VLSI Lab - Politecnico di Torino
Department of Electronics and Telecommunications
https://www.vlsilab.polito.it/

Authors
Yuri Ardesi - yuri.ardesi@polito.it
Giuliana Beretta - giuliana.beretta@polito.it

SCERPA DOCUMENTATION

1 Introduction

This file explains how to write input scripts for SCERPA. SCERPA consists of three parts: **layout**, **algorithm** and **viewer**. The general command to call SCERPA is `SCERPA(command,option1,option2)`, where `command` specifies which part of the code the user want to run, and `option1` and/or `option2` MUST be *struct* variables containing information for SCERPA. The handled possibilities are listed in Table 1.

Table 1: Possible commands to start SCERPA.

COMMAND	OPTION 1	OPTION 2	DESCRIPTION
'generate'	circuit	-	run only the **layout**
'launch'	algorithm settings	-	run only the **algorithm**
'generateLaunch'	circuit	algorithm settings	run both the **layout** and the **algorithm**
'plotSteps'	viewer settings	-	run only the **viewer**

The *struct* variables for SCERPA options contain several fields that are described in this document: Section 2 deal with *circuit definition*, Section 3 explains the possible *algorithm settings* and Section 4 lists the available *viewer settings*. At the end of this document some examples are provided.

2 Circuit definition

There are two possibilities to describe the circuit layout:

1. using only a MATLAB script (Section 2.1);
2. using MagCAD[1] and extracting the *.qll* file (Section 2.2).

[1] https://topolinano.polito.it/the-project/

In the following, the two possibilities are handled separately. In both situations, there are some fields that are **mandatory**, and they will be identified by the ⚠ symbol. The reference system is shown in Figure 1.

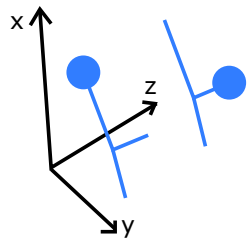

Figure 1: Reference system for the circuit layout.

2.1 Layout built with MATLAB

The following items are optional/mandatory fields of a unique structure.

magcadImporter ⚠ is a variable used to specify whether circuit is described through MATLAB (set to 0) or MagCAD (set to 1).

structure ⚠ is a matrix which specify the circuit layout. In each position one can insert a driver, an output, or a molecule. Drivers are identified by the string *Dr*, outputs are identified by the string *out*, molecules are identified by the number of the clock zone to which they belong.

Values_Dr ⚠ is a matrix in which for each driver (row) is specified its input voltage for each time step (column). Must be consistent with **structure**.

clockMode is a variable which specifies which clocking mode to use. The two possible values for this variable are 'phase' and 'map'. In phase mode, the user MUST include **stack_phase**. In map mode, the user MUST include **ckmap.coords** and **ckmap.field**. [*default* 'phase']

stack_phase ⚠ is a matrix in which for each clock zone (row) is specified the clocking electric field for each time step (column). Must be consistent with **structure**.

ckmap.coords ⚠ is a matrix which specifies the coordinates in Ångström in which the values of the clocking electric field are provided. First column is for z-axis coordinates, second column is for y-axis coordinates. Must be consistent with **ckmap.field**.

ckmap.field ⚠ is a matrix in which for each coordinates couple (row) specified in **ckmap.coords** is specified the clocking electric field for each time step (column). Must be consistent with **ckmap.coords**.

dist_z specifies the intermolecular distance along the z-axis in Ångström [*default* 10 Å]

dist_y specifies the intermolecular distance along the y-axis in Ångström [*default* 20 Å [2]]

molecule is a char vector containing the name of the molecules, chosen from the Molecule list below, even those in brackets [*default* bisfe_4]

components is a matrix, with the same dimensions of the **structure**, containing in each position the molecule type expressed with the corresponding number in the Molecule list below [*default* all elements equal to **molecule**] [3]

rotation_x is a matrix with the same dimensions of the **structure**, containing in each position the molecule rotation around the x-axis expressed in degree [*default* 0°]

[2] In previous versions of SCERPA the default value was **dist_z** + inter-dot distance of the bis-ferrocene molecule.

