

CogniMem Technology

Reference Guide

Version 2.1
Revised 12/16/2011

COGNIMEM
Technologies, Inc.

1 TABLE OF CONTENTS

1	Table of Contents	2
2	Introduction.....	4
3	What is a neuron?	5
3.1	Neuron part 1: A memory holding a pattern	6
3.2	Neuron part 2: A distance evaluation unit.....	6
3.2.1	Definitions of L1 and Lsup.....	6
3.2.2	Examples of usage of L1 versus Lsup	7
3.3	Neuron Part 3: An associative recognition logic	7
3.3.1	Firing stage.....	7
3.3.2	Identified or Uncertain recognition	7
3.3.3	Unique Search and Sort logic.....	8
3.4	Neuron Part 4: A learning logic	8
3.5	Neuron Part 5: An element in an infinite chain	8
3.6	Neuron Part 6: The behavior of a KNN or RBF classifier	9
4	Network architecture	10
4.1	A chain of identical neurons.....	10
4.2	Network segmentation into context.....	11
4.2.1	Multiple networks for combined decision criteria	12
4.2.2	Multiple networks for hierarchical decision	12
4.3	Save and Restore of the neurons	13
5	Neuron registers.....	14
5.1	Neuron registers.....	14
5.2	Global registers	16
6	Functional diagrams	17
6.1	Vector broadcasting	17
6.2	Vector Recognition.....	17
6.2.1	KNN, always best match classification.....	18
6.2.2	RBF, more subtle classification	18
6.2.3	RBF Response Type 1: Classification status	19
6.2.4	RBF Response type 2: Best-match	19
6.2.5	RBF Response type 3: Multiple matches.....	19
6.3	Vector Learning	20
6.3.1	Global settings prior to learning	20
6.3.2	Waiving dependency from the teaching sequence	20
6.4	Save and Restore sequences.....	21
6.5	Simple Example	22
7	CogniMem, high speed KNN classifier.....	23
7.1	Loading the training examples	23
7.1.1	Smart Category assignment.....	24
7.1.2	Optional Context usage.....	24
7.1.3	Learning, not loading, the examples.....	24

7.2	Closest K neighbors in microseconds	26
7.2.1	Broadcast the vector to all neurons.....	26
7.2.2	Search and Sort in a fixed amount of time	26
7.2.3	Limitations of the CM1K chip (first generation)	27
7.3	CM1K Timing Benchmarks	27
7.3.1	Step 1: Broadcast a vector with N dimensions	27
7.3.2	Step 2: Readout of the K Nearest Neighbors	27
7.3.3	Example.....	28
8	What is new in this manual ?	29
8.1	Revision from 09/23/2011	29

2 INTRODUCTION

Neural networks have been extensively discussed in the late eighty's and DARPA's recommendation in a survey published in 1987 was to "go silicon". While it has been claimed by many publications that neural networks could benefit from a parallel architecture, most current implementations are running on standard computers, which suffer from the following limitations: (1) they deal with processing and memory separately; (2) They execute one instruction at a time (or up to four with quad cores); (3) A good portion of the data bandwidth is dedicated to fetching and decoding instructions prior to executing them; (4) Architectures based on multiple processors require synchronization schemes and cache management which can become a serious hurdle. As a result, a neural network implemented by software and running on computer(s) cannot be defined as truly parallel.

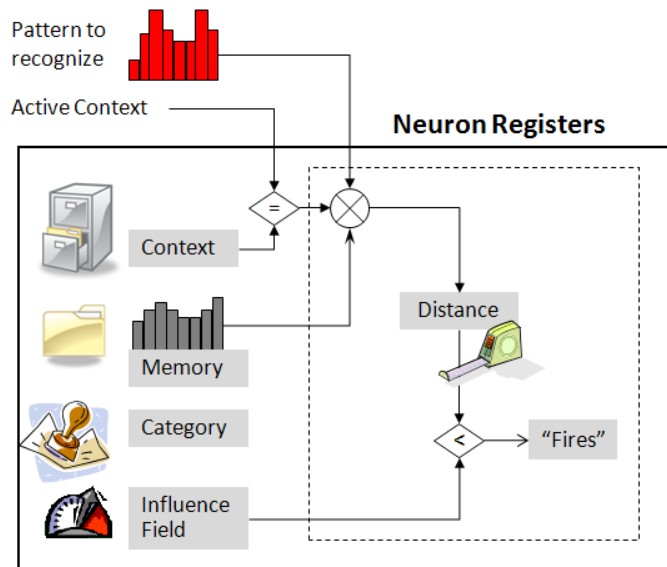
The CogniMem chip is a fully parallel silicon neural network: it is a chain of identical elements (i.e. neurons) which can store and process information simultaneously. They are addressed in parallel and have their own "genetic" material to learn and recall patterns without running a single line of code and without reporting to any supervising unit. In addition, the neurons fully collaborate with each other through a bi-directional and parallel neuron bus which is the key to accuracy, flexibility and speed performance. Indeed each neuron incorporates information from all the other neurons into its own learning logic and into its response logic. This mechanism prevents the learning of any redundant information, but also the immediate detection of novelty or potential conflicts. Another resulting achievement of the parallel architecture of the CogniMem neural network is a constant learning and recognition time regardless of the number of connected neurons. A second achievement is the ability to expand the size of the network by cascading chips. CogniMem is not the first of its kind.

CogniMem Key Features

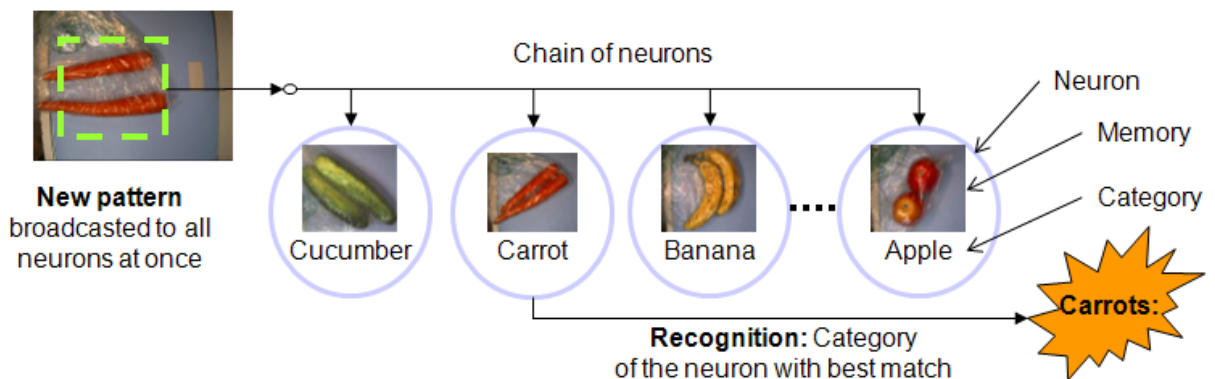
- Choice of two non-linear classifiers:
 - Radial Basis Function Network (RBF)
 - K-Nearest Neighbor classifier (KNN)
- Adaptive model generator
- Save and Restore of the contents of the neurons
- Parallel architecture
 - One thousand neurons on a chip recognizing the same input pattern in parallel
 - Unlimited number of neurons by cascading many chips in parallel
 - High-speed learning and recognition (<3 μ sec)

3 WHAT IS A NEURON?

A neuron is a cognitive and reactive memory which can autonomously evaluate the distance between an incoming pattern and a reference pattern stored in its memory. If this distance falls within a range called the influence field, the neuron returns a positive classification which consists of the distance value and the category of its reference pattern.



