NeuElab soft Ip Design Specifications

History:			
Revision	**Date**	**Author**	**Description**
v1_00_a	May 30th 2013	F. Diotalevi	First **Draft Release**
v1_01_a		F. Diotalevi	Adapted to be compliant with the SiCode project. Added the control of PowerDown in the CTRL register.
V2_00_a		F. Diotalevi	Added chip_type bit in CTRL_REG
V2_01_a	April 13rd, 2015	F. Diotalevi	Reduced clock of mn256 bias module to (about) 1MHz. Added details in Latch time field description
V2_02_a	May 6th, 2015	F. Diotalevi	Added settling time register and settling time in stimuli and reception of data. This modification fixes the requests of Vito De Feo after Capocaccia meeting.
V2_03_a	May 25th, 2015	F. Diotalevi	Added Read/Only TimeStamp register.

TABLE OF CONTENTS

1 Introduction

The NeuElab Ip is aimed to interface up to 4 NeuroMorphic Chip (NMC) to an embedded microprocessor using AXI 4 lite protocol and Axi Stream I/f for connecting the NeuElab to a DMA. The interconnection of the NMCs to the NeuElab is performed in a priority way. The NMCs are connected in such a way to share the most number of signals aimed to minimize the number of required FPGA I/O. To go deeper in the details of the number of IO pins used by the NeuElab architecture to interfacing with a NeuroMorphic Board, see Section 5.

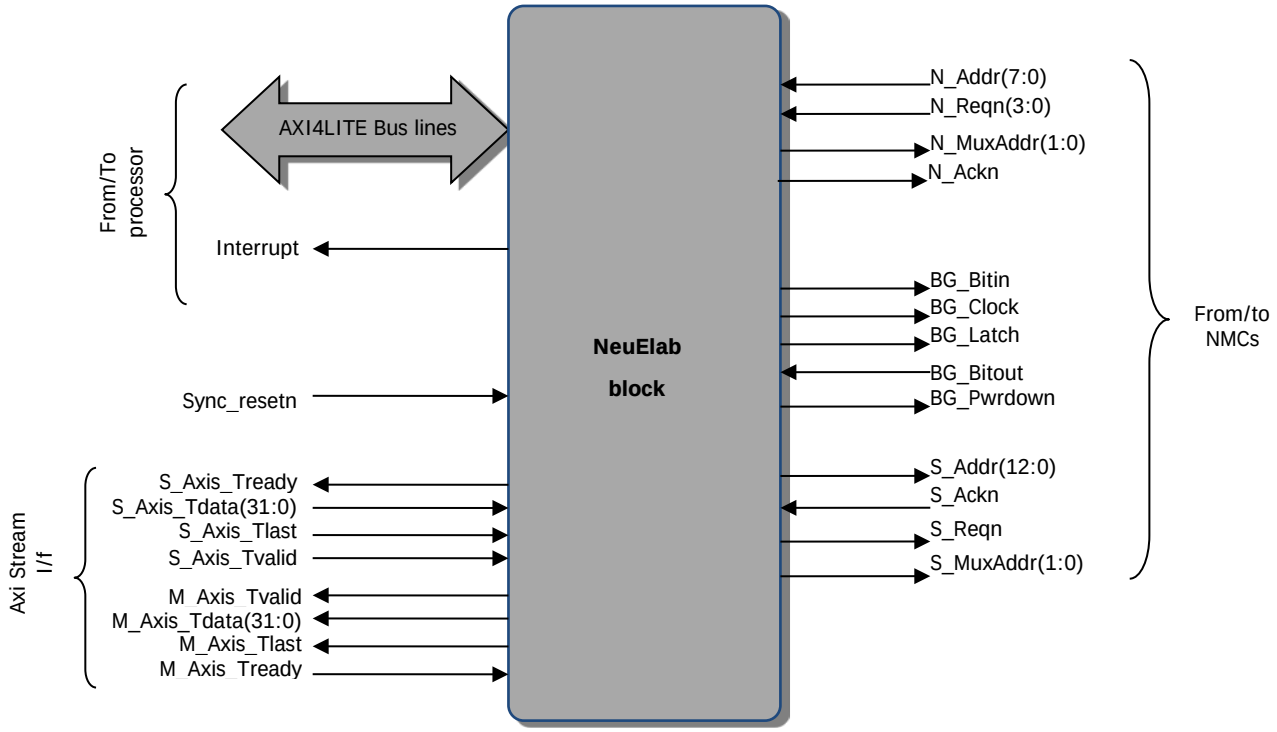


Figure 1 The NeuElab module interface

The list of the ports and their description is shown in Table 1.

Table 1 FocAxiIf interface signals description

Comment	Port name	Width	Dir	Description
NMCs signals	N_Addr	8	I	Address of the spiking Neural
	N_Reqn	4	I	Active low requests coming from the 4 external NMCs
	N_MuxAddr	2	O	Address of the external multiplexer used to get the correct Address of the spiking Neural from the 4 different NMCS. It is used also to deliver the N_Ackn signal to the selected NMC.
	N_Ackn	1	O	Active low Acknowledge signal to deliver to the selected NMC
	S_Addr	21	O	The meaning of this signal follows the MN256R1 AER Input decoding table [2].
	S_Reqn	1	O	Active low request signal for the selected NMC
	S_Ackn	1	I	Active low acknowledge signal coming from the selected NMC.

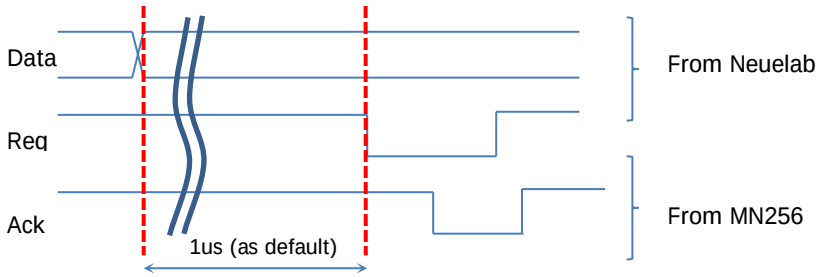
Comment	Port name	Width	Dir	Description
	S_MuxAddr	2	O	Address of the external multiplexer used to select the correct NMC that needs to be stimulated or Bias programmed.
	Interrupt	1	O	Interrupt line for host processor
Bias signal for the NMC	BG_Bitin	1	O	Bias serial BitIn
	BG_Clock	1	O	Bias serial Clock
	BG_Latch	1	O	Bias Latch
	BG_Bitout	1	I	Bias serial BitOut
	BG_Pwrdown	1	O	Power Down signal
	Sync_resetn	I	1	Synchronous reset signal
AXI4 Lite Bus lines	S_AXI_ACLK	1	I	AXI Clock, System clock line
	S_AXI_ARESETN	1	I	AXI Reset active low line
	S_AXI_AWADDR	32	I	AXI Write address
	S_AXI_AWVALID	1	I	Write address valid
	S_AXI_WDATA	32	I	Write data
	S_AXI_WSTRB	4	I	Write strobes
	S_AXI_WVALID	1	I	Write valid
	S_AXI_BREADY	1	I	Response ready
	S_AXI_ARADDR	32	I	Read address
	S_AXI_ARVALID	1	I	Read address valid
	S_AXI_RREADY	1	I	Read ready
	S_AXI_ARREADY	1	O	Read address ready
	S_AXI_RDATA	32	O	Read data
	S_AXI_RRESP	2	O	Read response
	S_AXI_RVALID	1	O	Read valid
	S_AXI_WREADY	1	O	Write ready
	S_AXI_BRESP	2	O	Write response
	S_AXI_BVALID	1	O	Write response valid
	S_AXI_AWREADY	1	O	Write address ready
Axi Stream I/f	S_Axis_Tdata(31:0)	32	I	
	S_Axis_Tlast	1	I	
	S_Axis_Tvalid	1	I	
	M_Axis_Tready	1	I	
	M_Axis_Tdata(31:0)	32	O	
	M_Axis_Tlast	1	O	
	M_Axis_Tvalid	1	O	

All the external inputs of the NeuElab module are synchronized as soon as they enter in the module in such a way to avoid metastability issues.

