



# The Perplexus project: Emulation of bioinspired spiking networks

Jordi Madrenas

Technical University of Catalunya (UPC)

Department of Electronic Engineering

Advanced Hardware Architectures (AHA) Group

<http://www-eel.upc.es/aha>



- UPC
  - 16 Schools and Faculties
  - 41 Departments
  - 28000 Students
  - 2700 Lecturers
- Department of Electronic Engineering
  - 6000 Students (40 PhD students)
  - 150 Lecturers
- AHA Group
  - 7 Lecturers (1 Prof., 3 Assoc. Prof., 4 Assist. Prof.)
  - 5 PhD Students
  - 10 MSc Students



# Expertise areas of the AHA group

- VLSI Design (Digital, analog and mixed-signal)
- High-level digital system design
- ***Soft computing*** models (ANN, Fuzzy Systems)
- Evolutive and Bio-inspired systems
- Microprocessor- and DSP-based design of embedded systems
- Design of FPGAs and programmable integrated systems
- (PSOCs) architectures
- FPGA- and PSOC-based system design
- Image processing



# Outline

- The Perplexus project
- Ubidule
- Spiking neural networks
- Ubichip architecture
- Ubicell
- Sequencer
- Development tools
- Ongoing and future work



# Outline

- The Perplexus project
  - Introduction
  - Objectives
  - General description
- Ubidule
- Spiking neural networks
- Ubichip architecture
- Ubicell
- Sequencer
- Development tools
- Ongoing and future work



# The Perplexus project

## Introduction

- Perplexus: Pervasive computing framework for modelling complex virtually-unbounded systems
- 6FP EU project on simulation of complex systems
- 3-year project (Sept. 2006 – Aug. 2009)
- Partners: HEIG-VD (coord, CH), UPC (ES), UNIL (CH), UJF (F), CNRS (F), TUL (P), EPFL (CH), SCIPROM (CH).



# The Perplexus project Objectives

1. Design and development of a **scalable hardware platform** made of **custom reconfigurable devices** endowed with **bio-inspired capabilities** that will enable the **simulation of large-scale complex systems** and the study of **emergent complex behaviors** in a **virtually unbounded wireless network** of computing modules (*ubidules*).
2. Simulation of complex phenomena in the domains of **realistic neural models**, **social sciences**, and **collective cooperative robotics**
3. To study the **emergent phenomena** arising from the strong **interaction** between our **ubiquitous computing modules and the real environment**.



# Outline

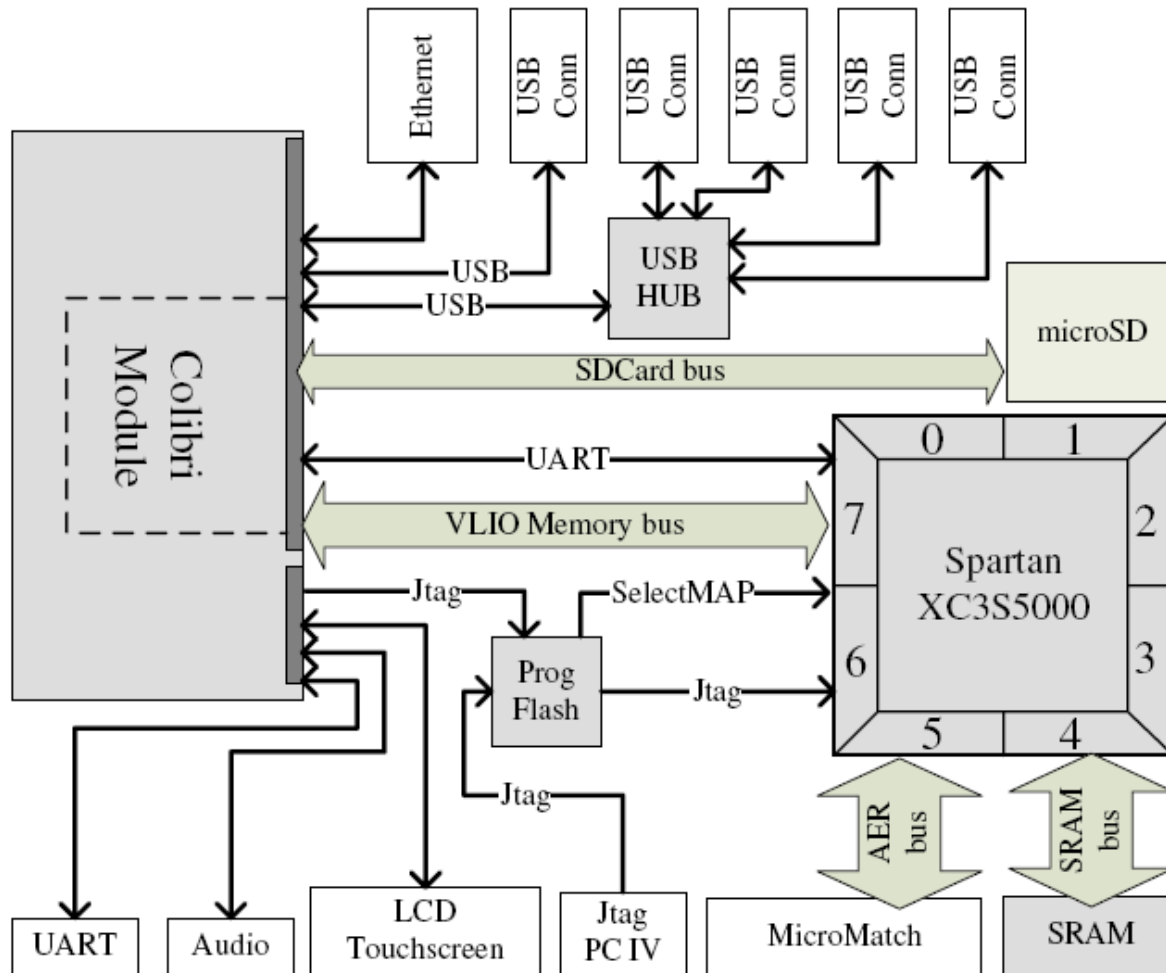
- The Perplexus project
- **Ubidule**
- Spiking neural networks
- Ubichip architecture
- Ubicell
- Sequencer
- Development tools
- Ongoing and future work



# Ubidule

- The ***ubidule*** is defined as an embedded ubiquitous element with computing and communication capabilities.
- Starting from Colibri board (Toradex). Support to Linux and Windows CE (app. programs, services, drivers...).
- Proposed implementation (for Perplexus):
  - Xscale PXA320 processor
  - FPGA (Spartan 3S5000)
  - SRAM (32x512k)
  - USB for communication and external sensors

# Ubidule diagram



# Ubidule PCB



Ubidule board



Ubidule with LCD and touchscreen interface

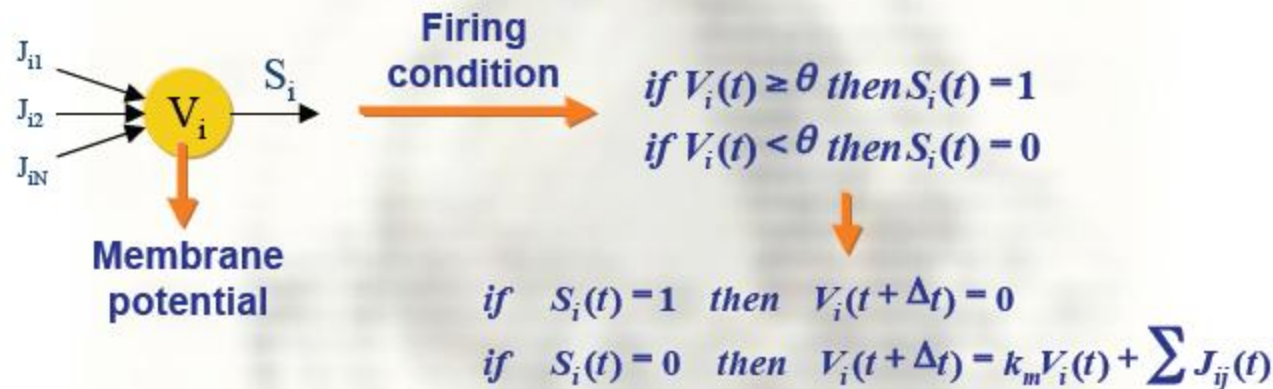


# Outline

- The Perplexus project
- Ubidule
- Spiking neural networks
  - Model
  - Scalability/connectivity issues
  - Address Event Representation (AER)
- Ubichip architecture
- Ubicell
- Sequencer
- Development tools
- Ongoing and future work

# Spiking neural networks Model

- Neuron model



- Synapse model

$$J_{ij}(t+1) = \begin{cases} J_{ij}(t) + (w_{RiRj} * A_{RiRj}(t)) & \text{when } S_j(t) = 1 \\ k_{syn} * J_{ij}(t) & \text{when } S_j(t) = 0 \end{cases}$$

$R_i$	$R_j$	$k_{syn}$	$w_{RiRj}$
E	E	↑↑	↓
E	I	↓	↑↑
I	E	0	↑
I	I	0	0

# Spiking neural networks Model (2)

$A_{ij}$ :

- Exc. - Exc. synapse: [0, 1, 2, 4]
- Any other synapse: 1

$$L_{ij}: L_{ij}(t+1) = k_{act} \times L_{ij}(t) + (YD_j(t) \times S_i(t)) - (YD_i(t) \times S_j(t))$$

