

V1KU-RTL-SDK

Register Transfer Level protocol
for the V1KU board

Version 1.3.1



Distributed by General Vision Inc.

Revised Nov 2009

License Agreement

The V1KU SDK is published by General Vision

This License Agreement is a legal agreement between you (LICENSEE) and General Vision Ltd for the enclosed computer software, and any associated media, printed materials, and "online" or electronic documentation ("SOFTWARE PRODUCT").

License, rights and limitations

General Vision grants to the LICENSEE a revocable, non-transferable, and non-exclusive license to use the SOFTWARE PRODUCT. This license may not be shared or used concurrently on different computers. You may install one copy of the SOFTWARE PRODUCT, or any prior version, on a single computer for use by a single user. LICENSEE may make one copy of SOFTWARE PRODUCT for backup purposes. No other copies shall be made without General Vision 's prior written consent.

Ownership and Copyright

The SOFTWARE PRODUCT, including all supporting documentation, is a proprietary product of General Vision. General Vision retains all title and copyrights in and to the SOFTWARE PRODUCT, the accompanying printed materials or electronic documentation and any complete or partial copies of the SOFTWARE PRODUCT.

Limited Warranty

General Vision warrants that the SOFTWARE PRODUCT will perform substantially in accordance with the accompanying written materials or electronic documentation and General Vision will make reasonable efforts to solve any problem issues arising out of the intended use of the SOFTWARE PRODUCT. To the extent allowed by applicable law, implied warranties on the SOFTWARE PRODUCT, if any, are limited to ninety (90) days. General Vision disclaim all other warranties and conditions, either express or implied, including, but not limited to, implied warranties of merchantability, fitness for a particular purpose, with regards to the SOFTWARE PRODUCT.

Limitation of Liability

To the maximum extent permitted by law, in no event shall General Vision be liable for any special, incidental, indirect, or consequential damages whatsoever (including, without limitation, damages for loss of business profits, business interruption, loss of business information, or any other pecuniary loss) arising out of the use of or inability to use the SOFTWARE PRODUCT or the provision of or failure to provide support services, even if General Vision has been advised of the possibility of such damages. In any case, General Vision' entire liability under any provision of the license agreement shall be limited to the amount actually paid by you for the SOFTWARE PRODUCT.

Terms and termination

This license agreement is effective from the date of receipt and shall remain in full force until terminated. Without prejudice to any other right, General Vision may terminate this LICENSE AGREEMENT if you fail to comply with any of the terms and conditions of this license agreement. In such event, you must destroy all complete and partial copies of the SOFTWARE PRODUCT in your possession.

istribution

The distribution of an application written with the SOFTWARE PRODUCT requires the purchase of one run-time license per system.

