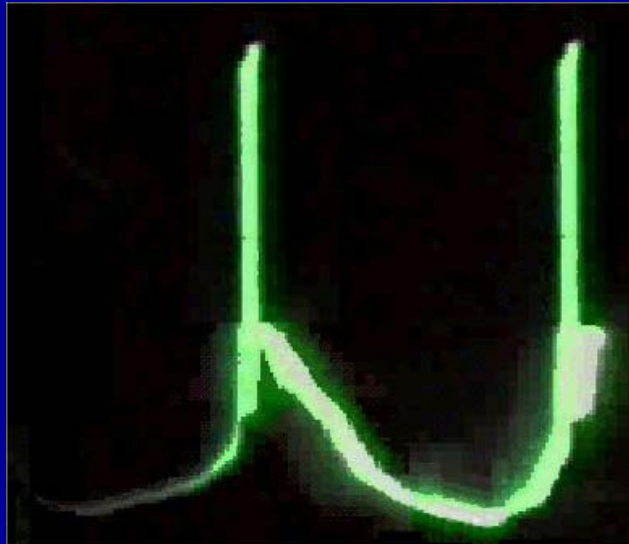


EDLUT: Event-Driven simulator based on LookUp Tables

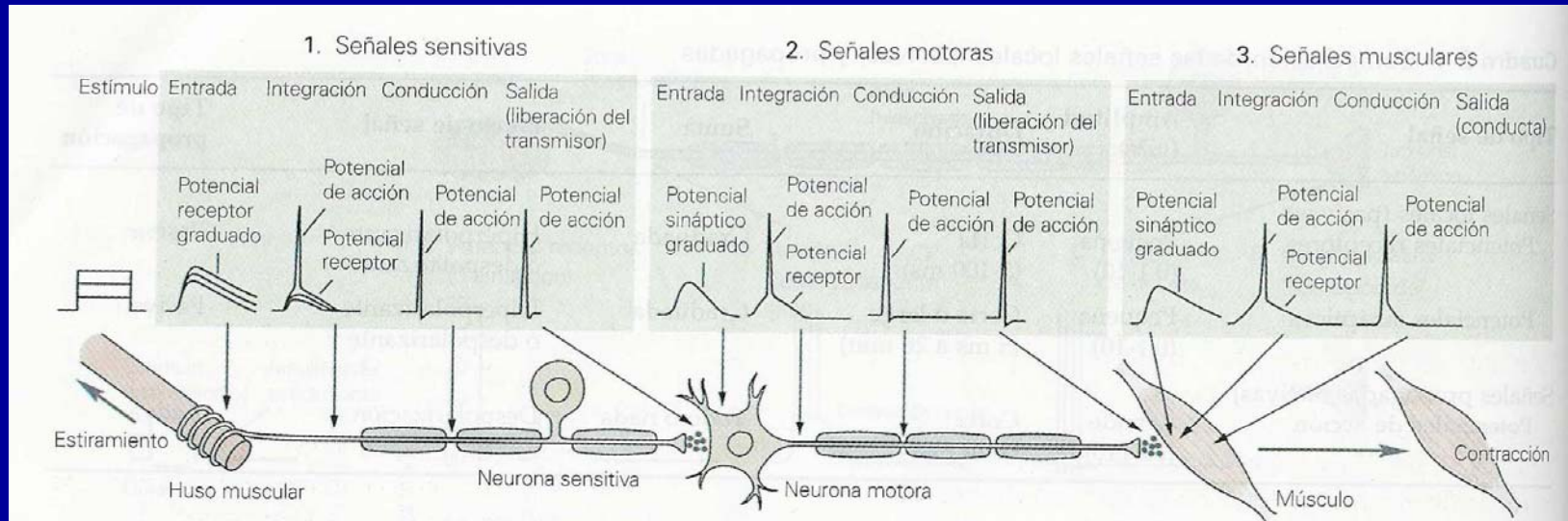
Richard Carrillo and Eduardo Ros
University of Granada