Although a neuron has its own processing unit to react to a pattern, it is the collective response of all the neurons which produces interesting diagnostics. When attempting to recognize a pattern, each neuron has the capability to spy the response of its counter-parts and to withdraw itself from the race if another neuron reports a smaller distance value.



3.1 Neuron part 1: A memory holding a pattern

The neuron has a memory with a capacity of 256 bytes. This means that it can store a pattern or prototype of 256 components ranging between 0 and 255.

The access to the memory cells is controlled by the neural network itself. All the neurons point to the same memory cell index at any time. The cell index is automatically incremented each time a new component is broadcasted to the neurons. It is reset to 0 when the last component is entered. Other subsidiary operations can change or reset the index as described under the chapter “Functional diagrams”.

3.2 Neuron part 2: A distance evaluation unit

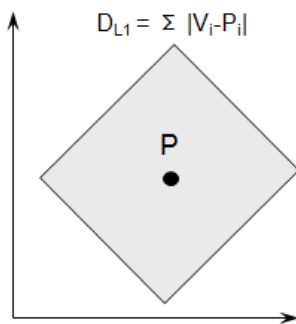
The distance evaluation unit computes the distance between the incoming vector (up to 256 components) and the pattern stored in the neuron memory. This occurs in parallel for all the committed neurons each time a vector component is broadcasted to the neuron bus.

The Distance value is automatically reset to 0 by writing the first component of a vector or by writing to the Network Status Register (to change the operation mode or the type of classifier).

The distance can be calculated using two norms: L1 (default) or Manhattan distance, and Lsup. The selection of a Norm depends on the application and in particular the type of patterns to classify, their possible variations between categories and the final intent of the recognition (identification, classification, anomaly detection).

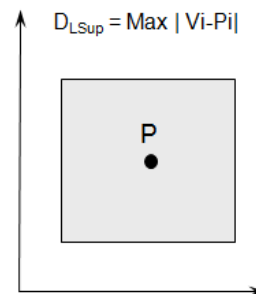
3.2.1 Definitions of L1 and Lsup

Norm L1 (Manhattan distance)



The L1 distance emphasizes the drift of the sum of the all components between V and P.

Norm L Sup



The Lsup distance emphasizes the largest drift of the same component between V and P.

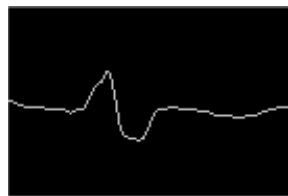
3.2.2 Examples of usage of L1 versus Lsup

Example 1: vector data with unrelated component values.

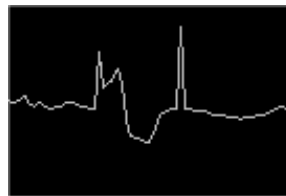
Let's take the example of a data mining application where the profile of customers is categorized based on attributes such as age, sex, weight, skin color, date of graduation, income bracket, etc. These attributes are expressed in different units, and some of them are actually codes rather than measurements. Still they can be assembled in a pattern vector to help classify people. In this case, the distance between an input vector and a stored prototype is not representative of any unit, but the L1 Distance gives an idea of the overall variations between them. On the other hand, the Lsup Distance is meaningless since depending on the index of the component with the highest difference, the unit can be years, dollars, codes, etc.

Example 2: Noise filtering.

Neurons can be used as a noise filter if they hold prototypes of non-noisy patterns and their Norm is set to Lsup.



Normal signal



Noisy signal

In the graphs above, the vector V_{noisy} shown to the right is a noisy version of vector V shown to the left. The Lsup distance between V and V_{noisy} is 50 when the L1 distance is 4900. Indeed the L1 distance increases dramatically when noisy peaks are superimposed onto the signal. Neurons trained with an L1 distance will not easily associate V_{noisy} to V . Neurons trained with the Lsup distance will make this association more easily.

3.3 Neuron Part 3: An associative recognition logic

3.3.1 Firing stage

The associative logic of the neuron is activated when the last component of the input pattern is received. The calculation of the distance between the input pattern and the pattern stored in the neuron is complete. If its value is less than a threshold called the neuron Influence Field, the neuron enters a status called "firing status" and outputs its category to the neuron bus.

3.3.2 Identified or Uncertain recognition

This output is multiplexed with the output of all the firing neurons and the result establishes immediately if the neuron is in agreement with the other firing neurons or not. If yes, the classification is labeled as Identified. If no, the classification is labeled as uncertain.

3.3.3 Unique Search and Sort logic

When a neuron has fired, it responds to subsequent requests for a distance and category value and when its own response becomes the smallest value. A patented “search and sort” mechanism retrieves the smallest value transmitted by all neurons at the same time on the neuron data bus. Each neuron can then use this feedback to determine if it can stay in the race of the best match in term of distance value or category value.

3.4 Neuron Part 4: A learning logic

The learning logic is activated after the associative logic when a category value is assigned to the last input pattern. Only the “firing” neurons react to a learning operation and if none of them recognizes the pattern correctly (that is with a category equal to the taught category), a new neuron gets committed to hold the input pattern and its associated category. Its influence field is set to the distance of the closest committed neuron.

If the category of a firing neuron is different from the category to learn, its Influence Field is reduced to its distance value or a Minimum Influence Field value whichever is greater. This reduction will prevent the neuron from making the same erroneous classification if the same pattern is broadcasted again. If the Influence Field of the neuron reaches the Minimum Influence Field value, it is flagged as degenerated.

Example 1:

In an OCR application, the U and V letters can have very similar signature. It would not be a surprise that the learning of examples of V and U degenerate some neurons because an area of uncertainty between the models of U and V exists (depending on the font of the characters).

Example 2:

A common cause for degenerating neurons is to send contradictory learning instructions. For example, if a knowledge already holds reference patterns of the character V and an operator teaches a new V pattern as an A character by mistake, this erroneous instruction will probably degenerate some of the neurons holding a correct V pattern.