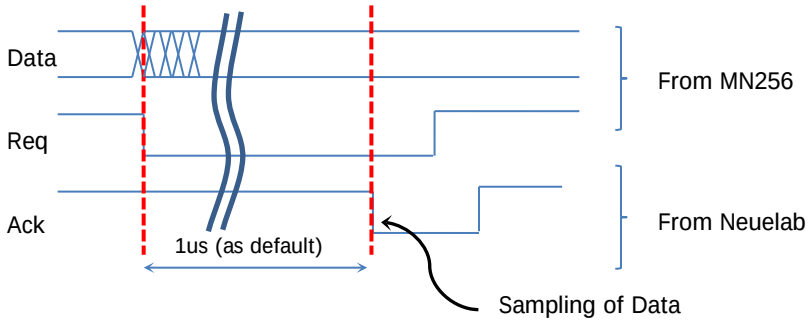
From a functional point of view, the NeuElab Soft Ip acknowledges any request from the NMCs. As soon as the NeuElab Soft Ip will be enabled, it starts to fill the internal Fifos with the data read from the NMC sticked with an internal 32bit time stamp.

Please note that after some empirical trials, this version of Neuelab has the following features.

- In the stimulation phase the *data* are changed and the *request* line is activated after the time specified by the Settling time register (SETTLINGTIME_REG):



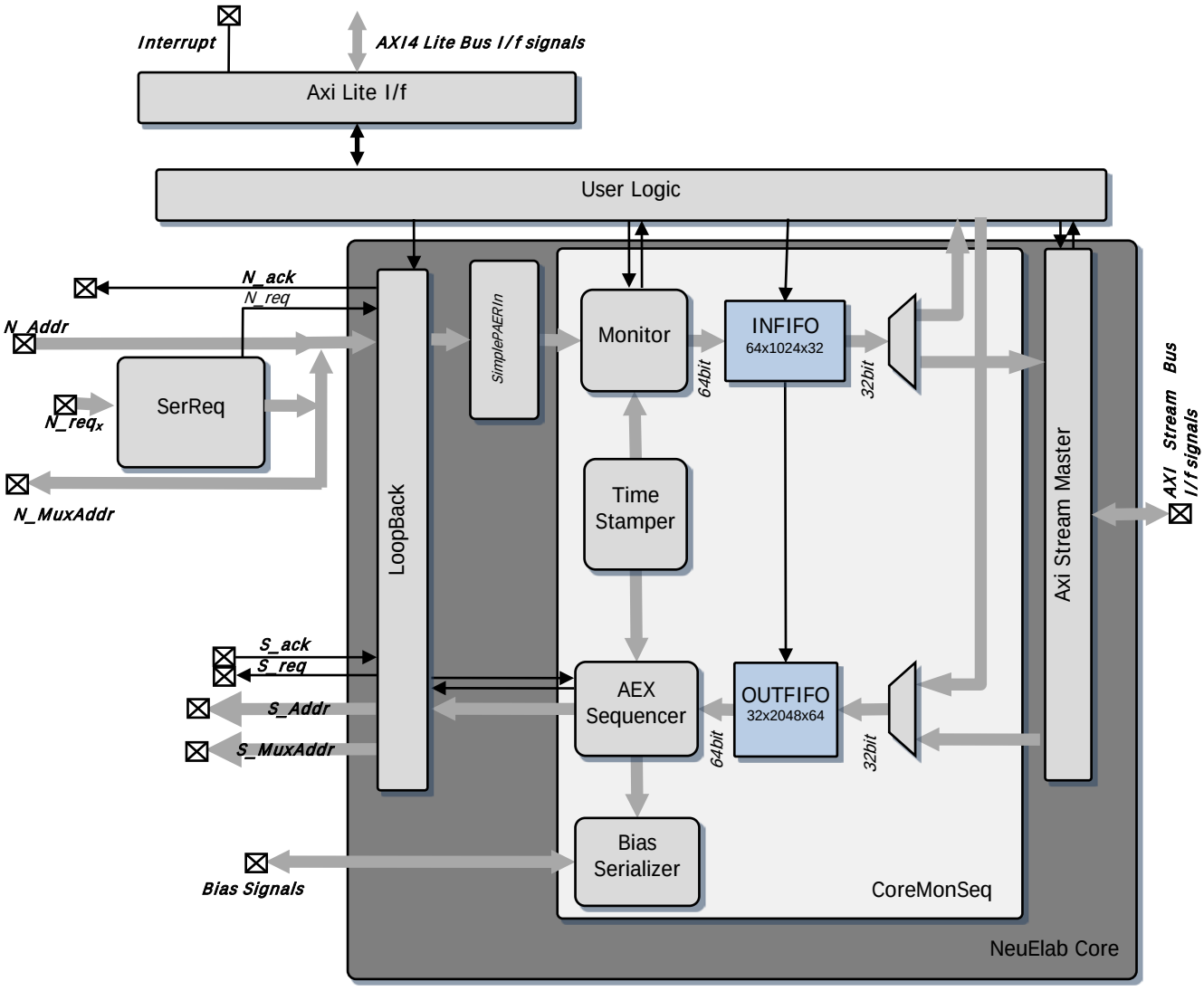
- In the reading phase the *acknowledge* signal is activated the time specified in Settling time register (SETTLINGTIME_REG) after that the *request* signal becomes active. The addresses are read by the Neuelab during the negative edge of the *acknowledge* signal



2 NeuElab Block Diagram

The NeuElab Block diagram is shown in Figure 2. It integrates the following fully synchronous blocks:

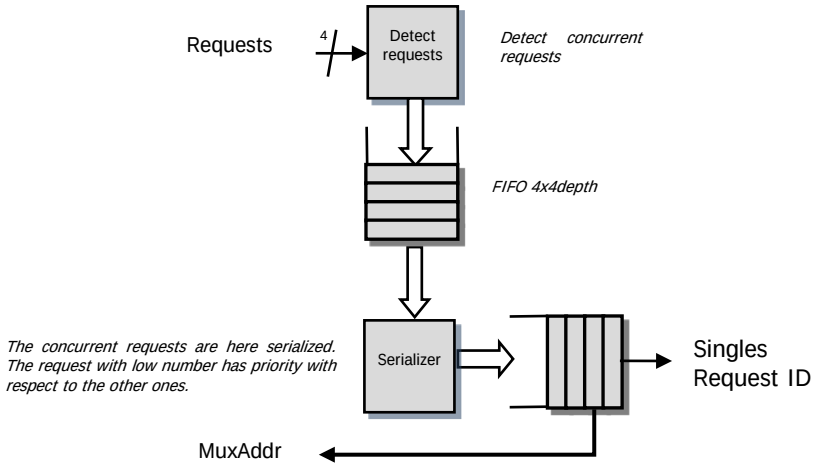
- SerReq
- Axi_lite_ipif,
- User_logic
- LoopBack
- Neuelab_axi_stream
- Neuelab_core



- **Figure 2 NeuElab core block diagram**

2.1 SerReq

The SerReq block is used to manage the requests coming from till 4 different NMCs in such a way to serialize them to use the unique shared Address port.



A first 4x4 depth Fifo is used to serialize the concurrent requests. The concurrent requests are then serialized, taking into account that request of NMC with low ID are served prior with respect to others NMCS concurrent. For instance, if NMC2 and NMC3 have requested the Address bus in the same time, the NMC2 will be acknowledged first with respect to the NMC3. The address bus coming from the NMC that will be acknowledged is addressed by the *MuxAddr*.

2.2 Axi_lite_ipif

This module is used to manage the Axi Lite protocol slave Interface. It comes directly from *Xilinx Create and Import peripheral* tool.

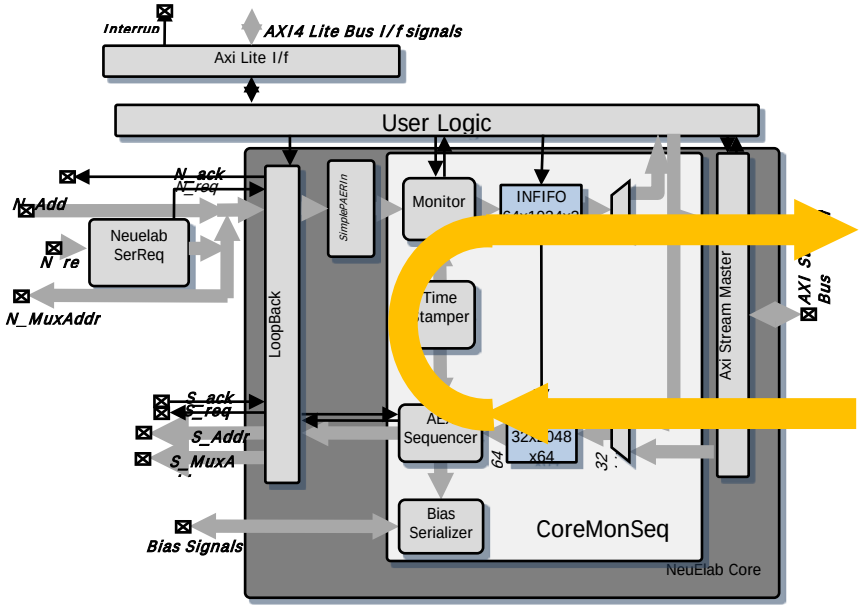
It generates useful control signals in way to correctly read and write user register instanced in *user_logic* module

2.3 User_logic

This is the module that physically contains the registers used in the NeuElab Soft Ip and fully described in Section 3.