[3] If **components** is present, the field **molecule** is not taken into account. If **components** is missing, the circuit is composed with the same molecule in each position, and the molecule used is the one specified in the **molecule** field (default or not).

rotation_y is a matrix, with the same dimensions of the **structure**, containing in each position the molecule rotation around the y-axis expressed in degree [*default* 0°]

rotation_z is a matrix, with the same dimensions of the **structure**, containing in each position the molecule rotation around the z-axis expressed in degree [*default* 0°]

shift_x is a matrix, with the same dimensions of the **structure**, containing in each position the molecule shift along the x-axis expressed in Ångström [*default* 0 Å]

shift_y is a matrix, with the same dimensions of the **structure**, containing in each position the molecule shift along the y-axis expressed in Ångström [*default* 0 Å]

shift_z is a matrix, with the same dimensions of the **structure**, containing in each position the molecule shift along the z-axis expressed in Ångström [*default* 0 Å]

Vext is a matrix, with the same dimensions of the **structure**, containing in each position the fixed external voltage applied onto the molecule expressed in volts [*default* 0 V]

plotLayout is a variable used to specify whether SCERPA should plot the layout of the circuit (1 is yes, 0 is no) [*default* 0]

Molecules list

Below the list of the available molecules with the identification number used in SCERPA. The terms in brackets specify which molecule is actually associated to the *circuit.molecule* field.

0 ⟶ bisfe4_ox_counterionOnCarbazole

1 ⟶ bisfe4_ox_counterionOnThiol (bisfe_4)

2 ⟶ bisfe4_ox_counterionOnThiol_orca (bisfe_4_orca)

3 ⟶ bisfe4_ox_noCounterion

4 ⟶ bisfe4_ox_noCounterion_TSA_2states

5 ⟶ bisfe4_ox_noCounterion_TSA_3states

6 ⟶ bisfe4_sym

7 ⟶ butane_ox_noCounterion (butane)

8 ⟶ butane_ox_noCounterion_orca

9 ⟶ butaneCam

10 ⟶ decatriene_ox_noCounterion (decatriene)

11 ⟶ linear_mol_w7_a2000 (linear_w7)

12 ⟶ linear_mol_w7_a3000

13 ⟶ linear_mol_w9_a3000 (linear_w9)

14 ⟶ linear_mol_w95_a3000 (linear_w95)

2.2 Layout built with MagCAD

The following items are optional/mandatory fields of a unique structure.

magcadImporter ⚠ is a variable used to specify whether circuit is described through MATLAB (set to 0) or MagCAD (set to 1).

qllFile ⚠ contains the path of the *.qll* file in which the circuit under test has been described.

doubleMolDriverMode ⚠ is a variable used to specify whether the drivers should be implemented with one molecule (set to 0) or two molecules (set to 1).

Values_Dr ⚠ is a matrix in which for each driver (row) is specified its input voltage for each time step (column). If **doubleMolDriverMode** is set to 1, in this matrix MUST be present the values for both molecules: the name of the added molecule MUST be the same of the other one followed by '_c'.

clockMode is a variable which specifies which clocking mode to use. The two possible values for this variable are 'phase' and 'map'. In phase mode, the user MUST include **stack_phase**. In map mode, the user MUST include **ckmap.coords** and **ckmap.field**. [*default* 'phase']

stack_phase ⚠ is a matrix in which for each clock zone (row) is specified the clocking electric field for each time step (column).

ckmap.coords ⚠ is a matrix which specifies the coordinates in Ångström in which the values of the clocking electric field are provided. First column is for z-axis coordinates, second column is for y-axis coordinates. Must be consistent with **ckmap.field**.

ckmap.field ⚠ is a matrix in which for each coordinates couple (row) specified in **ckmap.coords** is specified the clocking electric field for each time step (column). Must be consistent with **ckmap.coords**.

3 SCERPA settings

The following items are optional fields of a unique structure [4].

