SIMSIDES
User Guide

José M. de la Rosa

Institute of Microelectronics of Seville, IMSE-CNM
(CSIC/University of Seville)
jrosa@imse-cnm.csic.es

September 2018

CONTENTS

1

SIMSIDES User Guide

SIMSIDES (SIMulink-based SIgma-DElta Simulator) is a time-domain behavioral simulator for $\Sigma\Delta$Ms that has been developed as a toolbox in the MATLAB/SIMULINK environment. SIMSIDES can be used for simulating any arbitrary $\Sigma\Delta$M architecture, implemented with both discrete-time and continuous-time circuit techniques. To this end, a complete list of $\Sigma\Delta$M building blocks (integrators, resonators, quantizers, embedded DACs, etc) is included in the toolboox. The behavioral models of these building blocks take into account the most critical error mechanisms of different circuit techniques including SC, SI, and CT circuits. These models, validated through transistor-level electrical simulations and by experimental measurements taken from a number of silicon prototypes, have been incorporated into the SIMULINK environment as C-MEX S-functions. This approach drastically increases the computational efficiency in terms of CPU time and accuracy of the simulation results.

The behavioral models included in SIMSIDES have been compiled and tested in a number of operating systems, including Mac OS X, UNIX (Solaris), Linux, and Microsoft Windows. Both 32-bit and 64-bit system platforms have been successfully tested in the majority of them. Although SIMSIDES was originally developed using version 6.5 of MATLAB, the toolbox has been updated and successfully used in a number of MATLAB/SIMULINK versions over the last years.

This document provides a user guide of SIMSIDES, giving an overview of the most significant features of the simulator. The interested reader can find more details in related publications [1, 2].

1.1 Getting Started: Installing and Running SIMSIDES

A free copy of SIMSIDES can be downloaded from the following web site:
`http://www.imse-cnm.csic.es/simsides`

This website contains also this user guide, examples and a link to a YouTube demo video about SIMSIDES.

After completing the online registration form and accepting the terms and conditions for using SIMSIDES, a zip file named `simsides.zip` is downloaded. The following steps must be followed to install the toolbox:

1. Uncompress the `simsides.zip` file to a directory of your computer hard disk. Let us assume that the directory is named `SIMSIDES` .
2. Start MATLAB.

Extracted from the book "Sigma-Delta Converters: Practical Design Guide (2nd Edition)," José M. de la Rosa

3. Set the MATLAB search path in order to add the `SIMSIDES` directory. To do this, go to `File` menu in MATLAB and select `Set Path`. The `Set Path` dialog box opens, listing all folders on the search path. From this dialog box, click the button `Add with Subfolders` and select the `SIMSIDES` directory to add to the search path. In order to reuse the newly modified search path including SIMSIDES directory and subdirectories, click `Save`, and finally click `Close`. This procedure–illustrated in Figure 1.1a–must be done only the first time SIMSIDES is installed in your hard disk.

In order to start SIMSIDES, type `simsides` at the MATLAB prompt and the SIMSIDES main window is displayed, as illustrated in Figure 1.1b.

1.2 Building and Editing ΣΔM Architectures in SIMSIDES

To create a new ΣΔM architecture in SIMSIDES, select `File` and then `New Architecture` in the main menu, and a new SIMULINK model window is displayed. Alternatively, an existing ΣΔM architecture can be opened by selecting `File -> Open Architecture` as illustrated in Figure 1.2.

In order to define a ΣΔM block diagram in SIMSIDES, the required building blocks can be incorporated from the `Edit` menu as shown in Figure 1.3. Both SIMULINK and SIMSIDES library models can be included by selecting `Edit -> SIMULINK Library` or `Edit -> Add block` , respectively. The latter option allows users to browse through all SIMSIDES library models. This way, clicking on `Edit -> Add block`a new window is displayed where the user can select either ideal or real building blocks, by choosing either `Add Ideal Block` or `Add Real Block` menus, respectively. In both cases, building-block models are organized in a set of sublibraries, namely: integrators, quantizers & comparators, D/A converters, resonators, and auxiliary blocks. The latter are only available in real libraries.

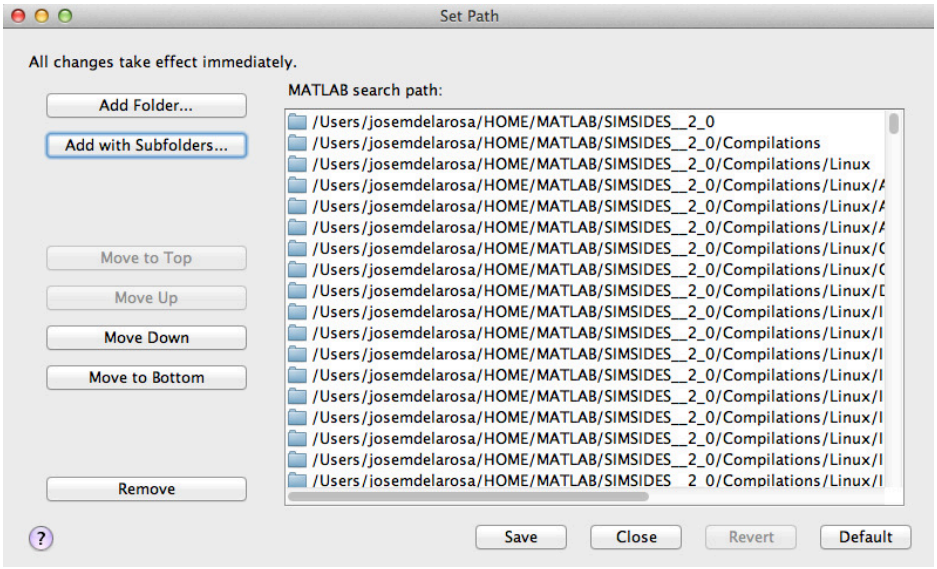
Some model libraries are grouped in sublibraries that contain different models corresponding to different kinds of circuit implementations. For instance, if library `Real Integrators` is selected, a new window is displayed where the user can select the circuit technique (CT, SC, or SI) as well as the type of integrator (i.e., either FE or LD in the case of SC and SI integrators, and Gm-C, Gm-MC, active-RC, MOSFET-C in the case of CT integrators). As an illustration, Figure 1.3 shows different sublibraries contained in the `Real Integrators` library. A complete list of model libraries and sublibraries is given in Chapter 2 of this user guide.

Once the ΣΔM block diagram is completed and the different building-block model parameters have been defined in the MATLAB workspace, the modulator can be simulated in SIMULINK following the same procedure as for the simulation of an arbitrary model in SIMULINK; i.e., choosing `Simulation -> Start` menu in the SIMULINK model window.