2.4 LoopBack

The loopback module is used to test the whole data path. It can be activated in way to loop back the data coming from the CPU, through the Stimuli interface to the acquired data path.



2.5 Neuelab_axi_stream

This block is deputed to manage the axi stream interface that drives a Direct Memory Access (DMA) module to write/read memory without any intervention of the CPU.

2.6 Neuelab_core

This module is the core of the whole system.
The LoopBack module is used to test the whole NeuElab Ip. If the *LoopBack* bit of the Control register (CTRL_REG) is set to '1', the data stored into the OutFifo can be used as input of the *DvsPaerDstRdyxS* module.

The *SimplePAERIn* is used to manage the AER protocol from/to the NMCs.

The *CoreMonSeq* module is used to add a time stamp to the events coming from the *SimplePAERIn* module and also to feed the 8Kbyte InFifo that contains data and relative time stamp ready to be processed. Events read with the same time stamp, means that they are arrived in the same 80ns time slot wide (indeed the clock of the time stamper is sysclock/8 and the sysclock is 100MHz, i.e. 10ns of period).

As soon as a data valid is passed to the NEUELAB, it passes through different FIFO, and then when it arrives to the Monitor, a TimeStamp value is attached to it and then fed into the InFifo.

The InFifo can be read from the processor by using the RX Time Buffer register (RXTIME_REG) and the RX Data Buffer register (RXDATA_REG) or by using the DMA I/f associated to this IP. In the same way the data from the processor can be written into the OutFifo both form using the TX Data Buffer register (TXDATA_REG) or by using the DMA I/f associated to this IP.

The *CoreMonSeq* module is also used to program the bias signals of the NMC and to stimulate a NMC thanks to the *AEXSequencer* module. The stimuli are time stamped, i.e., the time stamp here described is the value in clock period unit elapsed which the data is delivered to bias or a neural chip.

A typical stimuli data is as follows:

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Data stimuli	0	0	NMCs		reserved																											

The module used to program the bias signals is the *BiasSerializer*. The data stored into the OutFifo are deliverd to the *BiasSerializer* if the Data[31] is 'high'. The NMC to be correctly biased is identified by the S_MuxAddr signal.

When Data[31:30] is “10” the data is managed as a 24 bit serial bias data.

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Bias Data	1	0	NMCs	reserved																												Data Value

If the Data[31:30] is “11”, the data is managed as a Latch command and the serializer automatically puts into the BiasBitin signal 4 bits (“1111”) for test cell and 8 bits (“11111111”) for setting bias buffer splitter current and then put to low for the time programmed into the Bias Timing Register (BIASTIME_REG) the Bias Latch signal. This closes the bias programming phase. See for further details the Bias Timing Register (BIASTIME_REG) section.

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Bias Latch	1	1	NMCs	Reserved																												

The Figure 3 and the Figure 4 shown the timing performed with the default value of the Bias Timing Register (BIASTIME_REG) and the Prescaler Value Register (PRVAL_REG).
The Figure 5 shows the ASM algorithm implemented into the *BiasSerializer* module.

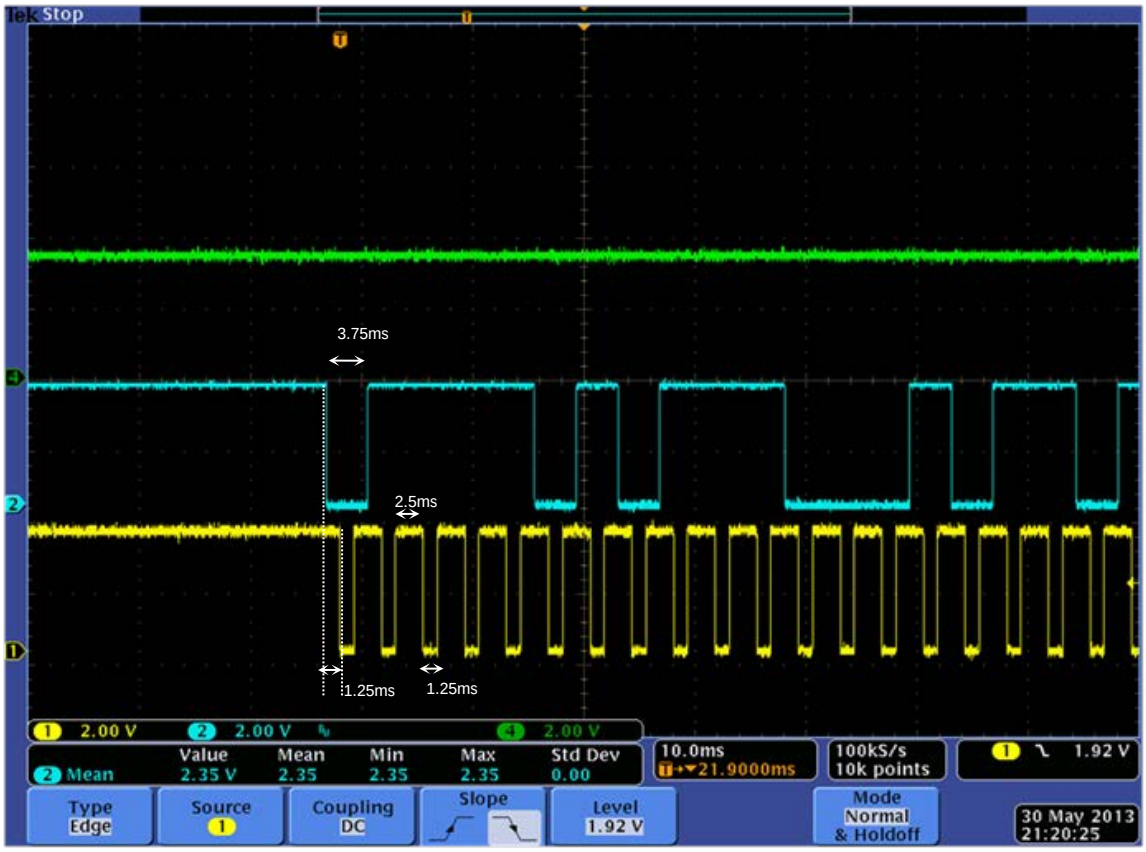


Figure 3 The signals timing during the Bias phase. Details for BG_Clock and BG_Bitout signals. In green the BG_LATCH signal, in cyan the BG_Bitout signal and in yellow the BG_Clock signal.