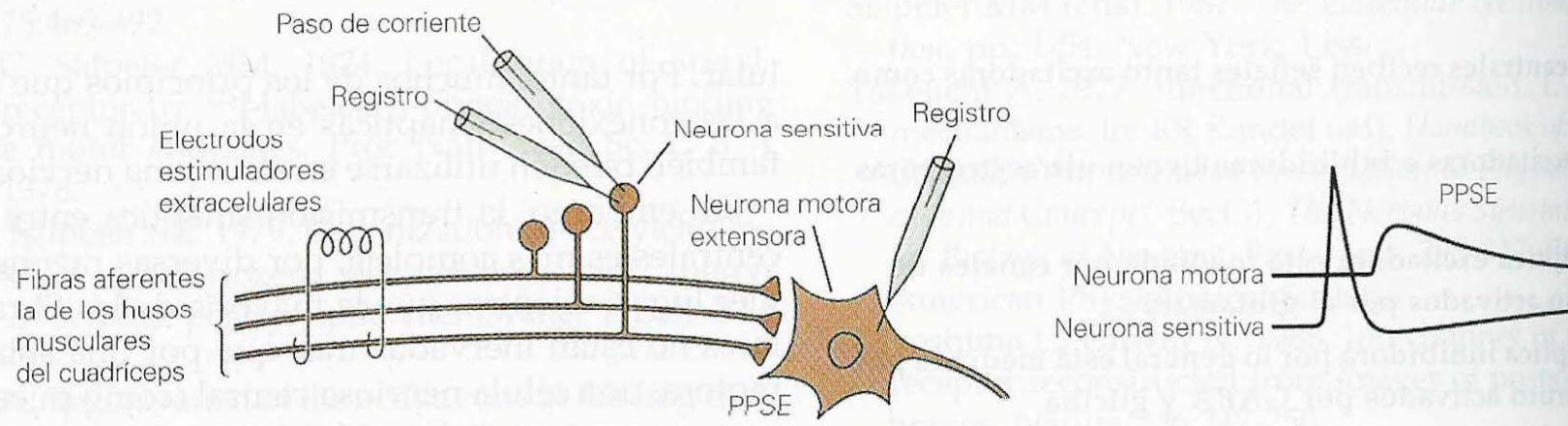
Simulator objectives

- Simulation of biologically plausible spiking neural structures
- Allow the simulation of different neuron models
 - Allow the incorporation of new neural features into neuron models without needing to modify the simulator code.
- Real-time simulation of middle-scale neural networks (thousands of neurons).