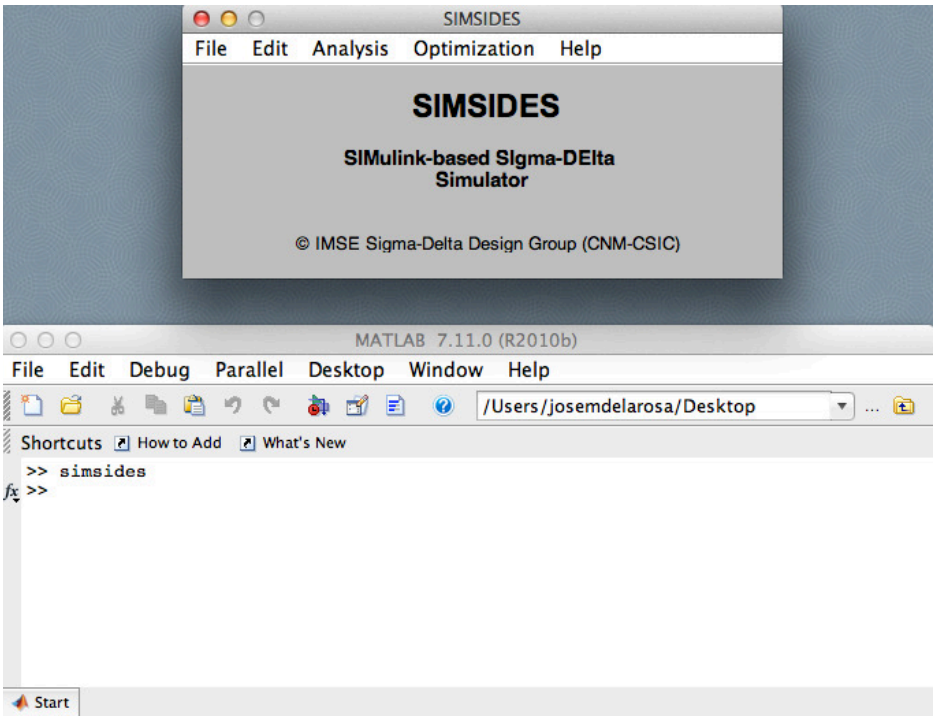
1.3 Analyzing ΣΔMs in SIMSIDES

Simulation output data can be post-processed in SIMULINK using the `Analysis` menu. As illustrated in Figure 1.4, the `Analysis` menu includes the following submenus:

- `Node Spectrum Analysis`, which computes and plots the FFT magnitude spectrum of a given signal.

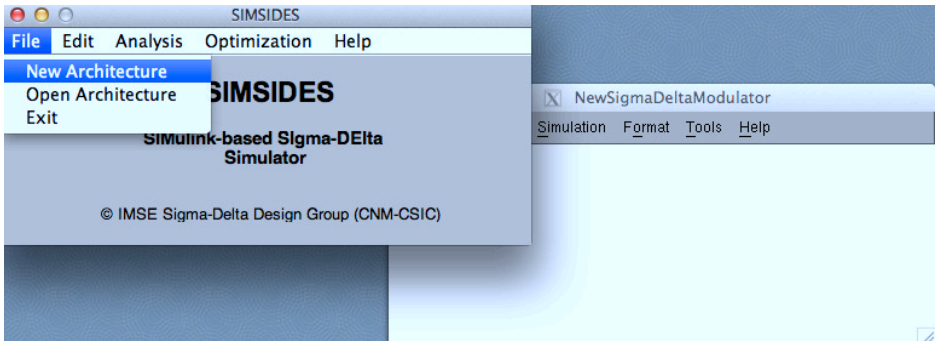


(a)

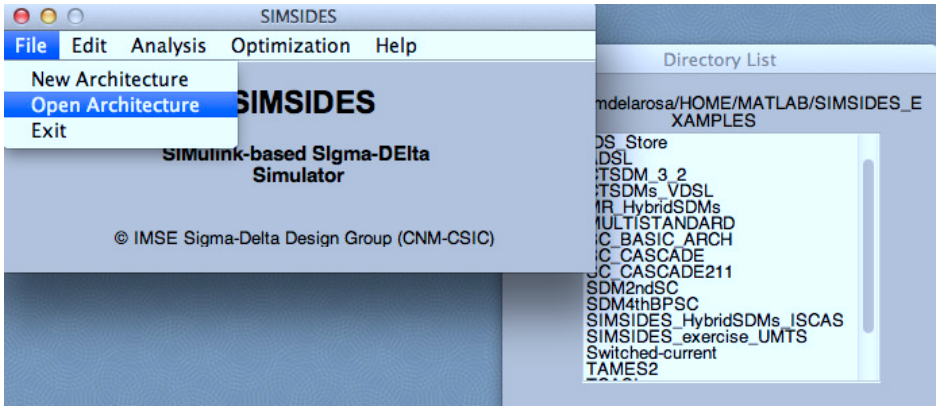


(b)

Figure 1.1 Installing and starting SIMSIDES: (a) Setting the MATLAB path. (b) Starting SIMSIDES at the MATLAB prompt (see also the demo video available in YouTube at https://www.youtube.com/watch?v=zT5XhBqysgU).



(a)



(b)

Figure 1.2 Building and editing ΣΔMs in SIMSIDES: (a) Creating a new ΣΔM architecture. (b) Opening an existing model.

- `Integrated Power Noise`, used for calculating and graphically representing the IBN within a given signal bandwidth.
- `SNR/SNDR` , which computes the SNR and/or SNDR within the band of interest, considering both LP- and BP-ΣΔMs.
- `Harmonic Distortion`, that computes dynamic harmonic distortion figures, like THD and intermodulation distortion figures.
- `Histogram`, used for representing histograms and analyzing the input/output swing in ΣΔM building blocks.
- `INL/DNL`, which calculates static harmonic distortion.
- `MTPR`, used for computing multi-tone power ratio (MTPR).
- `Parametric Analysis` , which allows to simulate the impact of a given model parameter on the performance of ΣΔMs.
- `Monte Carlo Analysis`, to do Monte Carlo simulations.

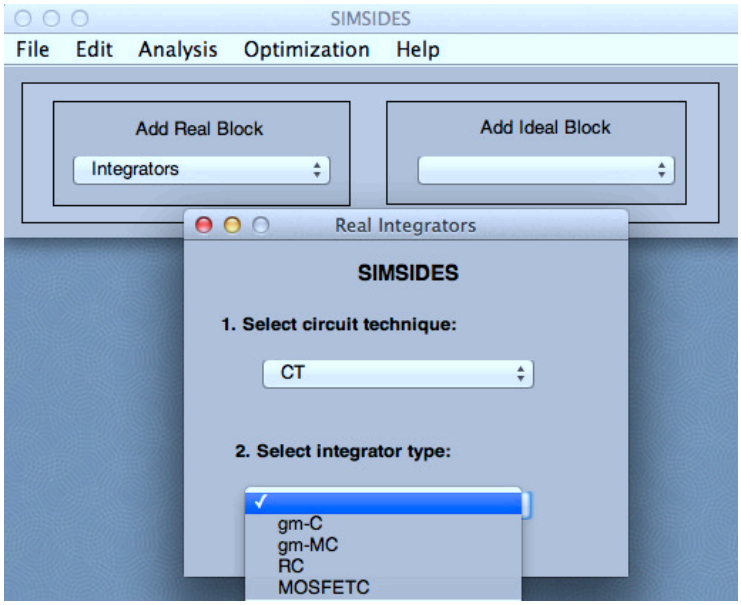


Figure 1.3 Illustrating different sublibraries included in the Real Integrators library.