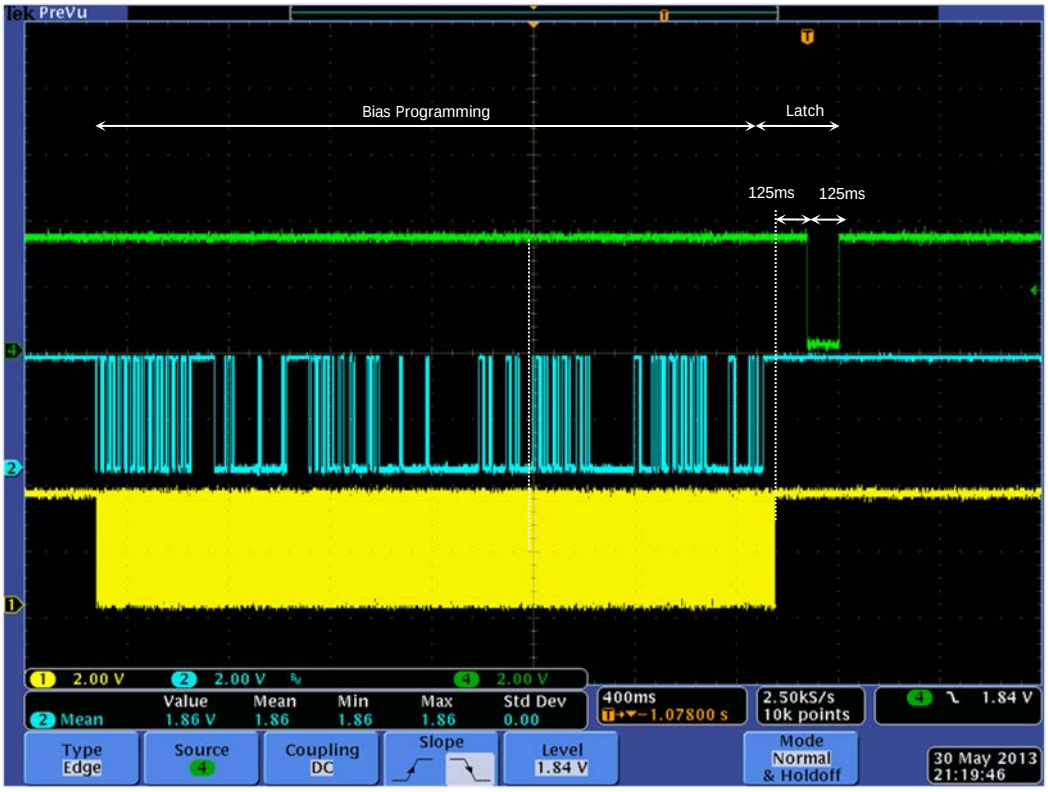
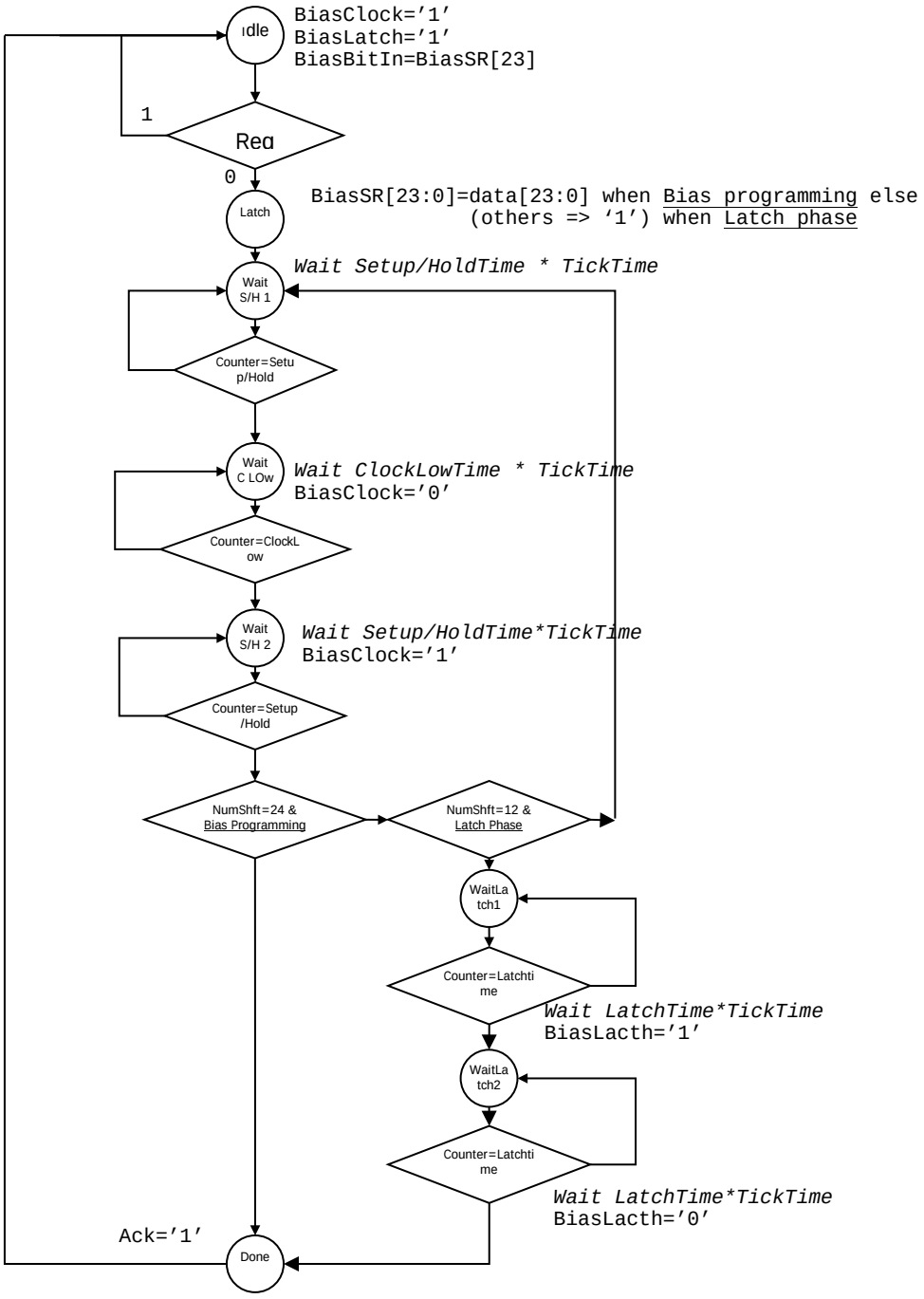


Figure 4 The signals timing during the Overall bias phase. In green the BG_LATCH signal, in cyan the BG_Bitout signal and in yellow the BG_Clock signal.



Bias programming when Data[31:0]="10"
Latch phase when Data[31:0]="11"

Figure 5 FSM Algorithm of the *BiasSerializer* module

3 NeuElab internal registers

The Register File of the NeuElab module has been generated starting from *Create and Import Peripheral* (CIP) tool of Xilinx starting from Platform Studio software.

The CIP tool creates a template of a generic register file that is interfaced to AXI4 Lite bus. Once having created the *USER_Logic* and the *Axi_Lite_Ipif* hdl modules, the HDL has been tailored to manage correctly the accesses to the created internal FocAxiIf registers.

In this Section a detailed view of the registers internal to the NeuElab is given.

The NeuElab block has an Axi Light Slave interface [1].

AXI is part of ARM AMBA, a family of micro controller buses first introduced in 1996. The first version of AXI was first included in AMBA 3.0, released in 2003. AMBA 4.0, released in 2010, includes the second version of AXI, AXI4. There are three types of AXI4 interfaces:

- AXI4—for high-performance memory-mapped requirements.
- AXI4-Lite—for simple, low-throughput memory-mapped communication (for example, to and from control and status registers).
- AXI4-Stream—for high-speed streaming data.

Xilinx introduced these interfaces in the ISE® Design Suite, release 12.3.

In the following the complete list of accessible FocAxiIf registers.

#	Offset	Mnemonic	Description	Type	Reset Value
0	0x00	CTRL_REG	Control register	R/W	0x00000000
			RESERVED		
1	0x08	RXData_REG	RX Data Buffer	R/O	0x00000000
2	0x0C	RXTime_REG	RX Time Buffer	R/O	0x00000000
3	0x10	TXData_REG	TX Data Buffer	R/W	0x00000000
4	0x14	DMA_BREG	DMA Burst Register	R/W	0x00000000
5	0x18	STAT_RAW_REG	Status RAW register	R/O	0x00000000
6	0x1C	IRQ_REG	IRQ register	R/C	0x00000000
7	0x20	MSK_REG	Mask register for the IRQ_REG register	R/W	0x00000000
8	0x24	BIASTIME_REG	Bias Timing Register	R/W	0x00640101
9	0x28	WRAPTimeStamp_REG	Wrapping TimeStamp Register	R/C	0x00000000
10	0x2C	Prescaler_REG	Prescaler register	R/W	0x0001E848
11	0x30	OUTNEUTHR_REG	Output neurons threshold register	R/W	0x00000000
12	0x34	SETTLINGTIME_REG	Settling time register for mn256	R/W	0x00630063
13	0x38	TIMESTAMP_reg	Time stamper value register	R/O	0x00000000
			RESERVED		
14	0x5C	ID_REG	Identification register	R/O	0x4e657523

3.1 Control register (CTRL_REG)

This register is used to control the behaviour of the NeuElab block.