- YD defines the time window after each pre- and post-synaptic spike:

$$\text{if } S_i(t) = 1 \quad YD_i(t+1) = YD_{MAX}$$

$$\text{if } S_i(t) = 0 \quad YD_i(t+1) = k_{learn} \times YD_i(t)$$

$L_{ij}$ :

- 4 thresholds ( $L_1, L_2, L_3, L_4$ )
- If  $L_{ij}$  lies between two consecutive thresholds
  - $L_{ij}$  is restored to the midpoint between thresholds
  - $A_j$  changes to the value associated to each region ( $A_0, A_1, A_2, A_4$ )



# Spiking neural networks

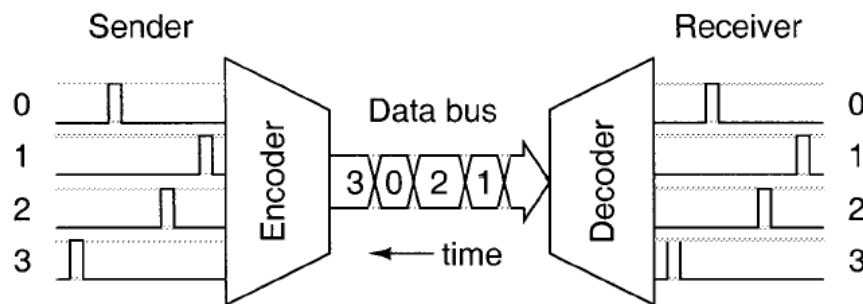
## Scalability/connectivity issues

- Target requirements:
  - Neural model emulation:
    - Average 300 inputs/neuron
    - 100 neurons/chip
    - 100 chips
    - At least (connectivity may be sparse)  $(300 - \text{neurons\_per\_chip}) \times \text{neurons\_per\_chip}$  dedicated inputs for dedicated connections between chips (~20K inputs!)
- Proposal
  - Packet switching mechanism: Address event representation.

# Spiking neural networks

## Address event representation (AER)

- Converts a sequence of spikes into a sequence of addresses corresponding to the spiking neurons:



- Requires an arbiter + encoder + decoder in every chip
- Parallel bus with an overall asynchronous behaviour
- Neural application: 24 address bus + handshake lines



# Outline

- The Perplexus project
- Ubidule
- Spiking neural networks
- Ubichip architecture
  - Bioinspired features
  - Hardware mechanisms
  - Granularity and memory mapping
  - Overall organisation
- Ubicell
- Sequencer
- Development tools
- Ongoing and future work



# Ubichip architecture

## Bio-inspired features

- **Phylogenesis:**
  - Support for **intrinsic**, **extrinsic**, **complete** and **open-ended** evolution
- **Ontogenesis:**
  - Support for **growth** and **self-replication**
- **Epigenesis:**
  - Support for neural friendliness (in terms of **plasticity** mechanisms and **connectivity** patterns)



# Ubichip architecture

## Hardware mechanisms

- **Dynamic routing:** Support for ontogenetic and epigenetic processes
- **Self-reconfiguration:** Support for ontogenetic and phylogenetic processes
- **Scalability:** Support for large-scale epigenetic processes (modified AER mechanisms)



# Ubichip architecture

## Granularity and memory mapping (1)

- Architecture proposal
  - Homogeneous granularity to ease scalability
  - Array of blocks: simple CPU + local RAM
    - Permits algorithm updating: self-placement, self-routing, evolution
    - Multiplexing synapses and neurons.
  - External massive RAM + cache memory system for access acceleration

# Ubichip architecture

## Granularity and memory mapping (2)

(Spiking neuron net application)

**Application parameters**

Neurons	N synapse/neuron
10000	300

Variables	Nvar/neuron	Nbits	Total bits
Ja	1	9	9
Jj	300	6	1800
wj	300	7	2100
Aj	300	2	600
YDj	300	4	1200
Lj	300	6	1800
sj	300	1	300
Vi	1	10	10
Si	1	1	1
Ydi	1	4	4
Lth	1	6	6
Theta	1	8	8
Tau	4	12	48

**Total #of bits: 7886**

# Ubichip architecture

## Granularity and memory mapping (3)

(Spiking neuron net application)

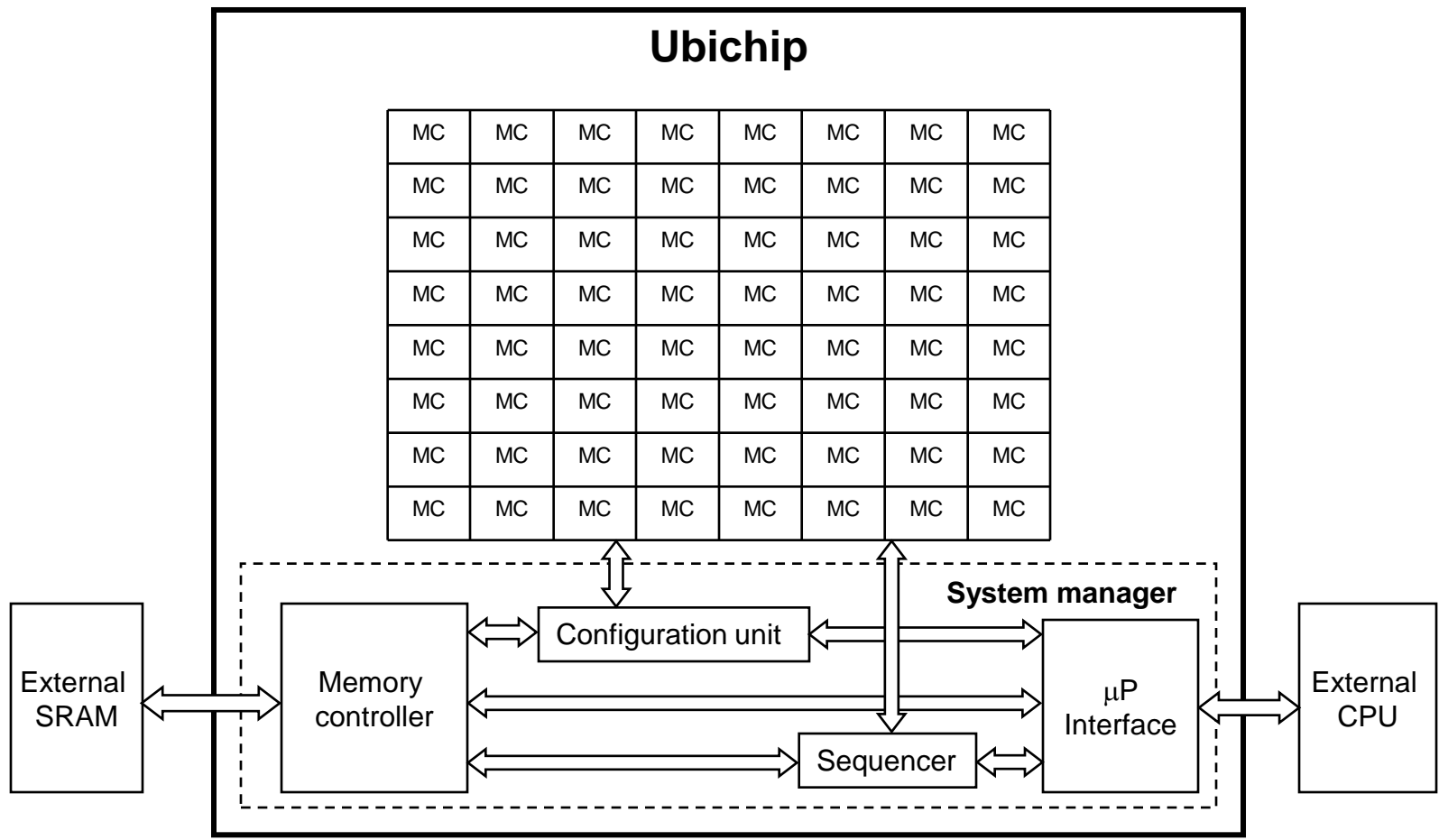
On-chip memory area (only parameters!)

$$A_{M\_PARAM} = \frac{\text{neurons}}{\text{chip}} \frac{\text{bits}}{\text{neuron}} \frac{\text{area}}{\text{bit}} =$$
$$= 100 \cdot 7886 \cdot 100 \mu\text{m}^2 = 78.86 \text{mm}^2 !!$$

↑  
Lower bound (0.35-micron technology)

