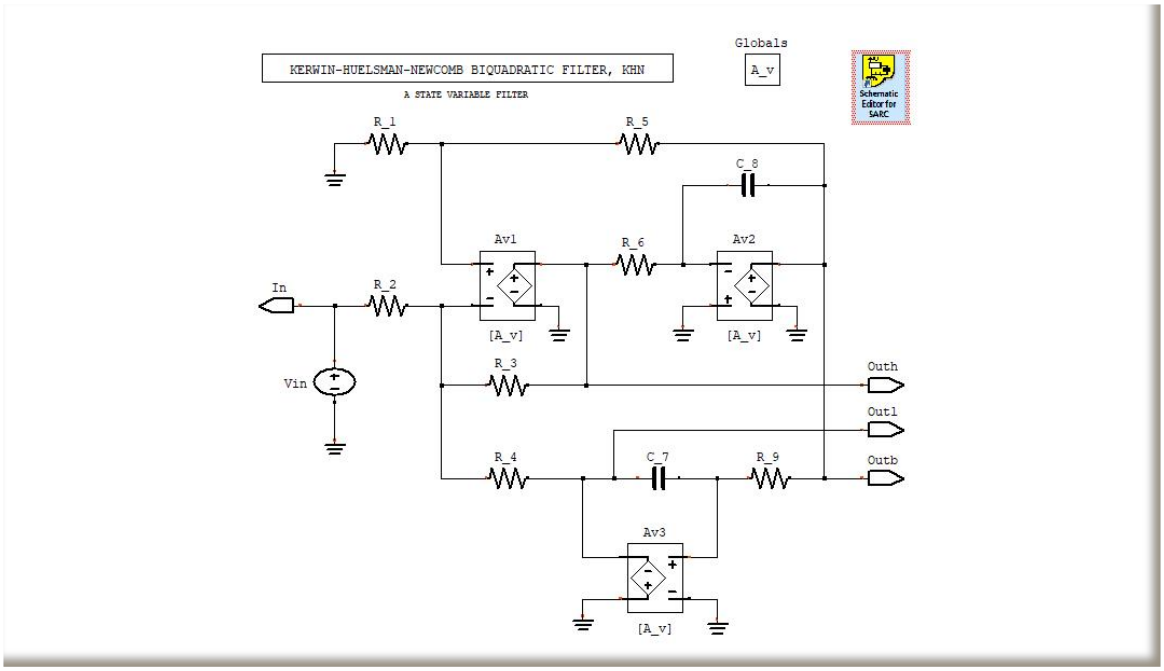


Transfer Functions, Modeling and Simulation using the SARC Schematic Editor

(Running on Windows OS, 64 bits)

Yves Leduc | July 2018

Introduction

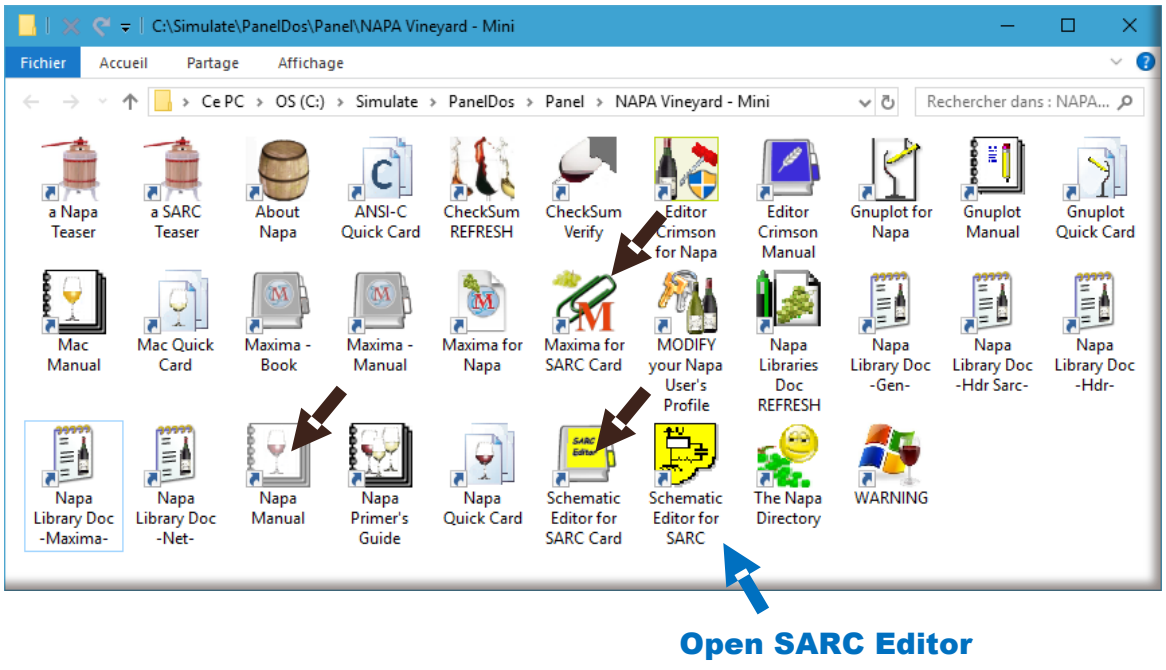
> **The SARC Schematic Editor, the symbolic computation of the transfer functions and the SARC simulations are the work of *Jacques Mequin*.**
> **The Napa Compiler and Package is the work of *Yves Leduc*.**
> ***http://www.borogoves.eu/NAPA_RELEASE_305e/***
>
> **Maxima is a descendant of Macsyma, the legendary computer algebra system developed in the late 1960s at the Massachusetts Institute of Technology.** ***http://maxima.sourceforge.net/***

The **SARC Schematic Editor** is the entry point to describe linear(ised) electrical circuits. It is closely linked to the symbolic programming tool Maxima which is used to compute, in the Laplace Domain, the transfer functions described by this editor and to mixed-signal simulations with the Napa package.

Maxima can be used to postprocess further the transfer functions. The functions produced by the pair SARC Schematic Editor/Maxima can be inserted in NAPA netlists.

The **Napa** Compiler is a compiler of netlists producing ad-hoc cycle-based ANSI-C simulators of mixed signal circuits. It can host continuous domain models described in the Laplace domain thanks to **SARC** (Semi Analytical Recursive Convolution).

The users' guides of these tools are not the object of this paper and can be found on the '**NAPA Vineyard**' Windows Panel created during the installation of the Napa package :

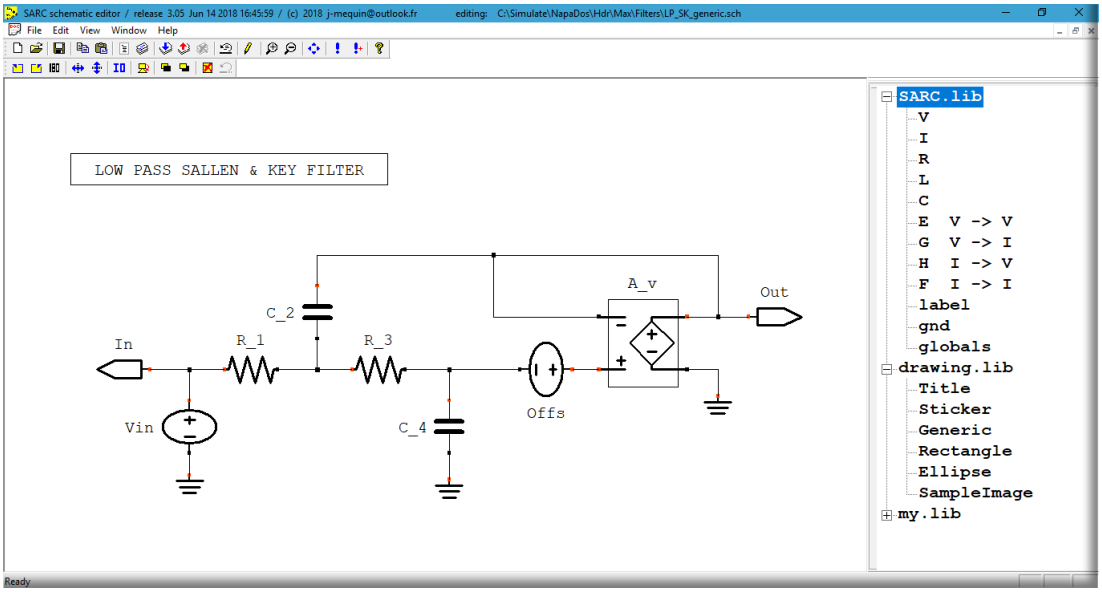


1. AN EXAMPLE OF SCHEMATIC

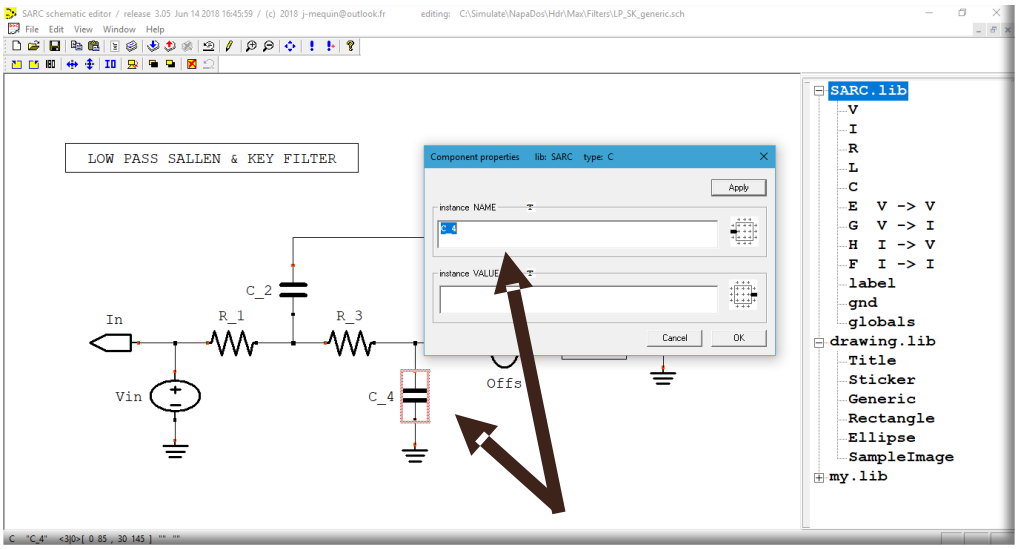
(*These examples use netlists located in directory '/**Simulate/NapaDos/Sch**'*)

We will use a simple filter to illustrate how to build a schematic and use it to produce transfer functions. We will limit our ambition for the moment to the simplest process.

This is the schematic of a **Sallen & Key Filter**. We have chosen in this example to limit the description to the main elements: 2 resistors, 2 capacitors, 2 independent voltage sources and 1 voltage-controlled voltage source selected in the right menu in the '**SARC.lib**' as respectively '**R**', '**C**', '**V**' and '**E**'.

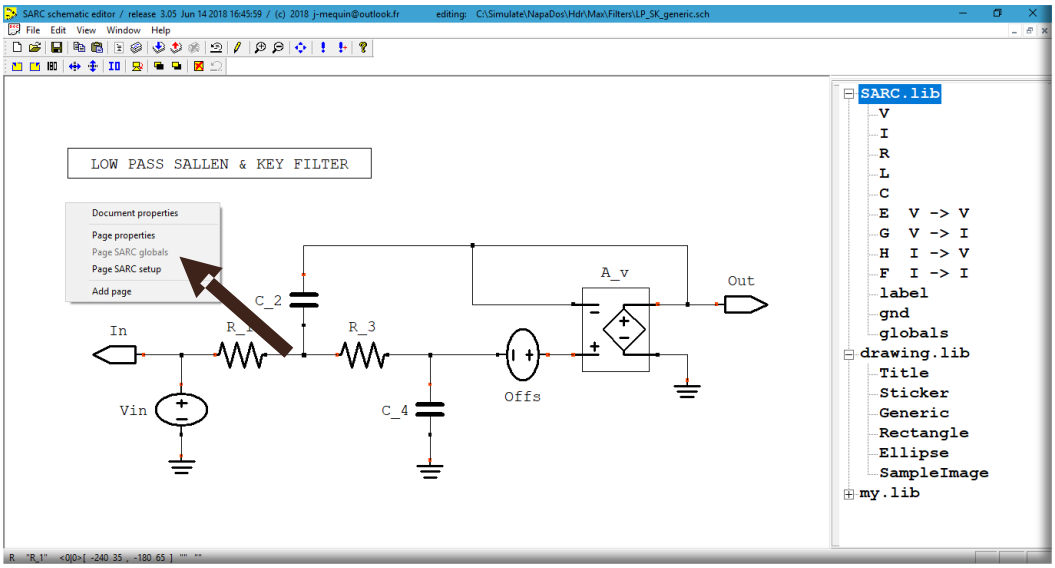


By a double click on an element, it is possible to enter its name and add some properties.



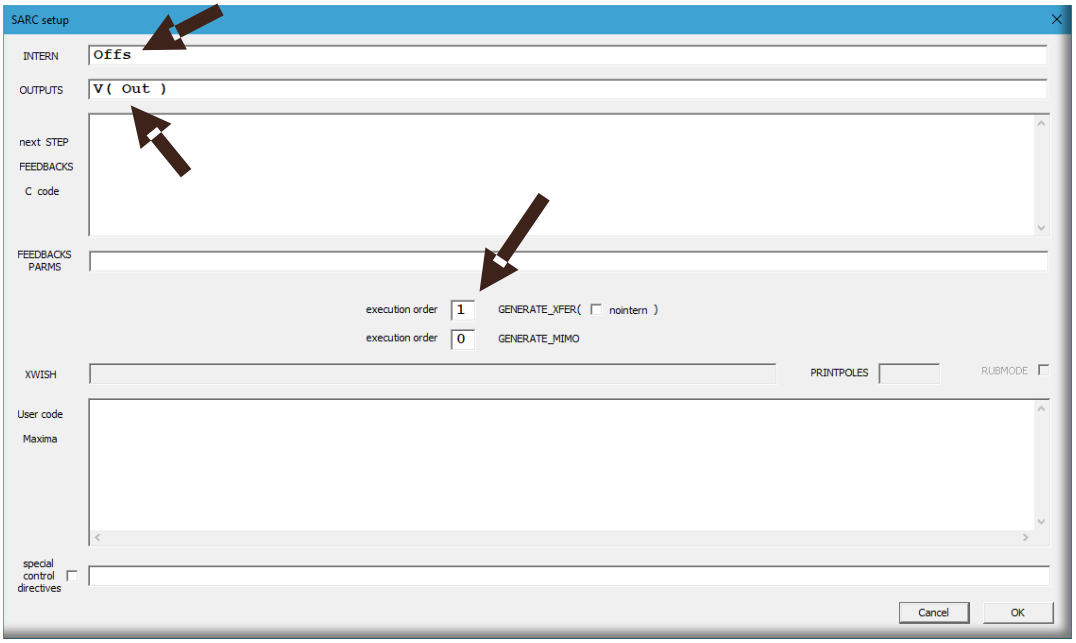
All constant sources (‘**V**’ or ‘**I**’) are by default the input of the circuits. We must indicate the output(s) we want to analyze: voltages or currents. (It is possible to output power and energy but only in a simulation context not for the transfer function)

By a single click on an empty zone of the schematic, a menu will appear, and we will select the item ‘**Page SARC setup**’



We then indicate the output(s) we want to compute, here the voltage of node ‘Out’. We choose to consider the offset ‘Offs’ as a parameter by declaring it as an ‘**INTERN**’. We select to produce a transfer function.

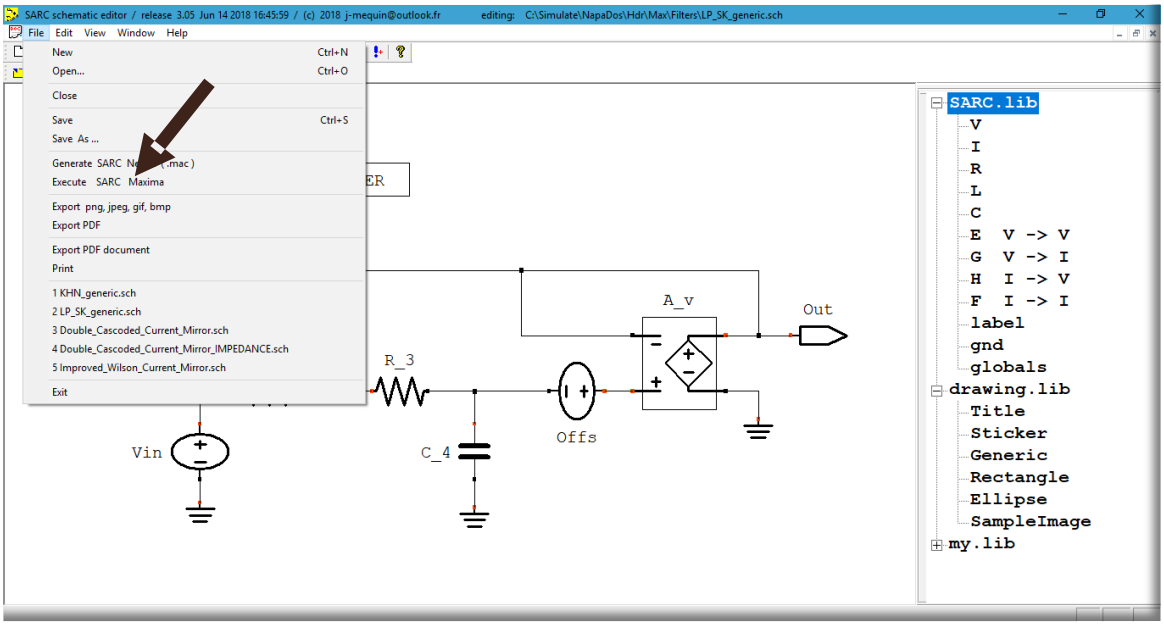
More details can be found in the ‘**Schematic Editor for SARC Card**’ and ‘**Maxima for SARC Card**’ documentations accessible from the ‘**NAPA Vineyard**’ Windows panel (represented page 1).