CTRL_REG (NEUELAB Base + 0x00) Reset Value: **0x00000100**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															Chip type

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			Rearm DMA	Reserved			BG PWRDWN	Reserved			Flush FIFOs	EN Loop Back	EN INT	EN DMA	EN IP
			w/c				r/w		s/c	s/c	s	r/w	R/W	r/w	r/w

- EN IP is the Enable of the whole NeuElab block
  - When ‘1’ the NeuElab block is enabled
  - When ‘0’ the NeuElab block is disabled
- EN DMA is the DMA interface Enable
  - When ‘1’ the DMA I/f is enabled
  - When ‘0’ the DMA I/f is disabled
- Enable Interrupt
  - When ‘1’ the Interrupt is enabled
  - When ‘0’ the Interrupt never rises up
- Enable LoopBack
  - When ‘1’ the Loopback is enabled, i.e. the data coming from OUTFIFO are redirected as input to the INFIFO.
  - When ‘0’ the Loopback is disabled, i.e. the data coming from OUTFIFO are redirected to the External Neural chip trough the EXTSTIM* signals.
- Flush FIFOS
  - When set to ‘1’ the FIFOS of NEUElab are flushed. This bit is automatically cleared.
- BG_PWRDWN
  - This bit drives the output signal BG_Pwrdown
- REARM DMA
  - This bit when set to 1 is used to issue a DMA request to the DMA. It’s automatically cleared
- Chip_type
  - This bit select the chip type.
    - When ‘0’ the mn256 chip is selected
    - When ‘1’ the ifslwta chip is selected

Note that the contributors of the Interrupt are all the bits listed in STAT_RAW_REG register.

3.2 RX Data Buffer register (RXDATA_REG)

This register contains the data (read from the INFIFO) coming from the selected by N_MuxAddr NMC. The format of the register is depicted into the figure below.

RXDATA_REG (NEUELAB Base + 0x08) Reset Value: **0x00000000**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						NMC ID		Spiking Neuron ID							
						ro		ro							

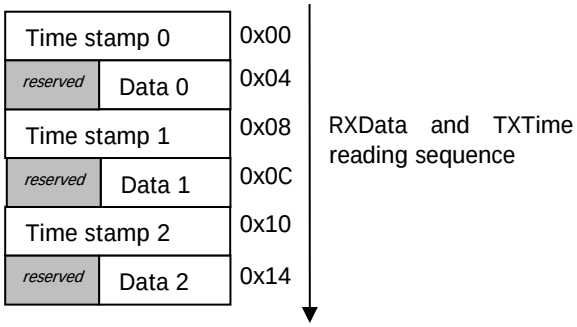
Where:

- NMC ID is the identifier of the NMC which has a neuron spiking.
- The Spiking neural ID is the Id of the spiking neuron of the NMC identified by bits 9 and 8.

The INFIFO is filled by the hardware in the following way:

- All meaning data are aligned to a 32bit address, i.e., address 0x00, 0x04, 0x08, 0x0C, 0x10 and so on…
- On the even addresses the time stamp can be read.
- On the odd addresses the data received at the time stamp read in the previous address can be read.

From the DMA/processor point of view, Time stamp and associated data are written in the same time.



NOTE: The reading of this register must follow the reading of the RX Time Buffer register (RXTIME_REG).

3.3 RX Time Buffer register (RXTIME_REG)

This register contains the time stamp associated to the received data (see RX Data Buffer register (RXDATA_REG)) from the INFIFO.

RXTIME_REG (NEUELAB Base + 0x0C) Reset Value: **0x00000000**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Time Stamp Value															
ro															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Time Stamp Value															
ro															

The Time Stamp value read from this register is the Time Stamp that the NeuElab Core sticks to the Received data available into the RX Data Buffer register (RXDATA_REG).

NOTE: After the reading of this register must follow the reading of the RX Data Buffer register (RXDATA_REG).

3.4 TX Data Buffer register (TXDATA_REG)

This register is used to fill the OUTFIFO.

TXDATA_REG (NEUELAB Base + 0x10) Reset Value: **0x00000000**

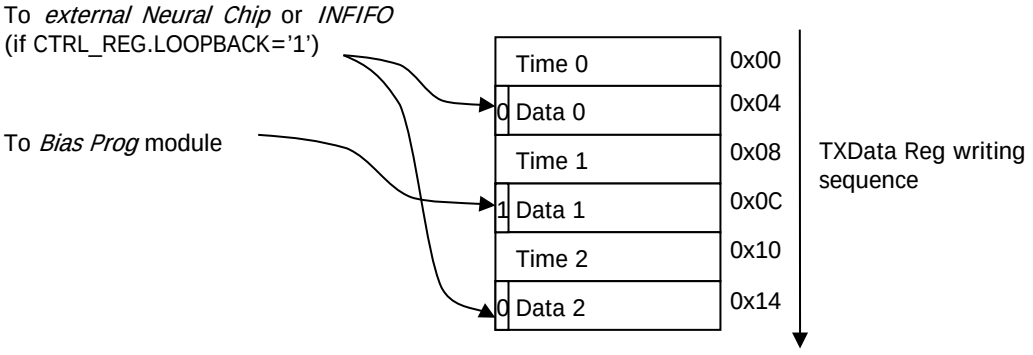
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TXData															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXData															
rw															

When writing to this register, keep in mind that it is used by the internal hw as follows:

The register needs to be written twice to enable the correct behaviour.

The first data written into the register represents the time, elapsed which, the second data written into the register is delivered to the *loopback* module (when bit 31th is '0') or to the *Bias Prog* module (when bit 31th is '1'). When written with a data, it has different meaning as explained in Section 2.6.



3.5 DMA register (DMA_REG)

This register is used set the behaviour of the Axistream interface. It represents the number of data (32 bit size length) sent to the DMA interface.

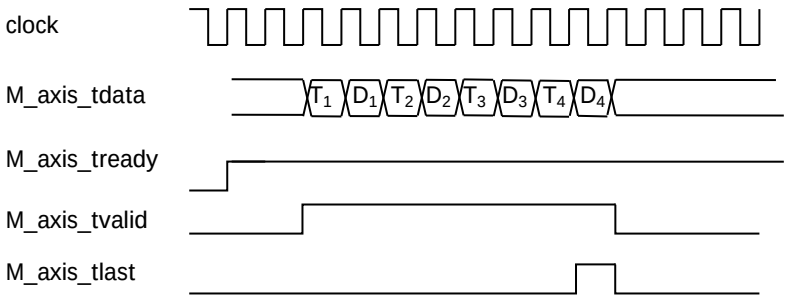
NOTE: This register can be written only if CTRL_REG.ENDMA='0'.

DMA_REG (NEUELAB Base + 0x14) Reset Value: **0x00000100**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
					DMA Length										
					r/w										

For example, if it is set to 8, the burst from/to the DMA I/f will be in terms of 8 data length.



3.6 RAW Status Register (STAT_RAW_REG)

When read, this register gives a snapshot of the status of warning or errors signals. It is a Read Only register.

STAT_RAW_REG (NEUELAB Base + 0x18) Reset Value: **0x00000000**

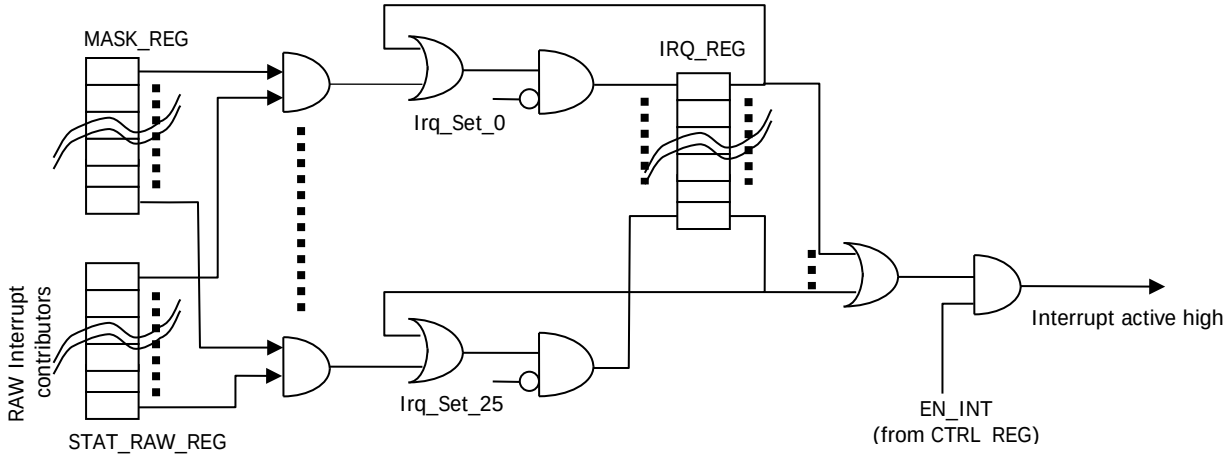
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						RX FIFO Not Empty	RxBufferReady	Time Stamp Wrapped	Bias Finished	TX Data Full	TX Data Almost Full	TX Data Empty	RX Data Full	RX Data Almost Empty	RX Data Empty
						ro	ro	ro	ro	ro	ro	ro	ro	ro	ro