3.5 Neuron Part 5: An element in an infinite chain

A neuron is an element in a chain of neurons. The neurons are daisy-chained to compose a network of neurons, but they all receive and decode commands in parallel as described in the next chapter.

Exceptionally, neurons can be accessed individually when their network is switch from the interactive Learn and Recognize mode to the passive Save and Restore mode.

3.6 Neuron Part 6: The behavior of a KNN or RBF classifier

The CogniMem neurons can have two different behaviors both important and field-proven in pattern recognition and machine learning: *K*-Nearest Neighbor algorithm (*k*-NN) and Radial Basis Function (RBF)

The ***k*-nearest neighbor algorithm (*k*-NN)** is a method for classifying objects based on closest training examples in the feature space. The parallel architecture of the CogniMem chip makes it the fastest candidate to retrieve the *K* closest neighbors of a vector of up to 256 bytes among ANY number.

- First of all, all the neurons update their distance value simultaneously as the component of the input vector are broadcasted on their parallel bus. Upon receipt of the last component of the input vector, all neurons have calculated its distance to the reference pattern they hold in memory. If an input vector of 200 bytes is broadcasted to a chain of 10 CM1K chips or 10,240 neurons, their 10240 distance values are calculated and ready to be read after the 200 clock cycles necessary to transmit the vector.
- The second advantage of the CogniMem network is that the neurons are then capable to order their response autonomously per increasing distance value as the host processor sends *K* successive read commands of the Distance register. Again, this unique feature pertains to the parallel architecture of the neurons and a patented Search and Sort process which allows them to know if other neurons have a smaller distance value without the need for a supervisor or controller. At each read command, only the neuron with the smallest distance outputs its value to the parallel bus after 19 clock cycles. If an application requires the use of a KNN with *K* equal to 50 for example, the distance values of the 50th closest neurons are read in 50 * 19 clock cycles.

A **radial basis function network** is a neural network that uses radial basis functions as activation functions. It is capable of representing complex nonlinear mappings and widely used for function approximation, time series prediction, and control.

- In RBF mode, the CogniMem neurons are capable of ranking similarities between input vectors and the reference patterns they hold in memory, but also reporting conflicting responses or cases of uncertainty, reporting unknown responses or cases of anomaly or novelty. The time necessary to obtain a response is independent from the number of committed neurons in the network and from their type of response.
- The model generator built-in the CogniMem network makes it possible to learn examples in real-time when they drift from the knowledge residing in the current neurons. The “novelty” examples can be stored in neurons assigned to a different context to allow a supervised verification and learning at a later time.
- The ability to assign the CogniMem neurons to different contexts or sub-network allows building hierarchical or parallel decision trees between sub-networks. This leads to advanced machine learning with uncertainty management and hypothesis generation.

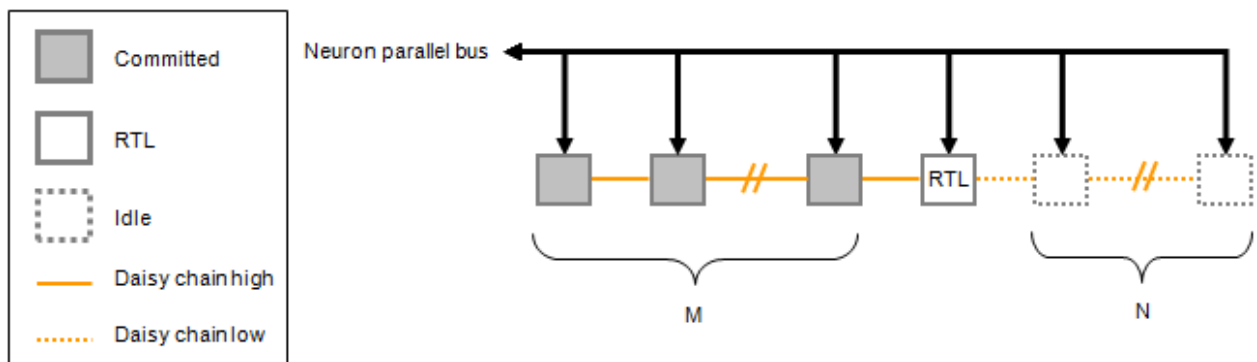
4 NETWORK ARCHITECTURE

The fully parallel architecture of CogniMem is made possible because all the neurons are identical and do not require any controller or supervisor to interact with one another. They receive the same instructions and data over the neuron parallel bus and execute them at the same time. The execution of certain instructions requires that the Ready-To-Learn (RTL) and Committed neurons consult the response of one another over the neuron parallel bus. This interaction is necessary to build a consistent and adaptive knowledge.

4.1 A chain of identical neurons

At initialization, the neurons are empty meaning that they do not have any knowledge. Their status is Idle except for the first one which is Ready-To-Learn (RTL). As examples are learned by the network, neurons are progressively used to store reference patterns along with their associated category and become Committed.

The state of a neuron in the chain can be Idle, RTL or Committed. It is defined by the status of its daisy-chain-in (DCI) and daisy-chain-out (DCO) lines. The DCO of a neuron rises if its DCI is high and its category register is different from 0. As a result, the commitment of neurons is propagated automatically as examples are taught and retained. The RTL neuron also moves along until no more Idle neuron is available in the chain.



The neural network is composed of $M+1+N$ neurons:

- M committed neurons holding a reference pattern and a category value
- 1 ready-to-learn (RTL) neuron
- N idle neurons

The behavior of a neuron is function of its state in the chain:

Neuron State	Idle	Ready-to-Learn	Committed
Behavior	Does not respond to input patterns, but updates its global registers such as Context, Minimum Influence Field and Maximum Influence Field.	Holds the last input pattern. If a category is taught and not recognized by any of the committed neurons, the RTL neuron stores the category and becomes committed. The next Idle neuron in the chain becomes the RTL neuron.	All committed neurons attempt to recognize an incoming vector. Once committed, a neuron can only shrink its Influence Field and set its Degenerated flag. Its status can return to Idle when it is instructed to forget.
Memory		x	x
Distance calculator		x	x
Associative logic			x
Learning logic		x	x
Save and Restore		x	x