Algorithm Output and Runtime Plot :

- out_path: this setting enables the specification of the path for SCERPA output files [*default* '/scerpaPath/OUTPUT_FILES']
- plot_plotAbsoluteCharge: if set to 1 it plots the absolute value of the dots charge in the 3D layout plot, otherwise it considers the actual value of the dots charge [*default* 1]
- plotIntermediateSteps: if set to 1, it opens a window during the calculation which shows the trend of charges and input voltages at each step of the iterative procedure. Setting to 1 this setting strongly slows down the execution [*default* 0]
- plotActiveRegionWindow: if set to 1, during the execution of the code, a figure shows the voltage of all the molecules an highligth the molecules inserted in the Active Region list. [*default* 0]
- plot_3dfig: if set to 1 it plots the 3D layout of the circuit for each time step [*default* 0]
- plot_voltage: if set to 1 it plots the input voltage on each molecule for each time step [*default* 0]
- plot_chargeFig: if set to 1 it plots the charge value of the logic dots for each molecule, and for each time step [*default* 0]
- plot_clock: if set to 1 it plots the clock field on each molecule for each time step [*default* 0]
- plot_molnum: if set to 1, it writes the molecule label in the 3D layout plot [*default* 1]

[4] Plots settings in this structure refers to runtime plots (they slow down execution).

- verbosity: this setting defines how much informations are written in the console during the simulation, if set to 0 means that no data are written, if set to 1 the convergence step is written for each time step, if set to 2 convergence info are written for each time step [*default* 0]
- pauseStep: this command, if set to 1, pauses the execution of scerpa at each step of the plotIntermediateSteps='1'. [*default* 0]

Convergence settings :

- max_step: set the maximum number of SCF iterations, after having reached this value, precision is lowered to LP and then to LLP if needed (see Precision settings below) [*default* 1000]
- immediateUpdate: if set to 1, the algorithm updates the charge value onto the dots at each substep, one molecule at a time. This let the algorithm to converge faster, but it is far from the physical behaviour, since molecules actually "update" all together. To speed up the convergence without loosing the physical meaning, one should appropriately choose the damping factor. Hint: set this setting to 0, unless strictly needed [*default* 0]
- damping: damping affects the rate and mode of convergence without influencing the final result, if correctly chosen. Value must be in range [0 - 1) [*default* 0.4]
- autodamping: if set to 1, it enables the automatic choice of the damping factor based on voltage variation value [*default* 0]

Convergence accelerations :

- enableRefining: if set to 1, once the algorithm reaches convergence it disables the active region and continues the evaluation, it stops otherwise [*default* 1]
- enableActiveRegion: if set to 1, the algorithm evaluates the interaction among a reduced set of molecule, avoiding the evaluation of molecules whose charge is not supposed to vary in the current step [*default* 1]
- activeRegionThreshold: this value defines which molecules belong to the active region, if the voltage variation is higher the this value, the molecule is added to the active region list [*default* 0.0015]
- enableInteractionRadiusMode: if set to 1, the algorithm evaluate, for each molecule, only the effects of molecules which are close to the one under evaluation (within a specific distance specified by the setting interactionRadius) [*default* 1]
- interactionRadius: this value defines the maximum distance (in Å) the algorithm might consider in the evaluation of electric field. Chosen a specific molecule, only the effects generated by the molecule with lower distance than this value will be considered in the evaluation [*default* 101]

DEBUG informations :

- printConvergenceTable: if set to 1, SCERPA prints the "Convergence Table". This table reports all the input voltages of each molecule and the subcomponents related to each intermolecular interaction. It allows detecting possible bugs and errors on SCERPA. [*default* 0]

Precision :

- LPmode: maximum number of steps to try to reach convergence in Low Precision Mode [*default* 200]
- LPPmode: maximum number of steps to try to reach convergence in Very Low Precision Mode [*default* 300]
- conv_threshold_HP: when the SCF error reaches the convergence threshold, the algorithm stops. If it is not reached, precision is lowered to LP and and the algorithm runs in LPmode. If convergence is not achieved even in this mode, precision is lowered to LLP and and the algorithm runs in LLPmode [*default* 0.000005]
- conv_threshold_LP: convergence threshold in Low Precision Mode [*default* 0.0005]