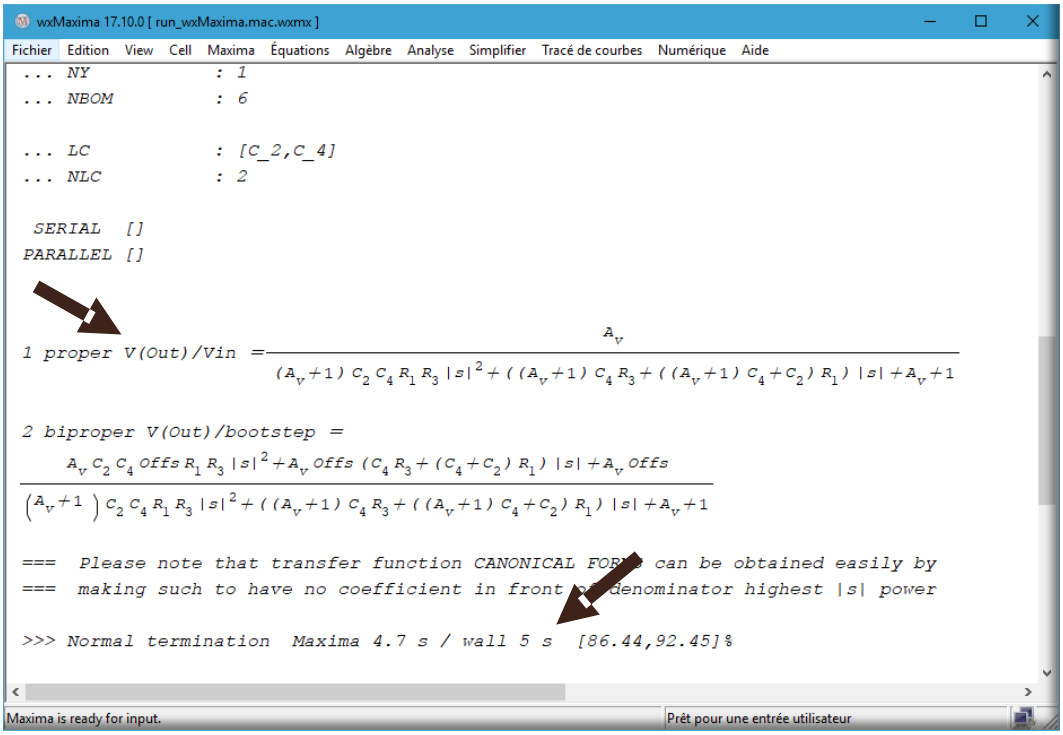
It is important to note that a simple '**Save**' is automatically producing a textual netlist (.mac) and a document (.pdf). This is a guarantee that schematic, netlist and documentation are always synchronized.

It is now time to produce the transfer function!

In the menu '**File**', we will select '**Execute SARC Maxima**'.



After a few seconds, a Maxima session starts and the transfer function automatically computed. Wait for the message ' >>> **Normal termination** '.

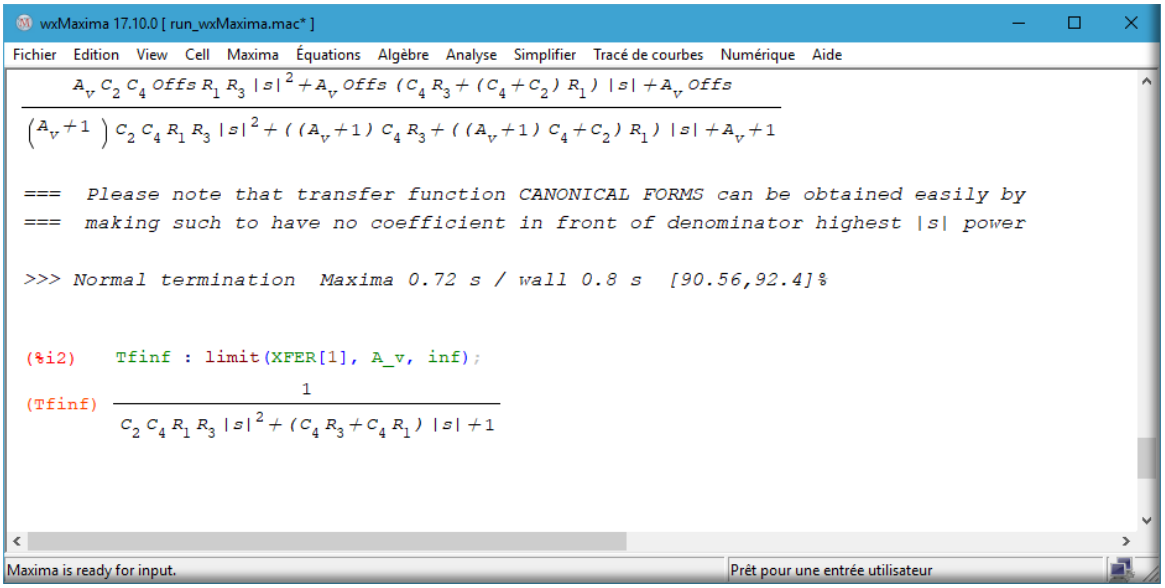


Here is the result:

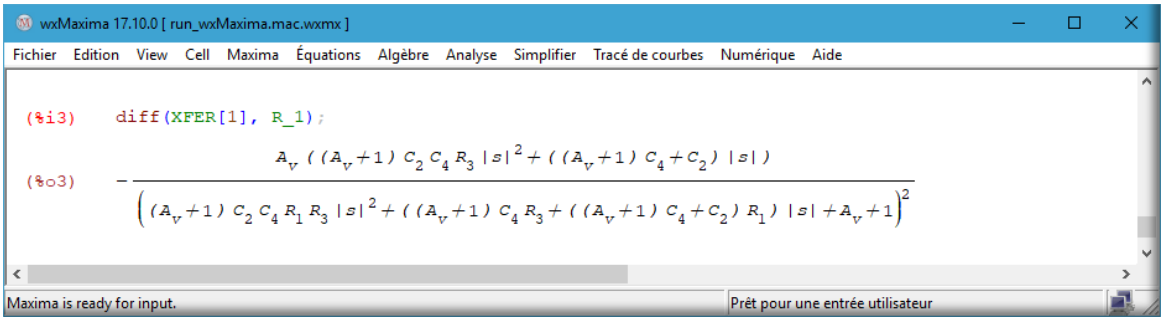
$$\mathbf{1\ proper\ V(Out)/Vin} = \frac{A_v}{(A_v+1)C_2C_4R_1R_3|s|^2 + ((A_v+1)C_4R_3 + ((A_v+1)C_4 + C_2)R_1)|s| + A_v + 1}$$

The results are stored in Maxima variables **XFER[1]** where '**1**' is the number in front of the output. The **2**nd transfer function represents the effect of the 'INTERN' input(s).

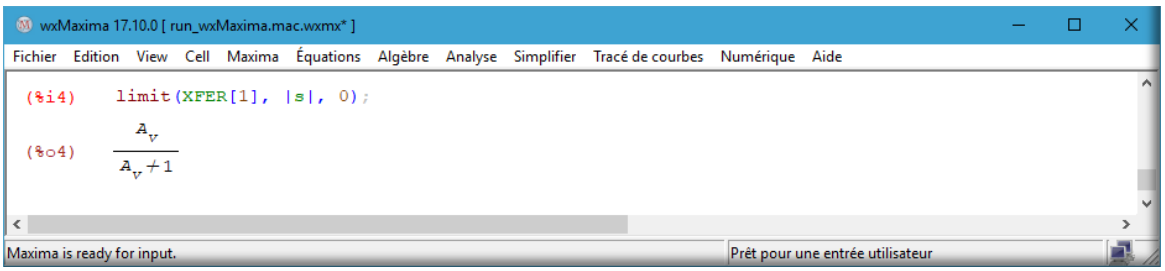
We can use Maxima to compute the transfer function for an ideal gain of the amplifier by applying a limit():



We can also use Maxima to postprocess the results to get the sensitivity of the transfer function to the value of an element, here the sensitivity to R_1:



We can also obtain the DC transfer function by applying a limit. The Laplace variable is |s|.

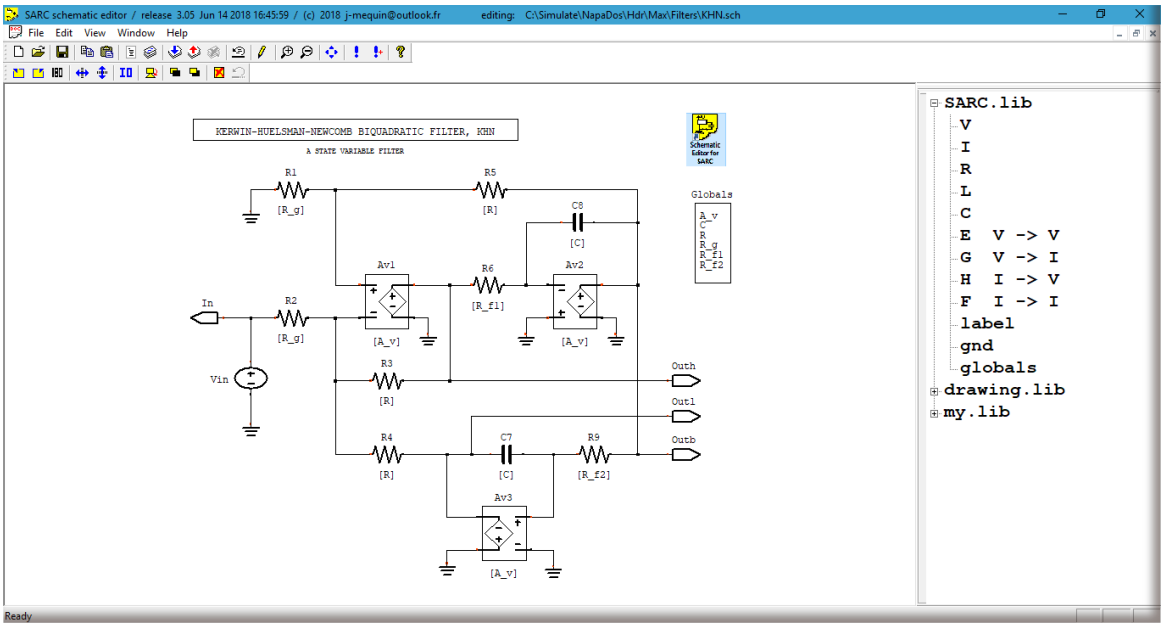


As expected, we can see that the DC gain of a Sallen & Key filter is about 1.

As it could be tedious to perform the same postprocesses again and again, we will add a few properties directly in the schematic.

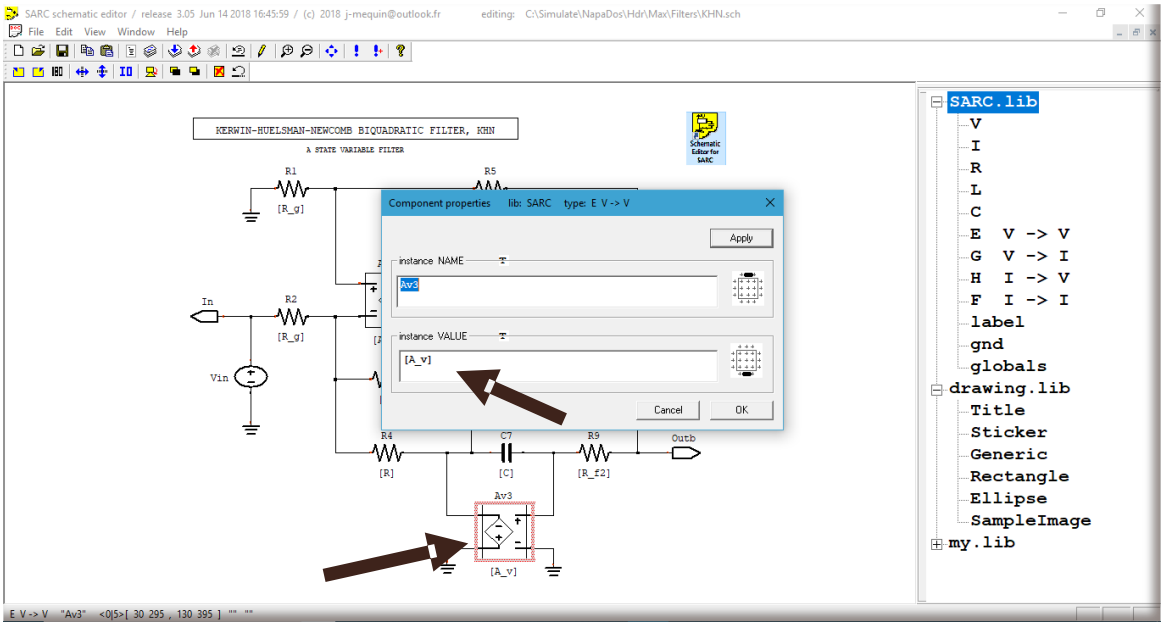
2. AN EXAMPLE OF SCHEMATIC, USING PROPERTIES

We use here the component properties to assign specific values to the elements.

