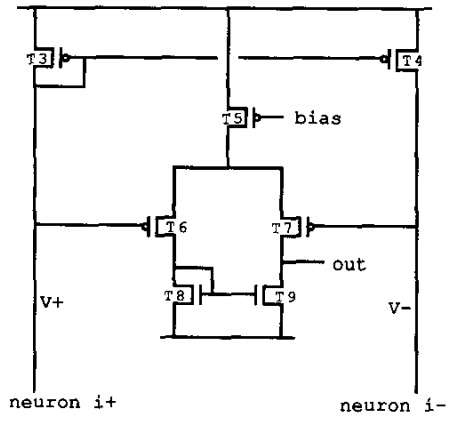


fig.3

The last problem we have to solve is the equality between the currents in the different synapses. With no particular device, the voltages *V+* and *V-* would be reduced when we increase the number of active synapses. With a great number of neurons, we will quickly reach a point where the addition of one synapse will have no more effect on the neuron current because of the *V+* and *V-* voltage diminution. To avoid this problem, we insert a negative feedback loop on the lines *i+* and *i-* between the synapses and the current reflector (figure 4). The voltages *V+* and *V-* are thus fixed to *Vref*, and the currents sunk in each synapse no more varies. While no high gain is needed for the feedback loop, the amplifier shown in figure 4 can be very simple (acceptable simulation results are obtained even with a single transistor amplifier).

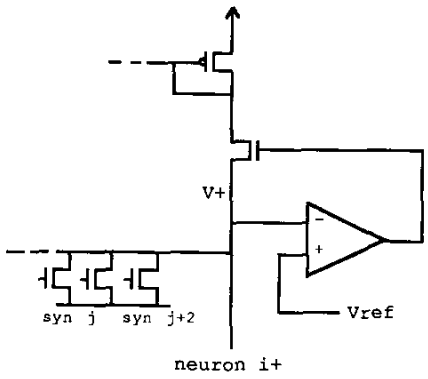


fig.4

Simulations showed that circuits based on this architecture can contain hundreds of neurons without any problem in the neuron value detection. Since this circuit is fully programmable, it can be used as a content-addressable memory ([5]), or can be programmed to solve optimization problems.

[1] LIPPMANN R. (1987) An introduction to computing with neural nets, IEEE ASSP Magazine, April 1987

[2] HOPFIELD J.J. (1982) Neural networks and physical systems with emergent collective computational abilities, Proc. Natl. Acad. Sci. USA, April 1982

[3] HOWARD R. and al. (1987) An associative memory based on an electronic neural network architecture, IEEE transactions on electron devices, July 1987

[4] GRAF H., DE VEGVAR P. (1987) A CMOS associative memory chip based on neural networks, proceedings ISSCC 1987

[5] SIRLETTI B., VERLEYSEN M., VANDEMEULE-BROECKE A., JESPERS P. (1988), A new learning algorithm for content-addressable memories using Hopfield's neural networks, paper submitted to Electronics Letters