- conv_threshold_LLP: convergence threshold in Very Low Precision Mode [*default* 0.005]

MATLAB optimization :

- enableJit: if set to 0, the MATLAB just-in-time compiler is disabled. This possibly slows down the calculation, yet allows performing more effective computational complexity analyses [*default 1*]

Driver saturation :

- driverSaturation: set to 1 whether the charge values of the driver should saturate, 0 otherwise [*default* 0]

Generation of the Additional Information TXT file :

- dumpClock: if set to 1, the TXT file reports the clock of each molecule in each timestep [*default* 0]
- dumpVout: if set to 1, the TXT file reports the voltage of each molecule in each timestep [*default* 0]
- dumpDriver: if set to 1, the TXT file reports the voltage of each driver in each timestep [*default* 0]
- dumpOutput: if set to 1, the TXT file reports the voltage of each output in each timestep [*default* 0]
- dumpComputationTime: if set to 1, the TXT file reports the time duration of each timestep in seconds [*default* 0]
- dumpEnergy: if set to 1, the TXT file reports energy values in each timestep [*default* 0]

4 SCERPA Viewer settings

The following items are optional fields of a unique structure.

Output path :

- out_path: MUST be the same path specified for the SCERPA algorithm, where the output files are already present [*default* '/scerpaPath/OUTPUT_FILES']

General Plot Settings :

- fig_saver: if set to 1, the viewer saves *.fig files of each picture [*default* 0]
- plotSpan: the viewer can print a limited number of steps, if plotSpan = N, the viewer plots a figure every N steps [*default* 1]
- plotList: is a vector that can be used to specify which timestep to print (plotList = [2 6 7] will print only steps 2, 6 and 7). If set to 0, the viewer plots all the available steps, considering the settings **plotSpan**. [*default* 0]

3D Plot settings :

- plot_3dfig: if set to 1, it enables the 3D plot [*default* 1]
- plot_3dfig_plotAbsoluteCharge: if set to 1, the viewer plots the absolute charge distribution [*default* 1]
- plot_3dfig_molnum: if set to 1, the viewer plots the name of each molecule on the figure [*default* 1]

1D Plot settings :

- plot_1DCharge: if set to 1, it enables the 1D plot [*default* 0]

Logic Plot settings :

- plot_logic: if set to 1, it enables the logic plot showing the information encoding on the cell according to the standard QCA definition [*default* 0]

Potential Plot settings :

- plot_potential: if set to 1, it enables the potential plot showing voltage generated by the molecular charge distribution [*default* 0]
- plot_potential_padding: this value allows enlarging the region where the potential is calculated [*default* 20]
- plot_potential_saturationVoltage: this value enables saturating the potential to improve graphical clarity [*default* 6]
- plot_potential_tipHeight: this value, in Å, defines the height of the virtual tip measuring the potential [*default* -5.5]

Waveform Plot settings :

- plot_waveform: if set to 1, it enables the plot of input/output waveform (Warning: this plot requires input/output information in the Additional Information file. Be sure the simulation was run with **dumpDriver** = 1 and **dumpOutput** = 1) [*default* 0]