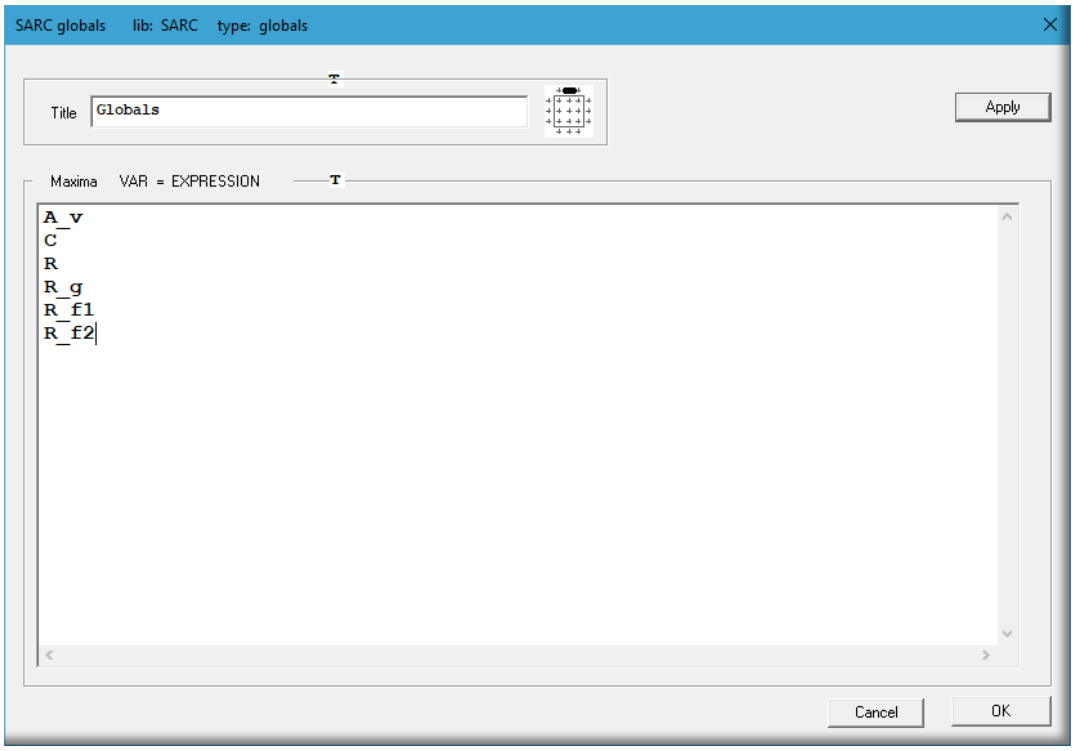


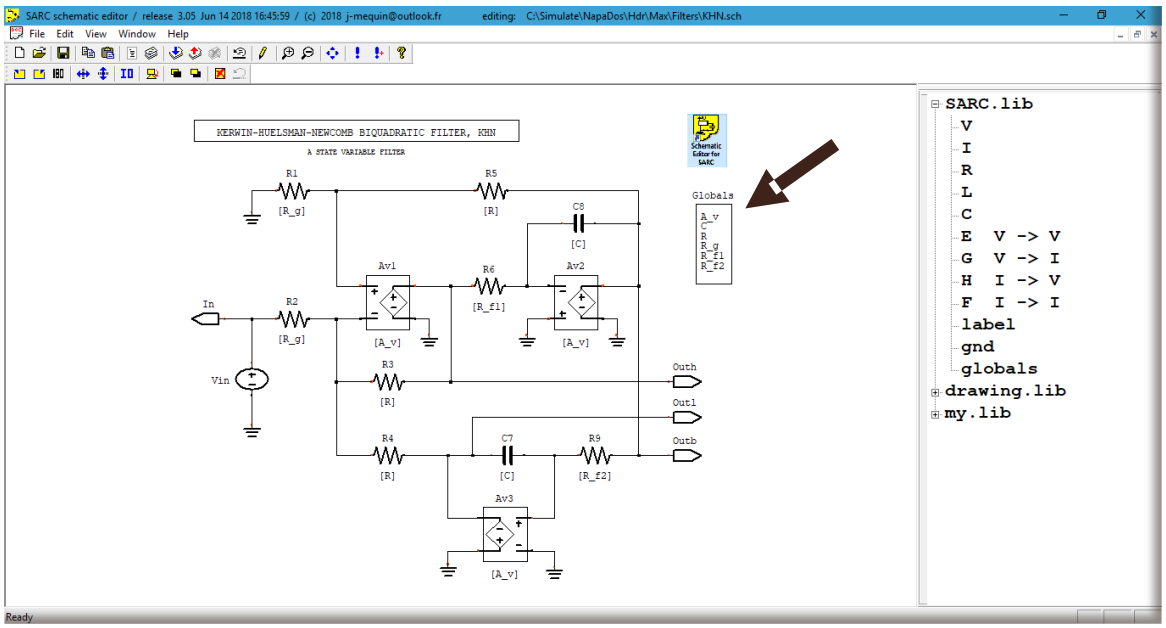
By double clicking on a component, we can fill its properties. The syntax uses brackets [ ].

We assign, for example, the same symbolic value for the gain of each amplifiers 'A_v'.

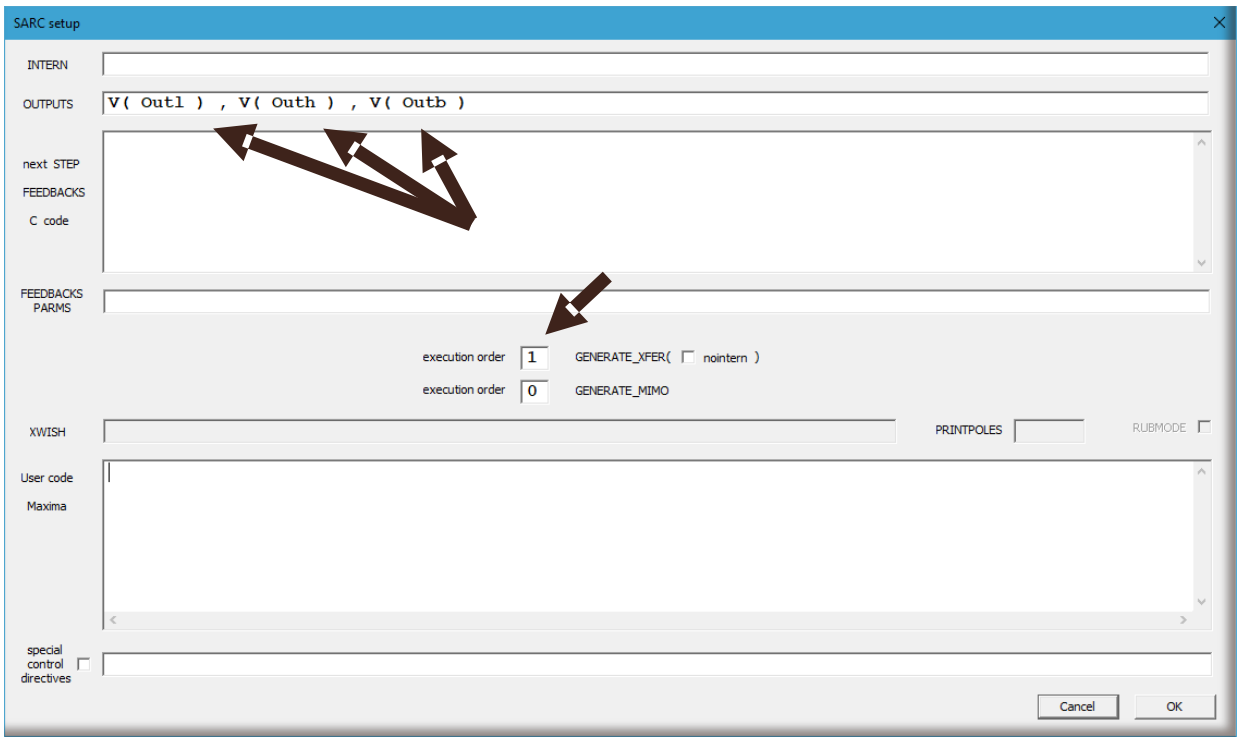


We need now to declare these new variables by selecting the component '**globals**' in the '**SARC.lib**' of menu on the right, instantiate it in the schematic, it is an empty rectangle, double click on it to access to the menu, and fill.

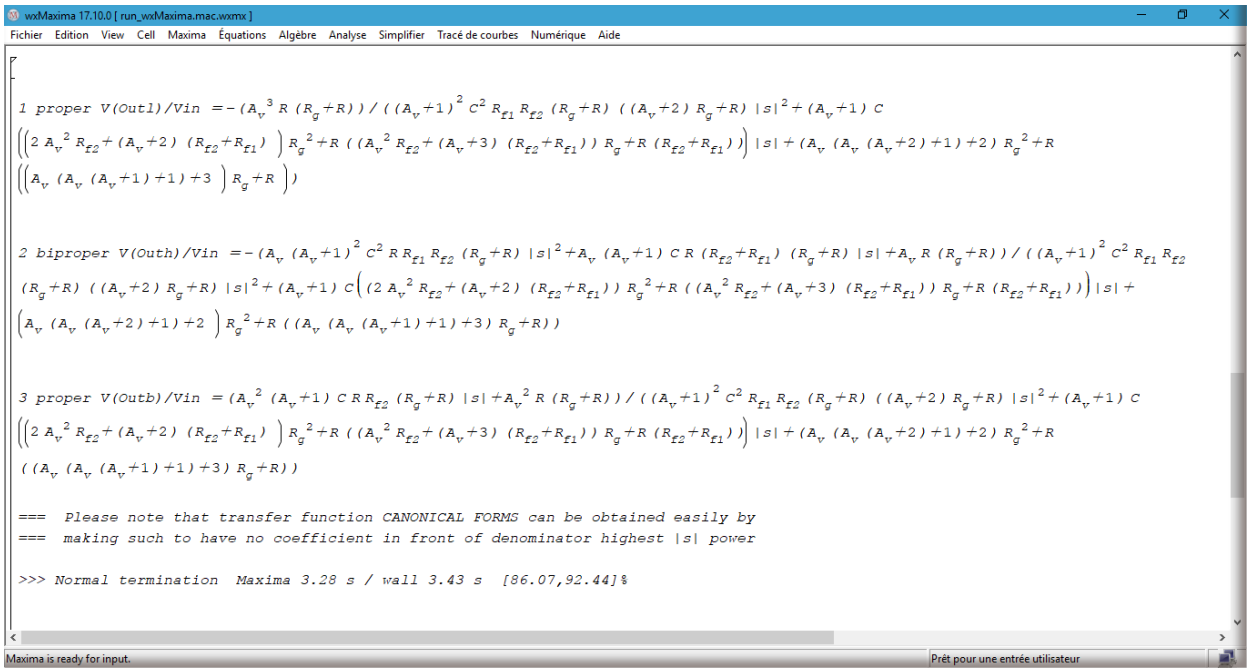




The 3 outputs are declared in the '**Page SARC setup**' and the transfer function is selected.



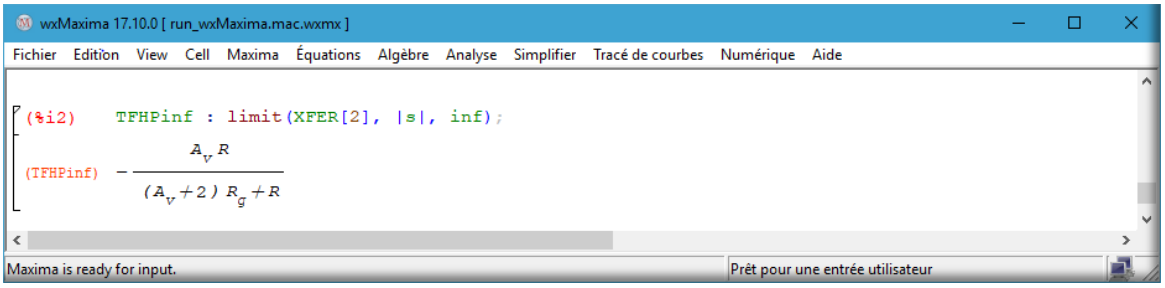
In the '**File**' menu, on the top, click on the '**Execute SARC Maxima**' to start the Maxima process.



The 3 transfer functions are produced. There is no need a postprocess to assign the values to the components. It is important to note that the complexity of the denominator is related to the complexity of the circuit.

The Maxima variables are respectively XFER[1], XFER[2], and XFER[3].

Again, we can use Maxima, to process more. Here we compute the gain at infinity of the high pass output (transfer function number 2 of the Maxima output):



It is possible to specify the postprocessing in the schematics.

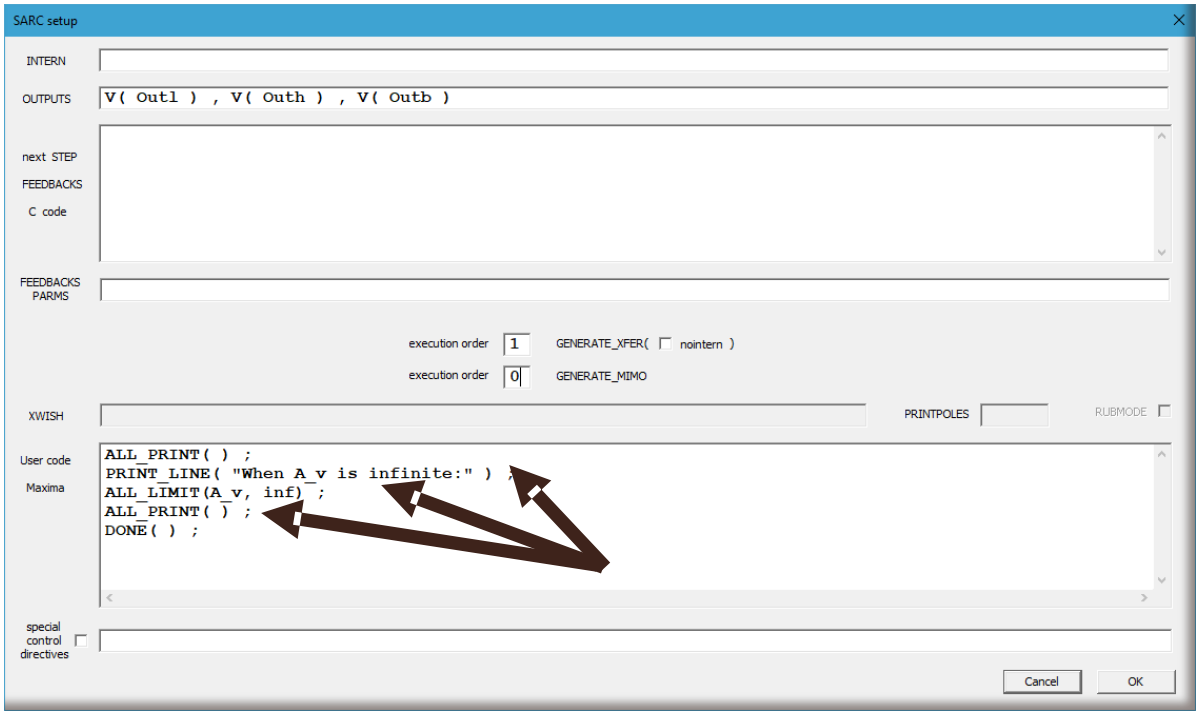
3. AN EXAMPLE OF SCHEMATIC, USING PROPERTIES AND POSTPROCESSING

In the previous schematics, we insert post processing in the schematics. Clicking in an empty space of the schematics, we access the '**SARC setup page**'. In the '**User code Maxima**' field, we can insert Maxima code.

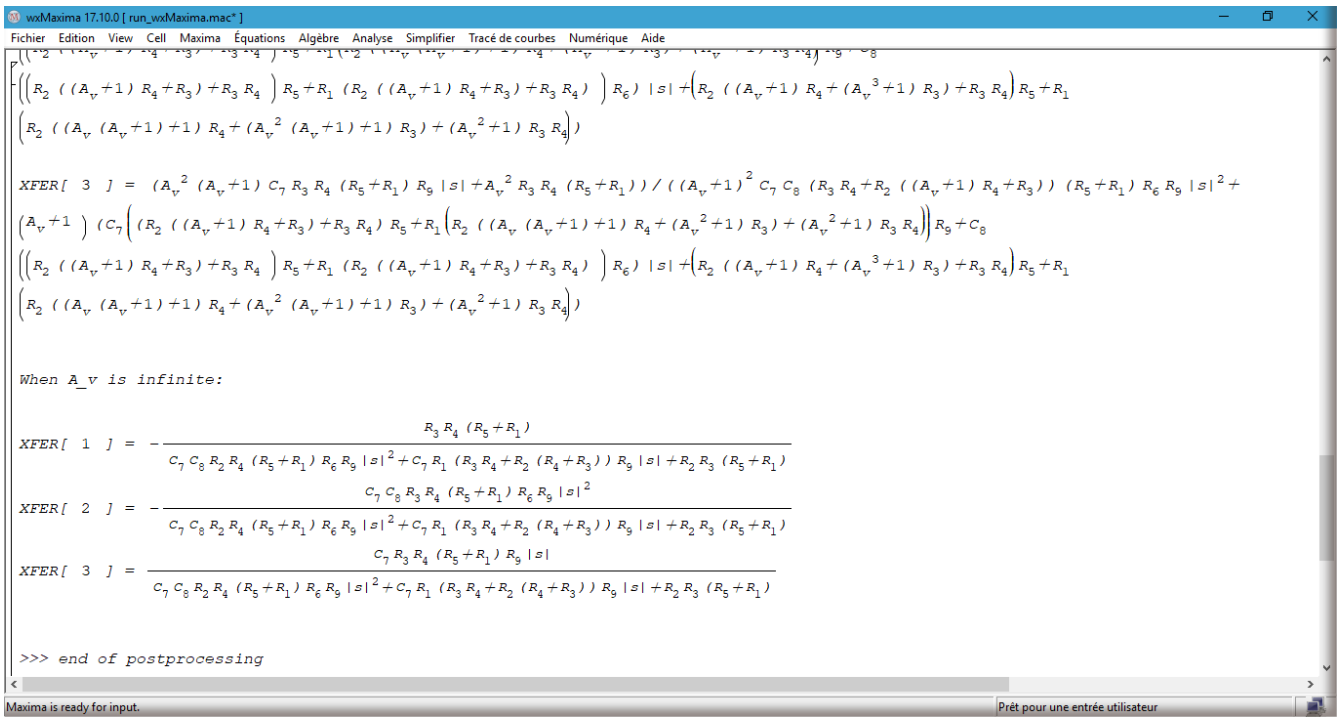
In addition to the standard Maxima functions, a few functions are currently available.
(located in "/Simulate/MaximaDos/Functions" directory)

```
Useful MAXIMA User's Functions           (used to process the transfer functions produced by the schematic editor)


PRINT_LINE(msg)           PRINT_LINE("line of text")  print the line
ALL_PRINT( )              ALL_PRINT( )                all transfer functions are printed
PRINT(id,xfer)            PRINT(IMPEDANCE,XFER[1])    return and print the transfer function "id = xfer"

Applied to all transfer functions        (modify the array of transfer functions XFER[]) :

ALL_NUMERIC( )            ALL_NUMERIC( )              insert numerical values in all transfer functions
ALL_SUBSTITUTE(parm,val)  ALL_SUBSTITUTE(gm01,10e-3)  substitute in all transfer functions
ALL_LIMIT(parm,lim)       ALL_LIMIT(gm01,inf)         apply limit in all transfer functions

Applied on a single transfer function    (does not modify the array of transfer functions XFER[]):

NUMERIC(xfer)             NUMERIC(XFER[2])            return a transfer function after inserting numerical values
SUBSTITUTE(xfer,parm,val) SUBSTITUTE(XFER[1],gds1,gn) return a transfer function after substitution
LIMIT(xfer,parm,lim)      LIMIT(XFER[1],gm01,inf)     return a transfer function after applying the limit

NUMERATOR(xfr)            NUMERATOR(XFER[1])          return the numerator of a transfer function
DENOMINATOR(xfr)          DENOMINATOR(XFER[3])        return the denominator of a transfer function

NCOEFF(xfer,n)            NCOEFF(XFER[2],2)           return the nth coefficient of the numerator of a transfer function
DCOEFF(xfer,n)            DCOEFF(XFER[1],3)           return the nth coefficient of the denominator of a transfer function

DC(xfer)                  DC(XFER[1])                 return the DC transfer function
INF(xfer)                 INF(XFER[1])                return the transfer function at infinite frequency

Useful to signal the end of the processing which could be in some cases (very) long

DONE( )                                               print a line "end of the processing"
```