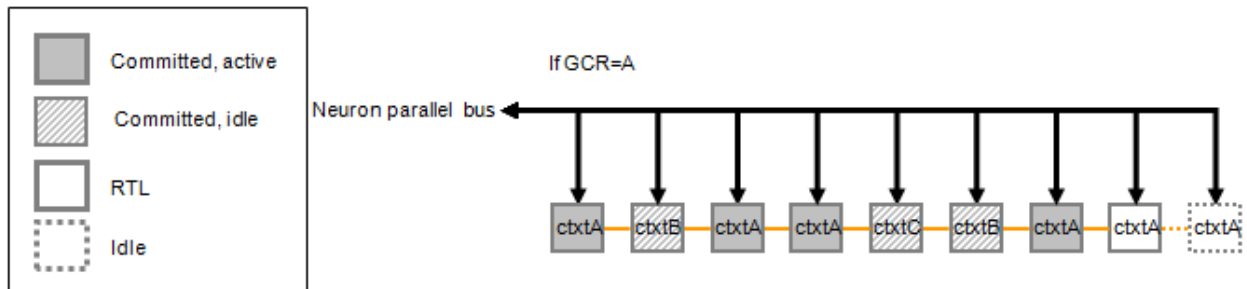
4.2 Network segmentation into context

The neurons can be associated to different contexts and their use can be enabled or disabled by selecting a context value. For example, if an application uses two sensors such as a microphone to record a voice signal and a camera to record a video signal, two contexts will be used to toggle between two sub-networks: the neurons trained to recognize input vectors deriving from a voice signal and the ones trained to recognize input vectors deriving from a video signal. Furthermore, if the video camera is an outdoor camera, it might be useful to train it differently as a function of time of the day. Indeed the images will have nothing in common between day and night and the sensor might require different filter, gain and shutter adjustments. A context Day and context Night will allow training two sub networks based on the time of the day. In summary, usage of the context allows segmenting the network per family of input data. This segmentation can be based on the model of the input sensor, the settings of the input sensor, the feature extracted from the sensor data, the data length, the time of collection of the data and more.

A context is selected by writing a context value to the Global Context Register (GCR) of the chip. Any committed neuron with its own context register different from the global context register turns idle and does not participate in any learning or recognition operation. One exception: the context value 0 enables all the neurons without regards of their context.

When a neuron gets committed, its Neuron Context Register (NCR) is set to the value of the Global Context Register (GCR). Whenever the GCR is changed, all the neurons with a different context value will not attempt to recognize any input vector, nor react to the learning of a new vector. They remain idle until the GCR is changed to a value matching their context value. A GCR equal to 0 activates all the neurons regardless of their context value.

The illustration below shows how neurons can be turned active or idle by writing the Global Context Register. In this example the GCR is set to the value A:



The neurons belonging to a given context make a recognition engine. If the network has neurons belonging to N different contexts, it means that it can switch between N different recognition engines prior to learning or recognizing an input pattern. Finally if the Global Context is set to the value zero, all engines will be used in conjunction to recognize the input pattern.

4.2.1 Multiple networks for combined decision criteria

Example 1: multiple zones of inspection in a part

If the good quality of a part can be decided based on the proper size, shape and position of N different components in a part, the neural network can be trained to recognize each component separately using one context per component. The final classification of the part is then based on the classification of each component and rules which can be more or less conservative or moderate. For example, a glass bottle can be shipped if its cap is properly twisted, the cap seal is in place, no contaminant is found at the bottom, and the filling level is above the minimum. It can then be diverted to a container for “Quality Assurance A” versus “Quality Assurance B” depending on its filling level.

Example 2: multiple features per object

If a material can be characterized by its colors and texture, two sets of vectors can be extracted from each sample: one describing its color distribution, and one describing its graininess. The classification of the material then relies on two contexts, or two decision spaces.

4.2.2 Multiple networks for hierarchical decision

The use of multiple networks trained on different features describing a same or different portions of an object allows generating hypotheses and building robust decision schemes. For example, an application with a high cost of mistake may require that at least N sub-networks produce a same classification in order consider the overall response as a positive identification.

Example 3:

A practical approach to recognizing moving targets such as vehicles in an outdoor scene may consist of learning relevant subsets of these targets such as a silhouette, a hood, headlights, a wheel, a tire, etc. Each subset is associated to a different network trained to deal with changes of scale and orientation. When analyzing the content of a new image, each network reports a map of the locations where it recognizes a pattern. The combination of these maps produces a “Transform” image which is much simpler than the original image and can be classified by a higher level context trained to verify the spatial distribution of the different subsets in order to produce a final decision. For example, it recognizes a car if it has a silhouette of type “Front View” which contains a hood, 2 headlights and 2 front views of a tire. It also recognizes a car if it has a silhouette of type “Side View” which contains 2 side views of a tire.

Example 4:

In predictive maintenance, the classification of anomalies can be processed hierarchically starting with the detection of any novelty by a top “conservative” engine (neurons of context#1) to make sure that nothing is discarded. The samples recognized as novelty trigger the use of a second engine trained to classify the data with a greater level of details. Based on its classification results, this engine can trigger other engines and so on until the novelty becomes a classified anomaly.

4.3 Save and Restore of the neurons

Once a decision space has been modeled and validated by recognizing many samples, the contents of the neurons can represent a valuable knowledge base and intellectual property.

A Save and Restore mode controlled through the Network Status Register allows to access the neurons sequentially and read/write their internal neuron registers and memory. The content of each neuron is saved in 263 bytes as follows: 256 bytes of vector, 1 byte of context, 2 bytes of minimum influence field, 2 bytes of influence field and 2 bytes of category. The knowledge which is defined as the contents of all the committed neurons occupies 261-bytes times the number of committed neurons. Note that it is important to associate the knowledge to the description of the feature extraction techniques producing the patterns stored in the neurons and also to the network settings such as the Minimum and Maximum Influence fields used during the training. This information can be stored in a header of the knowledge file.

A knowledge file can be built in advance using a system interfaced with a CogniMem silicon network or a simulation of the CogniMem network. The 263 bytes of information necessary per neuron can be loaded at a later time. This transfer should be preceded by a clear of the neurons whenever possible. Otherwise, it becomes a merger between two knowledge bases (the one residing in the neurons and the one to load) and it must be considered cautiously since their neurons could contradict each another. If the two knowledge bases use different contexts then merging them is always safe.

5 NEURON REGISTERS

The following table lists the neuron registers, their data type and their read/write access depending if the network is in normal or Save/Restore mode.