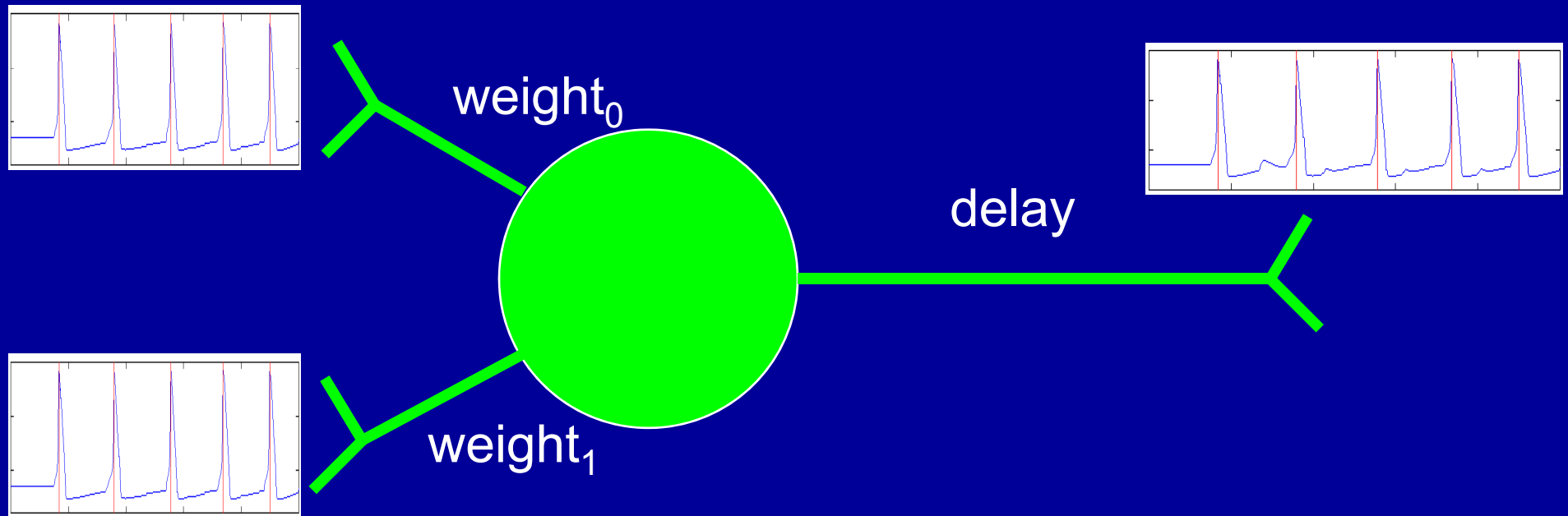
The biological neuron



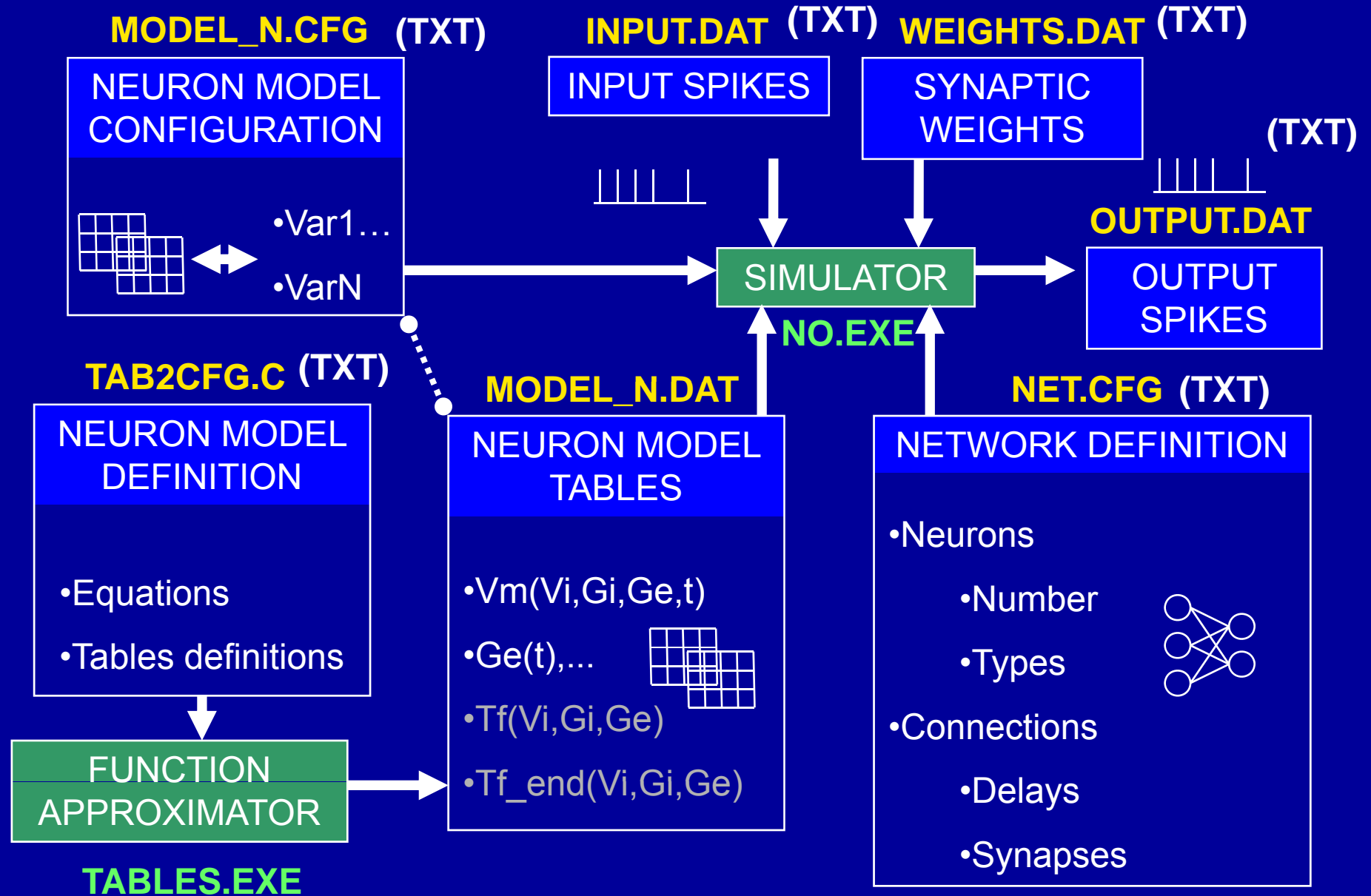
B Dispositivo experimental para el registro de las células del circuito