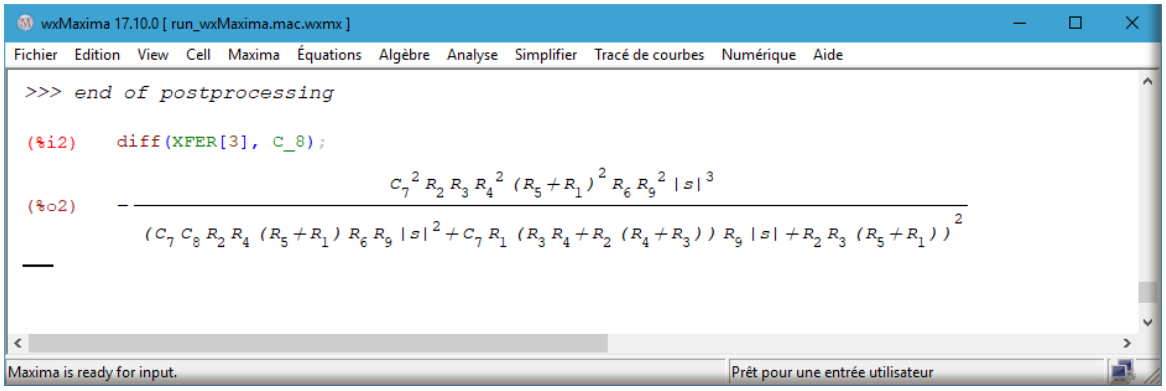
Here is the postprocessing code we have chosen; the syntax is the syntax of Maxima.



And the results are:



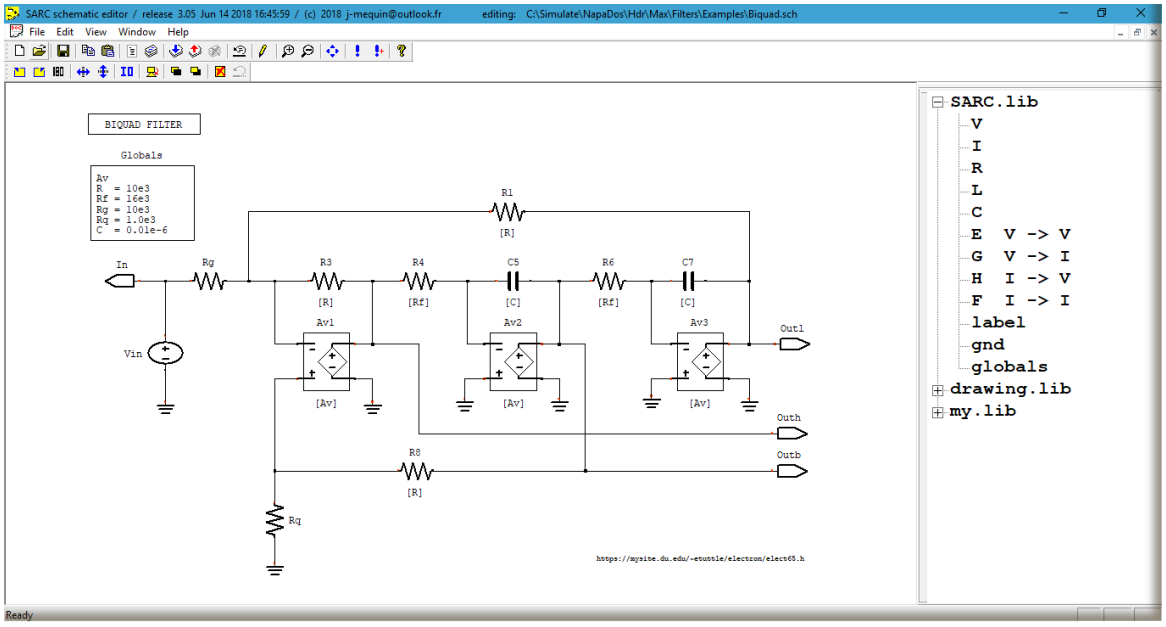
Again, we can 'play' with the solutions and use the plain symbolic capability of Maxima.



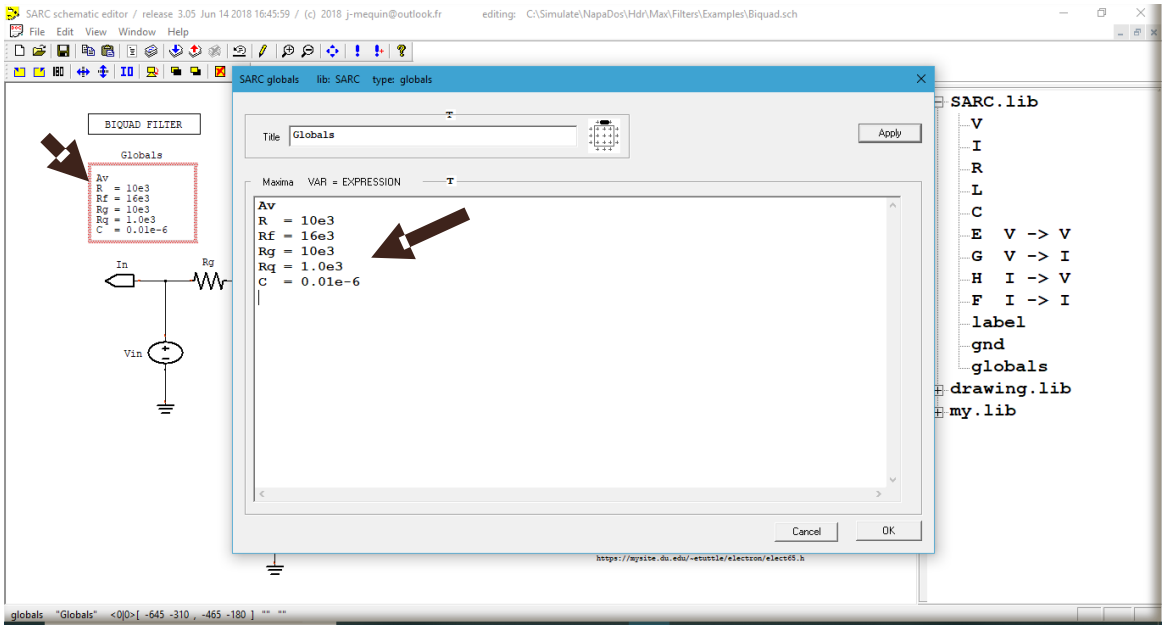
4. FROM SYMBOLIC RESULTS TO NUMBERS AND BODE PLOTS

After the symbolic analysis of transfer functions, it could be interesting to apply numerical values to evaluate the solutions.

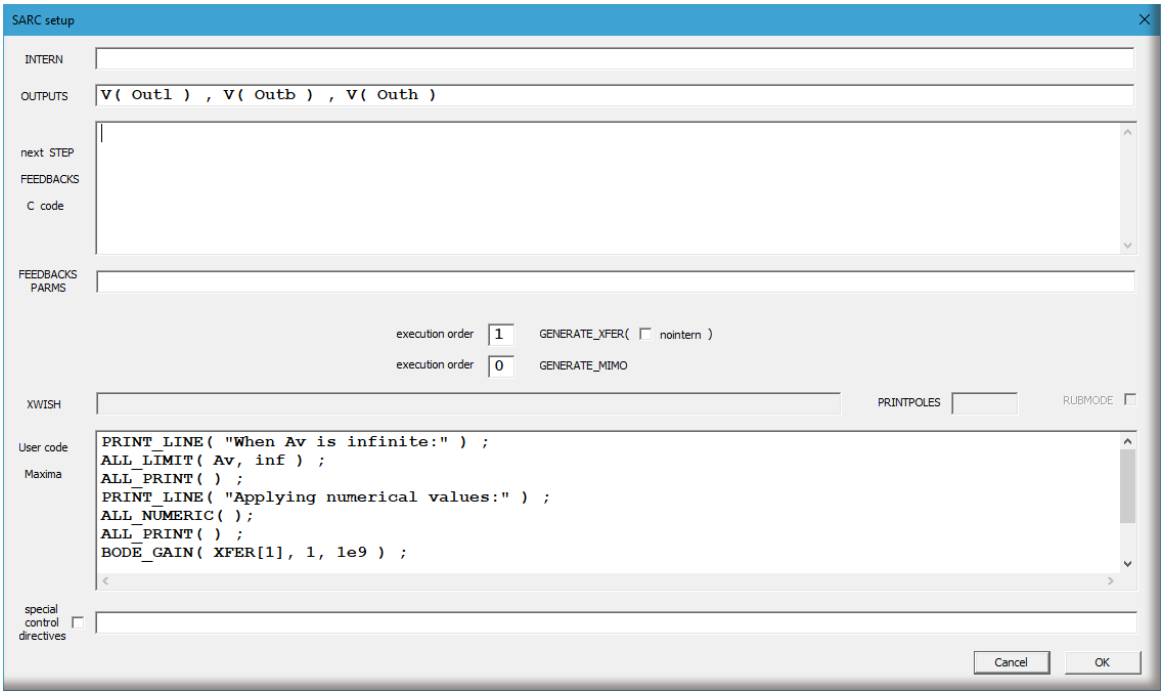
In this example, we apply symbolic values for parameters and add numerical values to the '**globals**' parameters and ask for bode plots.



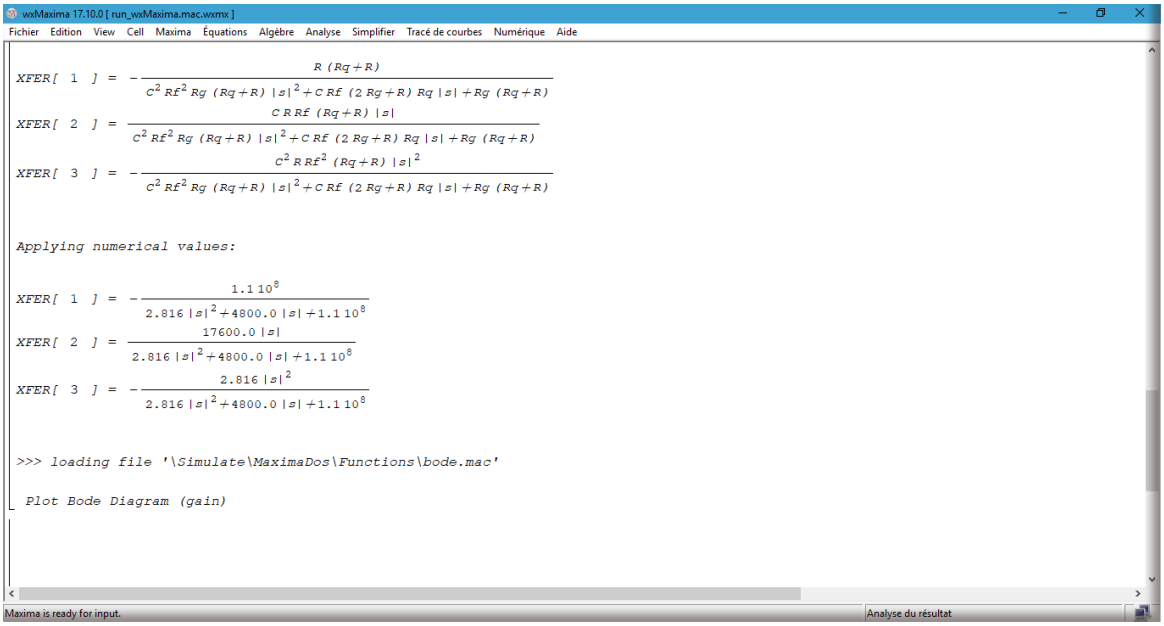
To modify the '**globals**' parameters, a double click on its instantiation and edit.



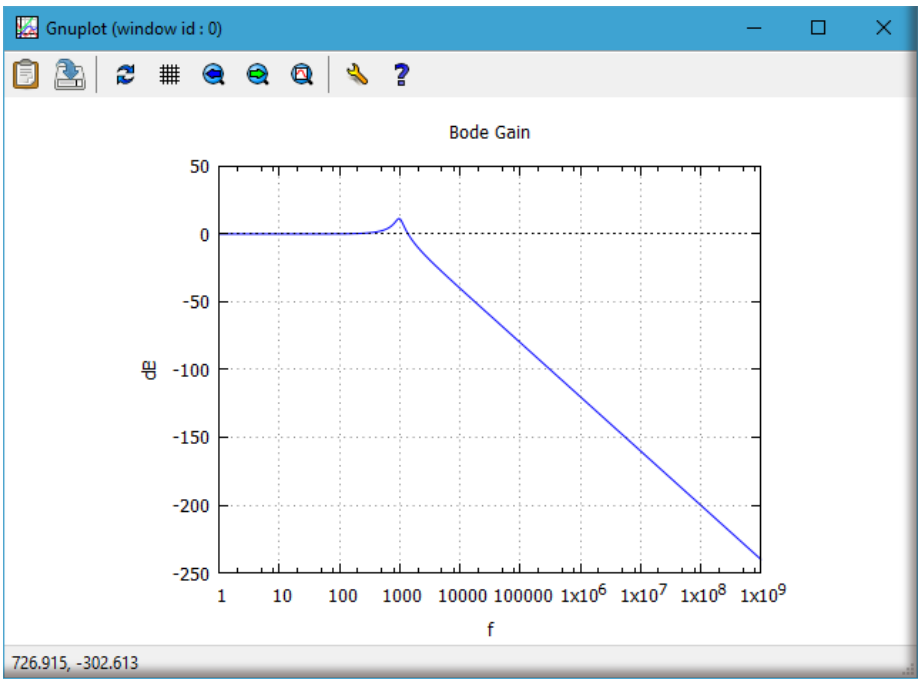
We edit the '**SARC page setup**' and add a few Maxima code to apply the numerical values and produce the Bode plot of the first transfer function.



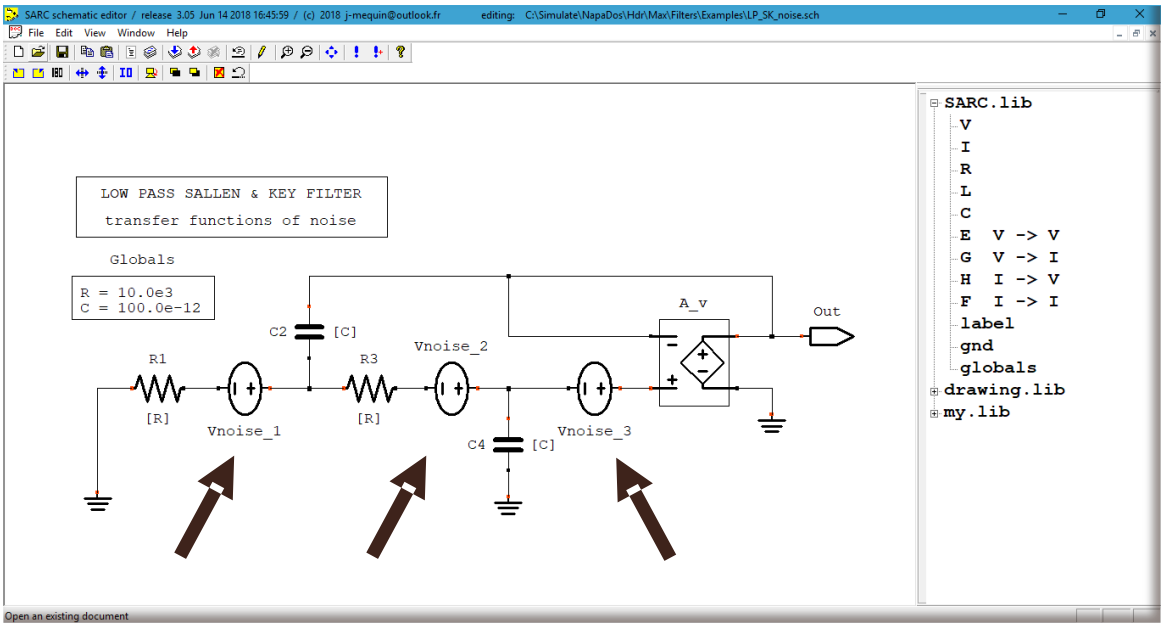
Here are the results of Maxima, the 3 transfer functions are now numerical.



