Chapter 8

Bus Architecture Implementation of the Hopfield Networks

8-1 Implementation Description

Hopfield network was adopted for this implementation because of its importance in the optimisation of some computational tasks such as the travelling salesman problem, content addressable memory...etc. However, there are two updating versions of this network: the synchronous version and the asynchronous version. It is well proven that the second version is the most stable and that it falls rapidly to the energy minimum. Consequently, the asynchronous version was adopted to achieve the best performance in both the network algorithm and the hardware speed up in order to minimise the time consumed during processing and to enhance the optimisation capability.

In order to implement the network interconnections, the bus architecture [1] was adopted. In this architecture, there are however, two kinds of units connected by a bus namely identical neuron units and one control unit called the arbiter unit. The bus is a group of wires that transport many signals at the same time between two or several digital systems. However, the bus in our architecture is used to transport the address of the "winner" neuron, to be updated, from the arbiter unit to all other neurons. Therefore, the neurons receive the same address more or less simultaneously and update their local field as is depicted by Figure (1).

The arbiter unit is a digital circuit that is designed to choose the winner neuron, which is the best candidate to be updated, based on a given criterion. The winner neuron in the point of view of our criterion, is however, chosen based on the absolute value of its local field. The reader is reminded that the local field is equal to the sum of the input signals from other neurons weighted by their corresponding local synaptic weights as shown in the following expression:

$$h_i(x_i) = \sum_{j=1}^{N} J_{ij} x_j$$

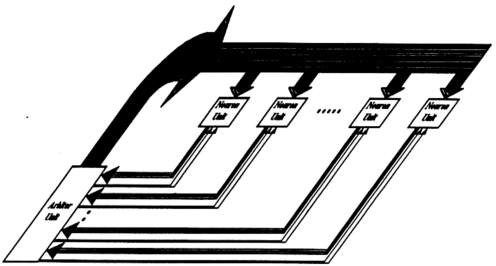

Figure (1). General description of the bus based architecture.

Therefore, the winner neuron is the neuron, which has the largest absolute of the local field. It will be updated to the firing state '1' if its local field is positive and to the quiescent state '0' in the opposite case.

8-2 Implementation VHDL Description

In this section, the different constituents of the top-level description (see the VHDL source code in Appendix F) are presented. Starting by the introduction of the neuron unit, then describing the arbiter unit, the final top-level programme in which the connection between the different above-mentioned units will be presented.

8-2-1 Neuron unit

Neuron' s input port busses (see Figure (2)) are key of two-bit width, sgl_in of eight-bit width, synapses of 32-bit width and winrn of four-bit width shared by the other neurons. The bus signal key is used to control the neuron different functionalities namely initialising its output, clearing its synapses, calculating and updating its initial local field.

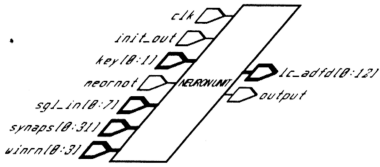

Figure (2). Input and Output Ports of a Neuron Unit.

The bus signal sgl_in dedicated for holding the initial configuration of the neural system is shared among all neurons. The synapse bus signal is used to transfer the value of the synapses loaded from outside the chip. Finally, the bus signal winrn holds the address bits of the updated neuron chosen by the arbiter unit to all neurons.

Beside the busses, there are single signals such as the clock signal clk, the signal init_out and the signal meornot. The signal clk is the clock signal that governs the neuron internal registers. Secondly, the signal init_out holds the initial value of the neuron which is stored at its output when the bus signal key gets the value "10". Finally, the meornot signal, indicating the winner neuron, is always on the state '0' except when the neuron is the winner neuron that it will hold the state '1' and then clips its state.

Besides, the neuron unit has output signals connecting it with other neurons units as well as with the arbiter unit. These signals are lc_adfd and output. The first signal is a bus signal used by the neuron unit to communicate the local field to the arbiter unit. The second is however, the output signal that holds the state of the neuron to be sent to the other neurons during the local-field calculation phase.

In Figure (3), a functional diagram is shown to describe the different neuron functionalities based on the state of the bus signal key.