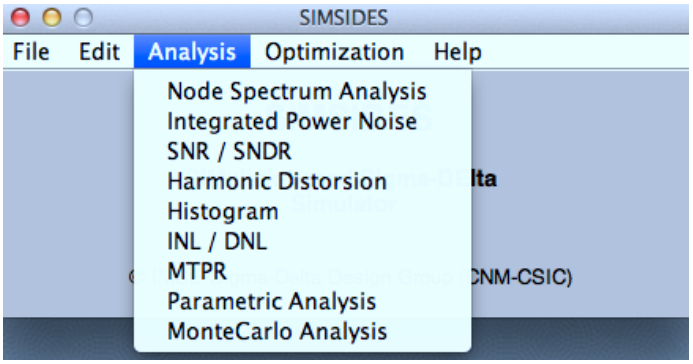


Figure 1.4 Analysis menu in SIMSIDES.

The required parameters and details involving the aforementioned analysis menus are described below.

Node Spectrum Analysis

Figure 1.5a shows the SIMSIDES `Node Spectrum Analysis` window. The following parameters are required to compute the FFT magnitude spectrum:

- `Name of the signal(s) to process`, where different variable names can be introduced, separated by commas. These variables can be output data generated in the simulations (for instance, the modulator output data stream), which have been previously saved in the MATLAB workspace by using the `To Workspace` SIMULINK block.
- `Sampling frequency`; i.e., the sampling frequency in Hz.
- `Window`, which defines the window function used for computing the FFT. The main window functions available in MATLAB can be selected, namely: Kaiser, Barlett, Blackman, Hamming, Hanning, Chebyshev, Boxcar, and Triangular.
- `Number of Points`; i.e., the number of points (`N` in Figure 1.5) for the selected window function and for FFT computation.
- `Window Parameters`, where other parameters required to define the window function are defined (like `Beta` parameter used in Kaiser windows).

Once these parameters have been defined, the output spectrum can be computed by clicking on the `Compute` button and then selecting the signal to be processed from the new window that is displayed (`Signal Spectrum` window shown in Figure 1.5b).

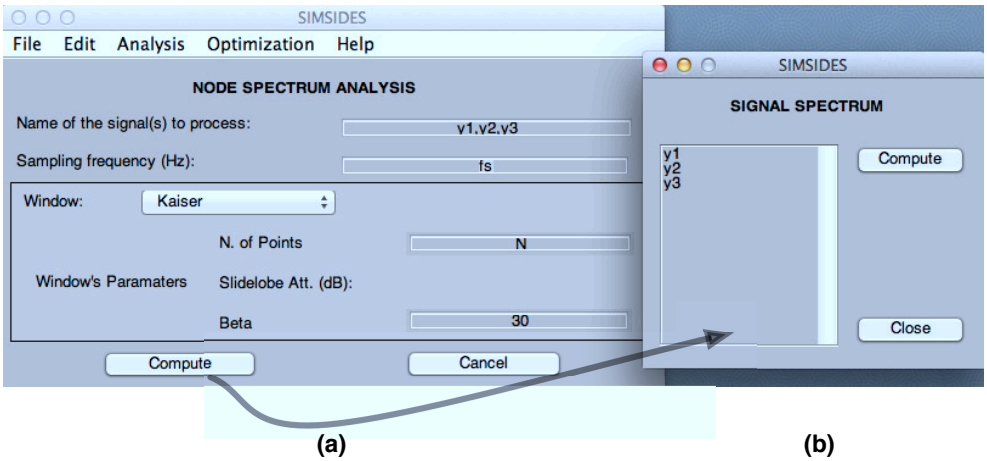


Figure 1.5 Node spectrum analysis menu.

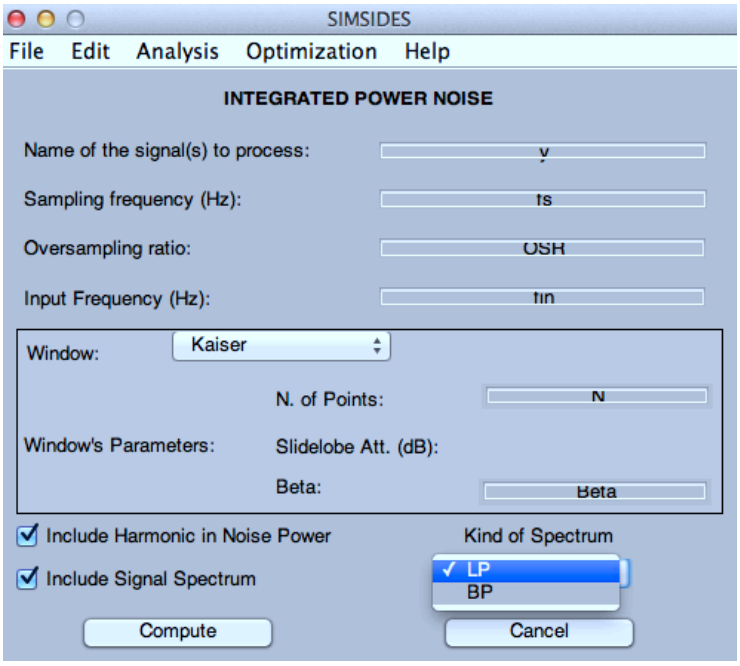


Figure 1.6 Integrated power noise menu.

Integrated Power Noise

Figure 1.6 shows the SIMSIDES `Integrated Power Node` window used for computing the IBN of any arbitrary data sequence obtained from simulations. To compute IBN, the following parameters are required:

- `Name of the signal(s) to process`.
- `Sampling frequency`; i.e., the sampling frequency in Hz.
- `Oversampling ratio`; i.e., the value of OSR that defines the signal bandwidth in which the IBN is computed.
- `Input frequency`, where it is assumed that a single-tone input signal is applied.
- `Window Parameters`; i.e., the parameters required to defined the window function used for computing the IBN.
- `Kind of Spectrum`, which specifies the signal nature; i.e., low-pass (LP) or band-pass (BP).

After defining all parameters described above, the IBN is computed by clicking on the `Compute` button. Harmonic distortion can be also taken into account in the calculation of the IBN by clicking the `Include Harmonic in Noise Power` button. The signal spectrum can be also plotted together with the IBN by choosing the `Include Signal Spectrum` option.