Spiking neuron



Simulator structure



Input spikes (INPUT.DAT)

FORMAT:

Number_of_total_spikes

{Time_of_first_spike Number_of_spikes Time_interval
First_neuron Number_of_neurons}

Example:

20

0.100 10 0.005 0 1

0.200 1 0 1 10

Network topology and learning rules (NET.CFG)

FORMAT:

[Definition of network neurons]

[Definition of learning rules (weight modifications)]

[Definition of the connections between neurons]

[Definition of network neurons]

FORMAT:

Number_of_neuron_types_in_the_network

Total_number_of_neurons_in_the_network

[Number_of_neurons Name_of_neuron_type Monitorized]

Example:

2

8

4 Granular 0

4 Golgi 1

[Definition of connections between neurons]

FORMAT:

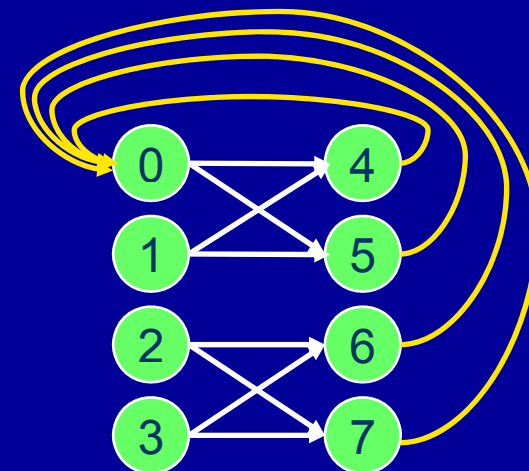
Total_number_of_connections_between_neurons

[Num_of_first_source_neuron Num_of_source_neurons
Num_of_first_target_neuron Num_of_target_neurons
Num_of_repetitions Delay(secs) Increment_of_delay Type
Max_conductance(nS) Num_of_learning_rule]

Example:

12

0	2	4	2	2	0.001	0.000	0	0.8	-1
4	4	0	1	1	0.002	0.001	0	0.8	-1



Synaptic weights (WEIGHTS.DAT)

FORMAT:

[Number_of_weights_to_define Conductance(nS)]

- If Number_of_weights_to_define=0, all the remaining weights are established
- If Conductance(nS)<0, The weight is chosen randomly between 0 and |Conductance(nS)|
- If Conductance(nS)> Max_conduc(nS), The weight is set up to the value Max_conduc(nS)

Example:

8 0.01

0 -0.5

Output file (OUTPUT.DAT)

FORMAT:

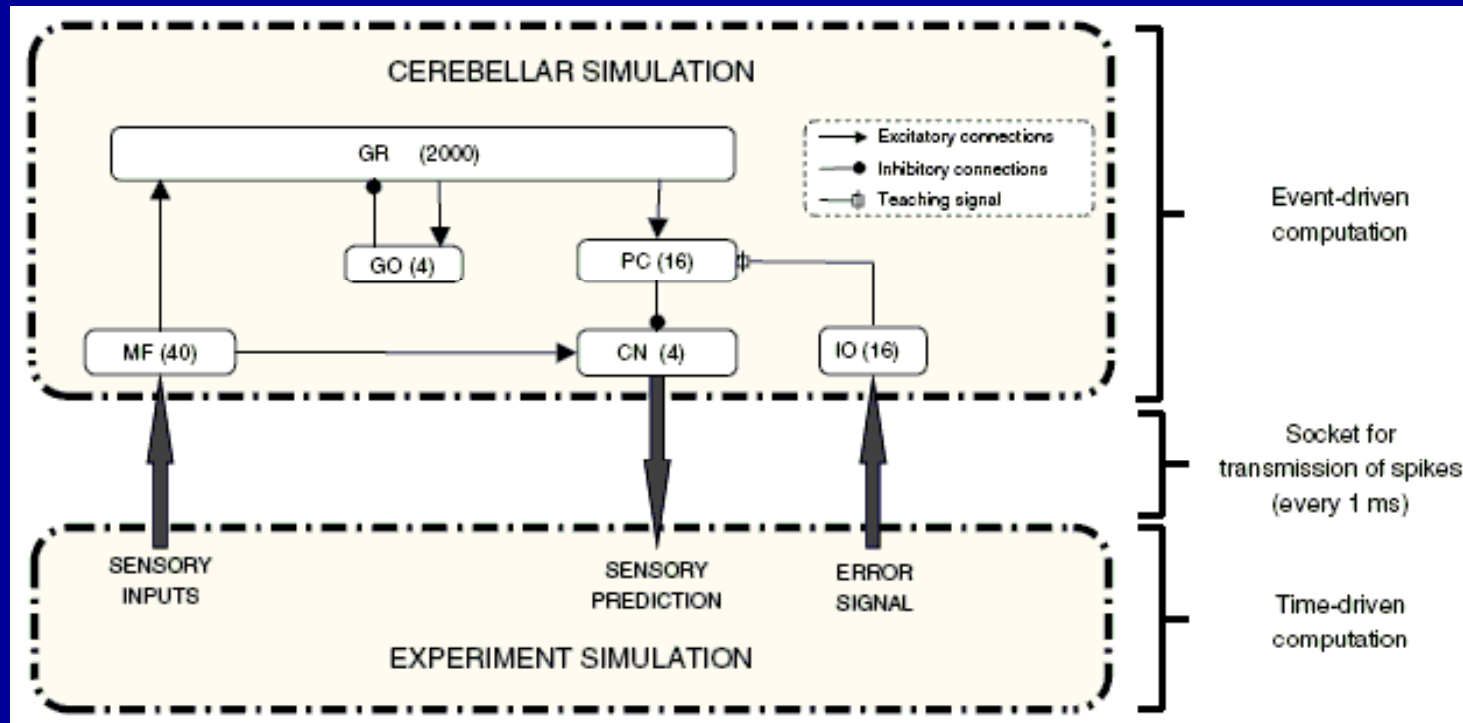
[Time Num_of_neuron State_variable(or 1.0)]