Contact

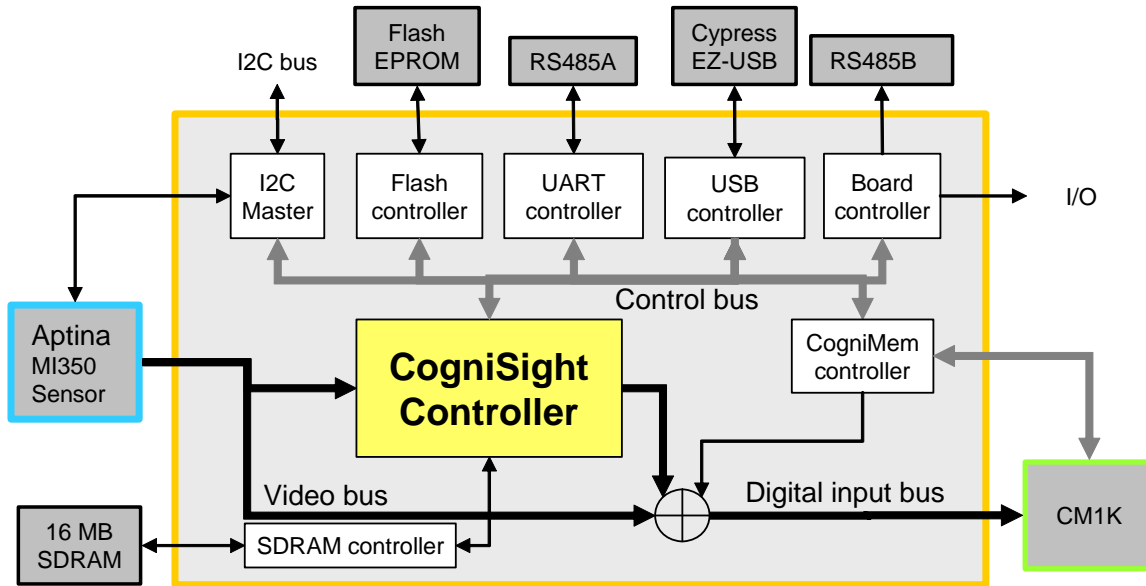
www.general-vision.com

Table of Content

LICENSE AGREEMENT	2
TABLE OF CONTENT	3
INTRODUCTION	4
THE COMMUNICATION PROTOCOL	5
HARDWARE SPECIFICATIONS	5
<i>RS485 interface</i>	5
<i>USB interface</i>	5
PACKET PROTOCOL DESCRIPTION	5
<i>Write command (Addr[31]=1)</i>	5
<i>Read command (Addr[31]=0)</i>	5
REGISTERS DESCRIPTION	6
TERMINOLOGY	6
COGNISIGHT CONTROLLER	7
COGNIMEM CONTROLLER	8
<i>Neurons and network registers</i>	8
<i>Recognition logic registers</i>	9
BOARD AND IO CONTROLLER	10
I2C MASTER CONTROLLER	10
MEMORY ACCESS CONTROLLER	10
FLASH CONTROLLER	11
EXAMPLES	12
LEARNING A REGION OF INTEREST AND RECOGNIZING IT IN THE FULL VIDEO FRAME	12
LEARNING AND RECOGNIZING A VECTOR WITH V1KU	13

Introduction

The block diagram below presents the different modules programmed in the FPGA of the V1KU board at factory settings. Their access is made through the map of registers presented below.



The various components of the V1KU board are easily accessible through Read and Write commands at the register level.

Address Range	Controller	Description
0x01000000 0x0100001F	CogniMem	Access to the CM1K neurons to learn and recognize vectors, save and restore knowledge. Also access to the recognition logic in bypass and video mode.
0x02000020 0x20000024	Board info	Access to information about the hardware and configuration of the opto-isolated lines.
0x03000030 0x03000036	Flash memory	Access to the Flash memory to read and write pages of data, but also to read and decode sequences of instructions for the other modules.
0x04000000 0x0400FFFF	I2C master	Access to the registers of the Aptina sensor.
0x05000000 0x05FFFFFF	Memory access	Access to the SRAM address bank. CogniSight uses the RAM to store video frames digitized by the Aptina sensor, or to images transferred from a host.
0x06000000 0x06000076	CogniSight	Access to the CogniSight recognition engine to learn and recognize a region of interest, find and report known patterns in a region of scan.

The communication protocol

The protocol to send Read and Write commands to the board is the same for both the USB controller and RS485 controller programmed in the board's FPGA.

Hardware specifications

RS485 interface

V1KU can interface to the RS485 port A at 921,600 baud with Data bit=8, Stop bit=1, Parity= None, Handshake=None.

USB interface

The USB bulk transfers of the V1KU board are handled by the Cypress EZ_USB chip (CY7C68014, 56-pin model). Packets from host to V1KU must be sent to the end point EP2_OUT. Packets from the device to the host are received on the end point EP6_IN. The EZ_USB chip features an CPU 8051 and internal RAM. The CPU implements the high-level USB protocol by servicing host requests over the control endpoint, but can also be used for general-purpose operations. Drivers and USB development tools can be downloaded from the Cypress web site. Cypress includes an evaluation version of the 8051 Keil Software Tools in the USB 2.0 development kit.

Packet protocol description

The communication protocol of the V1KU board is based on the following packet sequence which is interpreted by both the USB and RS485 controllers:

1 byte	4 bytes	3 bytes	N words
Device ID*	bit[31]= RW command 0, Read 1, Write bit[30-24]= Controller ID bit[23-0]= Register	Datalength expressed in bytes. Default is 2.	Sequence of N data words N must be equal to Datalength ÷ 2. Sent if bit[31]=1, Received if bit[31]=0

A single read or write command takes 10 bytes. A multiple read or write command takes $8+N*2$ bytes

*In the case of the RS485 protocol, the Write command returns one byte of acknowledgement equal to the DeviceID.