SNR/SNDR

Figure 1.7 shows the SIMSIDES `SNR/SNDR` window. The parameters required to calculate the SNR/SNDR of a given signal are essentially the same as those used for computing IBN–described in the previous section. In this case either the SNR or the SNDR is computed depending on the `Figure of merit` selected. Note that this kind of analysis calculates the SNR/SNDR for a given value of the input signal amplitude. If a SNR-versus-amplitude curve is required, a parametric analysis should be chosen as will be described later.

Harmonic Distortion

Figure 1.8 shows the SIMSIDES `Harmonic Distortion` window, which is used for computing the harmonic distortion power. Two different figures of merit can be calculated, namely: THD and third-order intermodulation distortion (`IM3` in Figure 1.8). The latter requires using a two-tone input signal. For that reason, there is an additional parameter named `Input2 Frequency` that defines the frequency of the second input tone.

Integral and Differential Non-Linearity

The `INTEGRAL AND DIFFERENTIAL NON-LINEARITY` menu, illustrated in Figure 1.9, is used for characterizing the static linearity in SIMSIDES. The analysis is based on either `Histograms` or `Input Ramp Waveform`–selected by the user. Other parameters required

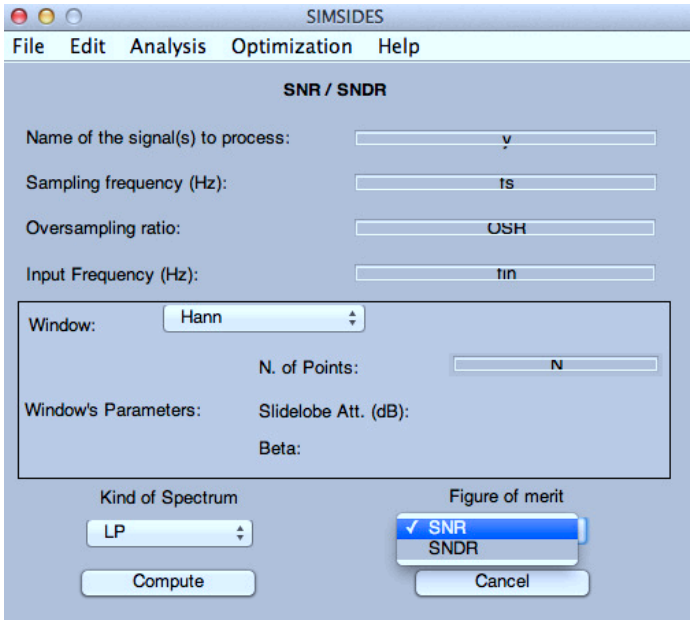


Figure 1.7 SNR/SNDR menu.

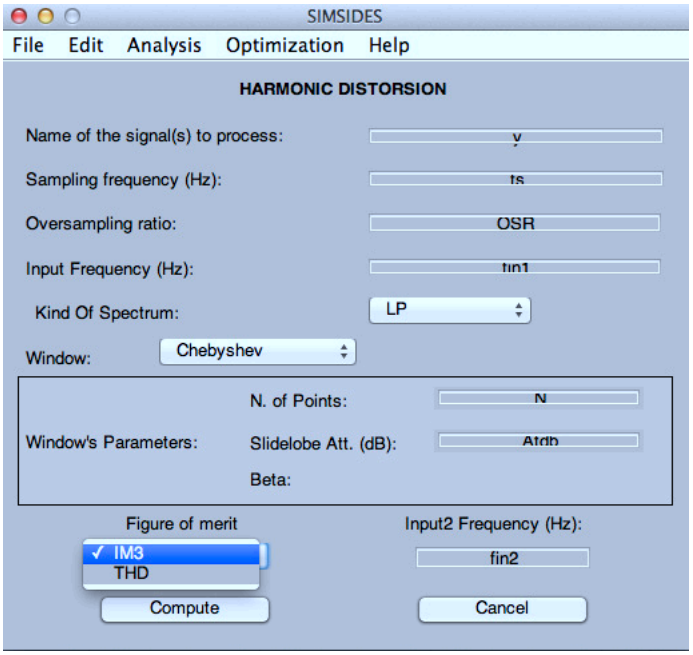


Figure 1.8 Harmonic distortion analysis menu.

to do this analysis are the `Input Amplitude` and the `Number of bits`, which specifies the ideal resolution of the A/D conversion, expressed in bits.

Multi-Tone Power Ratio

SIMSIDES can also analyze the harmonic distortion in some communication applications where a discrete multi-tone (DMT) signal is used. In this case, the linearity of the system is measured by a figure named multi-tone power ratio (MTPR). The corresponding SIMSIDES menu–shown in Figure 1.10–allows to compute MTPR for DMT input signals of different types:

- `Supressing 1 carrier of each 16`; i.e., 1 out of 16 carrier channels are suppressed.
- `Supressing 8 carrier of each 128`; i.e., 8 out of 128 carrier channels are suppressed.
- `Supressing 16 carrier of each 256`; i.e., 16 out of 256 carrier channels are suppressed.

In addition, the following parameters are also needed to compute MTPR:

- `Number of carriers`, which stands for the number of carrier channels in which the DMT signal is divided.

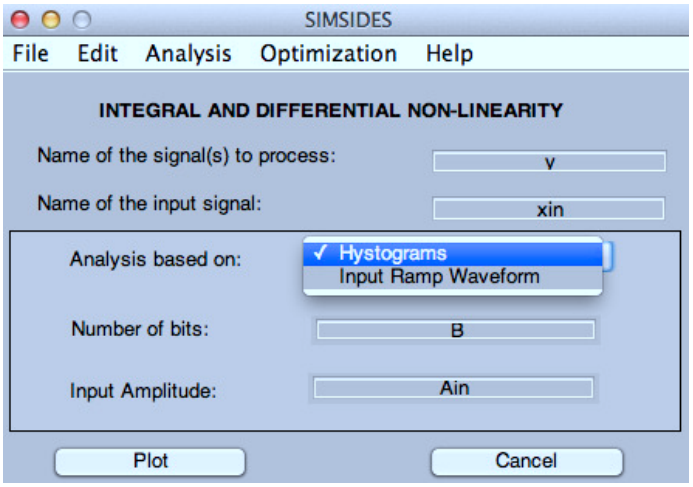


Figure 1.9 Integral and differential non-linearity analysis menu.

- `Bins by carrier`; the number of bins assigned to each carrier channel in the FFT.

Histogram

Histograms of signals that have been previously saved on the MATLAB workspace can be computed using the `HISTOGRAM` menu (illustrated in Figure 1.11), where the `Number of bins` specifies the number of intervals in which the signal range will be divided to compute the histogram.