- If State_variable=1.0, the neuron Num_of_neuron has emitted a spike at Time

Example:

```
0.001 0 -0.070
0.002 0 -0.065
0.003 0 -0.055
0.004 0 1.0
0.004 1 1.0
```

Learning rules

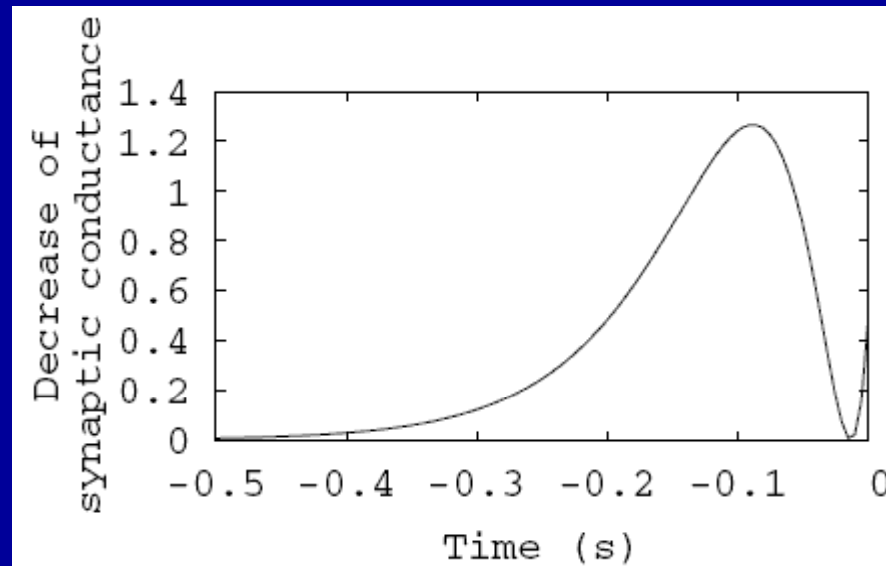


$$\text{if } X_{target}(t) < 0, I(t) = \alpha (0.15 - a X_{pred+}(t))$$

$$\text{if } X_{target}(t) \geq 0, I(t) = \alpha (0.15 + b (X_{target}(t) - X_{pred+}(t)))$$

Learning rules

$$W_i = W_i - \int_{-\infty}^{t_{IO}} K(t - t_{IO}) \delta_i(t) dt$$



LTD

(Long-term depression)

$$W_i = W_i + \delta_{LTP}$$

LTP

(Long-term potentiation)

[Definition of the rules for weight changing] (NET.CFG)

FORMAT:

Number_of_learning_rules

[Change_trigger Localization_of_peak(in_secs)

Ind_change_coefficient Dep_change_coefficient]