Write command (Addr[31]=1)

Example: Write the value 0x0055 to the register 0x0B of the controller 0x01 of the device 0x01

0x01 0x0100000B 0x000002 0x0055

Returns None in the case of a USB communication
0x01 in the case of an RS485 communication

Read command (Addr[31]=0)

Example: Read the value of the register 0x23 of the controller 0x02 of the device 0x01

0x01 0x82000023 0x000002

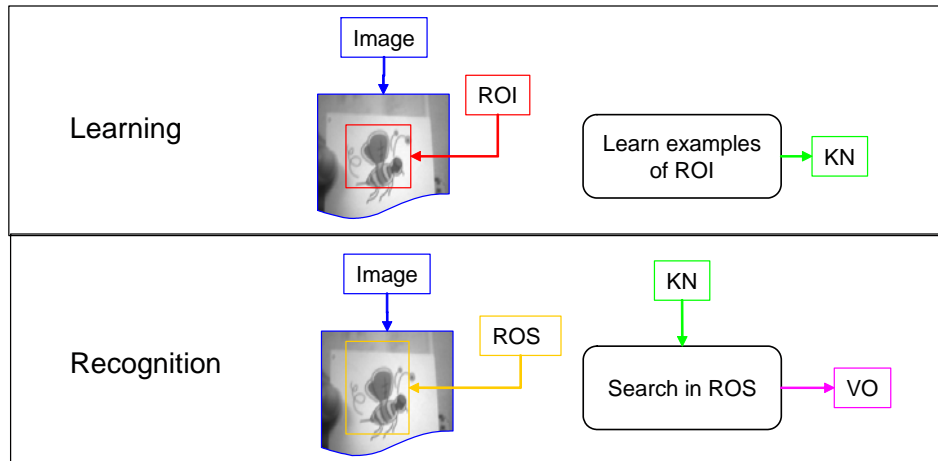
Returns 0x0040 (i.e. board's serial number)

Registers description

Terminology

The CogniMem and CogniSight controllers of the V1KU manipulate five simple components:

- Image
- ROI or Region of Interest
- KN, or Knowledge
- ROS, or Region of Search
- VO, or Visual Objects



Learning examples selected in images triggers the automatic model generator of the CogniMem chip. The contents of the neurons produces a knowledge which can be saved and displayed.

Reading the response of the CogniMem neurons over many different positions in an image produces a map of "recognized" Visual Objects which can be further studied and consolidated to produce a final decision.

For more information about the components Image, ROI, ROS, KN and VO, please refer to the introduction of the V1KU_CS_SDK manual.

CogniSight controller

Register	Hex	Description	Default	Access
CS_CSR	0x60	Control Status Register Bit [0], grab Bit [1], recognize ROI Bit [2], learn ROI Bit [3], scan the ROS and append the position and category of all recognized ROIs in a "hit" list.	0	R/W
CS_LEFT	0x61	Left position of the ROI	200	R/W
CS_TOP	0x62	Top position of the ROI	120	R/W
CS_RECODIST	0x63	Distance of the last processed* ROI	0xFFFF	R
CS_RECOCAT	0x64	Category of the last processed* ROI	0	R
CS_CATL	0x65	Category to learn	1	R/W
CS_ALEFT	0x66	Left position of the ROS	0	R/W
CS_ATOP	0x67	Top position of the ROS	0	R/W
CS_AWIDTH	0x68	Width of the ROS	752	R/W
CS_AHEIGHT	0x69	Height of the ROS	480	R/W
CS_HITCOUNT	0x6A	Number of identified ROIs in the ROS		R
CS_HITLEFT	0x6B	Left position of the next ROI in the hit list. Reading this register triggers the update of the associated registers HITTOP, HITDIST and HITCAT.		R
CS_HITTOP	0x6C	Top position of the next identified ROI. Must be read after HITLEFT.		R
CS_HITDIST	0x6D	Distance of the next identified ROI. Must be read after HITLEFT.		R
CS_HITCAT	0x6E	Category of the next identified ROI. Must be read after HITLEFT.		R
CS_INIT	0x6F	Reset all the above to their default values		R
CS_PAGENUM	0x70	Page number in the memory map to store a video frame. Can range from 0 to 63. Default=0	0	RW
CS_FWIDTH	0x71	Width of the image in memory	752	RW
CS_FHEIGHT	0x72	Height of the image in memory	480	RW
CS_STEPX	0x73	Horizontal scanning step	16	RW
CS_STEPLY	0x74	Vertical scanning step	16	RW
CS_RSR	0x75	Recognition Status Register Bit[0]=0, report all recognized objects during a search (CS_CSR=8) Bit[0]=1, report all unknown objects Bit[1]=1, report objects recognized with certainty Bit[2]=1, report objects recognized with or without uncertainty Bit[8]=1, do not use the CogniSight reco-logic, but use the CogniMem reco-logic (results must be read from the RTDIST and RTCAT registers of the CogniMem controller, automated Search mode is not available)		RW