- RxDataEmpty
  - When ‘0’, the INFIFO is not empty
  - When ‘1’ the INFIFO is empty
- RxDataAlmostEmpty
  - When ‘1’ the INFIFO has 1 or 0 data to be read.
  - When ‘0’ the INFIFO has more or equal two data to be read.
- RxDataFull
  - When ‘1’ the INFIFO is full.
  - When ‘0’ the INFIFO is not full.
- TxDataEmpty
  - When ‘0’, the OUTFIFO is not empty
  - When ‘1’ the OUTFIFO is empty
- TxDataAlmostFull
  - When ‘1’ the OUTFIFO has 2047 or 2048 data within himself.
  - When ‘0’ the OUTFIFO has less than 2047 data within himself.
- TxDataFull
  - When ‘1’ the OUTFIFO is full.
  - When ‘0’ the OUTFIFO is not full.
- Bias Finished
  - When ‘1’ the Bias signals have been latched
  - When ‘0’ no Bias signals have been latched
- Time stamp wrapped (this bit is high for one clock period only, when the counter wraps its value)
  - When ‘1’ the counter inside the TimeStamp module has wrapped its value.
  - When ‘0’ the counter inside the TimeStamp module has not yet wrapped its value
- RXBufferReady
  - When ‘1’ the Rx Fifo has at least DMA_REG value of data available
  - When ‘0’ the Rx Fifo has less than DMA_REG value of data available
- RXFifoNotEmpty
  - When ‘1’ the RX Fifo is not empty.
  - When ‘0’ the RX Fifo is empty

3.7 IRQ Register (IRQ_REG)

When read, this register gives the status of the collected warning or errors signals. It is a Read/Set register, i.e., to clear the warning/error bit the user has to write '1' on the corresponding bit position.



IRQ_ REG (NEUELAB Base + 0x1C) Reset Value: **0x00000000**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						RX FIFO Not Empty	RxBufferReady	Time Stamp Wrapped	Bias Finished	TX Data Full	TX Data Almost Full	TX Data Empty	RX Data Full	RX Data Almost Empty	RX Data Empty
						r/s	r/s	r/s	r/s	r/s	r/s	r/s	r/s	r/s	r/s

The meaning of the masked contributors of this register is the same of the RAW Status Register (STAT_RAW_REG).

3.8 Mask Register (MSK_REG)

This is the Mask register used to mask contributors for the interrupt signal.

MSK_ REG (NEUELAB Base + 0x20) Reset Value: **0x0000000B**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						RX FIFO Not Empty	RxBufferReady	Time Stamp Wrapped	Bias Finished	TX Data Full	TX Data Almost Full	TX Data Empty	RX Data Full	RX Data Almost Empty	RX Data Empty
						r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

The meaning of the masked contributors of this register is the same of the RAW Status Register (STAT_RAW_REG).

3.9 Bias Timing Register (BIASTIME_REG)

This register is used to set the timing of the signals involved into the bias programming.

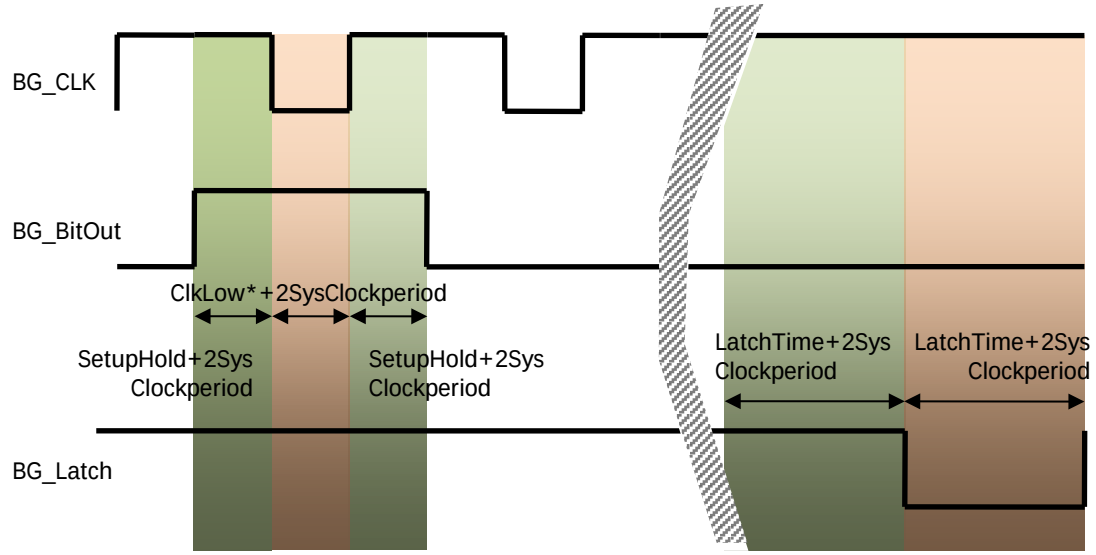
BIASTIME_ REG (NEUELAB Base + 0x24) Reset Value: **0x00640101**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								Latch time							
								r/w							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	CLK LOW time							Reserved		Setup/Hold Time					
	r/w									r/w					

Please, note that the values are expressed as tick time value as expressed in the Section related to the Prescaler Value Register (PRVAL_REG).

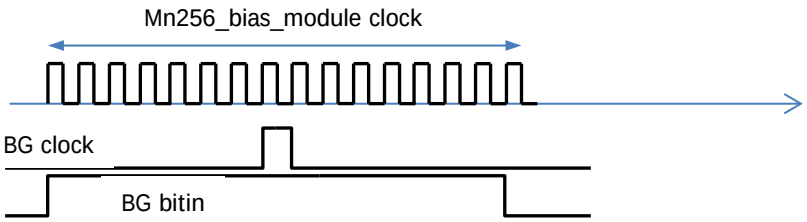
How this register acts on the bias signals timings is depicted in the Figure below.



For instance, when using both the default values of the Bias Timing Register (BIASTIME_REG) and of the Prescaler Value Register (PRVAL_REG), we have the following real timings:

SetupHold+2= $PRVAL * SysCLock\ period * SetupHoldValue + 2 * SysClockperiod = 1.25002$ ms
ClkLow+2= $PRVAL * SysCLock\ period * ClkLowValue + 2 * SysClockperiod = 1.25002$ ms
LatchTime+2= $PRVAL * SysCLock\ period * LatchTimeValue + 2 * SysClockperiod = 125.00002\ ms$

NOTE:
Only if chip_type of Control register (CTRL_REG) if high, the “Latch time” filed has the meaning of the number of clocks before and after an impulse of BG_Clock. I.e., if Latch time is 0xF then:


Mn256_bias_module clock
BG clock
BG bitin

3.10 Wrapping TimeStamp Register (WRAPTimeStamp_REG)

This register is used to read how many times the internal 32bit counter of the TimeStamp module has wrapped its value.

In case the user writes any value in this register, it will be cleared and also the internal 32bit counter of the TimeStamp module will be cleared.

WRAPTIMESTAMP_ REG (NEUELAB Base + 0x28) Reset Value: **0x00000000**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Wrapping times															
r/c															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Wrapping times															
r/c															

3.11 Prescaler Value Register (PRVAL_REG)

This register is used to set the value of the prescaler used as time bases of the counter that manage the bias timing (see Bias Timing Register (BIASTIME_REG)).