Parametric Analysis

Figure 1.12 shows the SIMSIDES `PARAMETRIC ANALYSIS` menu. This menu is used for analyzing the impact of varying a model parameter on the performance of $\Sigma\Delta$Ms. Either one parameter or two parameters can be varied simultaneously by selecting the `Second Parameter` option. For each parameter, the following data must be specified:

- `Parameter Name`; i.e., the name of the model parameter to be varied. This model parameter can be a variable used in a $\Sigma\Delta$M building block model (like for instance I_o, g_m, etc) or a simulation parameter, like the input signal amplitude, sampling frequency, etc.
- `Range [vi,vf]`, which defines the variation range, defined by an interval with a lower value given by `vi` and a upper value of `vf`.
- `N. of points`; i.e., the number of points in which the variation interval is divided.
- `Scale`, that specifies if the variation range is either linear or logarithmic.
- `Analysis`, that specifies the type of analysis to be carried out, including output spectrum, IBN, SNR/SNDR, INL, MTPR, harmonic distortion, histograms, etc.

Monte Carlo Analysis

Figure 1.13 shows the SIMSIDES menu to run a Monte Carlo analysis. This is a particular case of parametric analysis, which has essentially the same functionalities and model parameters. The only difference is that the variation of the parameters involved in the Monte Carlo analysis are randomly varied according to a probability distribution with a mean value and a standard deviation which are specified in the analysis menu. Different types of probability distributions can be chosen, including `Normal`, `Log-Normal`, `Exponential`, and `Uniform` distributions.

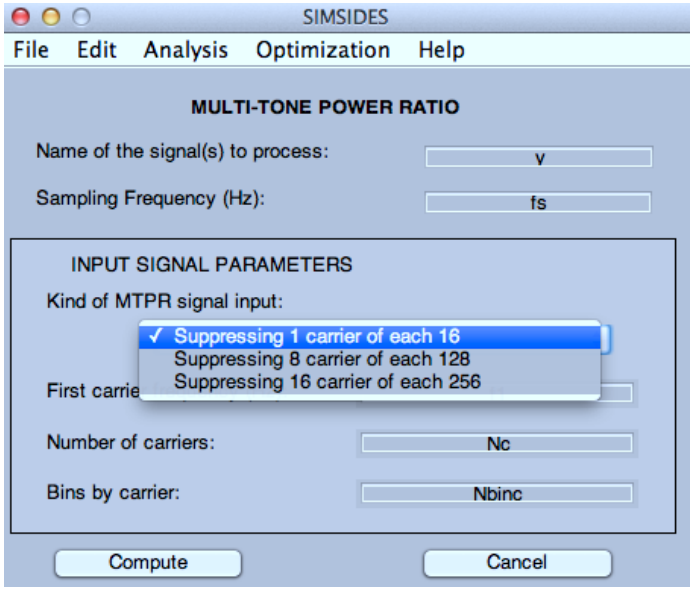


Figure 1.10 Multi-tone power ratio analysis menu.

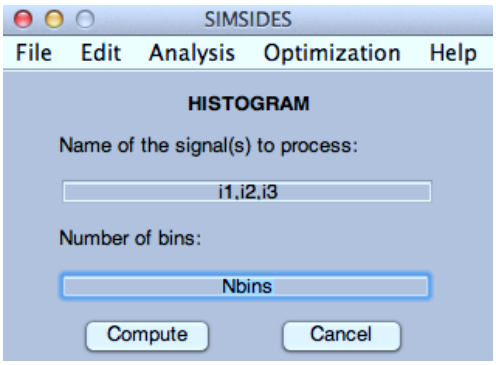


Figure 1.11 Histogram analysis menu.

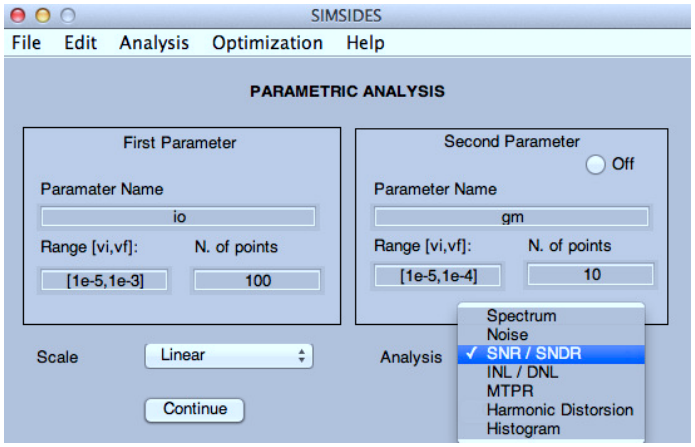


Figure 1.12 Parametric analysis menu.

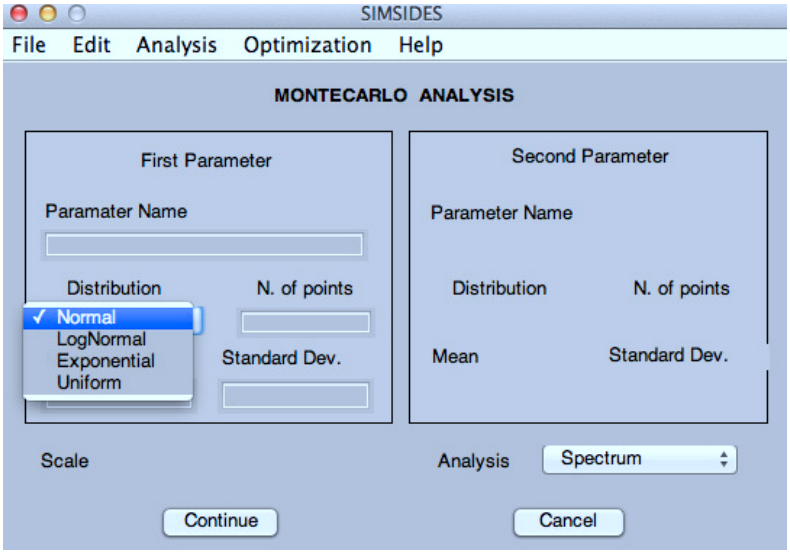


Figure 1.13 Monte Carlo analysis menu.