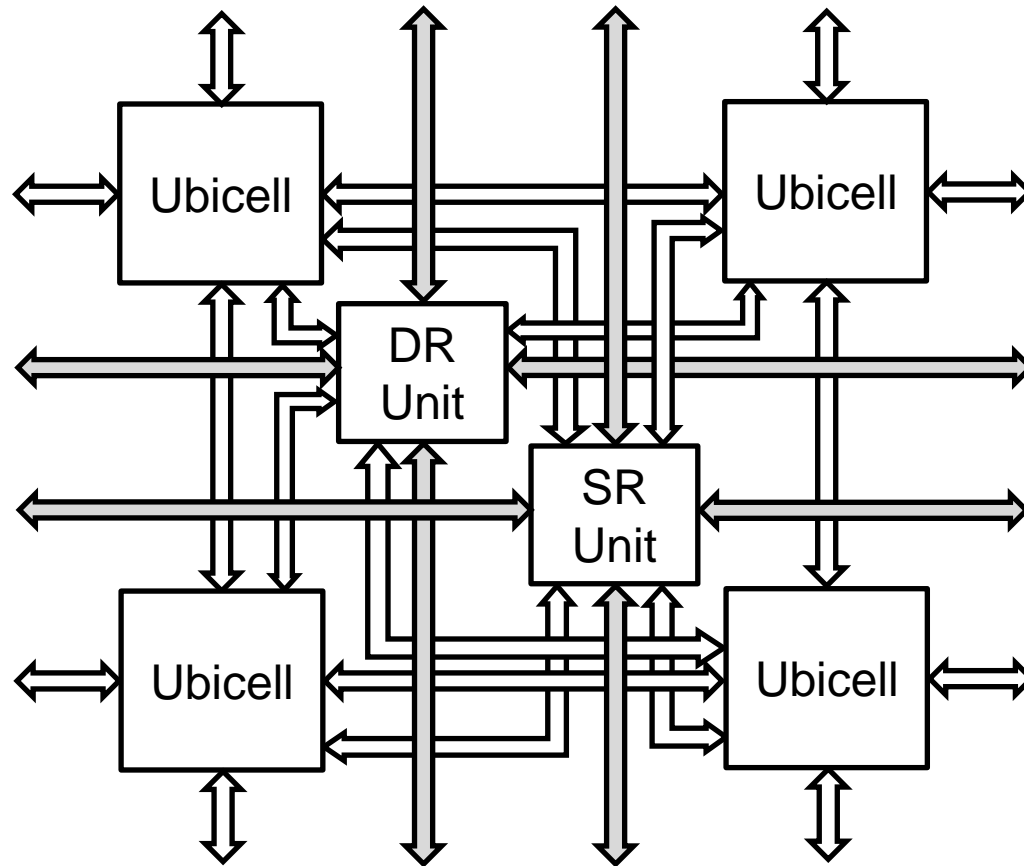
# Ubichip architecture

## Overall organisation



# Ubichip architecture

## Overall organisation (macro cell)

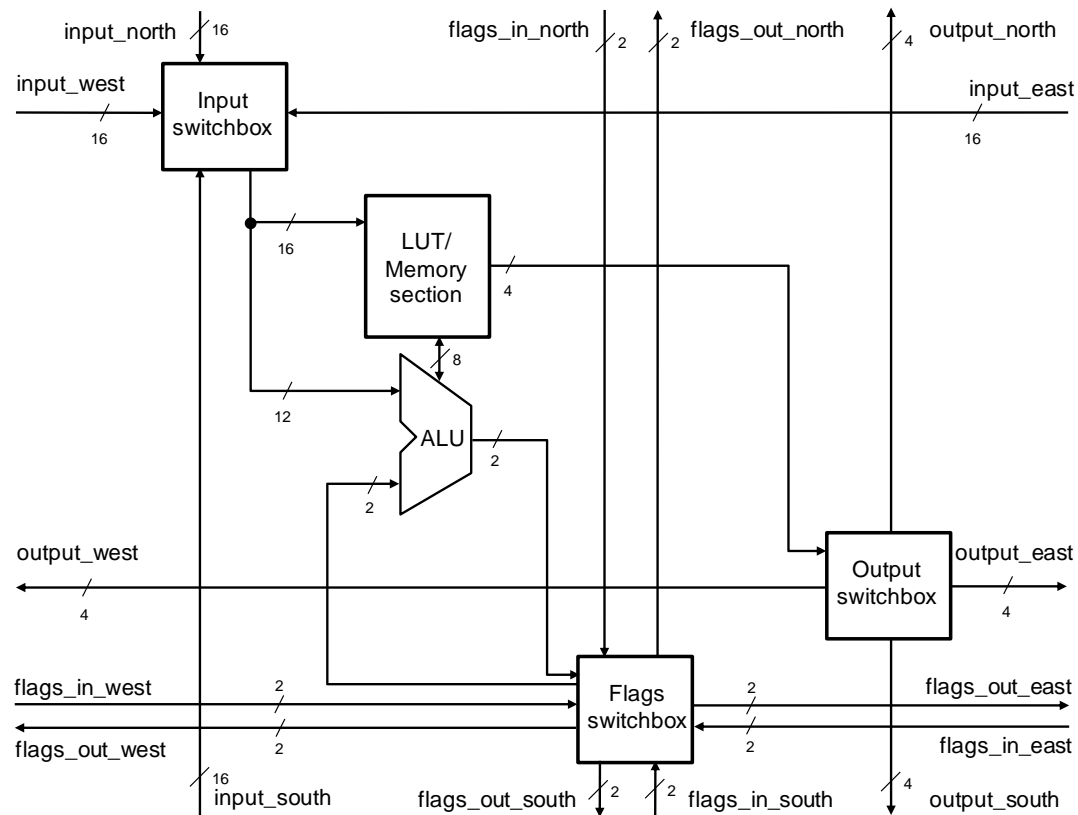




# Outline

- The Perplexus project
- Ubidule
- Spiking neural networks
- Ubichip architecture
- Ubicell
  - Organization
  - LUT/memory section
  - ALU section
  - Communication resources
  - Configuration
  - Self-Replication unit
  - Dynamic Routing unit
- Sequencer
- Development tools
- Ongoing and future work

# Ubicell Organization





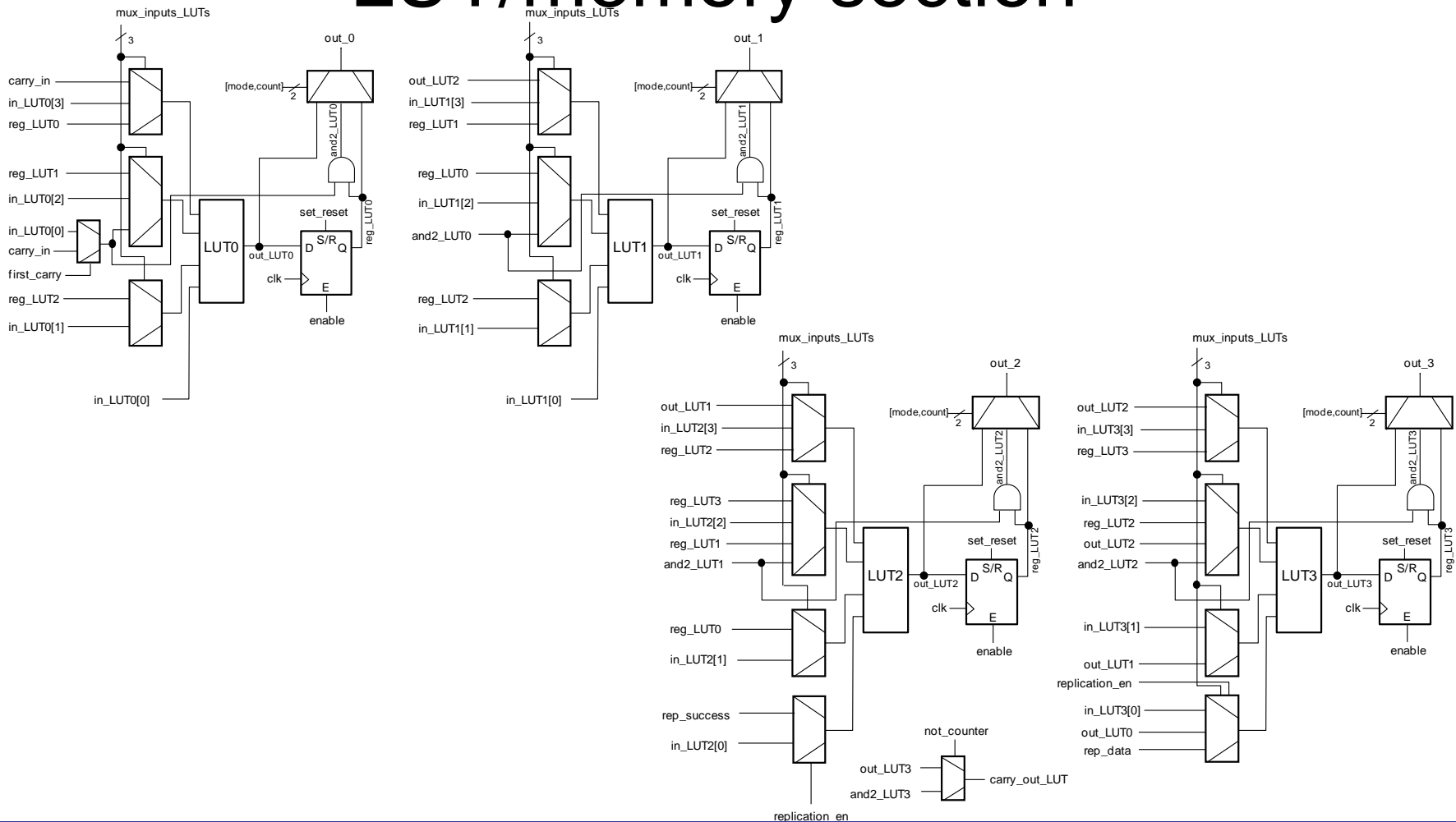
# Ubicell

## LUT/memory section (operating modes)

- 16 x 4-bit synchronous SRAM
- Four independent 4-input functions
- Wide decoder/high fanin function
- Two-level logic
- 4-bit counter
- Configurable registers
- 2 x 2-bit state machine (2 independent inputs per bit)
- 3-bit state machine
- Shift register
- 64 –bit LFSR