- $W_i = W_i + \text{Ind_Change_coefficient} * (\text{Max_}W_i - W_i)$
- $W_i = W_i + \text{Dep_change_coefficient} * F_n(\text{previous_activity})$

Example:

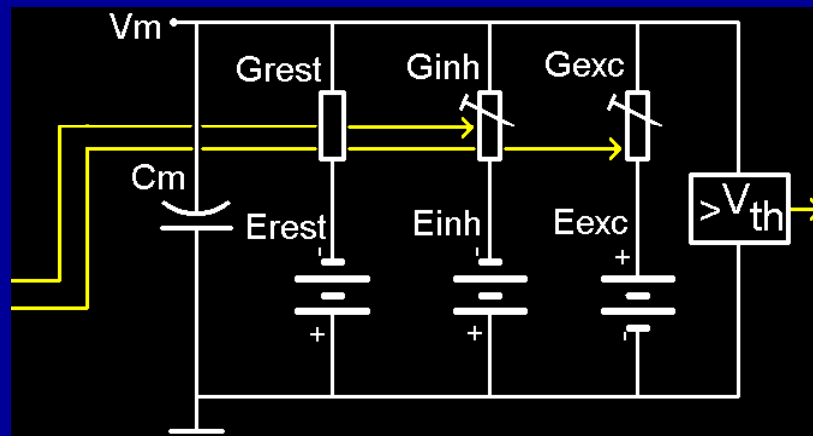
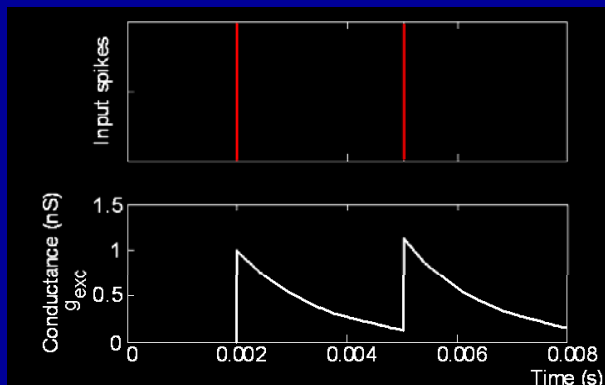
2

0 0.1 0.005 -0.03

1 0.0 0.0 0.0

Neuron model

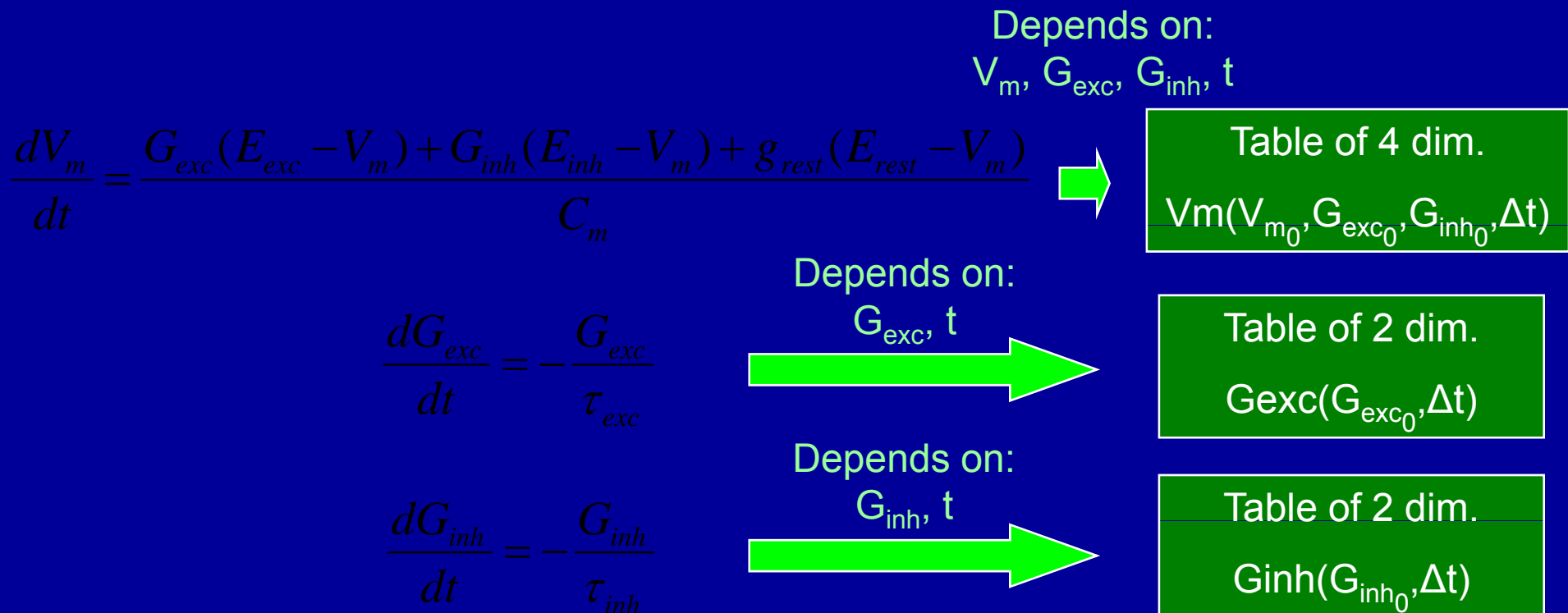
- Neural state
- Functions to calculate the value of each state variable
 - $V_m = f_{V_m}(V_{m_0}, G_{exc_0}, G_{inh_0}, \Delta t), G_{exc} = f_{G_{exc}}(G_{exc_0}, \Delta t), \dots$
- Functions for the firing prediction (output events)
 - $T_{dis} = f_{T_{dis}}(V_{m_0}, G_{exc_0}, G_{inh_0}), T_{fin_dis} = f_{T_{fin_dis}}(V_{m_0}, G_{exc_0}, G_{inh_0})$