1.4 Optimization Interface

SIMSIDES includes an optimization interface to assist designers to combine time-domain behavioral simulation with optimization in order to automate and systematize the high-level design of ΣΔMs [3]. Figure 1.14a shows a snapshot of the optimization menu. A number of solvers such as `Fmincon` , `Fminsearch` , `Patternsearch` , search

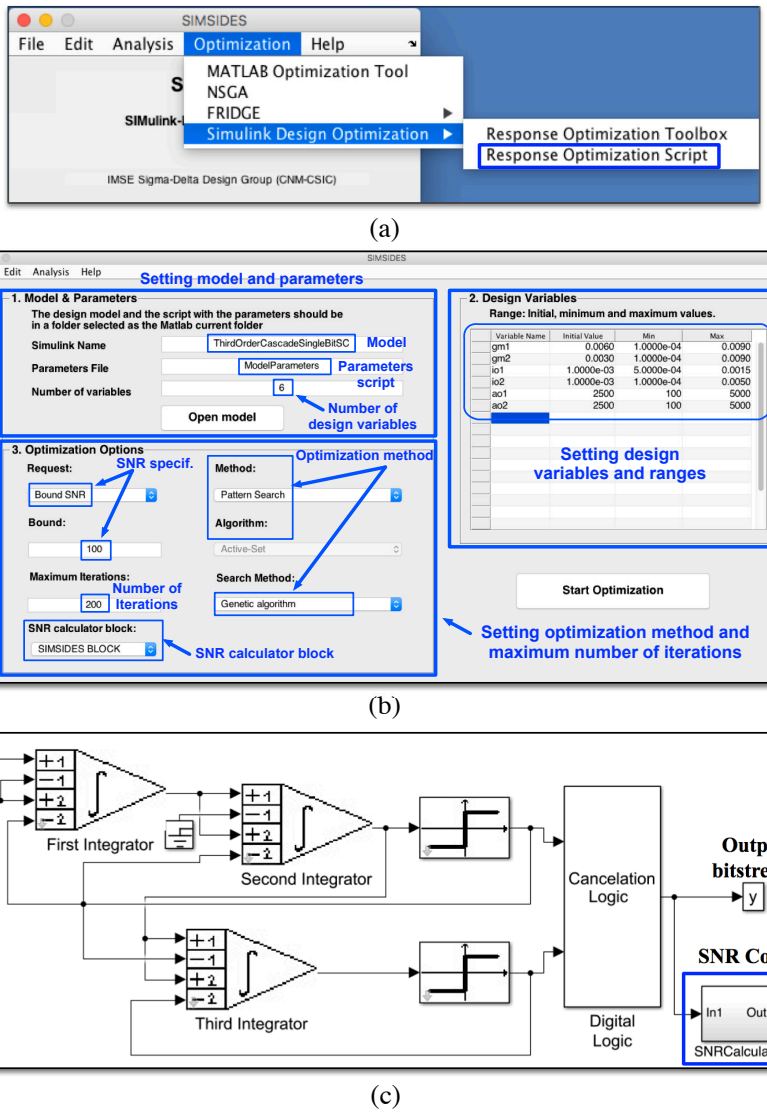


Figure 1.14 SIMSIDES optimization menu. (a) Main window. (b) GUI for setting optimization. (c) Model including the additional block to compute SNR while running optimization.

methods like `Gradient Descent` ,`Simplex Search` ,`Pattern` , etc. as well as diverse optimization algorithms, namely: `Neider-Mead` , `Genetic` , etc. can be used. In addition, some optimization engines–such as Multi-Objective Evolutionary Algorithm (MOEAs) like NSGA-II–is also included.

The `SIMULINK Design Optimization` toolbox can be used with any arbitrary SIMULINK model, and hence, SIMSIDES models can be also combined with this toolbox. However, although a powerful and user friendly GUI is provided, the formulation of the optimization problem is not an easy task when applied to the design of ADCs, and particularly to $\Sigma\Delta$Ms. The reason is that a suitable performance metric– like for instance the SNR– cannot be directly set from the optimization toolbox, which is mostly focused on optimizing the frequency response of filters based on fine tuning their Bode diagrams.

In order to address these problems, a dedicated interface has been developed and embedded in SIMSIDES in order to help designers combine the benefits of its time-domain behavioral models for $\Sigma\Delta$Ms with the different optimization methods and algorithms available in MATLAB. Figure 1.14b illustrates an excerpt of the SIMSIDES optimization interface. This optimization menu allows designers to customize their optimization problem by setting all required pieces of information, namely: name of the SIMSIDES model, MATLAB script including main simulation parameters, number of design variables, initial values and ranges of variables and the optimization method. To this end, designers need to follow these three basic steps:

- Build a model of the $\Sigma\Delta$M in SIMSIDES
- Create a MATLAB script with all design variables to be optimized as well as the parameters required to simulate the $\Sigma\Delta$M model
- Enter the information corresponding to the SIMSIDES model and the MATLAB script in the SIMSIDES optimization interface.

Different performance metrics can be used for optimization purposes. Without loss of generality, the SNR has been considered. Therefore, the design objective is to maximize the SNR while optimizing the $\Sigma\Delta$M building-block design variables in order to minimize the power consumption. However, the optimization solvers and algorithms available in MATLAB are intended to minimize a given function, rather than to maximize it – as it is the case here. Therefore, in order to overcome this limitation, an additional block is added to the SIMSIDES model–depicted in Figure 1.14c– which calculates the SNR at the output of the $\Sigma\Delta$M output bitstream and obtains the negative value of the computed SNR, so that the optimization problem can be formulated as:

$$\text{maximize } [f(\bar{x})] = \text{minimize } [-f(\bar{x})] \tag{1.1}$$

where $f(\bar{x})$ is the performance metric to be optimized–SNR in this case–and $\bar{x}$ denotes the vector of design variables involved in the optimization, i.e. the $\Sigma\Delta$M building-block specifications to be minimized/maximized in order to get the maximum SNR with the minimum power consumption. For the design variables, initial values and ranges need to be entered as depicted in Figure 1.14b. Also, the algorithm solver and search method used in the optimization are set as well as the maximum number of iterations to be considered in order to limit the CPU time in case the synthesis process does not converge to any solution. During the optimization procedure, the toolbox gives information about both the number

of iterations and the number of simulations. The former refers to the number of times the optimization algorithm is run whereas the later stands for the number of time the $\Sigma\Delta$M is simulated. The interested reader can find more detailed information about the combined used of SIMSIDES with MATLAB optimizers in [3].

1.5 Tutorial Example: Using SIMSIDES to Model and Analyze $\Sigma\Delta$Ms

This section illustrates the use of SIMSIDES through a simple example in which several kinds of analysis will be carried out to show the main features of the simulator[1]. Figure 1.15 shows the block diagram of the modulator under study, which consists of a third-order cascade 2-1 DT-$\Sigma\Delta$M with single-bit quantization in both stages.

Creating the Cascade 2-1 $\Sigma\Delta$M Block Diagram in SIMSIDES

The modulator block diagram shown in Figure 1.15 can be implemented by using the model libraries available in SIMSIDES. To this end, the same procedure as described in Section 1.2 is followed:

- Go to SIMSIDES main menu, select `File -> New Architecture` and introduce a name for the new $\Sigma\Delta$M architecture.
- Include the integrators and comparators from the SIMSIDES model libraries. To do this, select `Edit -> Add Block`. In this example, the FE integrators in Figure 1.15 are implemented by using the `SC_FE_Integrator_All_Effects` blocks from the `Real Integrators` library, whereas single-bit quantizers are modeled by the `Real_Comparator_Offset&Hysteresis` comparator block available in `Quantizers&Comparators` library. These building blocks can be incorporated in the new architecture by simply dragging and dropping the models from their corresponding SIMSIDES libraries, as illustrated in Figure 1.16a.