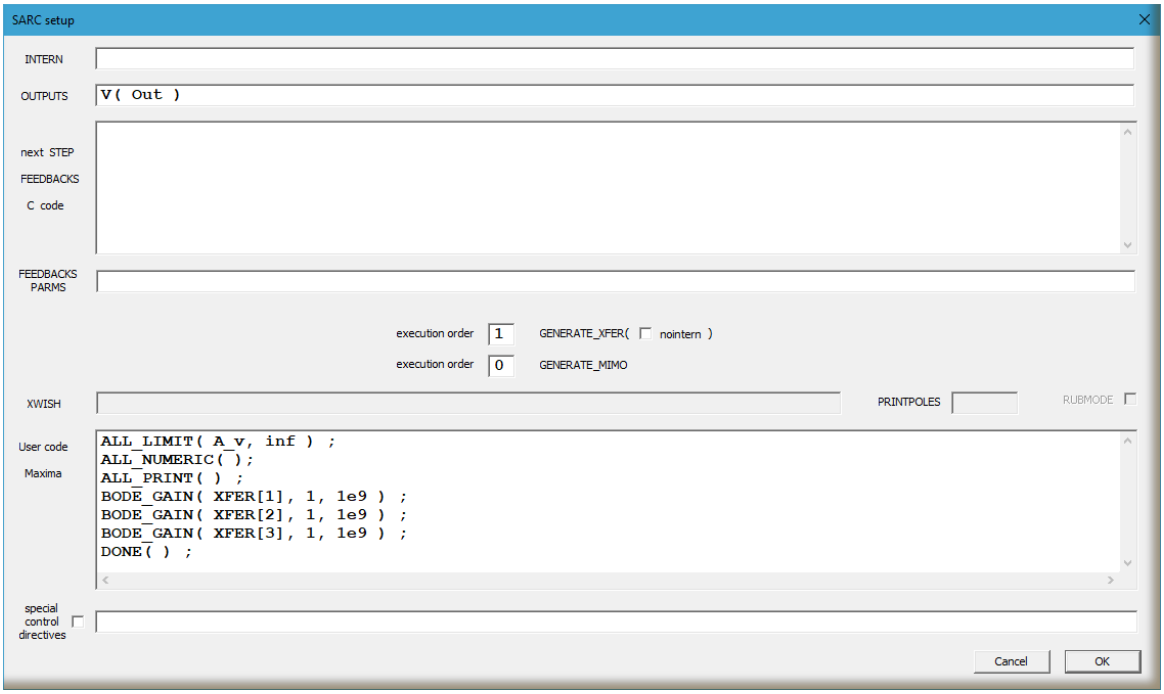
A Bode plot is then produced by Maxima:



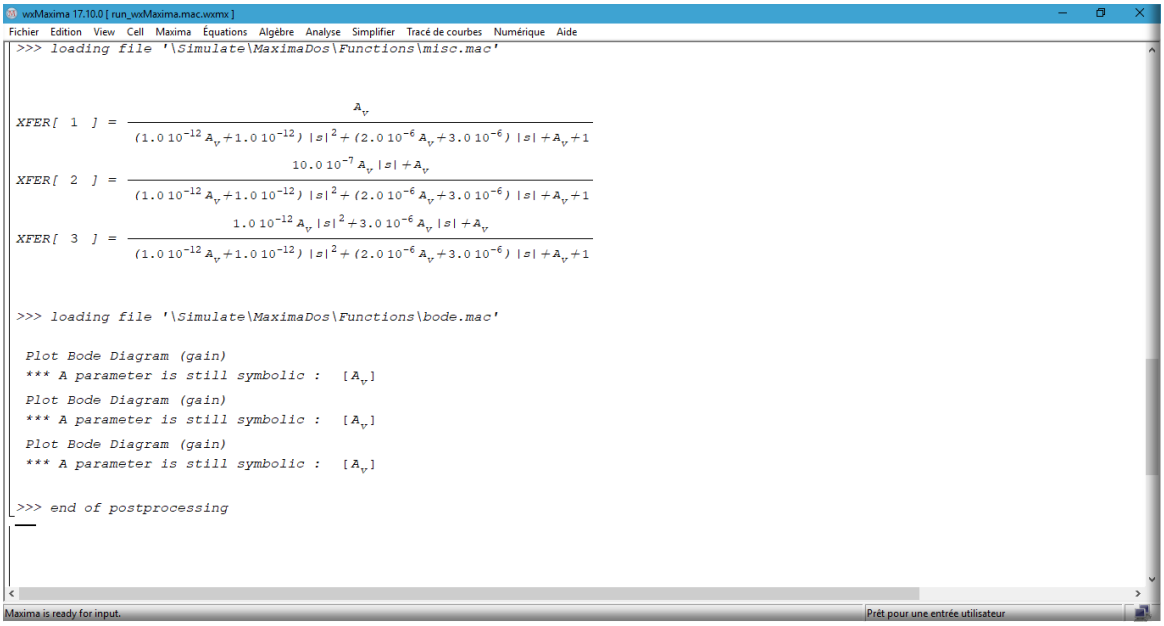
Another example produces the transfer function of noise sources, resistors and amplifier.



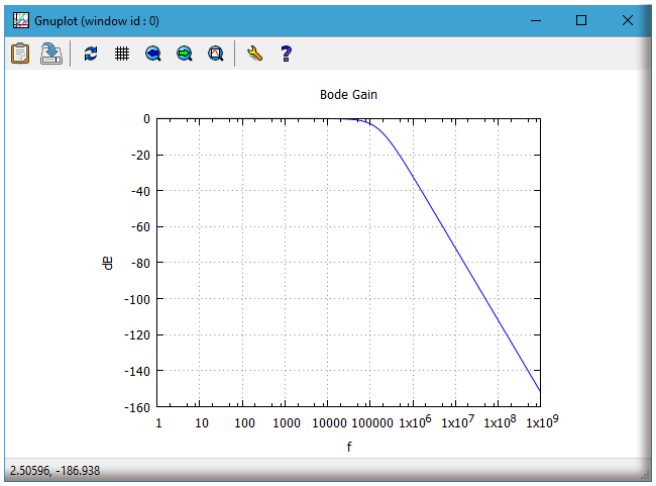
The setup for transfer functions and Bode plots when the gain of the amplifier is infinite.



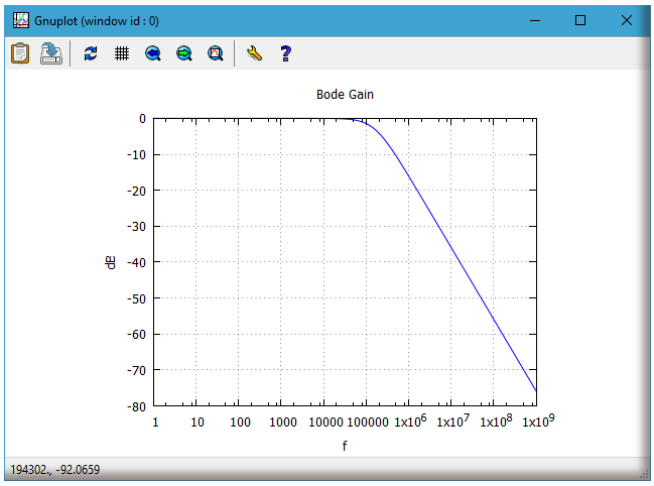
And the results:



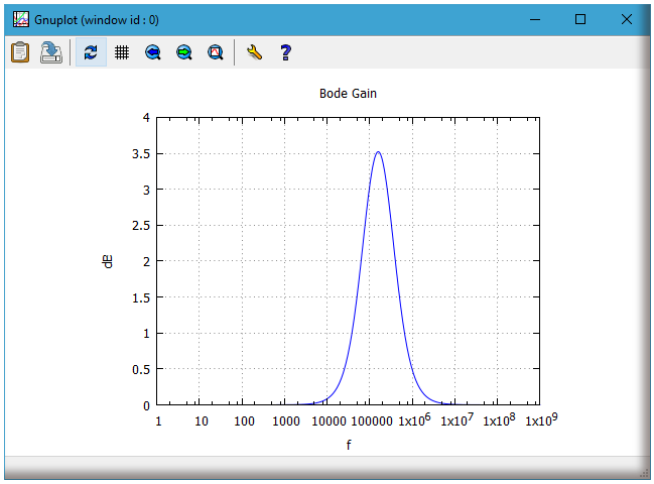
The Bode plot (noise 1→ output) :



The Bode plot (noise 2→ output) :



The Bode plot (noise 3→ output) :



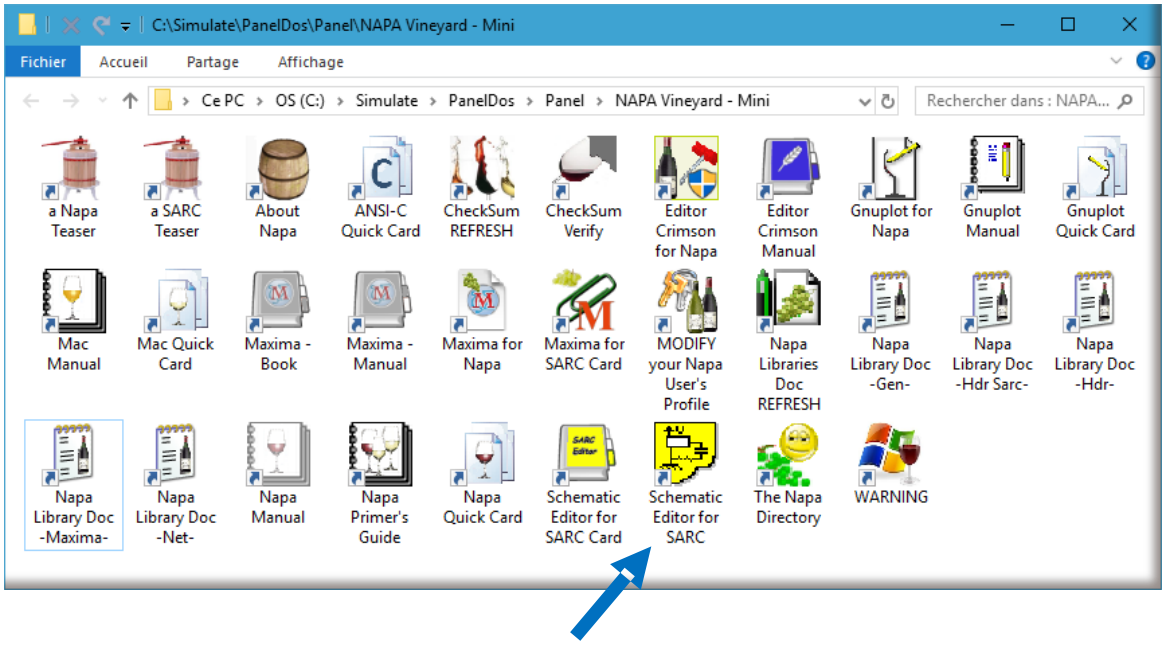
5. DO NOT BE TOO AMBITIOUS

Symbolic analysis plays with (smart) manipulations of equations of Kirchhoff.

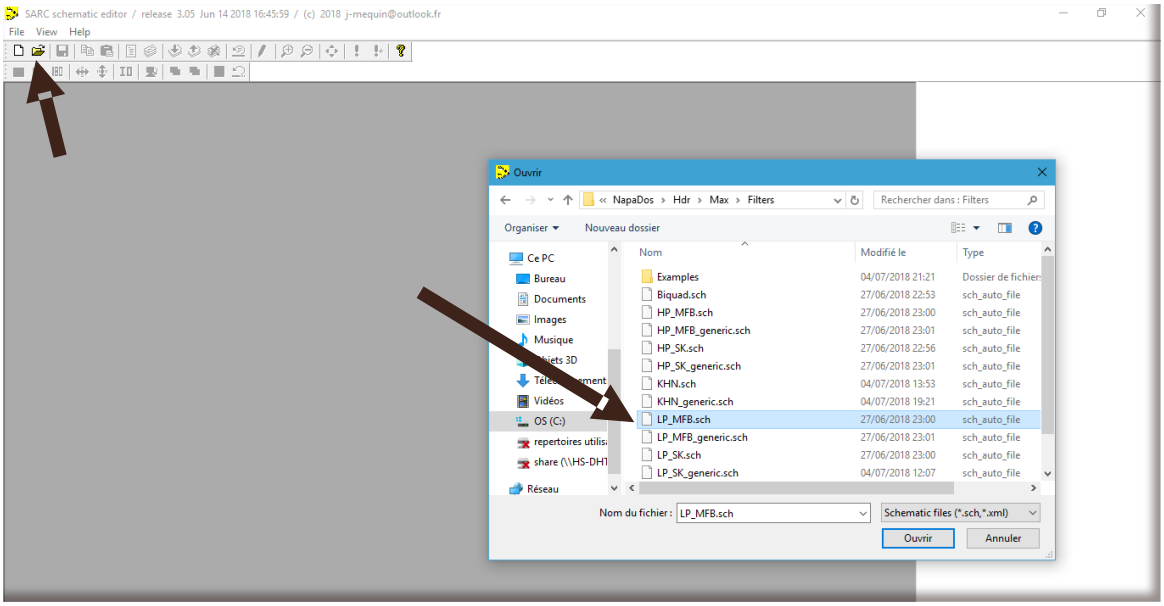
Examples are in directory '**/Simulate/Napados/Hdr/Max**'. You will notice that this software handles impressive problems. But this is a real challenge to handle too complex schematics, there is a limit in the complexity.

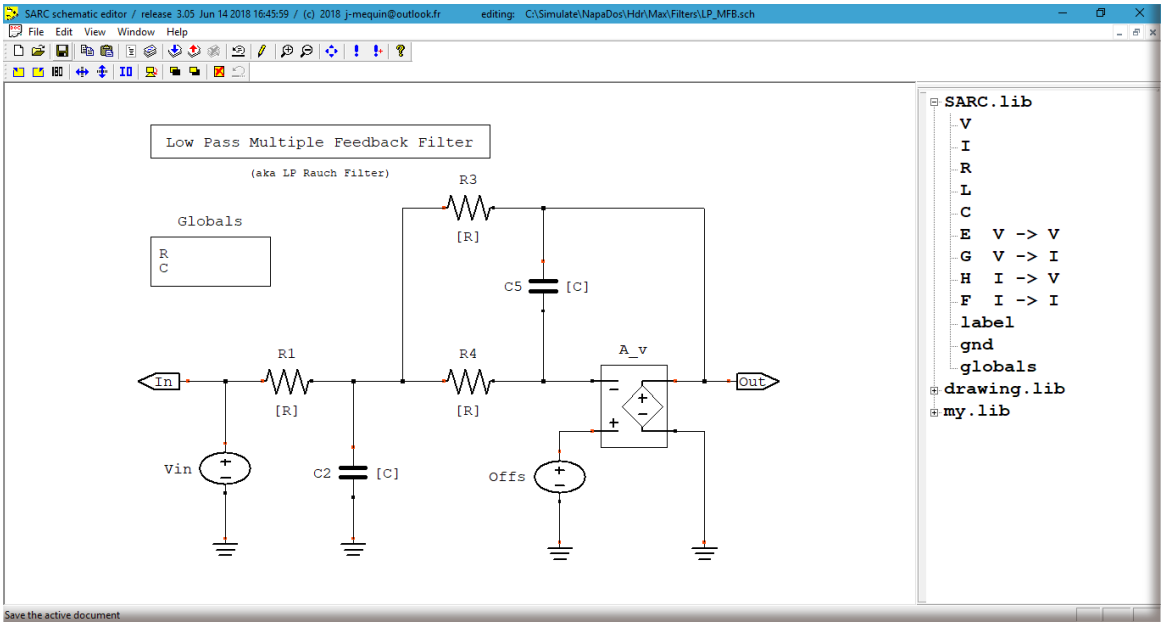
It is time to compute your first transfer function.

Open the '**Sarc Schematic Editor**' from the '**NAPA Vineyard**' panel



Open (or create) a schematic...


