How to interface to a NeuroMem chip?

A NeuroMem neural network chip can learn and recognize patterns, save and restore the knowledge built by the neurons. One of its key features is that the programming and latency of these operations remain the same whether your network is composed of 10, 100, 1000 or more neurons.

Finally NeuroMem is incredibly easy to use!

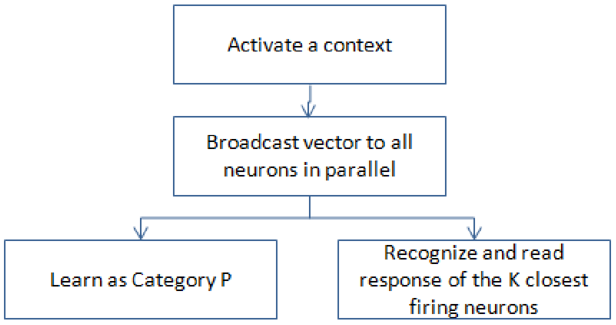
The interface to a chain of one or multiple NeuroMem chips comes down to 4 main operations:

- Broadcast a vector
- Recognize and/or Learn it
- Save the knowledge built by the neurons
- Load a knowledge into the neurons

THE NORMAL MODE

1.1 BROADCAST A PATTERN

Whether you intend to learn or recognize a vector, the first step consists of broadcasting its components to the neurons. As these components are written sequentially, the neurons receive them simultaneously and update their Manhattan distance between the vector and the model stored in their memory. The last component is written to a special register called LCOMP to notify the neurons that they can now fire if they find the vector similar to their model, that is if their calculated distance is less than the current value of their Influence Field.


Activate a context
Broadcast vector to all neurons in parallel
Learn as Category P
Recognize and read response of the K closest firing neurons

Assuming that the input vector has a length L which is less than or equal to 256, the broadcast is programmed as follows:

```
For (i = 0; i<L-1, i++) Write COMP, Vector(i);
Write LCOMP, Vector(L-1)
```

Optionally, you can immediately read the Network Status Register and find out if the vector is recognized by any neuron or not, and in the former case if some neurons disagree with the recognized category or not.

Also note that prior to broadcasting a vector, you have the option to change the value of the Global Context register in order to segment dynamically the network into sub-networks so neurons can learn vectors representing un-related data type or dimensions (for example, Context 1 can refer to audio data and Context 2 to pixel data).

1.2 BROADCAST TO TEACH

Learning the last broadcasted vector consists of assigning to it a category Category-to-Learn. This is where the firing neurons with a category other than Category-to-Learn will reduce their Influence Fields to exclude the vector from their similarity domain and not repeat the misfiring if this vector is broadcasted again. This is also where a new neuron will be committed if the vector is a novelty and there are no firing neurons with the category Category-to-Learn.

```
Write CAT, Category-to-learn
```

Optionally you can read the NCOUNT register and get the total number of committed neurons after this last learning operation.

1.3 BROADCAST TO CLASSIFY

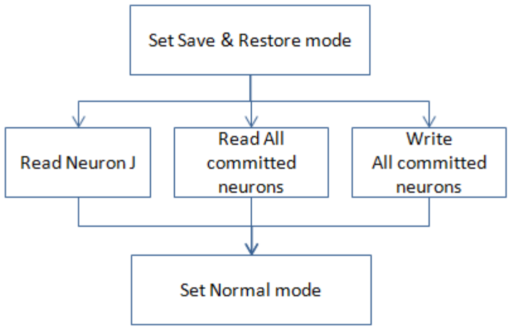
Once a vector has been broadcasted, reading the response of the “firing” neurons is made orderly with successive Read of the distance (DIST), category (CAT) and optionally the neuron identifier (NID) registers. This is where the “Winner Takes All” inter-connectivity between the neurons takes place so they know when they hold the next smallest distance and their model is consequently the next closest match to the broadcasted vector. Depending on your application, you can decide to stop the iteration at a value K of 1 or more.

```
Do Loop
{
    Read DIST, Output-distance
    Read CAT, Output-category
    Read NID, Output-identifier
```

```
      k++;
 } Until (Output-distance == 0xFFFF or k=K)
```

THE SAVE AND RESTORE MODE

The Save and Restore mode allows to access the neurons sequentially (as standard memories) and read or write their contents for the purpose of saving or restoring the knowledge that they have built under Normal mode. Do not forget to switch the network back to Normal mode when done.



1.4 SAVE THE KNOWLEDGE

Reading the knowledge (to save it to file for example) consists of switching the neurons to Save-and-Restore mode, pointing to the first neuron of the chain, and reading the neurons' models and internal registers such as their Context, Category and Influence Fields. After each Read CAT, the next neuron of the chain become in focus. The following code works seamlessly whether you have a chain of 1 or N NeuroMem chips.

```
Write NSR, 16
Write RESETCHAIN, 0
Do
 {
      For (i = 0; i<L, i++) Read COMP, Vector(i);
      Read NCR, Neuron_Context
      Read AIF, Neuron_InfluenceField
      Read CAT, Neuron_Category
} while (Neuron_Category!=0)
Write NSR, 0
```

1.5 LOAD A KNOWLEDGE

Loading the neurons with a pre-existing knowledge consists of switching the network to Save-and-Restore mode, pointing to the first neuron of the chain and writing sequentially to their memory and registers. Each Write CAT switches the neuron in focus to the next one of the chain. The following code works seamlessly whether you have a chain of 1 or N NeuroMem chips.

```
Write FORGET, 0
Write NSR, 16
Write RESETCHAIN, 0
For (j=0; j<Number-of-Inputs; j++)
{
      For (i = 0; i<L, i++) Write COMP, Vector(j,i);
      Write NCR, Context(j)
      Write AIF, InfluenceField(j)
      Write CAT, Category(j)
}
Write NSR, 0
```

SIMPLE SET OF 15 REGISTERS

All operations are a series of Read and Write commands accessing the 15 registers of the neurons which are described in the NeuroMem Technology Reference Guide. The most common are listed in the table below.

Register	Description	Addr	
COMP	Component	0x01	RW
LCOMP	Last Component	0x02	RW
DIST	Distance	0x03	R
CAT	Category	0x04	RW
AIF	Active Influence Field	0x05	R
NSR	Network Status Register	0x0D	RW
GCR	Global Context Register	0x00	RW
FORGET	Clear the neurons	0x0F	W
NCOUNT	Committed neurons	0x0F	R
RESETCHAIN	Reset the neuron chain	0x0C	W

CONCLUSION

All APIs delivered by General Vision include these basic functions and more. Our evaluation boards have these functions pre-programmed in FPGA for faster throughput. Read more about our NeuroMem SDK and CogniSight SDK.