*Processed ROI means either recognized (CS_CSR[1]) or learned (CS_CSR[2]) or last scanned (CS_CSR[3])

Remark: Whether you are launching a learning, recognition or scanning operation, the position of the ROI is automatically transmitted/generated from the CogniSight controller to the CogniMem controller. However, its size must be set in advance in the CogniMem controller through the registers CM_NWIDTH, CM_NHEIGHT, CM_BWIDTH, CM_BHEIGHT.

CogniMem controller

Neurons and network registers

Register	Hex	Description	Default	Access
CM_NCR	0x00	Bit [6:0], neuron context Bit [7], norm Bit [15:8], neuron identifier[23:16]	0	R/W in SR mode
CM_COMP	0x01	Component	0	W, R/W in SR mode
CM_LCOMP	0x02	Last Component	0	W, R/W in SR mode
CM_DIST	0x03	Distance. Can range between 1 and 65535. A distance 0 means that the vector matches exactly the model stored in the top firing neuron. The higher the distance, the farther the vector from the model. A distance of 32768 (0xFFFF) means that no neuron has fired and the vector is not recognized. Must be read after the Write CM_LCOMP.	0xFFFF	R
CM_CAT	0x04	Bit [14:0], category value ranging between 1 and 32767. Bit[15], Degenerated flag which indicates that the vector is recognized, but close to a zone of uncertainty. If category is equal to 65535 (xFFFF), the vector is unknown. If category is greater than 32768, it indicates that the firing neuron is degenerated. Must be read after the CM_DIST register.	0xFFFF	R/W
CM_AIF	0x05	Active Influence Field	0x4000	R/W in SR mode
CM_MINIF	0x06	Minimum Influence Field	2	R/W
CM_MAXIF	0x07	Maximum Influence Field	0x4000	R/W
CM_NID	0x0A	Neuron Identifier[15:0] (upper byte is stored in NCR) Must be read after the CM_CAT register.	0	R
CM_GCR	0x0B	Global Norm and Context	1	W
CM_RESET CHAIN	0x0C	Point to the 1 st neuron in SR mode		W
CM_NSR	0x0D	Network Status Register Bit [2], Uncertain status Bit [3], Identified status Bit [4], SR mode (default=LR mode) Bit [5], KNN classifier (default=RBF classifier)	0	R/W
CM_FORGET	0x0F	Clear the neuron registers, but not their memory. Also reset the Minif, Maxif and GCR global registers. Does NOT reset the NSR register.		W
CM_NCOUNT	0x0F	Return the number of committed neurons		R

Recognition logic registers

CM_RSR	0x1C	Recognition Status Register Bit [0], real-time recognition logic ON Bit [1], output bus enable Bit [2], Uncertain recognition status Bit [3], Identified recognition status Bit [4], Frame valid if VI_EN=1, Feature valid if VI_EN=0 Bit [5], recognition in progress	0	R/W
CM_RTDIST	0x1D	Real-Time Distance, or distance of the neuron with the best match. This value is updated after the recognition of each vector received on the digital input bus and ready when the DIST_VAL pulse rises.	0xFFFF	R
CM_RTCAT	0x1E	Real-Time Category, or category of the neuron with the best match. This value is updated after the recognition of each vector received on the digital input bus and ready when the CAT_VAL pulse rises.	0xFFFF	R
If VI-EN...				
CM_LEFT	0x11	Left position of the ROI in the digital input video frame	200	R/W
CM_TOP	0x12	Top position of the ROI in the digital input video frame	120	R/W
CM_NWIDTH**	0x13	Width of the ROI*	340	R/W
CM_NHEIGHT	0x14	Height of the ROI*	220	R/W
CM_BWIDTH	0x15	Width of the inner block*	20	R/W
CM_BHEIGHT	0x16	Height of the inner block*	20	R/W
CM_ROIINIT	0x1F	Reset the ROI to default		W

**Requirement:

If $NWIDTH = n * BWIDTH$ and $NHEIGHT = m * BHEIGHT$ then $n \times m$ must be less than or equal to 256. Optimally, the region $[NWIDTH, NHEIGHT]$ should be divided into 256 blocks of $[BWIDTH, BHEIGHT]$. The pixels of block #i are averaged to produce the ith component of the signature vector.