SARC schematic editor / release 3.05 Jun 14 2018 16:45:59 / (c) 2018 j-mequin@outlook.fr
File View Help
Ouvrir
NapaDos › Hdr › Max › Filters
Rechercher dans : Filters
Organiser
Nouveau dossier
Ce PC
Bureau
Documents
Images
Musique
Vidéos
OS (C:)
Réseau
Nom
Modifié le
Type
Examples
Biquad.sch
HP_MFB.sch
HP_MFB_generic.sch
HP_SK.sch
HP_SK_generic.sch
KHN.sch
KHN_generic.sch
LP_MFB.sch
LP_MFB_generic.sch
LP_SK.sch
LP_SK_generic.sch
Nom du fichier : LP_MFB.sch
Schematic files (*.sch,*.xml)
Ouvrir
Annuler


SARC schematic editor / release 3.05 Jun 14 2018 16:45:59 / (c) 2018 j-mequin@outlook.fr editing: C:\Simulate\NapaDos\Hdr\Max\Filters\LP_MFB.sch
File Edit View Window Help
Low Pass Multiple Feedback Filter
(aka LP Rauch Filter)
Globals
R
C
R3
[R]
C5
[C]
R1
[R]
R4
[R]
A_v
In
Out
Vin
C2
[C]
Offs
SARC.lib
V
I
R
L
C
E V -> V
G V -> I
H I -> V
F I -> I
label
gnd
globals
drawing.lib
my.lib
Save the active document

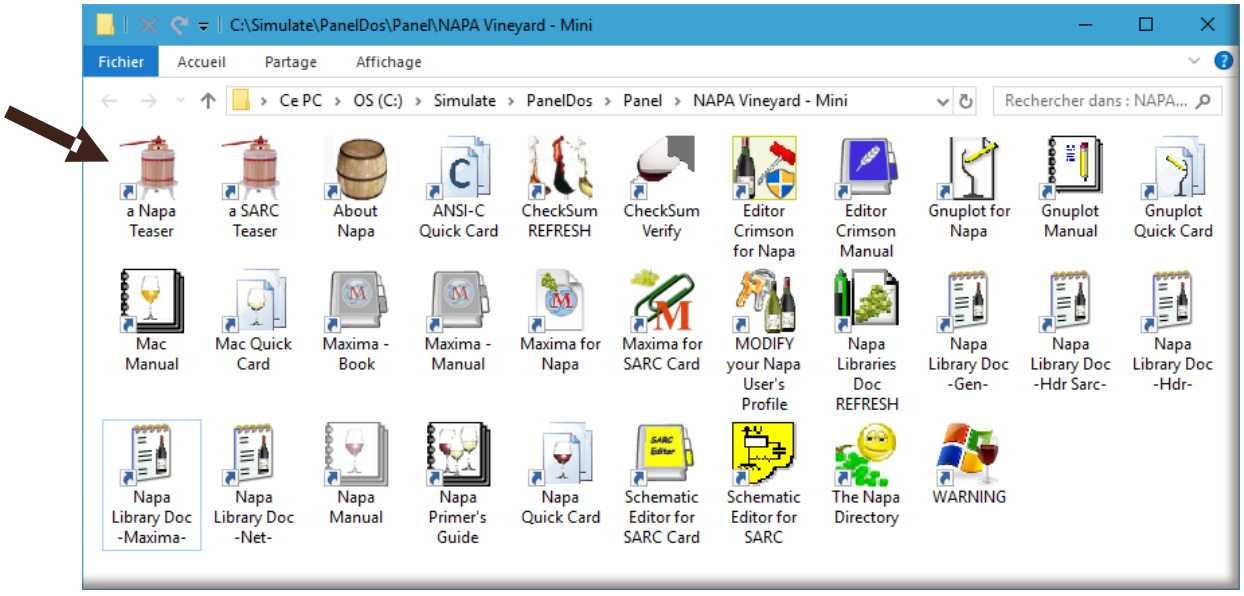
And enjoy !

6. MIXED-SIGNAL SIMULATIONS IN TIME-DOMAIN

This SARC Schematic Editor is also the entry point to insert continuous time modules in the Napa mixed-signal simulator netlist. The same schematics can produce the C code compatible to this simulator. The simulations are running in the time-domain together with sampled domain sub-circuits like switched capacitor network and/or digital circuitry.

The technology behind the simulation is SARC (Semi-Analytical Recursive Convolution) allowing to simulate linear and piece-wise linear circuits. A piece-wise linear circuit is a linear circuit with a few percussions which change on the fly the value of some elements while preserving strictly the flux of inductances and the charges of capacitances.

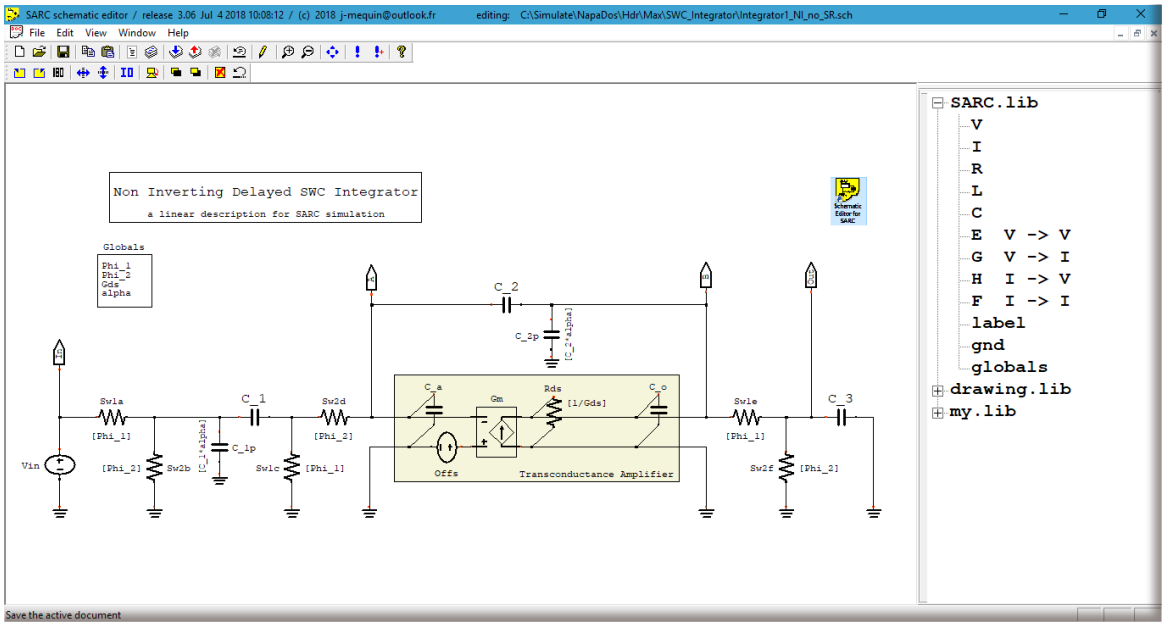
Napa is a mixed-signal simulator, a teaser is available in the '**NAPA Vineyard**' panel :



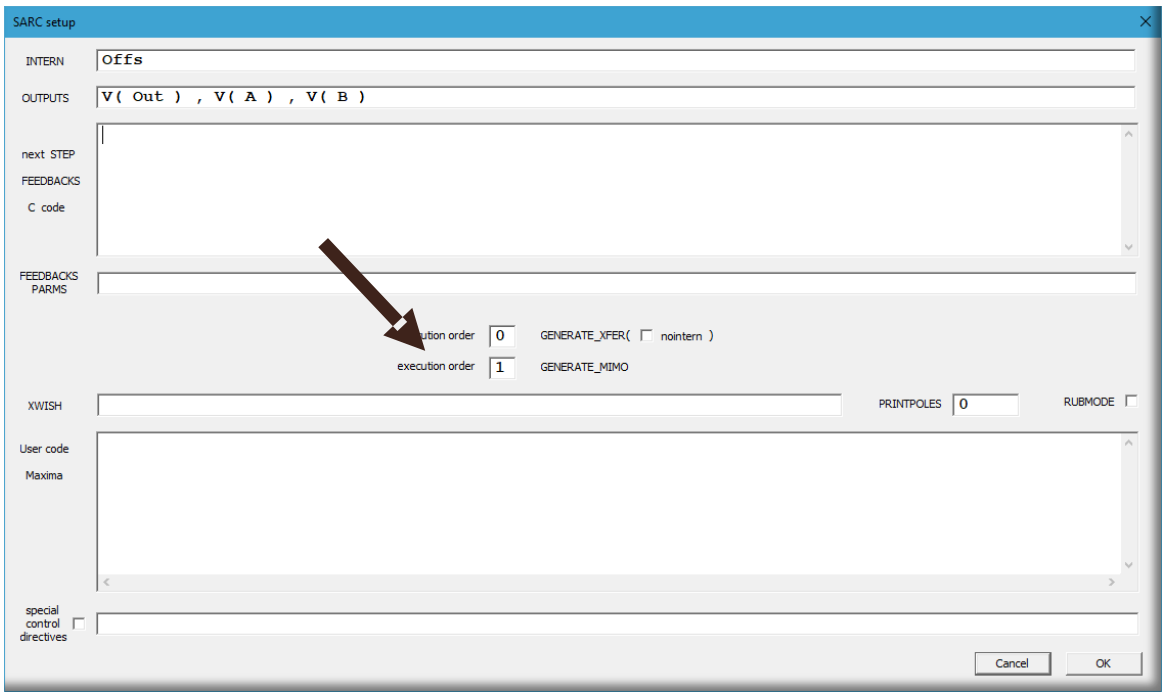
First, we will describe a non-inverting switched-capacitor integrator using the SARC Schematic Editor. All switches are described as resistors with a symbolic value 'phi1' or 'phi2'. During the simulations the value of these resistors will set to large or small. This change of value is a percussion which will be handled properly by the SARC algorithm.

Symbolic values of components and '**globals**' variables are set. The model includes parasitic capacitances ad offset. We will add slew rate later.

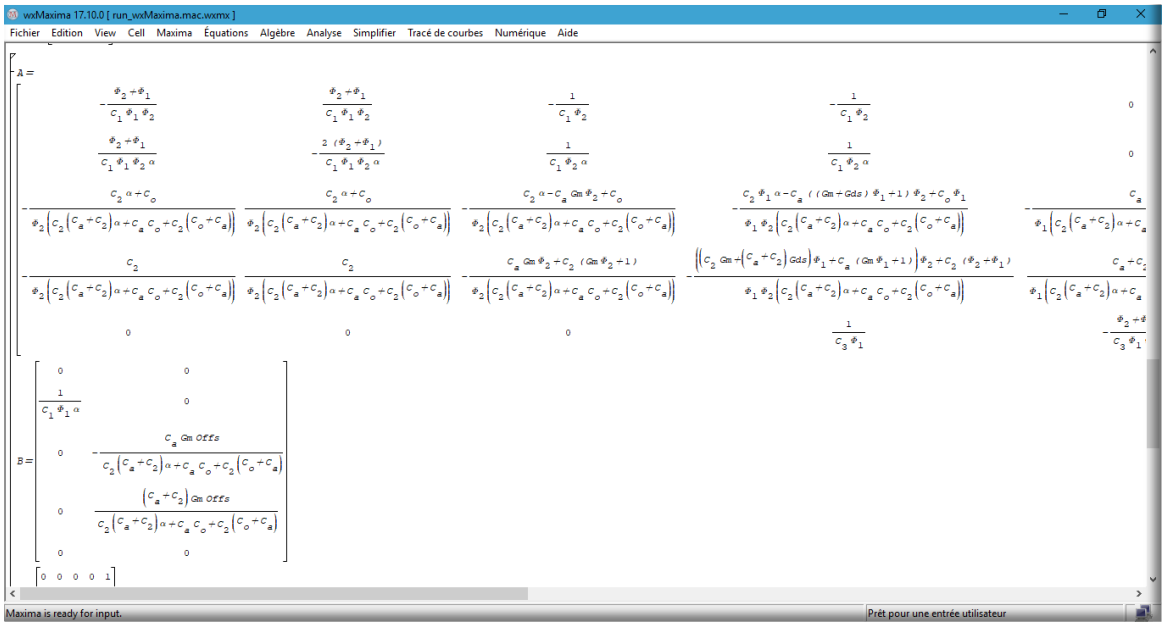
The schematic is in "/Simulate/Napados/Hdr/Max/SWC_Integrator/Integrator_NI.sch"



This schematic described a piece-wise linear circuit and there is no meaning to compute a transfer function as it has no utility at this point. We will prefer to output a C code for the SARC algorithm. We select a 'GENERATE_MIMO' instead of a 'GENERATE_XFER'.

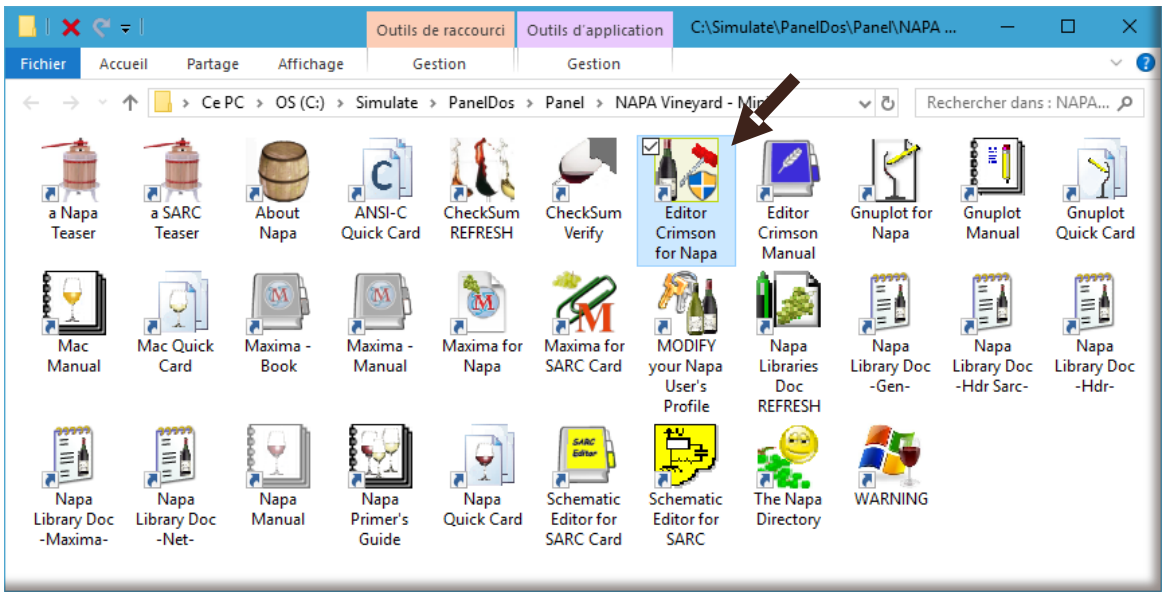


And we start Maxima from the '**File**' menu on the top of the SARC Editor. Maxima outputs the matrices A, B, C, and D for the SARC editor.



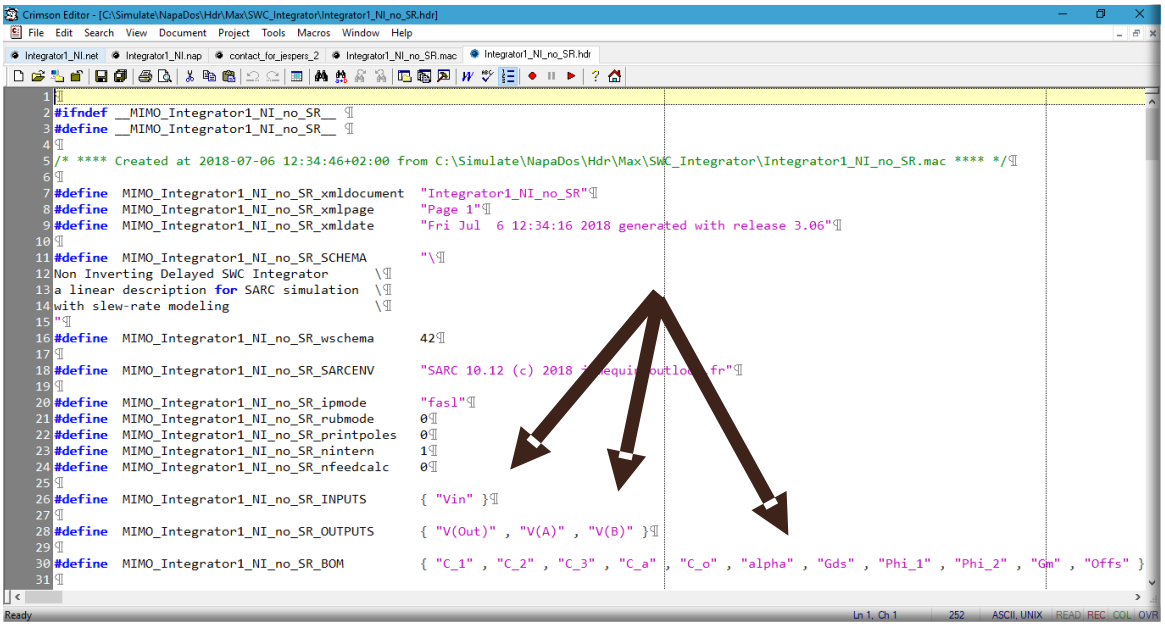
Maxima produces also a text file, an ANSI-C header file ('.hdr') compatible with the Napa simulator. The list of inputs, outputs and parameters of this model will be used to create a Napa cell.