Model equations

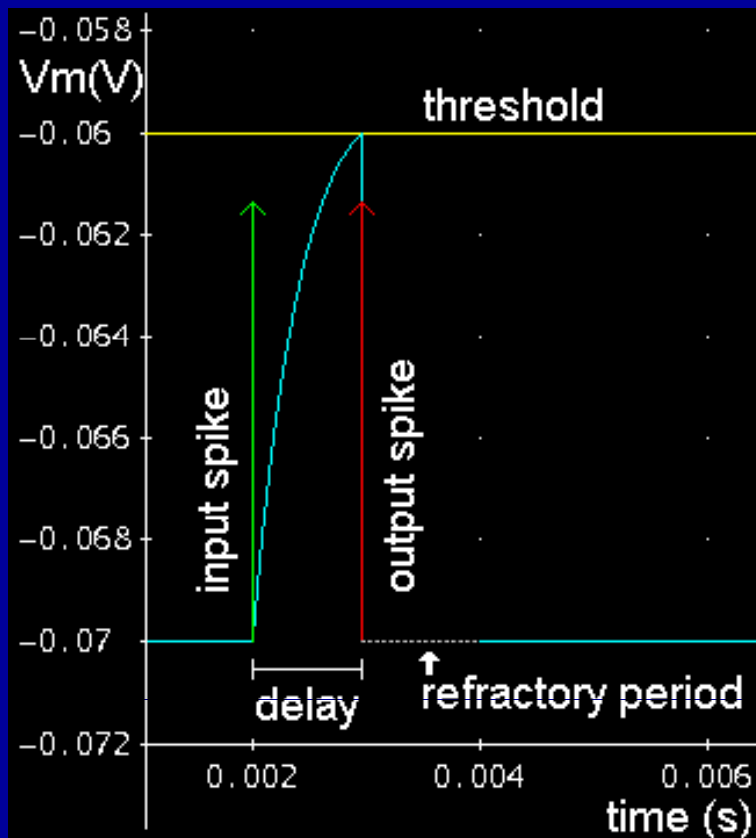
Example: integrate and fire with conductance-based synapses

- Neuron parameters: C_m , E_{exc} , τ_{exc} , E_{inh} , τ_{inh} , E_{rest} , g_{rest}
- Neuron state variables: V_m , G_{exc} , G_{inh}



Model tables

- In this neuron model the membrane potential takes a time to reach the firing threshold after receiving a spike.



- This delay depends on the neural state (V_m , G_{exc} , G_{inh}) in the instant the input spike is received

Euler / Runge-Kutta method

Table of 3 dim.

$Tf(V_{m_0}, G_{exc_0}, G_{inh_0})$

Euler / Runge-Kutta method

Table of 3 dim.