# Ubicell

## LUT/memory section





# Ubicell

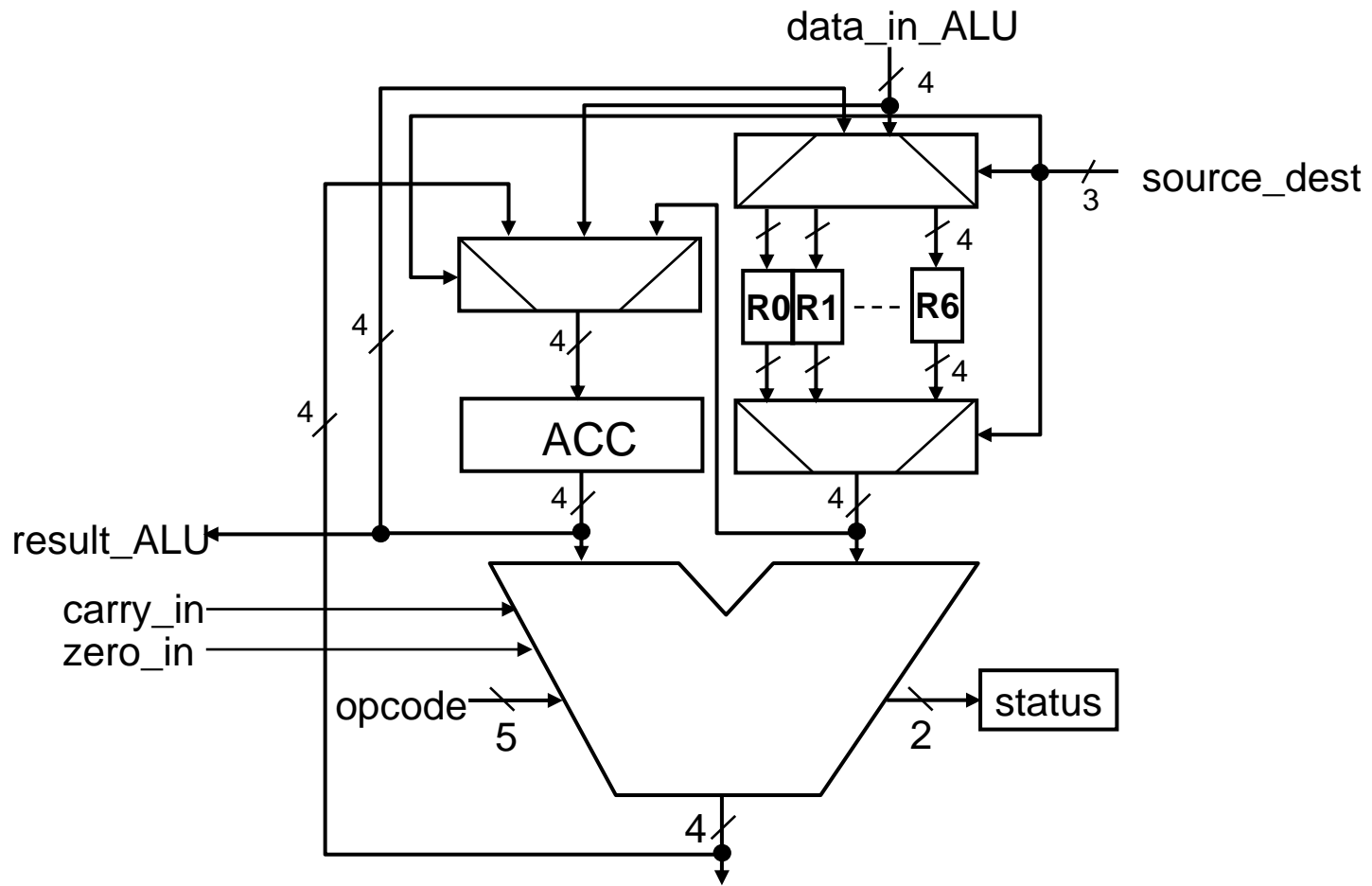
## ALU section (Instruction set)

Opcode	Mnemonic	Function
0000	NOP	No operation
01000	ADD	ACC <= [ACC] + [source_dest]
01001	SUB	ACC <= [ACC] - [source_dest]
01010	SHL	ACC <= [ACC]<<, carry = ACC[3]
01011	SHR	ACC <= [ACC]>>, carry = ACC[0]
01100	MOVA	ACC <= [source_dest]
11100	MOVR	source_dest <= [ACC]
01101	AND	ACC <= [ACC] AND [source_dest]
01110	OR	ACC <= [ACC] OR [source_dest]
11110	XOR	ACC <= [ACC] XOR [source_dest]
01111	INV	ACC <= NOT [source_dest]
00100	RST	source_dest <= "0000"
10100	LOAD	source_dest <= data_in_ALU
10001	SET	source_dest <= "1111"
10010	SWAP	source_dest <= [source_dest_shadow] source_dest_shadow <= [source_dest]
10101	MOVTS	source_dest_shadow <= [source_dest]

Opcode	Mnemonic	Function
10110	MOVFS	source_dest <= [source_dest_shadow]
10111	FREEZEC	Disables the registers of the ALU if the carry bit is set
11000	FREEZENC	Disables the registers of the ALU if the carry flag is not set
11001	FREEZEZ	Disables the registers of the ALU if the zero flag is set
11010	FREEZENZ	Disables the registers of the ALU if the zero flag is not set
11011	UNFREEZE	Enables the registers of the ALU

- The CLA can be used as an adder/subtractor
- In external CLA mode the result is not stored in the accumulator

# Ubicell ALU section





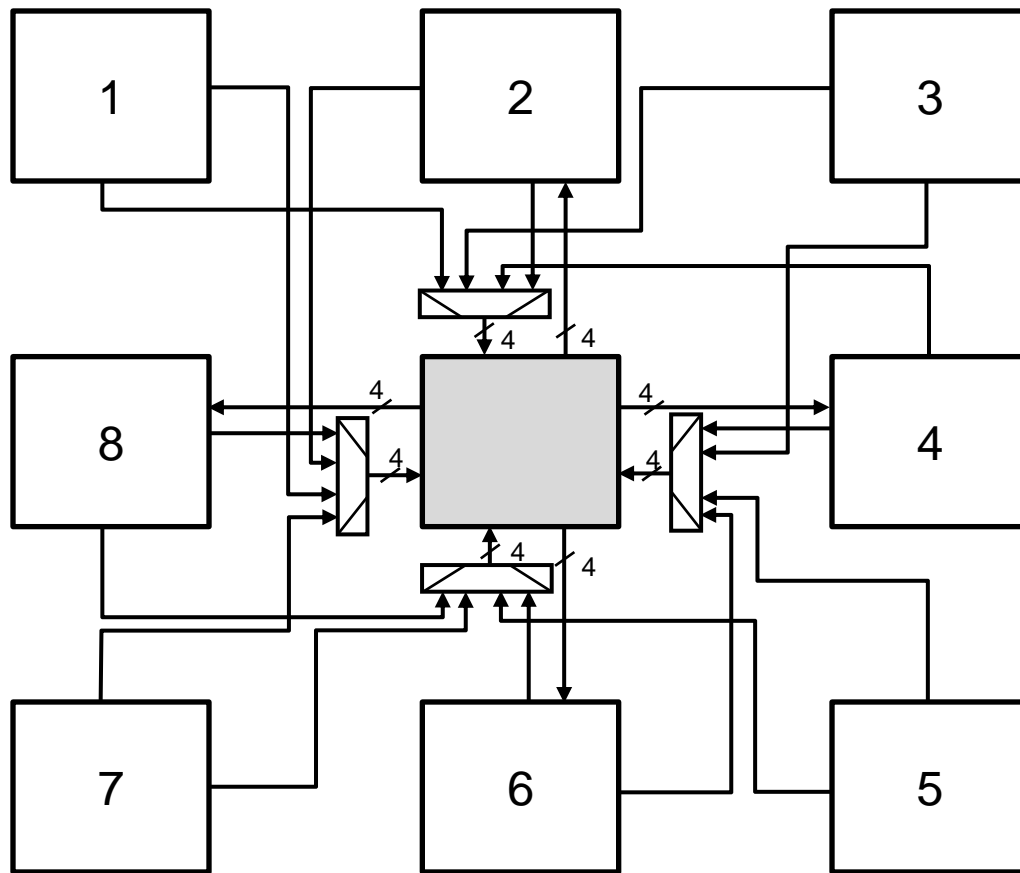
# Ubicell

## Communication resources

- **Input switchboxes** provide the actual inputs to the Ubicell from any of its input ports
- **Output switchboxes** permit to drive any output port with the output of the Ubicell, and can also be used as local routing resources
- **Carry and zero flags** have also dedicated routing resources
- **Dedicated multiplexers** permit to define a flexible connectivity pattern of a Ubicell with its 8 direct neighbours

# Ubicell

## Communication resources (Connectivity pattern)





# Ubicell Configuration

- 128 configuration bits per Ubicell
- Configuration bits can be read/written using either a serial or parallel (16-bit) interface
- Partial dynamic reconfiguration possible using the parallel interface
- In multi-processor mode the outputs of the 4 Ubicells in a Macro cell are grouped into a 16-bit data bus. The inputs for the Ubicells can also be written using a 16-bit data bus



# Ubicell

## Self-replication unit

- Elementary hardware unit that permits:
  - Configuration: Definition of a given functionality for a part of the system that is initially blank
  - Recovering: Scanning process of the configuration bits that permits self-replication
  - Remote configuration: Long-distance replication process
  - Kill: Destruction of a functionality already set in a given part of the system