Open the Crimson Editor from ten '**Napa Vineyard**' panel:



We use here files contained in the package. The explanations concerning the Napa simulator can be found in the corresponding teaser. We will explain briefly and focus to start quickly a Napa simulation.

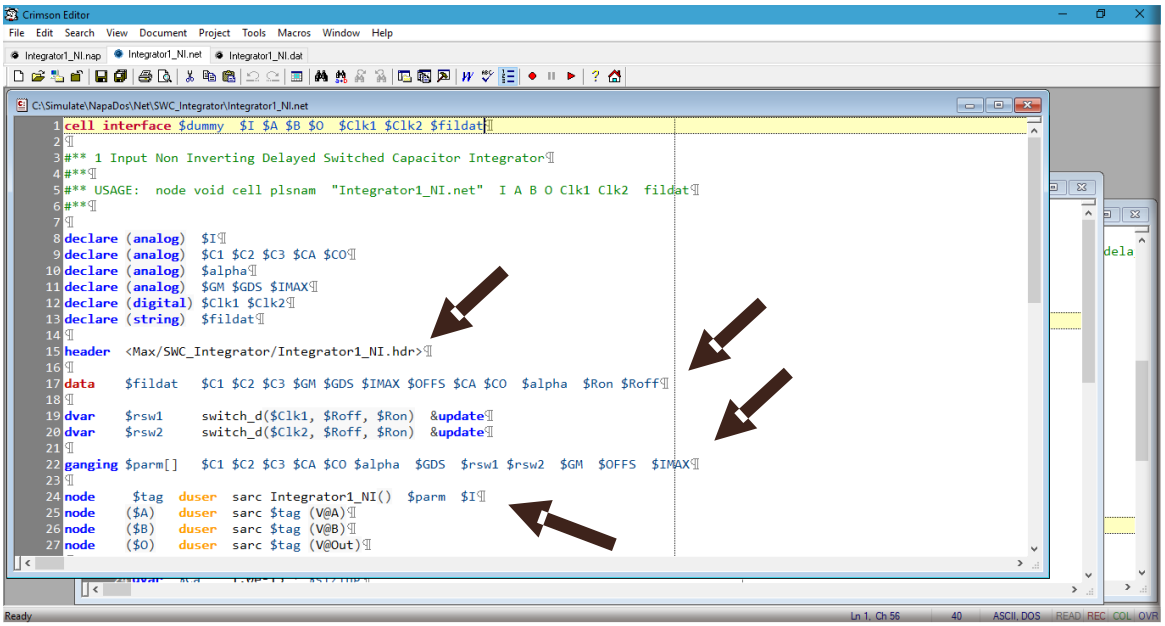
The header produced by Maxima '/Simulate/Napados/Hdr/Max/SWC_Integrator/Integrator_NI.hdr'



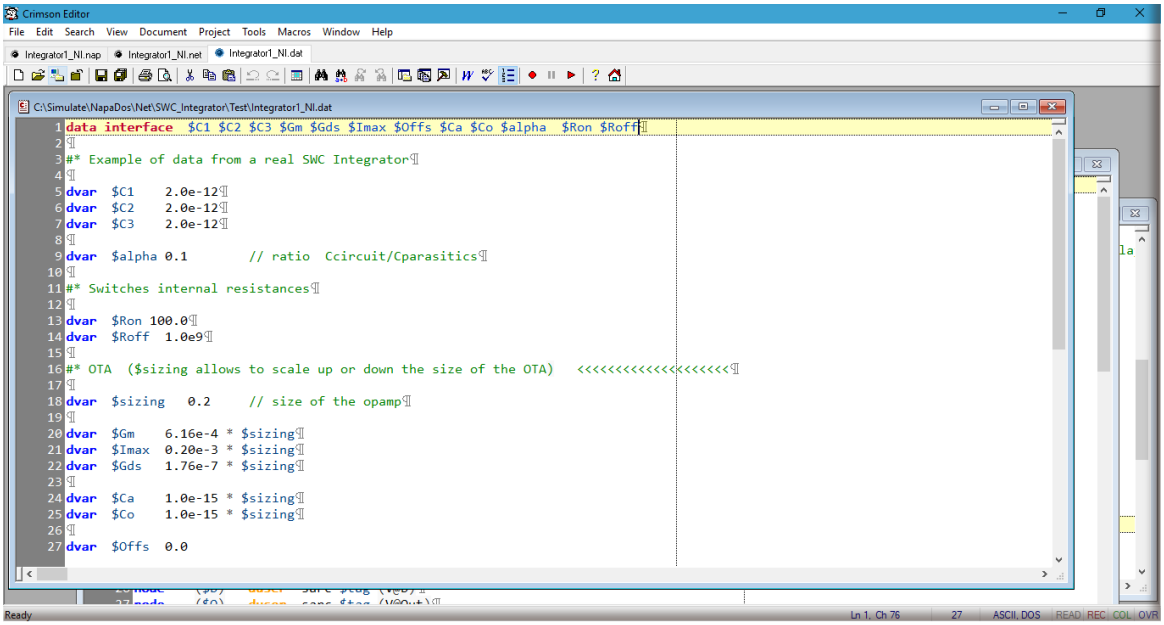
This model has 1 input, 3 outputs and 11 parameters. This file is in the directory containing the corresponding schematic ('.sch'), netlist ('.mac') and pdf files.

We prepare now a cell containing the call of the code produced by Maxima ('.net'), a data file containing the numerical values of the components ('.dat') and finally the file containing the simulation '(.nap)'.

The cell file '/Simulate/Napados/Net/SWC_Integrator/Integrator_NI.net'



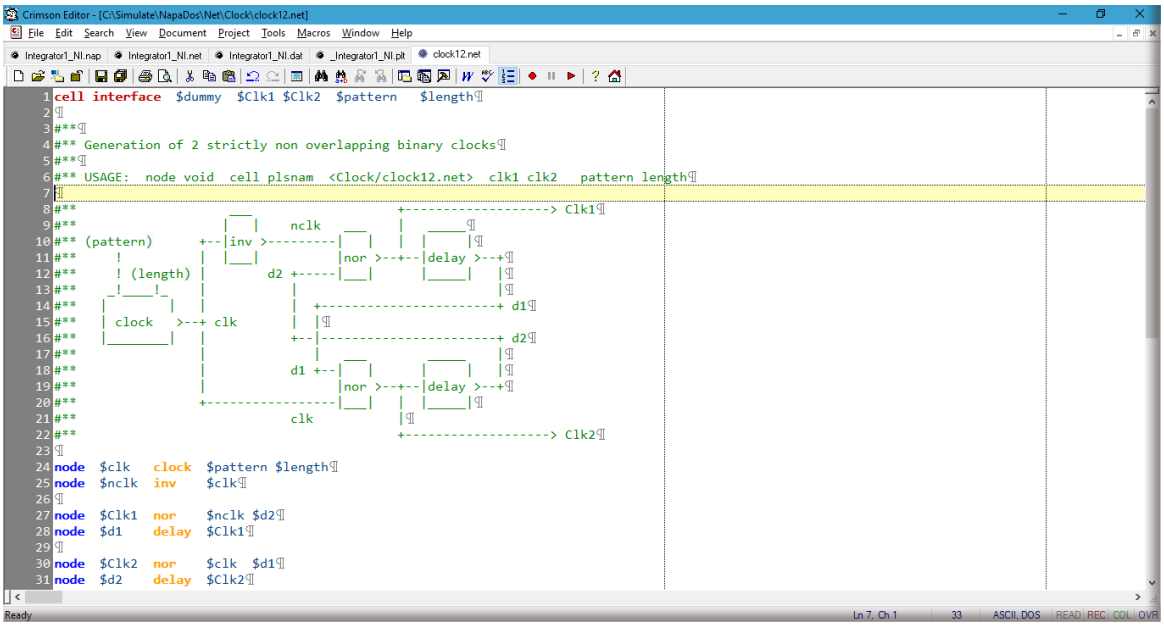
A second file is used to assign values to the components and is instantiated inside the cell. This data cell file is '/Simulate/Napados/Net/SWC_Integrator/Test/Integrator_NI.dat'



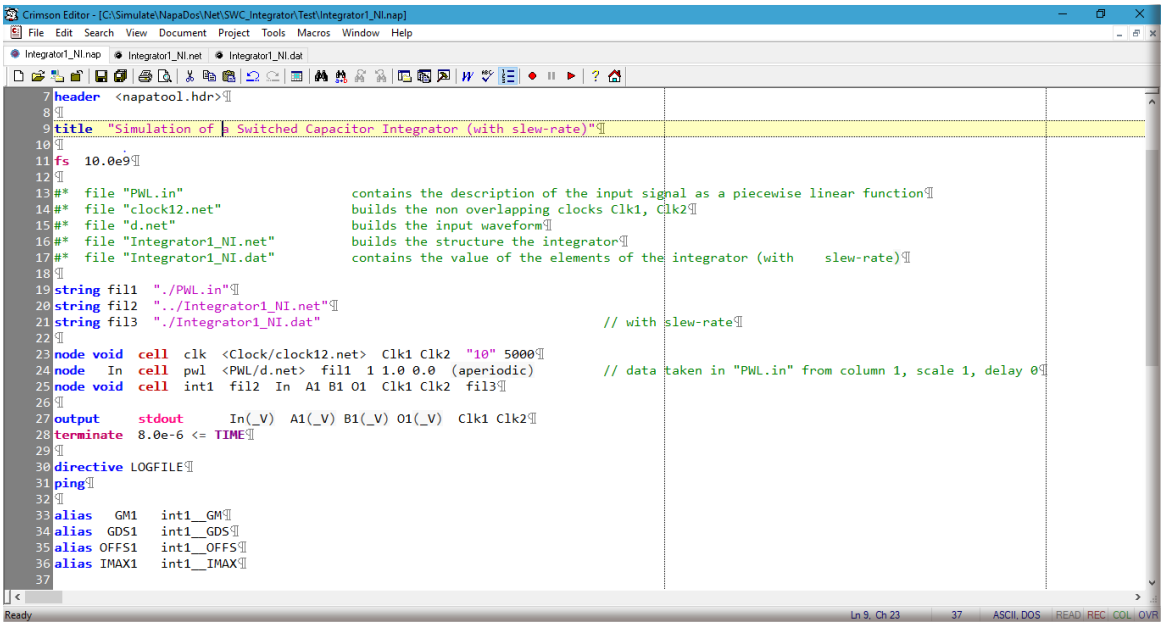
The last file is the netlist of the simulation.
The simulation file is '/Simulate/Napados/Net/SWC_Integrator/Test/Integrator_NI.nap'

The file contains the instantiation of a non-overlapped clock generator, the cell describing the integrator, a 'pwl' cell describing the input, and the location of the data file containing the parameters. A change in clocks percutes the value of the switches resistances.

The instantiated pwl cell is '/Simulate/Napados/Net/Clock/clock12.net'



The simulation file:

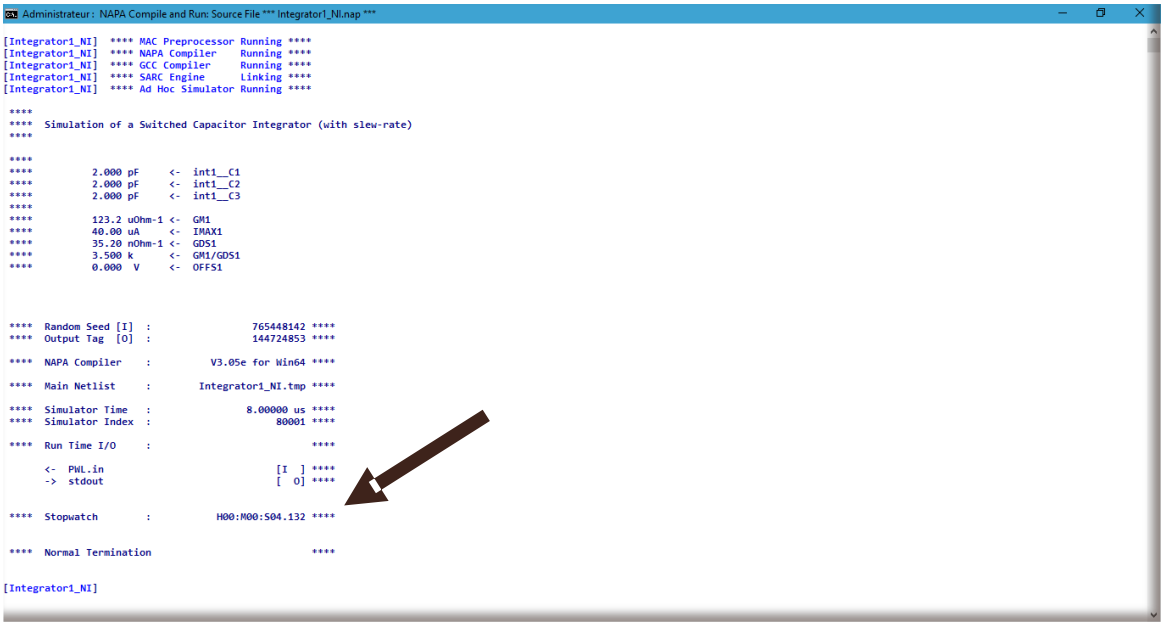


To start the simulation, from the file in the editor, press **<left alt> <R>**. The simulation flow is then automatic.

- The '**Mac preprocessor**' processes the simulation file and creates '/Simulate/Napados/Net/SWC_Integrator/Test/Integrator_NI.temp'.
- From this file, the '**Napa Compiler**' produces an ANSI-C file representing the simulator to run.
  The ANSI-C file is '/Simulate/Napados/Net/SWC_Integrator/Test/Integrator_NI.c'.
- This file is compiled by the '**mingw gcc Compiler**'.

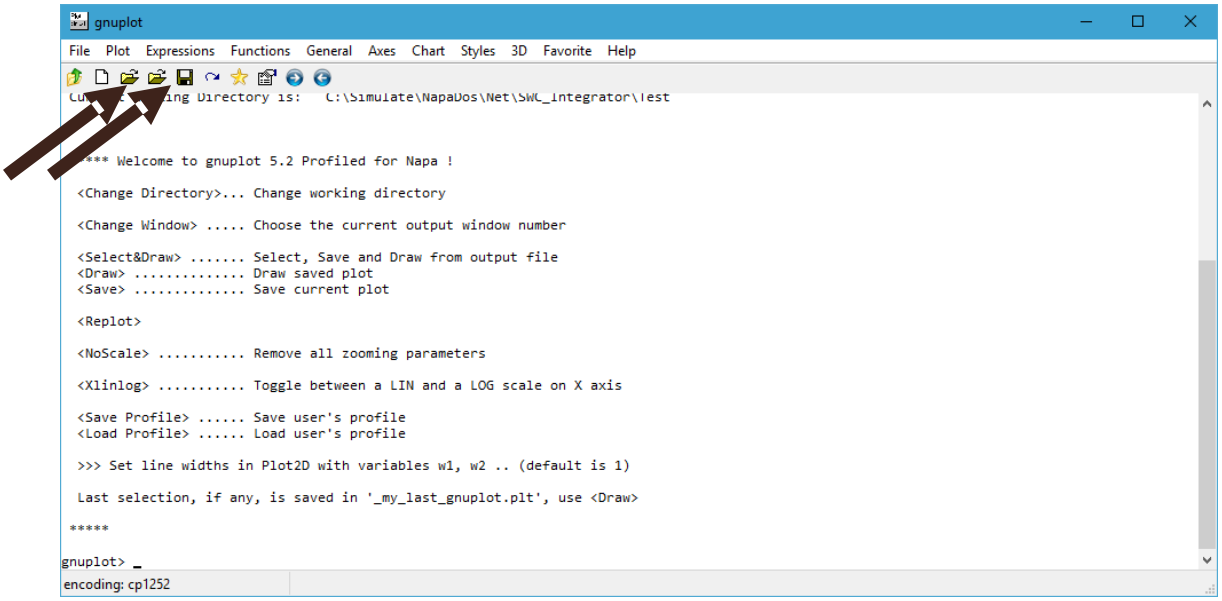
The executable is '/Simulate/Napados/Net/SWC_Integrator/Test/Integrator_NI.exe'.