$Tf_end(V_{m_0}, G_{exc_0}, G_{inh_0})$

Table generator

- Input file for the generator: TAB2CFG.C

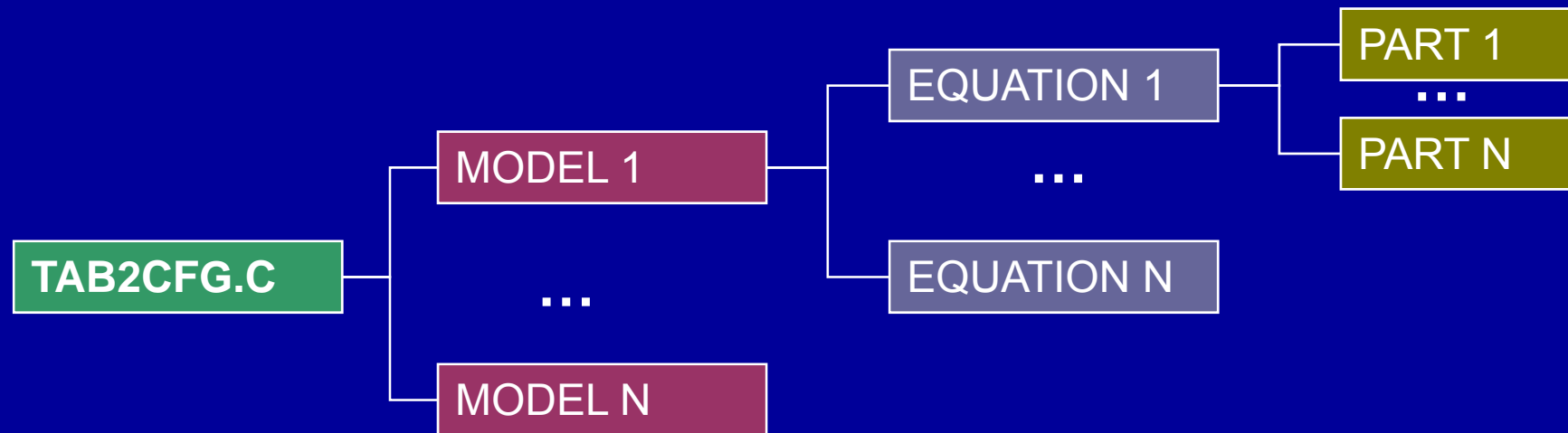


Table definition file (TAB2CFG.C)

FORMAT:

[Declaration of equations and neural variables]

[Definition of functions and function's selectors]

[Definition of tables]

[Declaration of equations and neural variables]

FORMAT:

```
#define NUM_EQS Number_of_equations

struct consts
{
    float Constants_used_in_the_equations;
};

union vars
{
    float list[NUM_EQS+1];
    struct
    {
        float t;
        float neural_variables;
    } named;
};
```

[Definition of functions and function's selectors]

FORMAT:

```
inline float Name_of_the_fn_calc(union vars *v, struct consts *c, float h)
{
    return(Computation_of_the_corresponding_neural_variable);
}
struct teq_sys
{
    float (*eq)(union vars *, struct consts *, float);
    int diff;
} Eq_sys[][NUM_EQS]={
    {{Name_of_Fn_linked_with_var1,Fn_diferential},...}
    ...
};

inline int Name_of_the_fn_selector(union vars *v, struct consts *c)
{
    return(0);
}
int (*(Eq_sel[]))(union vars *, struct consts *)=
    {Name_of_the_fn_selector,...};
```

[Defintion of tables]

FORMAT:

- Num_of_files_to_generate
 - Name_of_file
 - Num_of_tables_in_the_file
 - Values_of_constants_declared
 - Initialization_of_state_variables
 - Num_of_equations_to_compute
 - List_of_equations
 - Num_of_dimensions_of_the_table
 - Num_of_the_variable_corresponding_to_the_dimension
 - Num_of_intervals
 - First_coord_value Last_coord_value
 - Num_of_coords Coord_Distrib_lin_or_log
 - Num_of_selector

Configuration of the neuron model (MODEL_N.CFG)

FORMAT:

Number_of_neural_state_variables

Num_of_tables_corresponding_to_each_variable

Initialization_of_the_state_variable

Num_of_table_for_firing_prediction

Num_of_table_for_firing_end_prediction

Number_of_synaptic_state_variables

Num_of_table_corresponding_to_each_variable

Num_of_used_tables

Num_of_dims Num_of_var_corresp_to_first_dim Interpolation_of_first_dim...N

...