5.1 Neuron registers

	Description	Addr	Normal mode	SR mode	Value range/ Default
NCR	Neuron Context Register Bit [6:0]= Context value between 0 and 128 Bit[7]= neuron Norm , 0 for L1, 1 for Lsup * Bit[15:8] are used to store bit [23:16] of the neuron identifier	0x00		RW	16-bit*/ 0x0001
COMP	Component Writes to the neuron memory at the current index, updates the distance register and increments the index.	0x01	W	RW	8-bit/ 0x00
LCOMP	Last Component Writes to the neuron memory at the current index, updates the distance register and launch the neuron associative logic. The fire, ID, UNC flags and NSR registers are updated.	0x02	W		8-bit/ 0x00
INDEXCOMP	Component index Set the neuron memory index to an input value which can range between 0 and 255.	0x03	W	W	8-bit/ 0x00
DIST	Distance register. This register is updated by the neuron. Can range between 0 and 65535 (0xFFFF) A distance 0 means that the vector matches exactly the model of a firing neuron. The higher the distance, the farther the vector from the model. A distance of 0xFFFF means that no neuron recognizes the last input vector. Must be read after writing CM_LCOMP and before reading CM_CAT	0x03	R	R	16-bit/ 0xFFFF

	Description	Addr	Normal mode	SR mode	Value range/ Default
CAT	<p>Category register Can range between 1 and 32766 (0x7FFE) Bit 15 is a flag indicating if the firing neuron is degenerated.</p> <p>A category of 0xFFFF means that no neuron is firing and recognizes the last input vector.</p> <p>If category is greater than 32768, it indicates that the firing neuron is degenerated.</p> <p>Must be read after the DIST register.</p>	0x04	RW	RW	16-bit/ 0xFFFF
AIF	<p>Active Influence Field In normal mode, this register is updated by the learning logic of the neuron.</p>	0x05		RW	16-bit/ 0x4000
NID	<p>Neuron Identifier This register can be read after the category register.</p> <p>*bit[23:16] of the neuron identifier are stored in the unused upper byte of the NCR register.</p> <p>It is the subject of an Erratum at the end of this manual.</p>	0x0A	R	R	24-bit*/ 0x000000

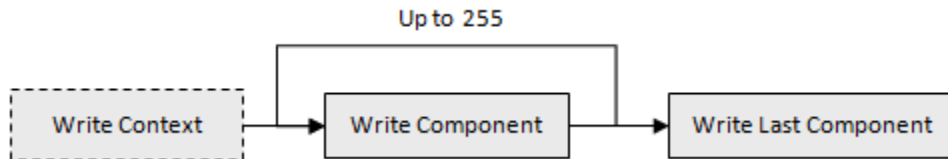
5.2 Global registers

	Description	Addr	Normal	SR	Value range/ Default
NSR	<p>Network Status Register</p> <p>Bit[1:0], reserved Bit[2], UNC (Uncertain) Bit[3], ID (Identified) Bit[4], SR mode (default is Normal mode) Bit[5], KNN classifier (default is RBF)</p> <p>ID and UNC are updated after each Write LCOMP command. ID is high if all firing neurons report the same category. UNC is high if several neurons fire but disagree with the category.</p> <p>When bit 4 is set to 1, the RTL neuron becomes the “indexed” neuron of the chain. When bit 4 is set to 0, the chain of neurons is re-indexed starting from identifier 1 and incremented by 1 until a neuron with category 0 is encountered.</p>	0x0D	R/W	W	16-bit/ 0x0000
GCR	<p>Bit [6:0]= Global Context Register Bit[7]= Norm , 0 for L1, 1 for Lsup</p>	0x0B	RW		16-bit/ 0x0001
MINIF	Minimum Influence Field	0x06	RW	R	16-bit/ 0x0002
MAXIF	Maximum Influence Field	0x07	RW		16-bit/ 0x4000
FORGET	Clear the neuron category (status becomes idle). Note that the neuron’s memory is not cleared, but its index is reset to point at the first component and this component will be overwritten by the next Write COMP.	0x0F	W		n/a
NCOUNT	<p>Normal mode: Number of committed neurons. Is equal to 0xFFFF if all neurons of the chain are committed.</p> <p>SR mode: Index of the neuron pointed in the chain. This index increments automatically after each Read or Write CAT, and is reset to 0 after a Write RESETCHAIN</p>	0x0F	R	R	16-bit
RESET CHAIN	Points to the first neuron of the chain	0x0C		W	n/a

6 FUNCTIONAL DIAGRAMS

6.1 Vector broadcasting

The memory of the neurons is 256 bytes long so the vectors to learn or recognize can be composed of up to 256 components of 8-bit value.



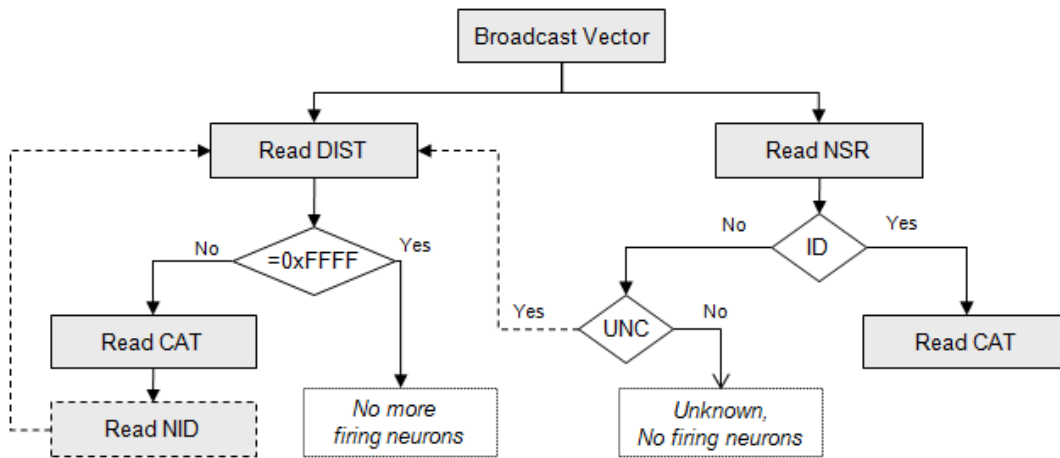
The broadcast of a vector to the neurons is made with the following sequence of operations:

- 1) Write Context
Optional, if the new vector must be associated to a context different than the current value of the Global Context or if the distance norm coded in bit 7 of the context must be changed
- 2) Up to 255 Write Component
Write all the components of the input vector but the last one in the Ready-To-Learn neuron. For all the committed neurons with a context equal to the Global Context, their distance register is updated after each Write Component according to the Norm in use.
- 3) 1 Write Last Component
For all the committed neurons with a context value equal to the Global Context, their distance register is updated and represents the distance between the input vector and the prototype stored in their memory. Furthermore, if their distance falls in their influence field, the neurons “fire” meaning that they are ready to respond to further inquiries about the vector such as reporting its distance and category.

6.2 Vector Recognition

The recognition of a vector is readily available after its broadcast to the neurons. Indeed the neurons which identify the vector as falling within the similarity domain of the vector stored in their memory have their “fire” flag raised.

The CogniMem network can exercise two types of classifiers: Radial Basis Function Network (RBF) or K Nearest Neighbor classifier (KNN). In either case, reading the response of the firing neuron can be done through various methods providing different levels of detail and speed performance are shown below.



If 2 neurons fire with the same distance but different category, their individual response are read as follows: Read Dist, Read Cat, Read Dist, Read Cat. The second Read Dist return the same value as the first Read Dist but is necessary to access the category register of the second neuron.