# Ubicell

## Dynamic routing unit

- Permits to construct on-line communication paths between different parts of the system
- Dynamic routing process is totally distributed
- 8-neighbour connectivity pattern
- Paths created by configuring multiplexers
- Paths already created can be reused for a new connection
- 4-bit width data lines
- Routing process started by a Ubicell
- Paths can also be destroyed if no longer used



# Ubicell

## Dynamic routing unit (components)

- Switchbox: Set of multiplexers used to construct a path
- Serial comparator: Needed to define a single master during a multiple routing process
- Propagation unit: Sends to the neighboring units the signals required to define a path
- Controller: Finite state machine that manages the dynamic routing algorithm



# Outline

- The Perplexus project
- Ubidule
- Spiking neural networks
- Ubichip architecture
- Ubicell
- Sequencer
  - Introduction
  - Multiprocessor (SIMD) architecture
  - Characteristics
  - Instruction set
  - Internal architecture
- Development tools
- Ongoing and future work



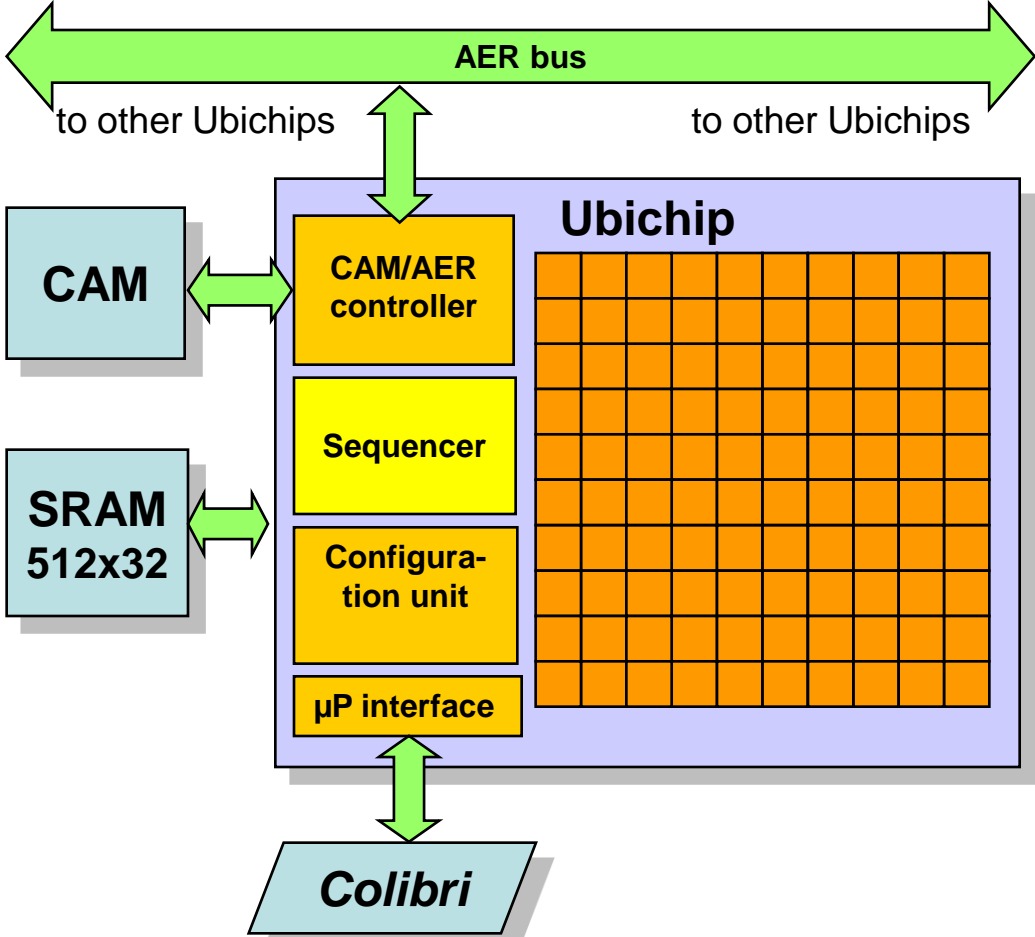
# Sequencer Introduction

The ubichip sequencer controls the program and data flow in the Ubichip in multi-processor mode (SIMD).

Tasks performed:

- Instruction fetching and decoding.
- Instruction broadcasting to the PE array.
- Sequencer-specific instruction execution.
- Data transfer between SRAM and PE array.
- Interfacing with CAM/AER controller. Access provided to SRAM and PE array.

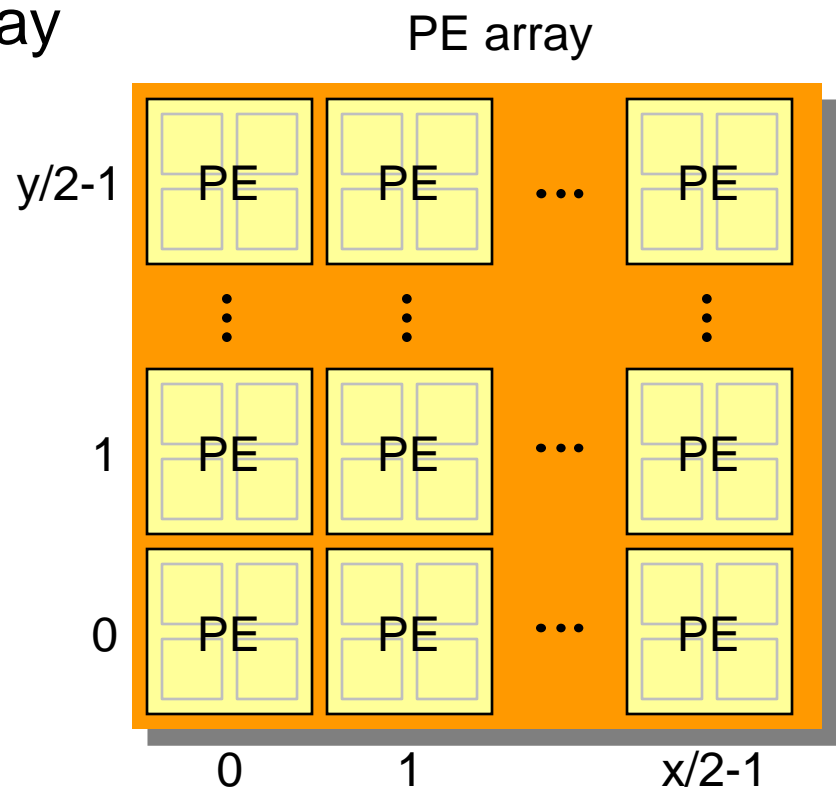
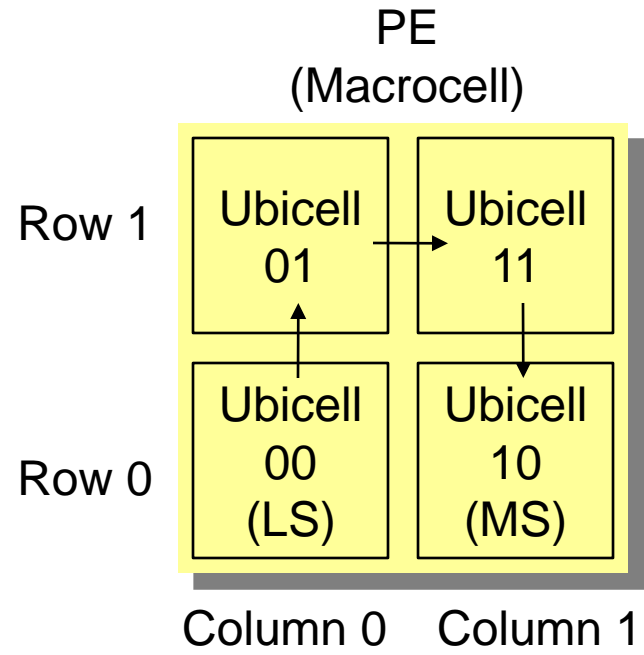
# Sequencer Multiprocessor (SIMD) architecture