Board and IO controller

The Board Controller is #2 and its registers range between 0x20-0x24. This controller reads the board information including its serial number and firmware version. It also allows to configure the board's opto-coupled output lines.

Register	Hex	Description	Default	Access
EB_CSR	0x20	Control Status Register Bit [0], opto lines output 0, disabled 1, enabled Relay0 = CM_RTCAT bit 0 Relay1= CM_RTCAT bit 1 Bit [1], opto lines format 0, sustained 1, pulsed (de-asserted at rising edge of CM_FV) Bit [8], disable UART controller	0	RW
EB_CHAINID	0x21	Reserved (chainID number)		
EB_SLA	0x22	Address of the I2C slave address	158	R
EB_SN	0x23	Board serial number		R
EB_FPGAREV	0x24	Version of the firmware in the FPGA		R

I2C Master controller

This I2C master controller is #4 and its registers can be any of the registers of the Aptina MI350 sensor. The most commonly used registers are listed below and the complete list is available in the MI350 user's manual (former Micron part number is MT9V022).

Register	Hex	Description	Default	Access
MI_RESET	0x0C	Sensor reset. Does not reset the I2C registers	0	RW
MI_GAIN	0x35	Gain between 16 and 64.	16	RW
MI_SHUTTER	0x0B	Exposure time in number of lines, between 1 and 480	480	RW
MI_AGC	0xAF	Automatic Gain (bit0) and Automatic Exposure (bit1)	3	RW
MI_TRIGGER	0x07	Bit 4, trigger	904	RW

Example 1: Read the shutter speed of the Micron sensor on the device 0x01
0x01 0x0400000B 0x000002 (returns 480 by default)

Example 2: Set gain to value 16 (0x10)
0x01 0x04000035 0x000002 0x00010
returns 0x01 in the case of RS485 protocol

Memory access controller

The memory controller uses the ControllerID 0x05. In this case the 24-bit register value is the 24-bit address of the SDRAM to access.

Example 1: Write a series of 6 bytes starting with the value 16 at the address 0x334455 on the device 0x01
0x01 0x85334455 0x000006 0x10 0x11 0x12 0x13 0x14 0x15 0x16
returns 0x01 in the case of RS485 protocol

Example 2: Read a series of 4 bytes at the address 0x112233 on the device
0x01 0x05112233 0x000004
return 4 bytes

Flash controller

The Flash Controller is #3 and its registers range between 0x30-0x3F. This controller allows to select a page of Flash memory to either read or write data. Two advanced Read modes are also available (1) to execute a script saved to Flash or (2) to restore the contents of neurons.

Register	Hex	Description	Default	Access
FLASH_STOP	0x30	Release the Chip_Select line of the Flash (to terminate one of the following transactions)		
FLASH_R_STA	0x31	Page number from which the next data will be read. This value can range from 0 to 2048 since the Flash memory is organized in 4096 pages of 264 bytes.		W
FLASH_R_DATA	0x32	Read the next 2 bytes of the Flash memory		R
FLASH_W_STA	0x33	Page number to which the next data will be written		W
FLASH_W_DATA	0x34	Write the next two bytes of the Flash memory		W
FLASH_R_EXEC	0x35	Read the page per increment of four bytes at a time and execute these bytes as a command with Byte0= RW bit + Controller[6:0], Byte1= Register[7:0] + Byte2-3= Data[15:0]. The instruction iterates until Controller=0xFF and then releases the Chip_Select line of the flash. This command must be preceded by the Flash_R_STA command.		W
FLASH_R_NEUR	0x36	Read the page per increment of 264 bytes at a time and restore these bytes as the content of a neuron with Byte0-255= CM_COMP, Byte 256-257= CM_NCR, Byte 258-259= CM_AIF, Byte 260-261= CAT. This function starts by clearing the neurons and setting CM1K in Save and Restore mode. It iterates through consecutive pages of Flash until Byte 262-263=0xFFFF. It then releases CM1K from its Save and Restore mode and then releases the Chip_Select line of the flash memory. This command must be preceded by the Flash_R_STA command.		W
FLASH_RESTORE	0x37	Restore a complete project or CKF, if properly saved to Flash memory with the function Flash_SaveCKF supplied in the Windows API.		W