If 2 neurons fire with the same distance and same category, only the response of the first one is read. The first Read Dist will notify both neurons to stay in query, but both will output their category at the following Read Cat and therefore exclude themselves from the next query. A second Read Dist will return the next higher distance value if applicable.

6.2.1 KNN, always best match classification

In KNN mode, a vector is always recognized since the classifier discards the relationship between the distance and influence field of a neuron. As a consequence, all the neurons fire and their distance and category can be read in sequence per increasing order of distance. The first distance quantifies the difference between the input vector and the neuron with the closest pattern. The category of this neuron is the category with the highest confidence level. The second distance quantifies the difference between the input vector and the neuron with the second closest pattern. The category of this neuron is the category with the second highest confidence level, and so on for **K iterations**.

Remark: Using the neurons as a KNN classifier does not necessarily require to learn the vectors, but rather to load them with set of reference vectors with or without category labels. This can be done in Save and Restore mode and executes in lesser time since it does not require the use of the model generator internal to the neurons.

For more information on how to use the CogniMem neurons as KNN classifier, please refer to the chapter “CogniMem, high-speed KNN classifier”.

6.2.2 RBF, more subtle classification

The RBF classifier uses the Influence Field of the neurons at the time of the recognition. A neuron fires only if the distance calculated between the input vector and its memory is less than its influence

field. Also if multiple neurons recognize the input vector, the recognition status can be detailed as follow:

- Unknown: no neuron recognizes the input vector
- Identified: one or several neurons recognize the vector and agree with its category value
- Uncertain: one or several neurons recognize the vector but do not agree with its category value.

(Response Type 1) Conformity detection	(Response Type 2) Best match	(Response Type 3) Detailed matches
<p>If an application needs to detect if a pattern is recognized, or not, reading the recognition status register (NSR) is sufficient and takes one clock cycle.</p> <p>This response can be sufficient in the case of presence/absence detection, pass/fail, etc.</p>	<p>If an application has a low cost of mistake, reading the distance and category of the neuron with the best match can be sufficient.</p>	<p>If an application has a high cost of mistake, the response of all the firing neurons might be of interest to obtain a better accuracy. A global response can then be established using probability functions, dispersion of the distances, minimum number of aggregates, etc.</p>

6.2.3 RBF Response Type 1: Classification status

As soon as a vector is broadcasted to the neurons, the following recognition status is known:

- Unknown: no neuron recognizes the input vector
- Identified: one or several neurons recognize the vector and agree with its category
- Uncertain: one or several neurons recognize the vector but disagree about its category

6.2.4 RBF Response type 2: Best-match

The first neuron to respond is the firing neuron with the smallest distance value is equivalent to a best match. If its distance is equal to 0, it means that the vector matches exactly the prototype stored in the neuron. If the Recognition Status is Identified, the Best match is the only possible response. On the other hand, if the Recognition Status is Uncertain, other responses can be read from the neurons as described in the next paragraph.

6.2.5 RBF Response type 3: Multiple matches

Examining the distance and category of all the firing neurons can be of interest to reinforce the accuracy of a decision, especially in the cases of uncertainty. The first distance quantifies the difference between the input vector and the neuron with the closest pattern. The category of this neuron is the category with the highest confidence level. The second distance quantifies the difference between the

input vector and the neuron with the second closest pattern. The category of this neuron is the category with the second highest confidence level, and so on until all firing neurons have reported their response and the distance reads 0xFFFF.

Rules have to be established on a “per application” basis depending on the cost of a mistake, the requirements for a minimum throughput, minimum false negatives, etc.

6.3 Vector Learning

If this combined information (vector and category) represents novelty to the existing neurons, the Ready-To-Learn neuron becomes committed. It stores the instructed category in its category register. Its influence field of the new neuron is set to the smallest distance register of the firing neurons or the Minimum Influence Field whichever is greater. If there are no firing neurons at all, its influence field is set to the current value of the Maximum influence field.



The next neuron in the chain turns from idle to RTL (ready-to-learn). If there are neurons which recognized the vector with a category other than the instructed category, they automatically reduce their influence field to prevent such erroneous recognition in the future.

6.3.1 Global settings prior to learning

The following global registers affect the way the neurons will learn new vectors and model a decision space. Changing them should be done prior to broadcasting the vectors to learn.

Global Context	(to segment the network)
Maximum Influence Field	(to adjust conservatism)
Minimum Influence Field	(to control uncertainty domain)

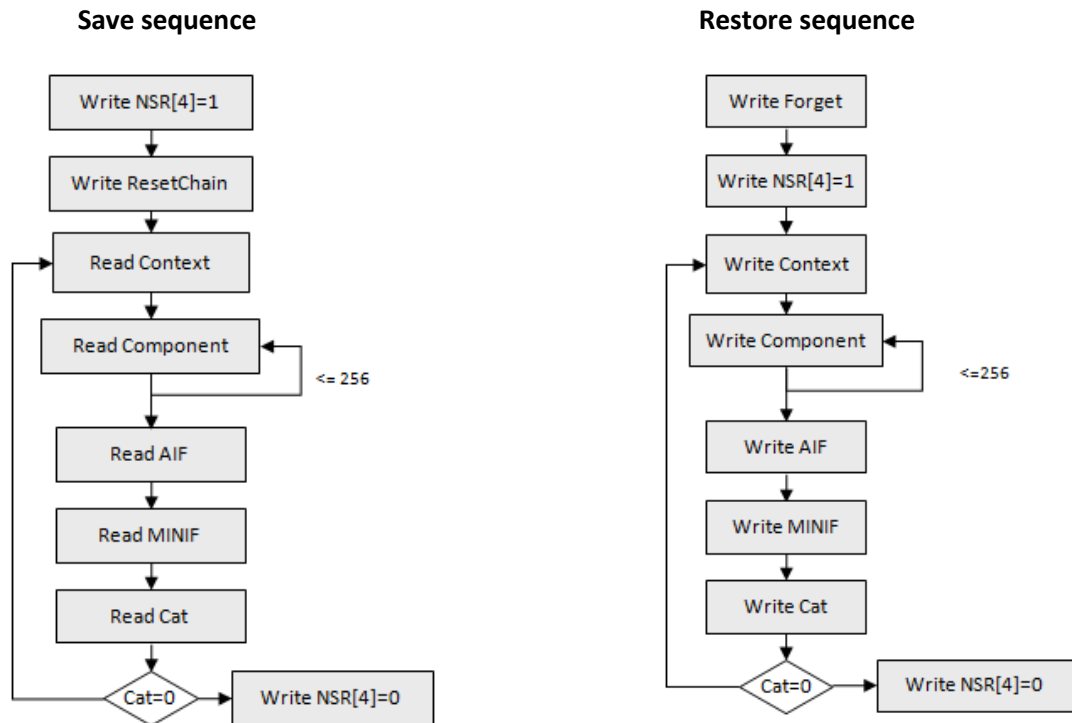
Remark: the minimum and maximum influence fields must be expressed in a dimension relevant to the dimension of the input vector.