[1] The interested reader can find more examples and details in [2].

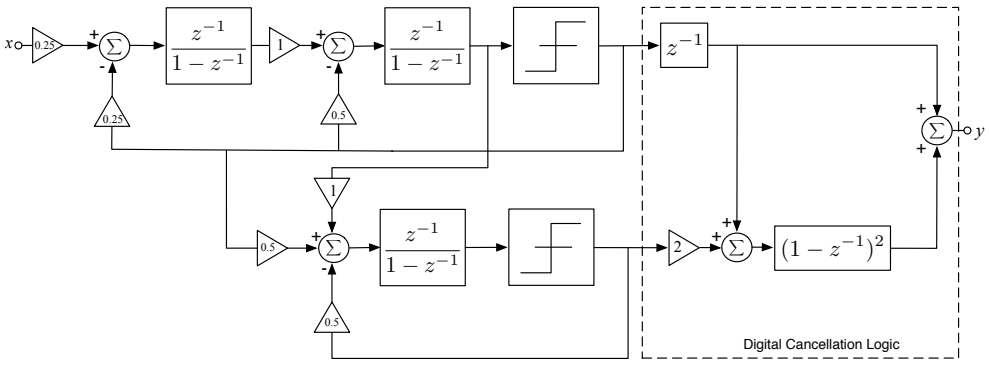
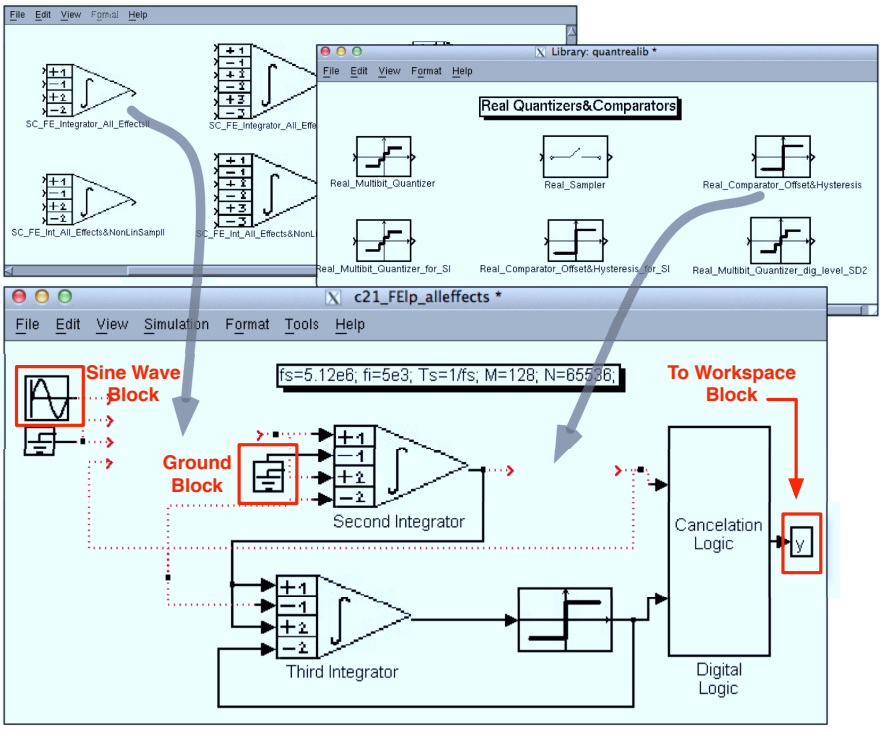
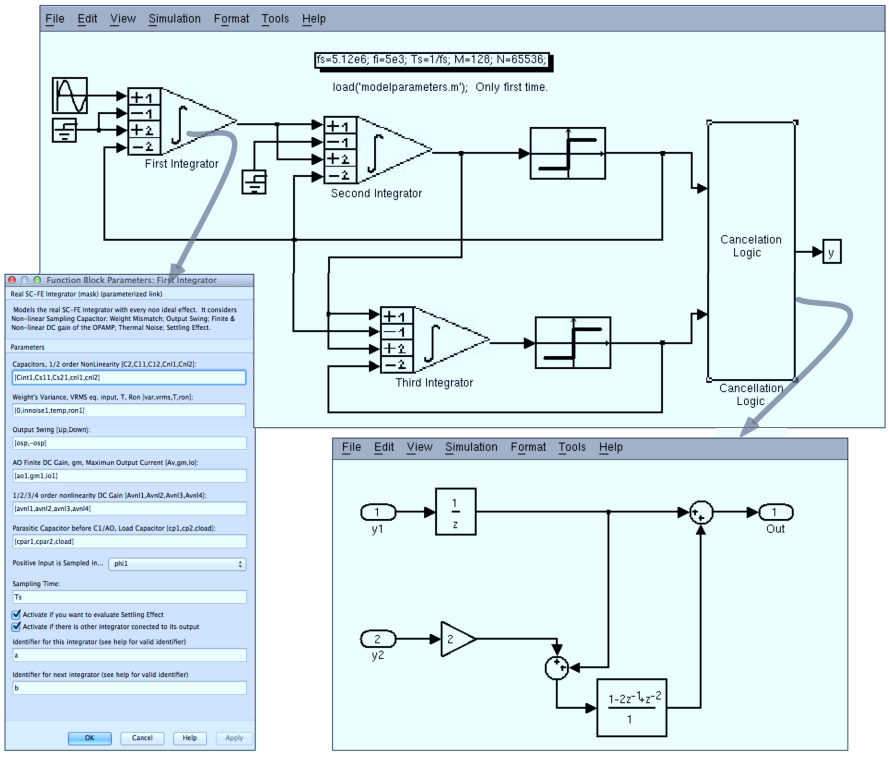


Figure 1.15 Z-domain block diagram of a cascade 2-1 DT-$\Sigma\Delta$M.



(a)



(b)

Figure 1.16 SIMSIDES block diagram of the ΣΔM shown in Figure 1.15: (a) Building and editing the block diagram. (b) Complete modulator block diagram in SIMSIDES.

- Incorporate the remaining building blocks from the SIMULINK model library. To do this, go to `Edit -> Simulink Library` and drag the required models. In this example the following blocks are required: `Sine Wave` and `Ground` blocks from `Sources` library, `Unit Delay` and `Discrete Filter` block from the `Discrete` library, and `To Workspace` from `Sinks` library.
- Finally, once all required blocks have been included in the new architecture, they are properly connected to implement the required ΣΔM architecture shown in Figure 1.16b.