# Sequencer

## Multiprocessor (SIMD) architecture (2)

- 16-bit PE (Macrocell) organization proposal
- y-column x-row ubicell array



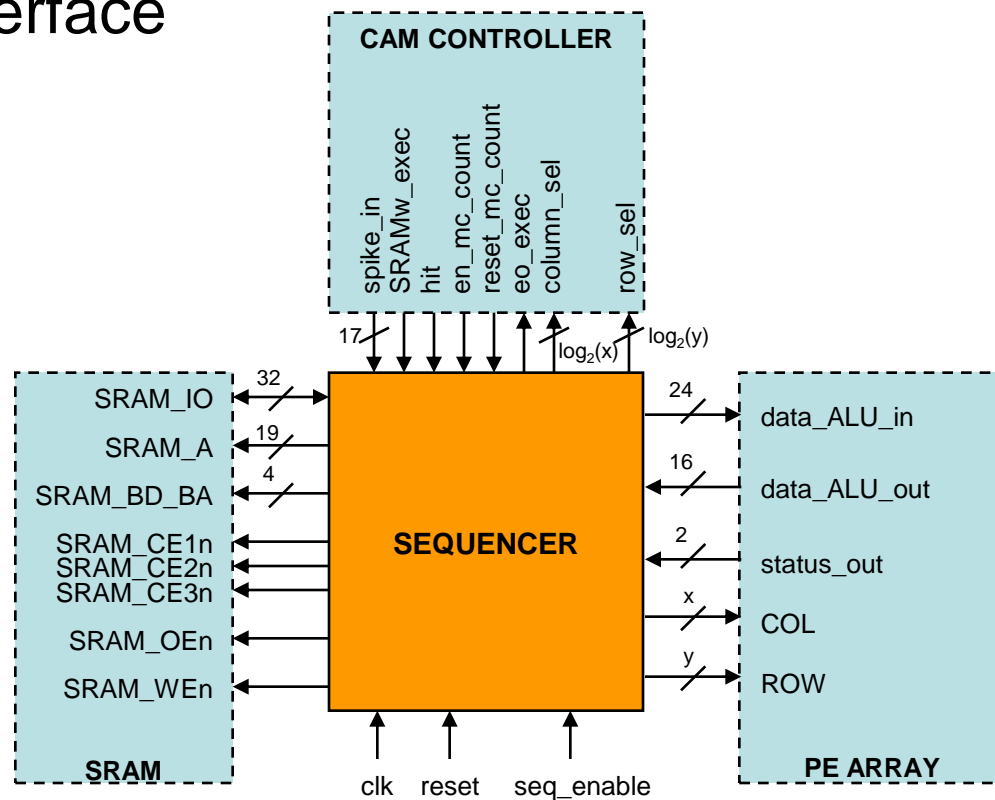


# Sequencer Characteristics

- Common data + instructions memory (SRAM).
- Von Neumann (vs. Harvard) architecture.
- ALU and register bank are external (PE array).
- SIMD processor. Conditional instructions:
  - Conditional store
  - FREEZE/UNFREEZE

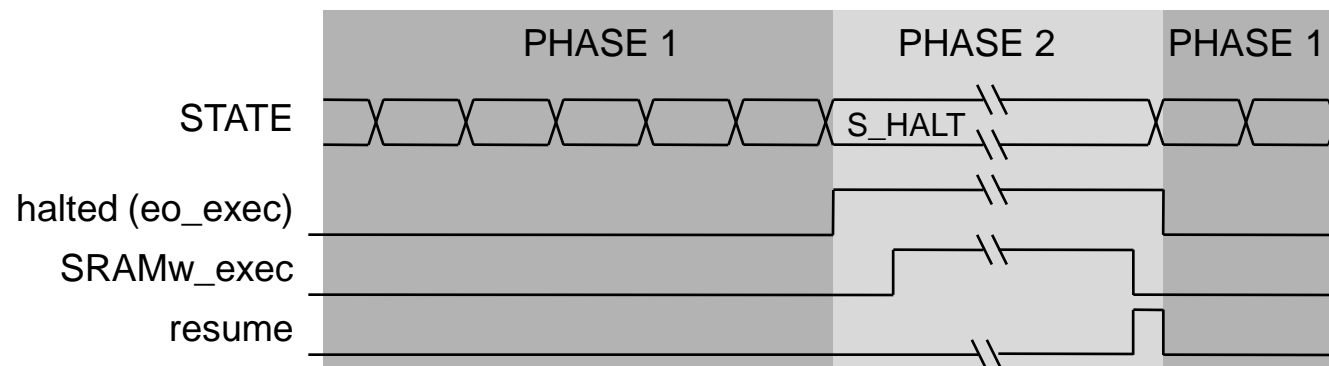
# Sequencer Characteristics (2)

- Signal interface



# Sequencer Characteristics (3)

- Operation phases
  - Phase 1: execution
  - Phase 2: spike transmission (sequencer halted)



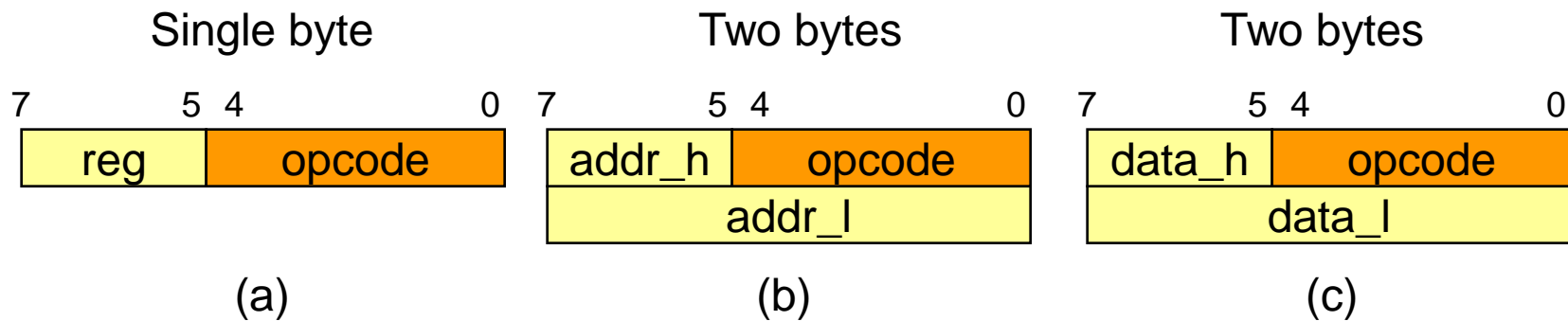
# Sequencer Instruction set

- Groups of instructions

Instruction group	Class	#bytes	Description
NOP	A	1	No operation
STOREC	A	1	Loop of conditional ACC store to SRAM
ALUREG	A	1	ALU register transfer
ALUOP	A	1	ALU arithmetic/logic operation
LOAD_ALL*	A	1	Simultaneous data load to macrocells
LOAD	A	1	Sequential data load to macrocells
SETMP	S	2	Select SRAM pointer
READMP	S	1	Read memory pointer from SRAM
LOOP	S	2	Start loop. Counter setting
ENDL	S	1	End loop. Counter decreasing
GOTO	S	2	Jump instruction
HALT**	S	1	Halt instruction for the CAM controller
RET**	S	1	Return instruction

# Sequencer Instruction set (2)

- Instruction formats



ALL except:

SETMP  
GOTO

LOOP

# Sequencer Instruction set (3)

- ALU operations

Mnemonic	Opcode	Group	Function	Flags
ADD reg	01000	ALUOP	ACC <= ACC + reg	Z, C
SUB reg	01001	ALUOP	ACC <= ACC - reg	Z, C
SHL	01010	ALUOP	ACC <= ACC<<, carry = ACC[msb**]	Z, C
SHR	01011	ALUOP	ACC <= ACC >>, carry = ACC[lbsb**]	Z, C
AND	01101	ALUOP	ACC <= ACC AND source_dest	Z
OR	01110	ALUOP	ACC <= ACC OR source_dest	Z
INV	01111	ALUOP	ACC <= NOT source_dest	Z
XOR reg	11110	ALUOP	ACC <= ACC XOR reg	Z
SET reg	10001	ALUOP	reg <= "1111"	Z
RST reg	00100	ALUOP	reg <= "0000"	Z

# Sequencer

## Instruction set (4)

- ALU register-related
  - FREEZE nesting depth: 3

Mnemonic	Opcode	Group	Function	Flags
MOVA reg	01100	ALUREG	ACC <= source_dest	Z
SWAP reg	10010	ALUREG	reg <=> shadow_reg	-
MOVTS reg	10101	ALUREG	shadow_reg <= reg	-
MOVFS reg	10110	ALUREG	reg <= shadow_reg	-
FREEZEC	10111	ALUREG	Disables the registers of the ALU if C=1 ****	-
FREEZENC	11000	ALUREG	Disables the registers of the ALU if C=0 ****	-
FREEZEZ	11001	ALUREG	Disables the registers of the ALU if Z=1 ****	-
FREEZENZ	11010	ALUREG	Disables the registers of the ALU if Z=0 ****	-
UNFREEZE	11011	ALUREG	Enables the registers of the ALU *	-
MOVR reg	11100	ALUREG	reg <= ACC	-

# Sequencer Instruction set (5)

- PE input-output

Mnemonic	Opcode	Group	Function	Flags
STC reg	00001	STOREC	SRAM $\leftarrow$ reg if C=1	-
STNC reg	00010	STOREC	SRAM $\leftarrow$ reg if C=0	-
STZ reg	00011	STOREC	SRAM $\leftarrow$ reg if Z=1	-
STNZ	00101	STOREC	SRAM $\leftarrow$ reg if Z=0	-
LDALL reg	10011	LOAD_ALL	reg $\leftarrow$ data_in_ALU (broadcast) ***	-
LOAD reg	10100	LOAD	reg $\leftarrow$ data_in_ALU (iterative)	-