Output on screen:



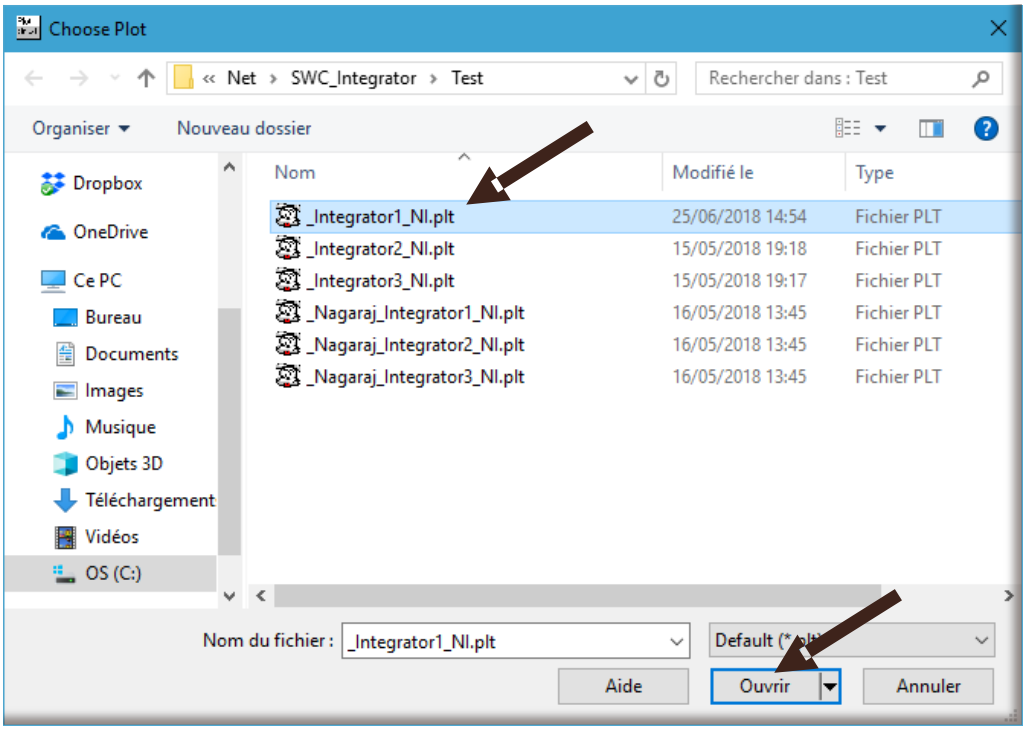
This **executable** produces here the simulation outputs in less than 4.2 seconds.

Now, from the file in the editor, press **<left alt> <G>** to call '**gnuplot**'.

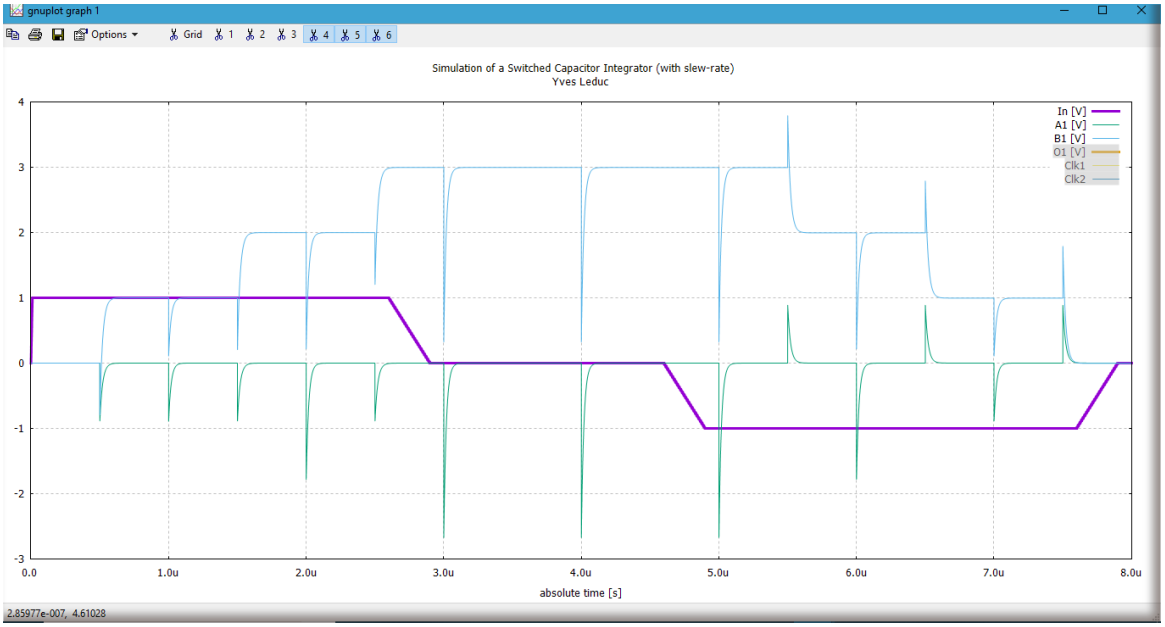


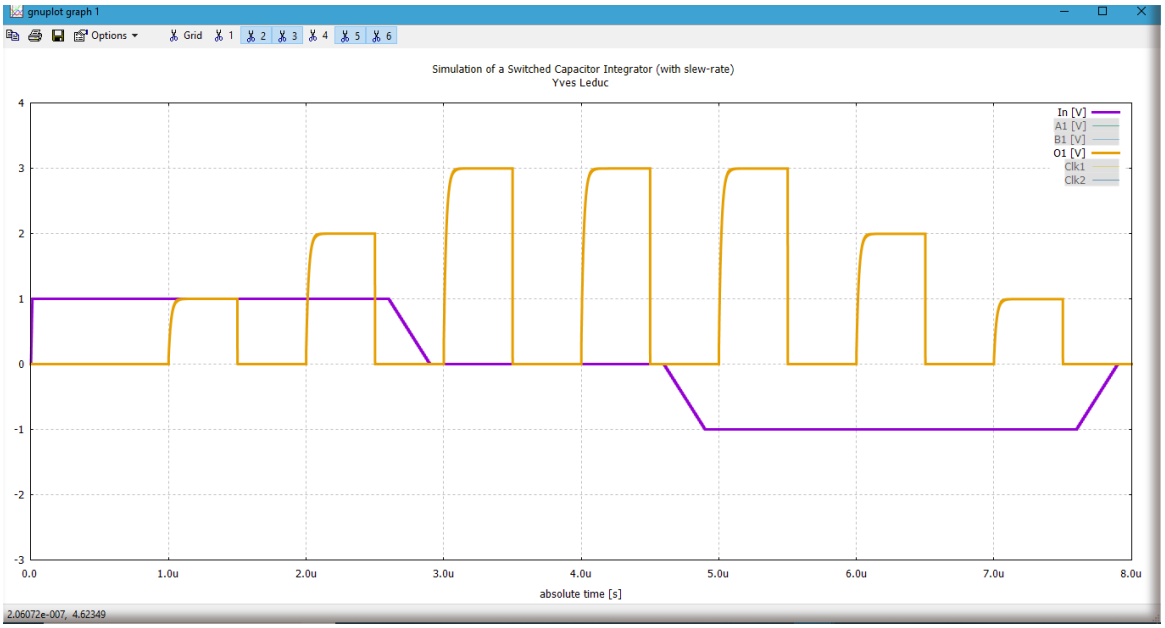
It is possible to configure the plot (icon pointed by the left arrow) or use a preexisting configuration (right arrow). The configuration was made available for this exercise.

This file is '/Simulate/Napados/Net/SWC_Integrator/Test/Integrator_NI.plt'



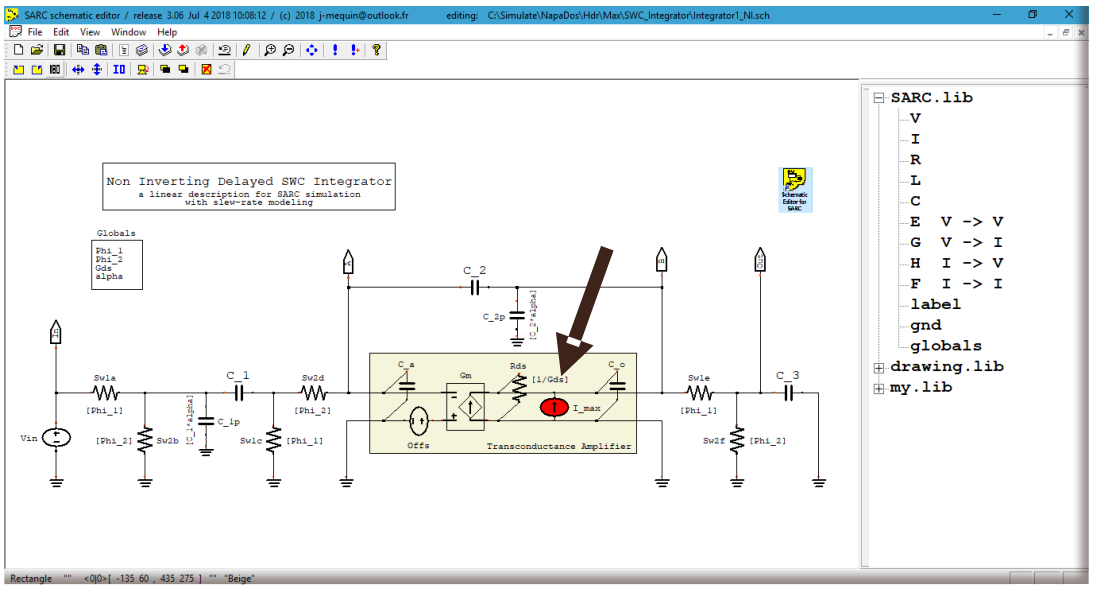
And '**gnuplot**' produces the graphic, 2 views of this output:



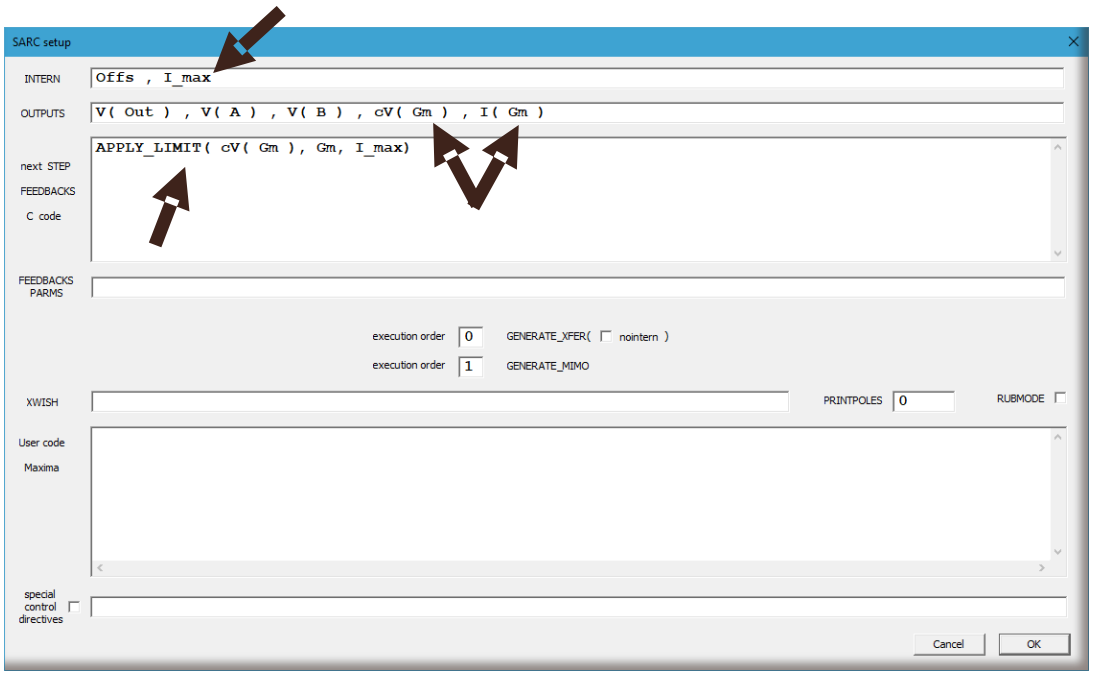


No slew-rate? We will add the appropriate model.

A current source is added in parallel to the transconductance.

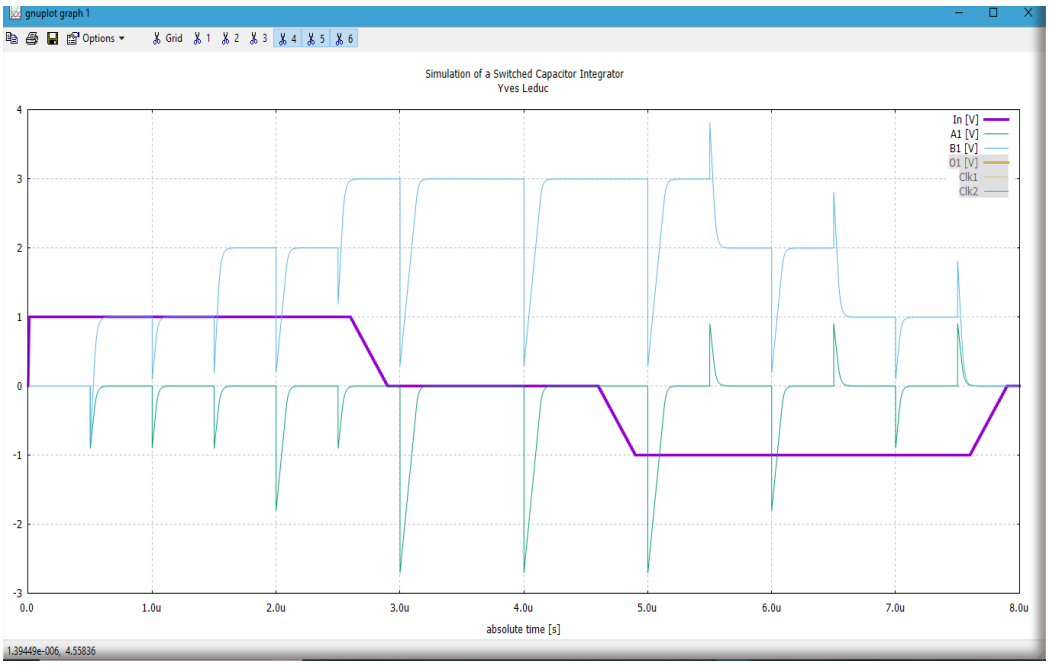


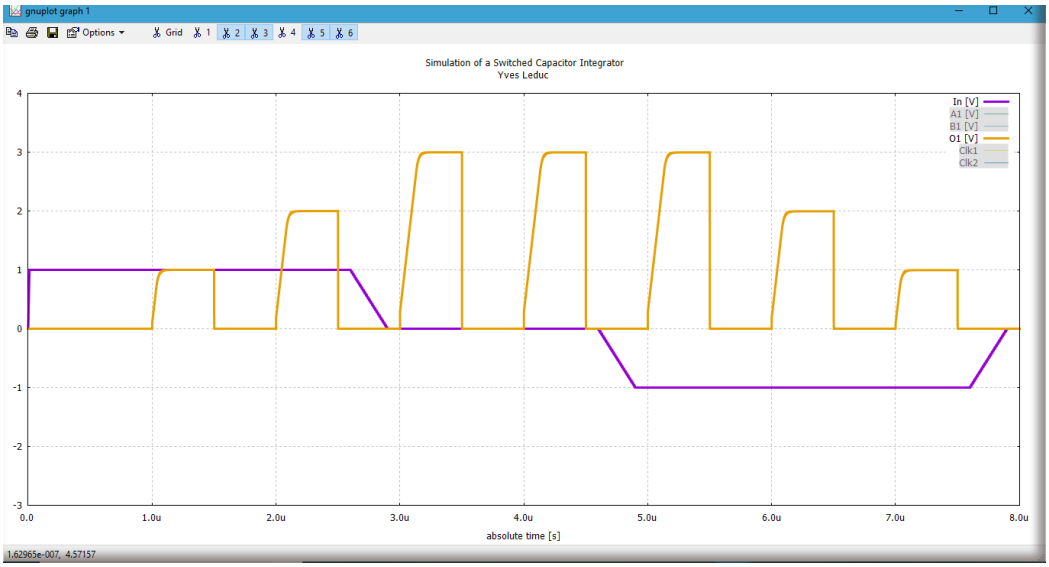
A Maxima user's function 'APPLY_LIMIT()' is used to implement the slew-rate model. The value of the constant current source must be declared as '**INTERN**', the current 'I(gm)' flowing in the transconductance and its voltage command 'cV(gm)' must be in '**OUTPUTS**'.



The numerical value of 'I_max' is in the data file and transmitted to the SARC function.

Using the same procedure, the slew-rate limitation appears now in the graphic.





The Napa Simulator has many features to analyze the results. Please consult the related teaser.