5 Examples

5.1 Three phase wire

```
clear all
close all

%clock values definition
clock_low = -2;
clock_high = +2;
clock_step = 3;


%layout (MagCAD)
file = 'threePhasesWire.qll';
circuit.qllFile = sprintf('%s\\%s',pwd,file);
circuit.doubleMolDriverMode = 1;
circuit.magcadImporter = 1;

%molecule
% circuit.molecule = 'bisfe_4';

%layout (MATLAB)
% circuit.structure = {'Dr1_c' 'Dr1' '1' '1' '1' '1' '1' '1' '1' '1' ...
%     '2' '2' '2' '2' '2' '2' '2' '2' '3' '3' '3' '3' '3' '3' '3' '3'};
% circuit.magcadImporter = 0;

%drivers and clock
D0 = num2cell(-4.5*ones(1,clock_step*4));
D1 = num2cell(+4.5*ones(1,clock_step*4));
Dnone = num2cell(zeros(1,clock_step*4));

circuit.Values_Dr = {
    'Dr1'   D0{:} D1{:} Dnone{:}
    'Dr1_c' D1{:} D0{:} Dnone{:}
};

%clock
pSwitch =  linspace(clock_low,clock_high,clock_step);
pHold =   linspace(clock_high,clock_high,clock_step);
```

```
pRelease =  linspace(clock_high,clock_low,clock_step);
pReset =   linspace(clock_low,clock_low,clock_step);
pCycle = [pSwitch pHold pRelease pReset];

circuit.stack_phase(1,:) = [pCycle pCycle, pReset pReset];
circuit.stack_phase(2,:) = [pReset pCycle pCycle, pReset];
circuit.stack_phase(3,:) = [pReset pReset, pCycle pCycle];


%SCERPA settings
settings.out_path = '..\threePhaseWire';
settings.damping = 0.6;
settings.verbosity = 2;
settings.dumpDriver = 1;
settings.dumpOutput = 1;
settings.plotIntermediateSteps = 0;

%PLOT settings
plotSettings.plot_waveform = 1;
plotSettings.plot_3dfig = 1;
plotSettings.plot_1DCharge = 1;
plotSettings.plot_logic = 1;
plotSettings.plot_potential = 1;
plotSettings.plotSpan = 3;
plotSettings.fig_saver = 1;
plotSettings.plotList = 0;

%copy outputh path from algorithm settings if specified by the user
if isfield(settings,'out_path')
    plotSettings.out_path = settings.out_path;
end

%%%%
this_path = pwd;
scerpa_path = '..\';
cd(scerpa_path)
generation_status = SCERPA('generateLaunch', circuit, settings);
                    SCERPA('plotSteps', plotSettings);
cd(this_path)
```

5.2 Three phase wire clock map

```
clear all
close all

%clock values definitions
clock_low = -2;
clock_high = +2;
clock_step = 3;

%layout (MagCAD)
file = 'threePhasesWire.qll';
circuit.qllFile = sprintf('%s\\%s',pwd,file);
circuit.doubleMolDriverMode = 1;
circuit.magcadImporter = 1;
```

```
%molecule
% circuit.molecule = 'bisfe_4';

%layout (MATLAB)
% circuit.structure = {'Dr1_c' 'Dr1' '1' '1' '1' '1' '1' '1' '1' '1' ...
%    '2' '2' '2' '2' '2' '2' '2' '2' '3' '3' '3' '3' '3' '3' '3' '3'};
% circuit.magcadImporter = 0;

%drivers and clock
D0 = num2cell(-4.5*ones(1,clock_step*4));
D1 = num2cell(+4.5*ones(1,clock_step*4));
Dnone = num2cell(zeros(1,clock_step*4));

circuit.Values_Dr = {
    'Dr1'   D0{:} D1{:} Dnone{:}
    'Dr1_c' D1{:} D0{:} Dnone{:}
};

%
circuit.clockMode='map';

%defined here (can be obtained by a csv file)
    z = 400*rand(1000,1)-20;
    y = 100*rand(1000,1)-20;
    Eclock1 = 4*exp((-z.^2-y.^2)/3000) - 2;
    Eclock2 = 4*exp((-(z-100).^2-y.^2)/3000) - 2;
    Eclock3 = 4*exp((-(z-200).^2-y.^2)/3000) - 2;

circuit.ckmap.coords = [z y];
circuit.ckmap.field = [Eclock1 Eclock2 Eclock3];

%SCERPA settings
settings.damping = 0.6;
settings.verbosity = 2;
settings.plotIntermediateSteps = 0;
settings.plot_clock = 1;

%PLOT settings
plotSettings.plot_3dfig = 0;
plotSettings.plot_1DCharge = 1;
plotSettings.plot_logic = 0;
plotSettings.plot_potential = 1;
plotSettings.plotSpan = 1;
plotSettings.fig_saver = 0;
plotSettings.plotList = 0;

%%%%
this_path = pwd;
scerpa_path = '..\';
cd(scerpa_path)
generation_status = SCERPA('generateLaunch', circuit, settings);
                    SCERPA('plotSteps', plotSettings);
cd(this_path)
```