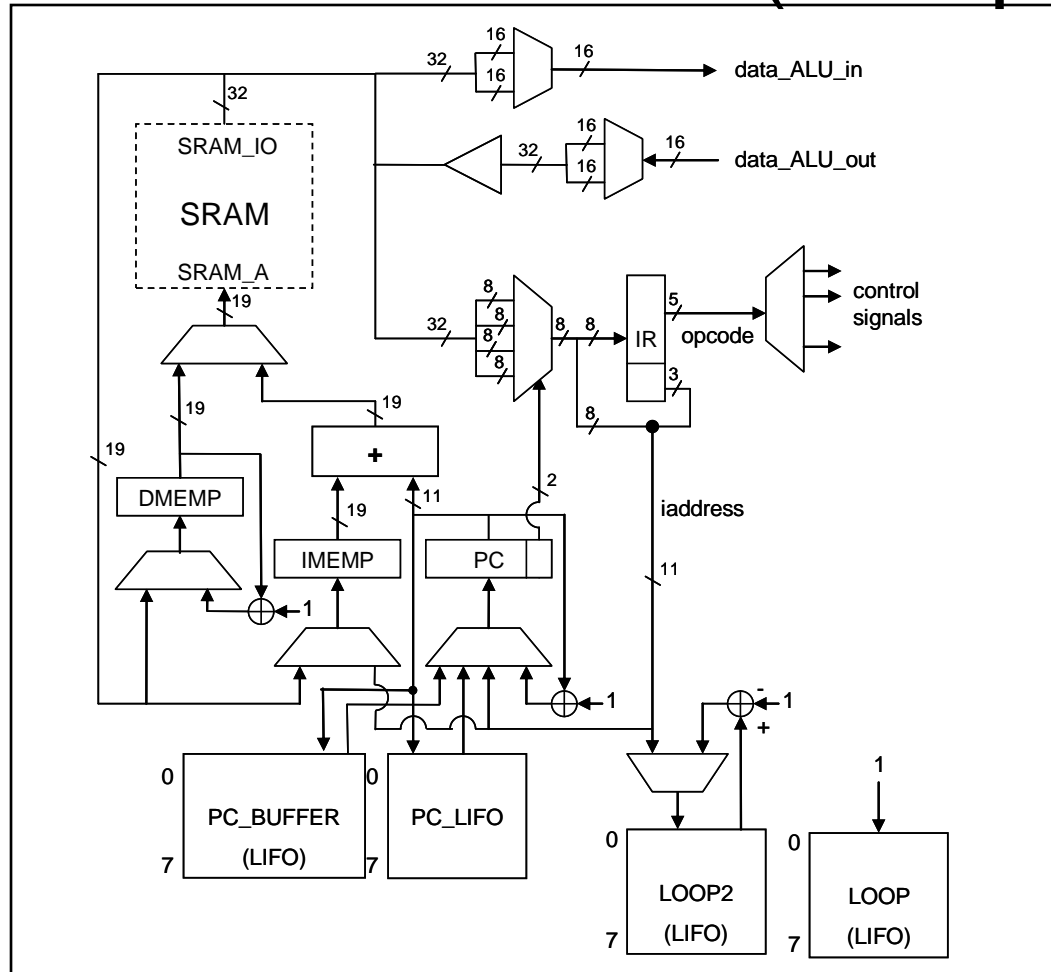
# Sequencer Instruction set (6)

- Sequencer-specific instructions & NOP
  - RET nesting depth: 8 (configurable)
  - LOOP nesting depth: 8 (configurable)

Mnemonic	Opcode	Group	Function	Flags
NOP	00000	NOP	No operation	-
HALT*	00000	HALT	Stops the sequencer until input signal SRAMw_exec = 0 (from CAM controller)	-
RET*	00000	RET	PC <= PC_BUFFER	-
SETMP data	00110	SETMP	DMEM_P <= data	-
READMP n	00111	READMP	if n=0, DMEM_P <= MEM(DMEM_P) if n>0, DMEMP <= MEM(DMEM_P + 2 <sup>(n-1)</sup> * LOOP_index)	-
LOOP data	10000	LOOP	push PC_LIFO, LOOP_LIFO, LOOP_LIFO2; LOOP_LIFO(0) <= 1; LOOP_LIFO2(0) <= data	-
ENDL	11101	ENDL	if LOOP_LIFO(0) = LOOP_LIFO2(0) then pop PC_LIFO, LOOP_LIFO, LOOP_LIFO2 else restore(PC); LOOP_LIFO(0) <= LOOP_LIFO(0)+1;	-
GOTO addr	11111	GOTO	PC <= addr; PC_BUFFER <= PC	-