Example 1: Write an array of data to a page of Flash memory
 Write FLASH_W_STA, page_num
 For i=0 to N: Write FLASH_W_DATA, data(i): Next i
 Write FLASH_STOP

Example 2: Read an array of data from a page of Flash memory
 Write FLASH_R_STA, page_num
 For i=0 to N: Read FLASH_W_DATA, data(i): Next i
 Write FLASH_STOP

Examples

Learning a region of interest and recognizing it in the full video frame

Define a region of interest with a size of 32 x 32 divided into internal blocks of 2 x 2

Sequence	Commands
Write CM_NWIDTH, 32	0x01 0x81000013 0x000002 0x0020
Write CM_NHEIGHT, 32	0x01 0x81000014 0x000002 0x0020
Write CM_BWIDTH, 2	0x01 0x81000015 0x000002 0x0002
Write CM_BHEIGHT, 2	0x01 0x01000016 0x000002 0x0002

Grab a video frame to memory and move the region of interest to the location (10,10). Learn as category 33 and read the number of committed neurons.

Sequence	Commands
Write CS_CSR, 1	0x01 0x86000060 0x000002 0x0001
Write CS_LEFT, 10	0x01 0x86000061 0x000002 0x0020
Write CS_TOP, 10	0x01 0x86000062 0x000002 0x0002
Write CS_CATL, 33	0x01 0x86000065 0x000002 0x0033
Write CS_CSR, 4	0x01 0x86000060 0x000002 0x0004
Read CM_NCOUNT	0x01 0x0100000F 0x000002 (returns 0x0001)

Grab a new frame and recognize the region of interest

Sequence	Commands
Write CS_CSR, 1	0x01 0x86000060 0x000002 0x0001
Write CS_CSR, 2	0x01 0x86000060 0x000002 0x0002
Read CS_RECOCAT	0x01 0x86000064 0x000002 (returns 0x0033)
Read CS_RECODIST	0x01 0x86000063 0x000002 (returns 0x0000)

In the same frame, define a region of search (5, 3, 128, 256), scan it with a step of 4 and read the number of identified regions.

Sequence	Commands
Write CS_ALEFT, 5	0x01 0x86000066 0x000002 0x0005
Write CS_ATOP, 3	0x01 0x86000067 0x000002 0x0003
Write CS_AWIDTH, 128	0x01 0x86000068 0x000002 0x0080
Write CS_AHEIGHT, 256	0x01 0x86000069 0x000002 0x0100
Write CS_STEPX, 4	0x01 0x86000073 0x000002 0x0004
Write CS_STEPY, 4	0x01 0x86000074 0x000002 0x0004
Write CS_CSR, 8	0x01 0x86000060 0x000002 0x0008
Read CS_HITCOUNT	0x01 0x0600006A 0x000002 (returns N)

Learning and recognizing a vector with V1KU

Let's take the example of an input vector equal to a series going from 00 to 99.

Broadcast the vector, learn as category 33 and read the number of committed neurons.

Sequence	Commands
For (i = 0; i<98, i++) Write CM_COMP, Vector(i);	0x01 0x81000001 0x000063 0x00 0x01 0x02 ...0x63
Write CM_LCOMP, Vector(99)	0x01 0x81000002 0x000002 0x0064
Write CM_CAT, 33	0x01 0x81000004 0x000002 0x0033
Read CM_NCOUNT	0x01 0x0100000F 0x000002 (returns 0x0001)

Broadcast the same vector and read successively the distance and category registers until the response is equal to xFFFF meaning that all firing neurons have reported their results.

Sequence	Commands
For (i = 0; i<98, i++) Write CM_COMP, Vector(i);	0x01 0x81000001 0x000063 0x00 0x01 0x02 ...0x63
Write CM_LCOMP, Vector(99)	0x01 0x81000002 0x000002 0x0064
Read response of 1 st neuron: Read CM_DIST	0x01 0x01000003 0x000002 (returns 0x0000)
Read CM_CAT	0x01 0x01000004 0x000002 (returns 0x0033)
Read response of 2 nd neuron: Read CM_DIST	0x01 0x01000003 0x000002 (returns 0xFFFF)
Read CM_CAT	0x01 0x01000004 0x000002 (returns 0xFFFF)