PRVAL_ REG (NEUELAB Base + 0x2C) Reset Value: **0x0001E848**

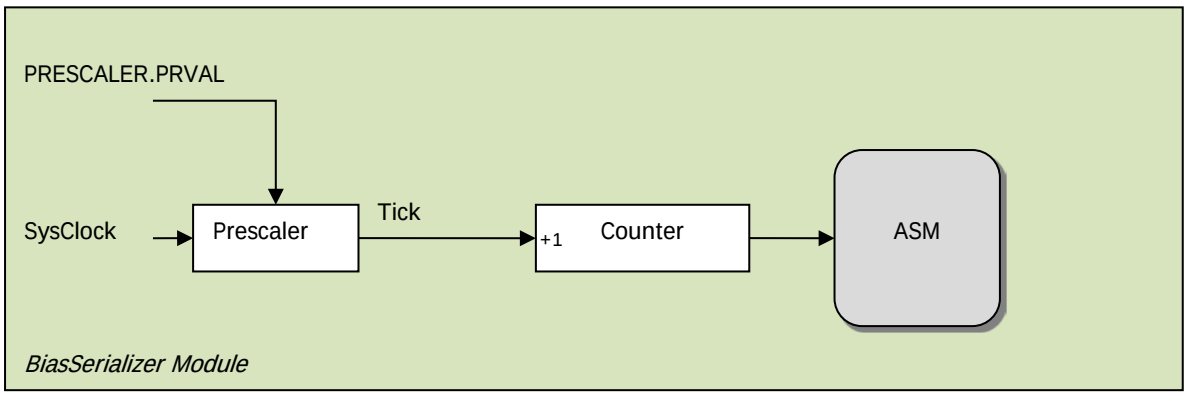
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PRVAL															
r/w															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PRVAL															
r/w															

For instance, when the PRVAL_REG is set to 0x1E848, this means that the bias timing counter is ticked every

$$PRVAL * SysCLock\ period = \ 0x0001E848 * 10ns = 125000 * 10ns = 1.25ms$$

From the hardware point of view, the register here described affects the Prescaler module as in Figure below.



3.12 Output Neurons threshold (OUTNEUTHR_REG)

This register is used to correctly feed the INFIFO fifo. When set to 0x00000000 any spiking neuron feed the fifo. If the user sets it to the number where the output neurons starts, only the output spiking neurons feed the Ffifo.

OUTNEUTHR_ REG (NEUELAB Base + 0x30) Reset Value: **0x00000000**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Output Neurons threshold value															
r/w															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Output Neurons threshold value															
r/w															

3.13 Settling time register (SETTLINGTIME_REG)

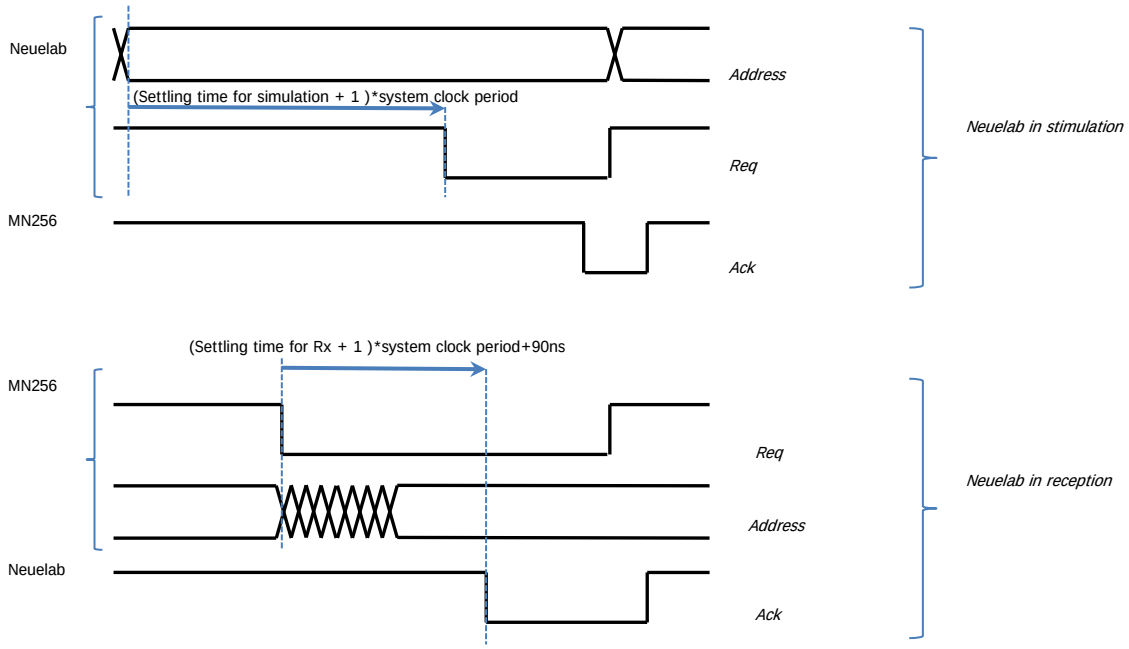
This register is used to set the settling time both in Rx and stimulation. In case of stimulation, the Neuelab changes the address and waits for (Settling time for simulation + 1)*system clock period (i.e. if system clock is 100Mhz, wait for 1us as default) before to activate the request.

In case of receiving spikes, the Neuelab wais for for (Settling time for rx + 1)*system clock period + 90ns (i.e. if system clock is 100Mhz, wait for 1.09us as default) before to sample the address and keeping low the Ack.

SETTLINGTIME_REG (NEUELAB Base + 0x34) Reset Value: **0x00630063**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Settling time for RX															
r/w															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Settling time for Stimulation															
r/w															



3.14 TimeStamp register (TIMESTAMP_REG)

This register is used to read the value of the Time stamper module used to identify the time of arriving of the spiking neurons. It's a read only register. It can be cleared by writing the Wrapping TimeStamp Register (WRAPTimeStamp_REG).

TIMESTAMP_REG (NEUELAB Base + 0x38) Reset Value: **0x--**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Time stamper value															
r/o															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Time stamper value															
r/o															

3.15 Identification register (ID_REG)

This register contains the ID of the NeuElab.

ID_REG (NEUELAB Base + 0x5C) Reset Value: **4E657523**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
N								e							
r/o								r/o							

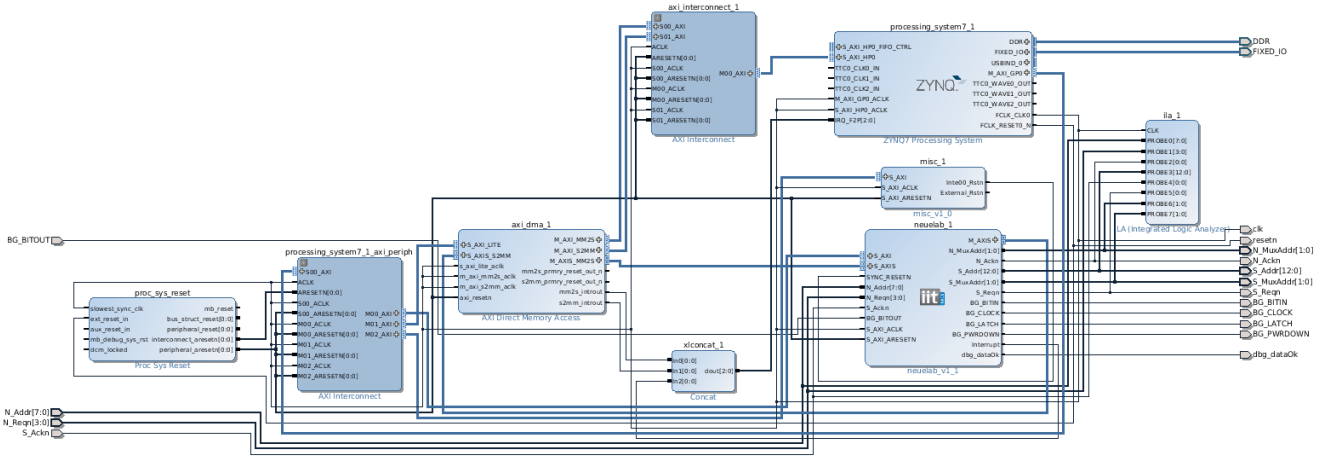
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
u								Major				Minor			
r/o								r/o				r/o			

Minor = 1;

Major = 1;

4 Example of NeuElab soft IP integration into the Zynq Architecture

In the following Figure an example of integration of the NeuElab soft Ip into the Zynq architecture.



5 Connection between ZedBoard and NeuroMorphicBoard

The ZedBoard will be connected to a custom designed board, here called NeuroMorphic Board (NMB in the following), which will be designed to hosts till 4 NMC.

Thanks to the developed architecture of the NeuElab Soft Ip, we match the needs to don't overcome the number of available IO of the Zedboard. Indeed, we share much pins as possible without reducing the functionality of the whole system by using the connection shown in

The number of IO used for the connection of the NCB to the ZedBoard can be easily computed by the Excel Table shown below. We choose to use a NMB with 4 NMC at the maximum.