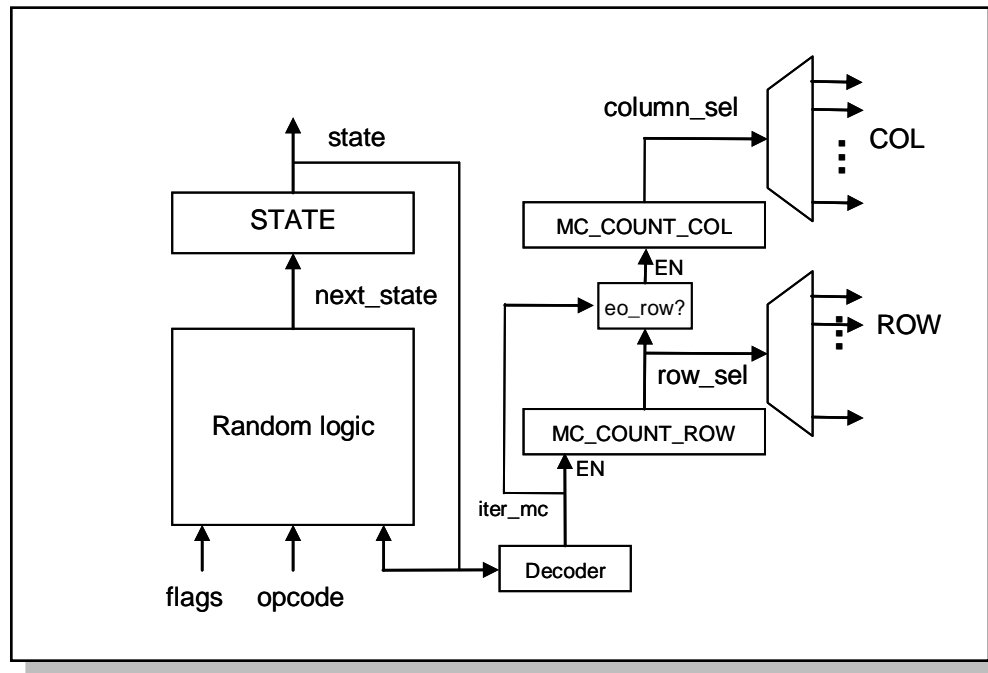
# Sequencer

## Internal architecture (Datapath)



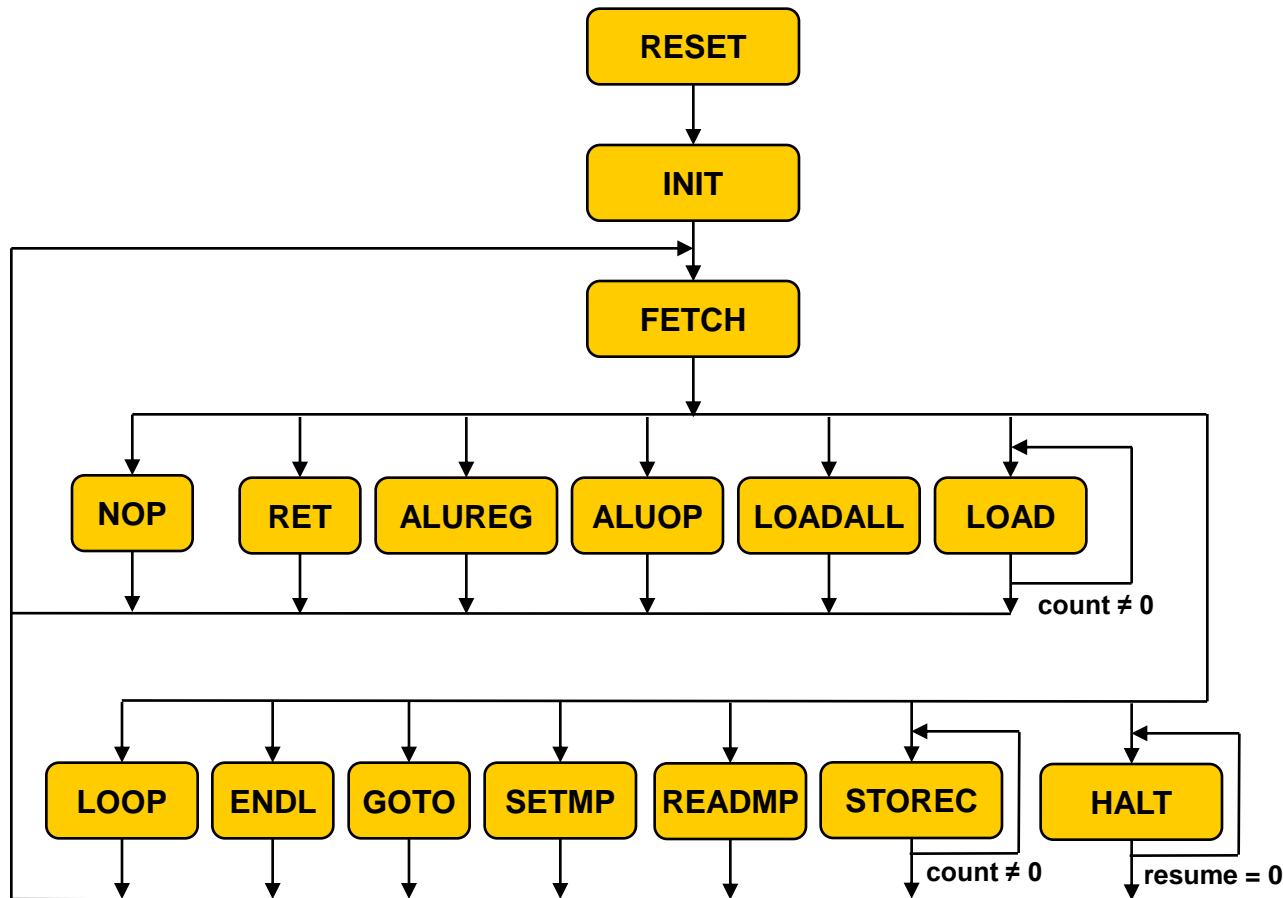
# Sequencer

## Internal architecture (Control unit)



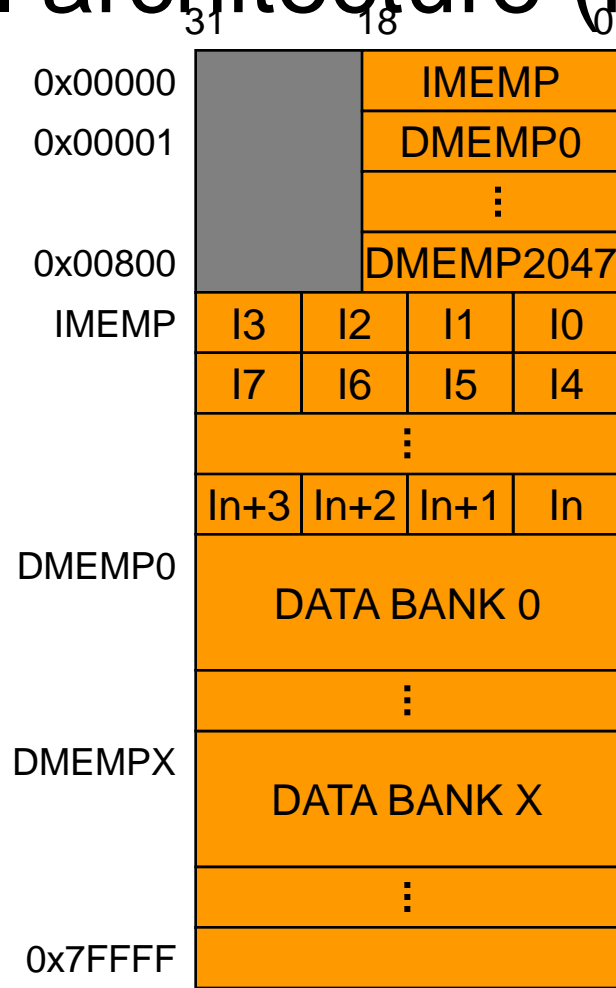
# Sequencer

## Internal architecture (Control unit FSM)



# Sequencer

## Internal architecture (memory map)





# Outline

- The Perplexus project
- Ubidule
- Spiking neural networks
- Ubichip architecture
- Ubicell
- Sequencer
- Development tools (SpiNDeK)
  - Objectives
  - Assembler
  - GUI
  - Spiking neural net modeling
  - Design flow
- Ongoing and future work



# Development tools (SpiNDeK)

## Objectives

- Automate the configuration of spiking neural networks.
  - Parameter tuning.
  - Structure generation.
- Develop an assembler to simplify the ubichip software programming.
- Automatic generation of the memory contents.
  - Code section (SRAM).
  - Data section (SRAM).
  - Synapse section (CAM).
- Provide a GUI to ease the development integration.
  - Simple graphic display of the generated network.
  - Connect to the digital simulation tool.
  - User-friendly presentation of register evolution.



# Development tools (SpiNDeK)

## Assembler

- Internally stores mnemonic and translated opcode lines.
- Java TreeMap structure (hash map) to store the line number.
- Assembling errors can be traced back and displayed.
- Expanded to write VHDL testbenches which could be simulated.
- Features
  - Comments (;)
  - .data label (data segment)
  - .code label (code segment)
  - Names of variables instead of data pointer (LOAD instructions)
  - Command line mode

# Development tools (SpiNDeK) GUI

Code frame  
(assembler editor)

The screenshot shows the SpiNDeK GUI with four main panels:

- Code Frame (Assembler Editor):** Displays assembly code for neuron connections, including instructions like `ldall r0, LMAX`, `swap r2`, `sub r2`, `shl`, `freezenc`, `movfs r3`, `movfs r1`, `sub r1`, `movr r3`, `ldall r0, LMAX`, `shr`, `movr r2`, `unfreeze`, `unfreeze`, `movfs r3`, `freezenc`, and `---`.
- Parameter Frame (NN parameter frame):** A configuration window for neuron parameters.
 

Dimension Of Neurons:	20	20
Neurons per Chip:	10	10
Type of Neuron:	Excitatory	Inhibitory
Proportion in Network:	0.8	0.2
Uniform Connection Probability:	0.02	0.0
Gaussian Maximal Probability:	0.6	0.2
<b>Neuron/Synapse Variables</b>		
Type of Neuron:	Excitatory	Inhibitory
Postsynaptic Potential (P):	84	-80
Membrane Resting Potential (VRest):	-7800	-7800
Membrane Treshold Potential (Theta):	-4000	-4000
Membrane Time Donator (Dmem):	61309	61309
Synaptic Plasticity Time Donator (Dsyn):	63918	63918
Activation Time Donator (Dact):	65530	65530
Real valued Maximum (LMax):	16383	
Max. Memory of last Spike (MMax):	1638	
- NN graphical frame:** A 2D grid of neurons represented by colored dots (green, red, blue) on a black background.
- Message console:** Displays simulation statistics.
 

--- Statistics ---			
Statistic	Total	Excitatory	Inhibitory
NumOfNeurons	400	305	95
NumOfSynapses	33772	25829	7943
average NumOfSyn/Neuron	84	64	19
peak NumOfSyn/Neuron	126	126	45

NN parameter  
frame

NN graphical  
frame

Message console



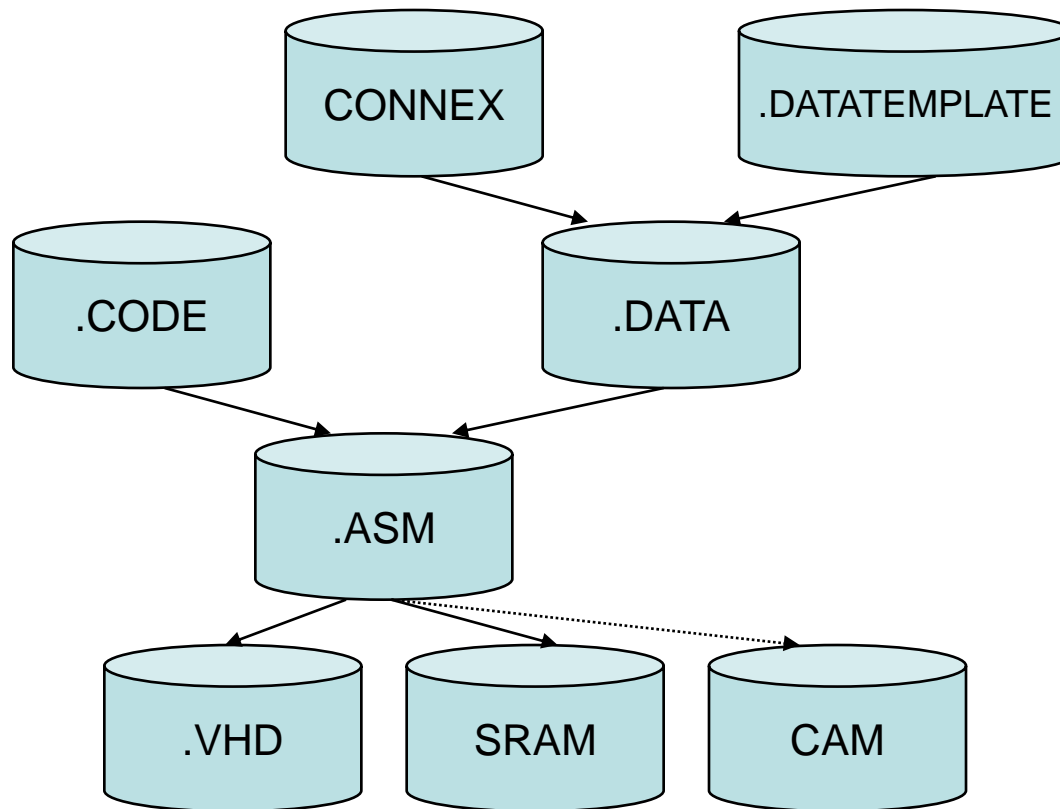
# Development tools (SpiNDeK)

## Spiking neural net modeling

- Mathematical model analyzed and compared.
- Initial pseudo algorithm has been corrected.
  - Model modifications from UJF.
  - Dependencies checked.
  - Still remaining: Full precision analysis.
  - Adopted solution: 16 bit for the unknowns.
- Checking dependencies.
- Parameter tuning.
- Parameter normalization (floating-point to integer).
- Structure generation.

# Development tools (SpiNDeK)

## Design flow





# Outline

- The Perplexus project
- Ubidule
- Spiking neural networks
- Ubichip architecture
- Ubicell
- Sequencer
- Development tools (SpiNDeK)
- Ongoing and future work



# Ongoing and future work

- Completion of two modules.
  - Multi-chip allocation.
  - CAM file generation.
- Full testing and debugging.
  - Single macrocell (2x2 ubicell).
  - Rectangular array.
  - Square 10x10 array.
- Full FPGA prototyping and Ubichip implementation.



## Ongoing and future work (2)

- Full integration with CAM/AER controller.
- Flag propagation.
  - Connection order in multi-processor.
- New instructions:
  - SETZ, SETC, RESZ, RESC
  - Unconditional STORE
- Support to data transfer between processors
- 1-cycle execution for single-byte instructions that do not use SRAM.