6.3.2 Waiving dependency from the teaching sequence

The decision space is modeled as examples are taught and its shape depends on their sequence. Indeed, the order of the examples determines the commitment of new neurons and the shrinking of existing committed neurons. This temporal dependency is not desirable to build an accurate knowledge and it is recommended whenever possible to learn the examples repeatedly until the decision space is stable. This condition is established when no new neuron is committed between two iterations. The ability to execute an iterative learning requires that the examples can be archived which is technically possible in most applications.

6.4 Save and Restore sequences

The table below describes the typical sequences to save or restore the contents of neurons.



In both sequences, once the neurons are set to the Save and Restore mode by writing bit 4 of the NSR to 1, the Category register must be the last register to be read or written per neuron since access to this register increments the neuron pointer automatically. The order in which the other neuron registers (COMP, AIF and Context) are read is not important.

Note that the LCOMP register is not used. Indeed in Save and Restore mode, the neurons are passive and simply write or return the value of the requested register. The notion of Write LCOMP which triggers the interaction between all the neurons in Learn and Recognition mode does not exist.

In the Restore sequence, the initial Write Forget which resets the category of all the committed neurons to 0 is optional. If you do not execute it, this means that you are not loading a new knowledge but rather appending knowledge to an existing one. Please note that this operation requires caution and a good understanding of the contents of the neurons. Indeed after writing bit 4 of the NSR register to 1, the neurons are set to the "passive" Save and Restore mode and do not interact with one another to learn data and adjust their influence fields. Unless the knowledge to restore describes neurons with a different context than the neurons already committed in the network, restoring without forgetting can be risky and create a corrupted or inconsistent knowledge base.

Saving and restoring the MINIF of each neuron is necessary if it is known that additional training will be done at a later time to complete or expand the knowledge. This will ensure that the degenerated status of the neurons is properly flagged if appropriate.

6.5 Simple Example

The following table describes a sequence of learning and recognition operations on a set of vectors with 10 components:

Vector	Cmd	Description	1 st best match	2 nd best match
Vector 1= 0,1,2,3,4,5,6,7,8,9	Learn as 1	The 1 st vector is stored in a first neuron		
	Reco	Its recognition generates an “exact match”	Cat=1, Dist=0	Cat=0, Dist=-1
Vector 2= 0,1,2,6,4,5,6,7,8,9	Reco	The 4 th components of the 2 nd vector is different by a value 3. All other components are identical. Still the 1 st neuron recognizes the second vector.	Cat=1, Dist=3	Cat=0, Dist=-1
Vector 3= 0,1,4,3,8,5,12,7,16, 9	Learn as 2	The 3 rd vector is stored in a second neuron		
	Reco	Its recognition generates an “exact match” with the second neuron.	Cat=2, Dist=0	Cat=0, Dist=-1
Vector 4= 0,1,2,3,4,5,12,7,16, 9	Reco	The 1 st half of the 4 th vector matches V1 and the 2 nd half matches V3. Both neurons knowing V1 and V3 recognize V4. Neuron 2 gives the best match because it is closer with a distance of 6.	Cat=2, Dist=6	Cat=1, Dist=14
Vector 5= 0,1,2,3,4,5,6,7,	Reco	2/3 of the components matches the one of V1 and 1/3 matches V3. Neuron 1 gives the best match with a distance of 8.	Cat=1, Dist=8	Cat=2, Dist=12

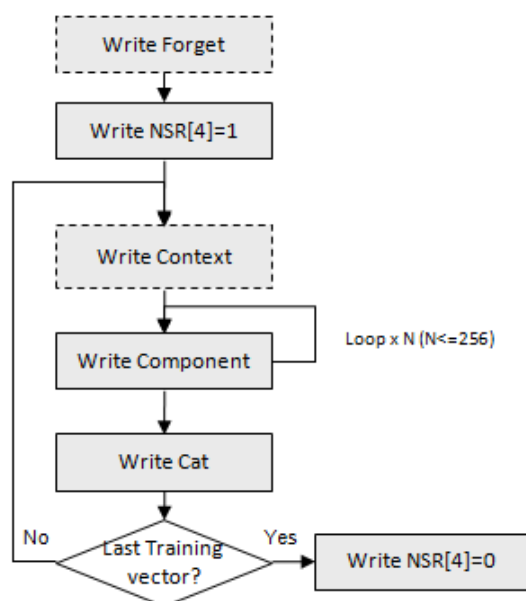
7 COGNIMEM, HIGH SPEED KNN CLASSIFIER

The k -nearest neighbor algorithm (k -NN) is a method for classifying objects based on closest training examples in the feature space. This paper explains how the parallel architecture of the CogniMem chip makes it the fastest candidate to retrieve the K closest neighbors of a vector among ANY number by (1) calculating distances in parallel and (2) sorting them in increasing order autonomously.

7.1 Loading the training examples

The training examples can be any number of vectors composed of up to 256 bytes. A vector can be a series of measurements with different dimensions and characterizing a population of objects. In some cases, the 256 bytes can be samples of the temporal evolution of a signal, or the spatial distribution of pixels, and more.

The training examples be loaded sequentially into the neurons using the Save and Restore (SR) mode of the CM1K chip. Under this mode, the neurons are passive and writing a neuron register takes one system clock cycle. The following diagram describes the simple sequence of commands loading the training examples to the neurons. Its execution time is solely proportional to the number of examples and independent of the architecture of the neural network whether it is composed of a single CM1K chip (1024 neurons) or a chain of 10, 100 or more CM1K chips daisy chained together.



The initial Write Forget resets the category of all the neurons to 0. If you do not execute this command, you will be appending the examples after the ones already loaded in the existing committed neurons.

The neurons are then set to the Save and Restore mode by setting bit 4 of the Neuron Status Register (NSR) to 1 and the first neuron of the chain becomes the "Ready-To-Load". The N bytes of the first example are loaded through a series of Write Component. After the N^{th} component, the Write Category assigns a value to the Category register of the neuron (default is 1). The latter becomes "Committed" and the next neuron in the chain becomes "Ready-To-Load". Once all examples have been loaded, the network is returned to its default Learn and Recognize mode by setting bit 4 of the NSR back to 0.

Depending on the execution, or not, of the two optional commands shown in dotted frames, the loading of the training examples will take $(N+1)$ or $(N+2)$ cycles per vector plus an overhead of 2 or 3 cycles.