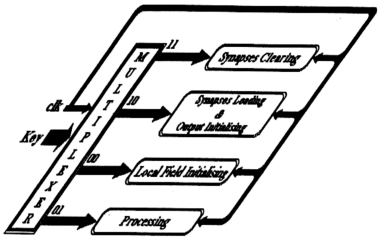

Figure (3). Functional Diagram of the Neuron Unit of the Bus-based Architecture

8-2-2 Arbiter Unit

The arbiter unit is the unit 'arbiting' between the neurons to choose the winner neuron among them to be updated. It has three single signals and one pre-defined bus. The signals are clk, rst and start. The signal clk and the signal rst govern the internal registers holding internal signals influencing the unit output. The signal start is a port that chooses between the different functionalities of the arbiter unit. However, if it is in the state '0' then the state of the neurons will be at the configuration initialisation phase and they remains in this phase until the signal start clips to '1'. In the last case, the arbiter directs the neuron towards two phases. Firstly, the local-field initialisation phase and then the processing phase. The arbiter unit controls the neuron functionalities through the output port load_wup, which is connected to all neurons through their input port signals key. The two other output ports are winad and winsig. The first output port send the winner neuron address to all neurons to update their local fields when being in the processing phase based on their local field values received through the input port recei_fld. The

signal bus winsig output port is connected to all neurons through their meornot signals. Figure (4) shows the input and output ports of the arbiter unit while Figure (5) represents the functional diagram of the mentioned unit.

8-2-3 Top-Hierarchical Level

At this description level, the above units are combined together in to build the whole system using the VHDL structural component statement. In this step, the major problem rising frequently is the synchronisation between the different processing units constituting the whole hierarchy. Both of the two previous units are using a clock clk signal because of their internal registers and this might therefore create, most probably, functional interference problem between entities dependant on the clk in both units.

Figure (4). Input and Output Ports of the Arbiter Unit of the Bus-based Architecture

At least two buses certainly are affected, lc_adfd, which is a bus signal generated from each neuron unit to the arbiter unit, and winrn which is a bus signal generated from the arbiter unit to all neurons. However, if both units use the same clock signal, then at the edge rising of the clock signal the outputs of the internal registers will get wrong data from each other. This occurs because it is the product of the same clock edge rising. Therefore, each unit will get its required values late form the other unit because of the delay time of the combinatorial circuit and hence the confusion occurs.

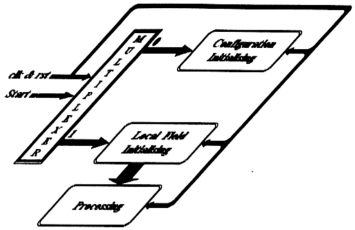

Figure (5). Functional Diagram of the Arbiter Unit Processing.

To overcome the problem, we propose that the arbiter unit uses the system clock signal while neuron unit uses the reversed system clock signal. Adopting this method, only one unit updates its internal registers at a time, tackling consequently the confusion between both units. Figure (6) shows VHDL simulation of the present implementation.

8-3 Conclusion

We have presented hereby an architecture that seems promising in the point of view of simplicity, communication economy and therefore energy and hardware economy. These criteria are very important for the implementation of artificial neural networks especially when the number of the implemented neurons is relatively high.

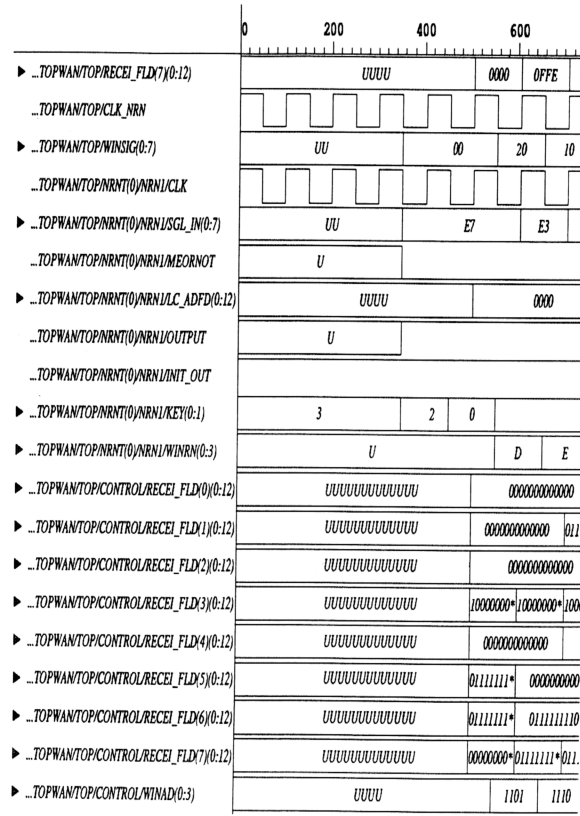

Figure (6). VHDL Simulation Diagram of the Bus Architecture Implementation of the Hopfield Network.