Value for each NeuroMorphic Chip

Number of Neurons (**NN**)
256
Neural Activity Address Bus length (**Sup(Log2(NN))**)
8
Number of bit representing a value of synapse (**NbSYN**)
4
Number of bit representing the synapse (**NbSYNVAL**)
9

Number of NeuroMorphic Chip	Number of Neurons	Number of IO managed by the FPGA
1	256	25
2	512	28
3	768	31
4	1024	32
5	1280	35
6	1536	36
7	1792	37
8	2048	38

Legend:
Modify only the values green backgrounded

Table 2 Meaning of IO signals used to connect the NMB to the ZedBoard. The direction is referred to the ZedBoard point of view.

Comment	Port name	Width	Dir	Description
Neural Activity	**N_addr**	**Sup(Log_2(NN))**	I	Address of the Neural activity. ***NN* is the number of neurons of a single NMC**. For instance, if a single NMC has 128 neurons, the size of this bus is 7
	N_MuxAddr	**Sup(Log_2(NNMCs))**	O	Signal used to drive the external multiplexer that connect all the *N* NMC to the ZedBoard. **NNMCs is the number of NMC**. If we manage 8 NMC, the size of this bus is 3.
	N_reqn_x	**NNMCs**	I	Request signal active low for NMCx
	N_ackn	**1**	O	Acknowledge signal active low for NMCx. The specified NMC to be addressed is chosen by the N_MuxAddr signal. Undriven acknowledge signals are tied high by pullup resistor in way to have an inactive signal on them.
Stimuli Activity	**S_addr**	**NbSYNVAL+ NbSYN**	O	Address of the couple Neuron/Synapse to be stimulated.
	S_MuxAddr	**Sup(Log_2(NNMCs))**	O	Signal used to drive the external multiplexer that connect all the *N* NMC to the ZedBoard. **NNMCs is the number of NMC**. If we manage 8 NMC, the size of this bus is 3.

Comment	Port name	Width	Dir	Description
	S_reqn	**1**	O	Request active low signal to be addressed to the NMC_x.
	S_ackn	**1**	I	Acknowledge signal active low coming from the NMC_x.
Total Number of requested IO		**$Sup(Log_2(NN)) + 2*Sup(Log_2(NNMcs)) + NNMCs + NbSYNVAL + NbSYN + 3$** **Where:** • **NN is the number of neurons for each NMC,** • **NNMCs is the number of NMCs,** • **NbSYNVAL is the number of bits used to represent a Synapse value** • **NbSYN=is the number of bits used to represent a Synapse**		

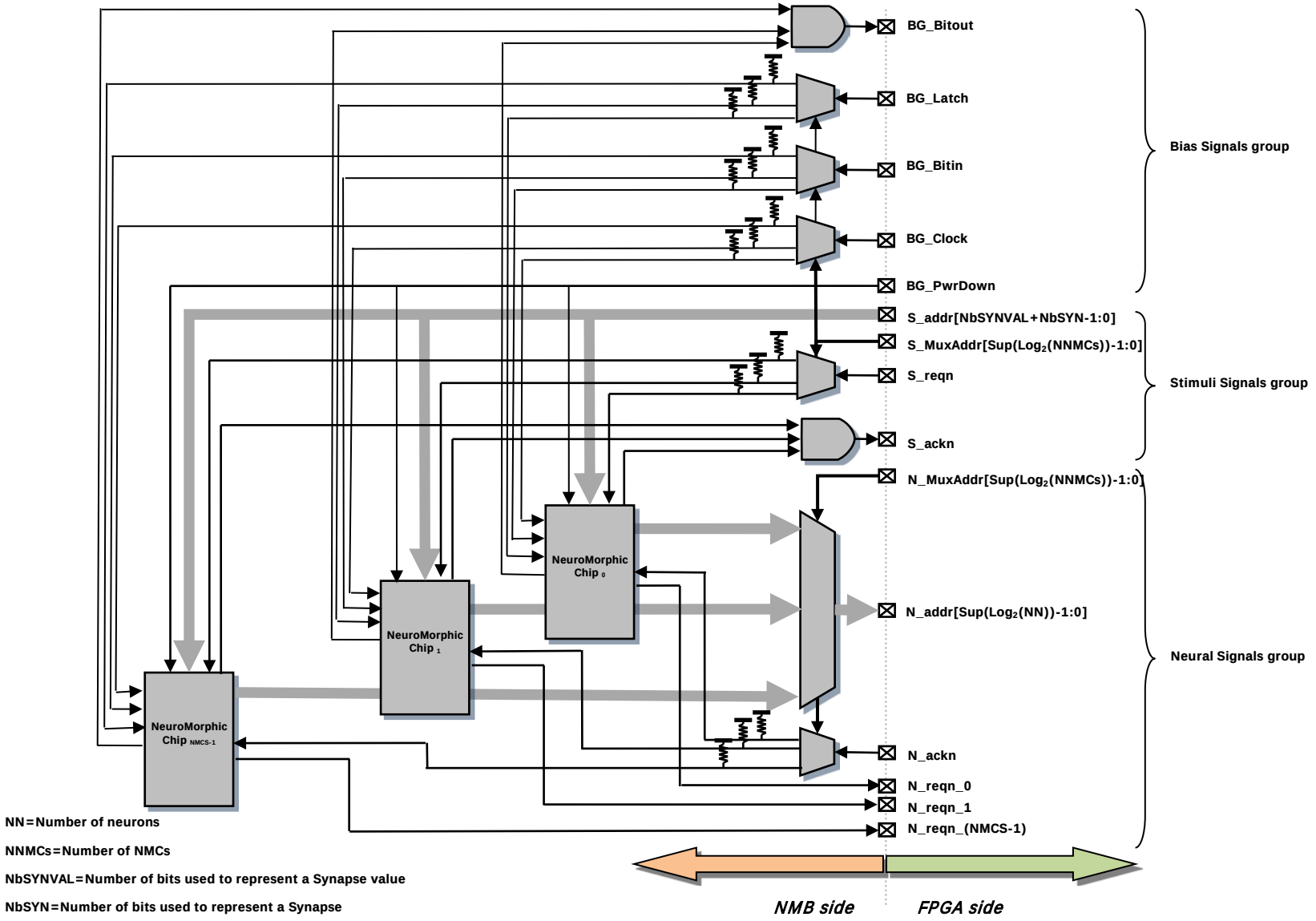
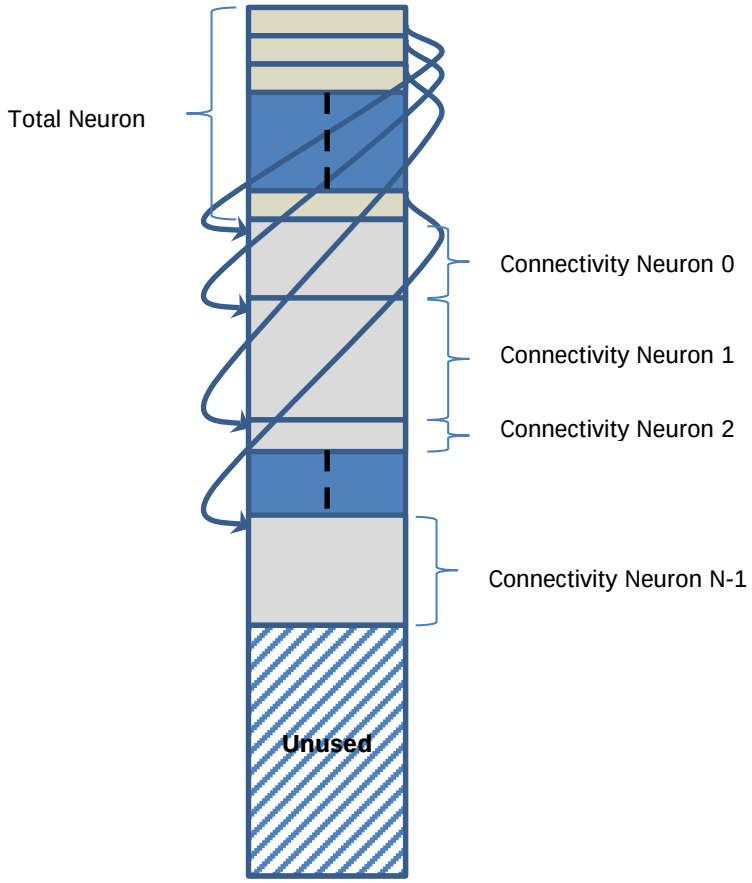


Figure 6. Zedboard connection to the NMB.

References

[1] ARM AMBA AXI protocol v2.0

[2] MN256R1 AER Input decoding (*I have a hard copy of this table…)*


Total Neuron
Connectivity Neuron 0
Connectivity Neuron 1
Connectivity Neuron 2
Connectivity Neuron N-1
Unused