5.3 Three phase wire multi-molecule

```
clear all
close all

%definitions
clock_low = -2;
clock_high = +2;
clock_step = 3;

%layout (MagCAD)
% file = 'threePhasesWireMultiMol.qll';
% circuit.qllFile = sprintf('%s\\%s',pwd,file);
% circuit.doubleMolDriverMode = 1;
% circuit.magcadImporter = 1;

%layout (Layout Generator)
circuit.structure = {'Dr1_c' 'Dr1' '1' '1' '1' '1' '1' '1' '1' '1' ...
    '2' '2' '2' '2' '2' '2' '2' '2' '3' '3' '3' '3' '3' '3' '3' '3'};

circuit.components = {'0' '0' '0' '0' '0' '0' '0' '0' '0' '0' '0' '0' ...
    '0' '0' '0' '0' '0' '0' '7' '7' '7' '7' '7' '7' '7' '7'};

circuit.magcadImporter = 0;

%drivers and clock
D0 = num2cell(-4.5*ones(1,clock_step*4));
D1 = num2cell(+4.5*ones(1,clock_step*4));
Dnone = num2cell(zeros(1,clock_step*4));

circuit.Values_Dr = {
    'Dr1'   D0{:} D1{:} Dnone{:}
    'Dr1_c' D1{:} D0{:} Dnone{:}
};

%clock
pSwitch =  linspace(clock_low,clock_high,clock_step);
pHold =   linspace(clock_high,clock_high,clock_step);
pRelease =  linspace(clock_high,clock_low,clock_step);
pReset =   linspace(clock_low,clock_low,clock_step);
pCycle = [pSwitch pHold pRelease pReset];

circuit.stack_phase(1,:) = [pCycle pCycle, pReset pReset];
circuit.stack_phase(2,:) = [pReset pCycle pCycle, pReset];
circuit.stack_phase(3,:) = [pReset pReset, pCycle pCycle];


%SCERPA settings
settings.doubleMolDriverMode = 1;
settings.damping = 0.6;
settings.verbosity = 2;

%PLOT settings
plotSettings.plot_3dfig = 1;
plotSettings.plot_logic = 1;
plotSettings.plot_potential = 1;
plotSettings.plotSpan = 3;
plotSettings.fig_saver = 1;
```

```
plotSettings.plotList = 0;
plotSettings.plot_potential_tipHeight = -10;

%%%%
this_path = pwd;
scerpa_path = '..\';
cd(scerpa_path)
generation_status = SCERPA('generateLaunch', circuit, settings);
                    SCERPA('plotSteps', plotSettings);
cd(this_path)
```

6 SCERPA Bibliography

- Y. Ardesi, G. Turvani, M. Graziano and G. Piccinini, "SCERPA Simulation of Clocked Molecular Field-Coupling Nanocomputing," in IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 29, no. 3, pp. 558-567, March 2021, doi: 10.1109/TVLSI.2020.3045198.
- Y. Ardesi, R. Wang, G. Turvani, G. Piccinini and M. Graziano, "SCERPA: A Self-Consistent Algorithm for the Evaluation of the Information Propagation in Molecular Field-Coupled Nanocomputing," in IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 39, no. 10, pp. 2749-2760, Oct. 2020, doi: 10.1109/TCAD.2019.2960360.
- R. Wang, M. Chilla, A. Palucci, M. Graziano and G. Piccinini, "An effective algorithm for clocked field-coupled nanocomputing paradigm," 2016 IEEE Nanotechnology Materials and Devices Conference (NMDC), 2016, pp. 1-2, doi: 10.1109/NMDC.2016.7777166.