7.1.1 Smart Category assignment

Whenever it is possible, a good usage of the category register consists of assigning a different category value to each training example. This will ensure that if the recognition of a vector causes two or more neurons to report a same distance their response will be discriminated from one another by their category value (refer to the paragraph “Limitations of the CM1K chip” later in this chapter).

If the training examples have pre-defined categories which can be encoded on less than 15 bits, it is a good idea to edit the unused bits to include some indexing into the neuron’s category register. Again, this will minimize the probability that later multiple neurons fire with the same distance and same category and be accounted as one neighbor.

7.1.2 Optional Context usage

Writing the Context register is optional and only useful if you wish to load at once several sets of training vectors which are not related. The vectors can for example derive from different data sources, or a same data source but different feature extraction techniques. Under these circumstances the context can be an index to a table describing how the vector was obtained. Another usage of the context can be to encode the length of the vector, a time stamp, or else.

At the time of the recognition, the context of the input vector has to be assigned to the proper value so only the neurons belonging to the same context participate to the recognition.

7.1.3 Learning, not loading, the examples

It is possible to load the neurons with training vectors using the default Learn and Recognize mode. This method will take longer with $N+19$ clock cycles per vector instead of $N+1$ and create a decision space where zones of unknown and uncertainties can exist. Nevertheless, these zones will be discarded when it is time to recognize a vector with the KNN classifier since all the neurons fire whatever their influence field.

When loading examples in Save and Restore mode, the number of committed neurons is equal to the number of examples. Furthermore since the neurons are passive and not reactive, erroneous inputs (if any) will not be detected including duplicate examples, inconsistent examples, etc.

When learning examples in the Learn and Recognition mode, the neurons are reactive and only retain examples which bring novelty at the time they are learned. This means that the committed neurons are directly related to the order of the examples. This dependency can be waived by learning them several times until no new neuron gets committed between two passes. Other factors affecting the learning behavior of the neurons are the Maximum and Minimum Influence Fields of the network. These parameters are global registers and help moderate the conservatism of the neurons.

7.2 Closest K neighbors in microseconds

7.2.1 Broadcast the vector to all neurons

A new vector can be broadcasted to all of the neurons in parallel through a series of Write Component and a Write Last Component commands. After each input component, the neurons update their distance registers automatically according to the norm defined by bit 7 of their context register. Upon receipt of the last component their distance register gives the distance between the input vector and the reference pattern they hold in memory.

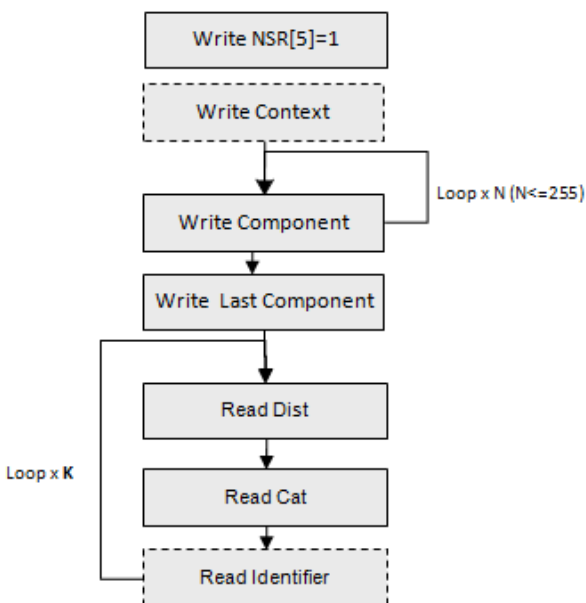
If an input vector is composed of 200 bytes and broadcasted to a single CM1K chip, the 1024 distances of the 1024 neurons in the chip are available after 200 + 2 clock cycles. Similarly, if the network is a chain of 10 CM1K chips or 10,240 neurons, the 10,240 distance values are also available after the 200 +2 clock cycles.

7.2.2 Search and Sort in a fixed amount of time

Once a new vector is broadcasted to all the neurons, the response of the top K neurons with the best matches is read out through K successive Read Distance and Read Category.

The autonomous sorting mechanism of the neurons is a key feature pertaining to their parallel architecture and a patented Search and Sort algorithm which allows each neuron to know if other neurons have a smaller distance value without the need for a supervisor or controller. In such case, the neuron holds its response letting the one with a smaller distance respond first.

The value K must be less than or equal to the number of committed neurons in the network.



Setting bit 5 of the Network Status Register to “1” switches the behavior of the neurons from the default Radial Basis Function classifier to the K-Nearest Neighbor classifier.

The initial Write Context command is optional as described in the previous chapter.

The Read Category command is mandatory even if its value has no particular interest. If it is not executed the neuron will remain in the race for the next Search and Sort thus preventing the retrieval of the following best matches.

The Read Identifier can be optional and returns the index of the firing neuron. This index is assigned internally by the neurons at the time they

get committed. Reading the identifier is useless if the category value has been set to the example's index number as recommended in the earlier paragraph "Smart category assignment".

7.2.3 Limitations of the CM1K chip (first generation)

The response of the firing neurons is ordered per increasing distance and for the same distance per increasing category. The CM1K chip does not provide for the subsequent sorting per identifier.

If one or more of these neurons have the same distance and same category, the readout of the Distance register followed by the Category register will exclude them all at once from the next search and sort. If you are interested in surveying the histogram of the distances and a probability density function, this means that the neurons with the same distance and same category will be accounted as one and produce incorrect results.

In conclusion, it is highly recommended to encode a different category value for each training example. Refer to the Erratum chapter of the CM1K Hardware Manual for more information about the limitations of the CM1K chip.

7.3 CM1K Timing Benchmarks

The following table reports the number of clock cycles to access the registers useful for a KNN vector classification:

Register	Read	Write
Context	n/a	1
Component	1	1
Last Component	n/a	3
Distance	18	n/a
Category	3 if identified, 19 otherwise	1 if identified, 19 otherwise
Identifier	1	

7.3.1 Step 1: Broadcast a vector with N dimensions

Shortest Timing: $N+2$ (without changing the Context register)
 Longest Timing: $N+3$ (with change of the Context register)

7.3.2 Step 2: Readout of the K Nearest Neighbors

Shortest Timing: $K * 22$ (all neurons loaded with a same category, request to read the identifier)
 Longest Timing: $K * 39$ (all neurons loaded with a different category, request to read the identifier)

7.3.3 Example

N=96, K=20

CM1K clocked at 27 MHz or 37 nanoseconds per clock cycle.

The broadcast of a single vector of 96 values takes 3.63 microseconds.

The “shortest” readout of the 20 nearest neighbors of a single vector takes 16.30 microseconds.

8 WHAT IS NEW IN THIS MANUAL ?

8.1 Revision from 09/23/2011

- Addition of a comment regarding the [0, 0x7FFE] value range of the registers CM_CAT and CS_CATL