Setting Model Parameters

The modulator parameters and model parameters required to simulate the block diagram of Figure 1.16 can be either set up in the MATLAB command window or they can be alternatively saved in an M-file that is loaded when needed. As an illustration, Figure 1.17 shows the M-file used for setting up all model parameters of Figure 1.16, that also includes a brief description of the different parameters and variables included. For the sake of completeness, Table 1.1 includes the values of all building-block parameters as they are described in the SIMSIDES user masks, as well as other auxiliary block parameters (such as those used in `Sine Wave` and `To Workspace` blocks) which are required during simulation. In addition to these model parameters, simulation parameters must be set up to run a simulation. To do this, go to `Simulation -> Simulation Parameters` menu and define the following parameters:

- Simulation Time: `Start Time: 0.0` ; `Stop Time: (N-1)*Ts`
- Solver options: `Type: Variable Step` ; `Max Step Size: Auto`

Note that integrator building blocks are identified in order to properly compute the equivalent load capacitances required for the incomplete settling error model.

Computing the Output Spectrum

The output spectrum of the ΣΔM can be computed in SIMSIDES by following the next steps:

- Set up model parameters by using the M-file shown in Figure 1.17.
- Simulate the modulator in Figure 1.16b from the menu `Simulation -> Start`.
- Once the simulation has finished, go to `Analysis -> Node Spectrum Analysis` menu in SIMSIDES.
- Define the parameters requested in that menu. In this example, the sampling frequency is defined as `fs` and a Kaiser window function is used with a number of points `N` and `Beta = 20`.
- Click on `Compute` and then `Plot`, and the output spectrum shown in Figure 1.18 is displayed.

Table 1.1 Building-block model parameters used for simulating the ΣΔM in Figure 1.16b.

Building Block	Parameter Description	Value/Variable
Input Sine Wave	Sine Type	Time based
	Amplitude	0.5
	Bias	0
	Frequency (rad/s)	2*pi*fi
	Phase (rad)	0
	Sample time	0
	Interpret vector parameters	Selected
First Integrator	Integration and Sampling Capacitors (Branch 1, Branch 2)	[Cint1,Cs11,Cs21]
	Capacitor nonlinear coefficients	[cnl1,cnl2]
	Weight's variance, rms eq.input noise, temperature	[0,innoise1,temp]
	OTA DC gain, transconductance, max. output current	[ao1,gm1,io1]
	Positive/Negative Output swing	[osp,-osp]
	Switch on-resistance	ron1
	OTA DC gain nonlinear coefs.	[avnl1,2,3,4]
	Parasitic capacitances before the OTA	[cpar1,cpar2]
	Load capacitance	cload
	Positive Input 1 is sampled at...	phi1
	Sampling Time	Ts
	Identifier for this integrator	a
	Identifier for the next integrator	b
Second, Third Integrators	Integration and Sampling Capacitors (Branch 1, Branch 2)	[Cint2,Cs12,Cs22]
	Capacitor nonlinear coefficients	[cnl1,cnl2]
	Weight's variance, rms eq.input noise, temperature	[0,innoise2,temp]
	OTA DC gain, transconductance, max. output current	[ao2,gm2,io2]
	Positive/Negative Output swing	[osp,-osp]
	Switch on-resistance	ron2
	OTA DC gain nonlinear coefs.	[avnl1,2,3,4]
	Parasitic capacitances before the OTA	[cpar1,cpar2]
	Load capacitance	cload
	Positive Input 1 is sampled at...	phi1
	Sampling Time	Ts
	Identifier for this integrator (second integrator)	b
	Identifier for this integrator (third integrator)	c
	Identifier for the next integrator	c
Comparators	Vhigh, Vlow	[vref, -vref]
	Offset, Hysteresis	[0,hys]
	Phase ON	phi1
	Sampling Time	Ts
	Identifier for this quantizer	quant1
To Workspace (*y*)	Variable name	y
	Limit data points to last	N
	Decimation	1
	Sample Time	Ts
	Save format	Array

SNR versus Input Amplitude Level

Figure 1.19 shows the SNDR versus input amplitude level (or SNDR curve) of the ΣΔM in Figure 1.16b. This figure has been obtained by using the `Analysis` menu and choosing `SNR/SNDR` analysis[2]. In this example, the following parameters are used:

- `Parameter Name: Ain`, where `Ain` is the `Amplitude` parameter defined in the `Input Sine Wave` block in Figure 1.16b.
- `Range [vi,vf]: [1e-6,2]`
- `N. of points: 50`
- `Scale: Logarithmic`

[2]Alternative, a specific SNDR-vs-amplitude analysis can be carried out in which the input amplitude is represented in dB referred to the Full-Scale (FS) range (dBFS). To do so, go to *Analysis− >Parametric Analysis− >SNR/SNDR vs Amplidute (dBFS)*.

```
% SDM parameters:
% Sampling Frequency(fs), Input Frequency (fi), Sampling Time (Ts)
% OverSampling Ratio (OSR=M); Number of points (N)
fs=5.12e6; fi=5e3; Ts=1/fs; M=128; N=65536;
% Model parameters
kt=0.026*1.6e-19; % Boltzmann constant

% First Integrator's parameters
Cint1=24e-12; % integration capacitor For gain=1
Cs11=6e-12; % sampling capacitor (branch 1)
Cs21=6e-12; % sampling capacitor (branch 2)
innoise1=0; % rms value of the input equivalent noise
ao1=2.63e3; % open-loop OTA DC gain
gm1=4.5e-3; % transconductance
io1=0.977e-3; % maximum OTA output current
ron1=60; % sampling switch-on resistance

% Second- and Third- Integrators
Cint2=3e-12;
Cs12=1.5e-12;
Cs22=1.5e-12;
innoise2=0;
ao2=1.38e3;
gm2=0.87e-3;
io2=0.25e-3;
ron2=650;

% Common integrator parameters
temp=175; % temperature
osp=2.7; % output swing
cnl1=0; % capacitor first-order non-linear coef.
cnl2=25e-6; % capacitor second-order non-linear coef.
avnl1=0; % DC gain first-order non-linear coef.
avnl2=15e-2; % DC gain second-order non-linear coef.
avnl3=0; % DC gain third-order non-linear coef.
avnl4=0; % DC gain fourth-order non-linear coef.
cpar1=0.6e-12; % parasitic (opamp) input capacitance
cpar2=0.6e-12;
cload=2.28e-12; % opamp (intrinsic) load capacitance
% Comparators
vref=2; % DAC reference voltage
hys=30e-3; % comparator hysteresis
```

Figure 1.17 M-file including all model parameters required to simulate the ΣΔM in Figure 1.16b.

- `Analysis: SNR/SNDR`
- `Second Parameter: Off`

Once the aforementioned parameters are set up, click on `Continue` and the `SNR/SNDR` window menu shown in Figure 1.7 is displayed. The requested parameters (i.e., sampling frequency, oversampling ratio, etc) are set up according to the values given in Figure 1.17, namely:

- `Name of the signal(s) to process: y`
- `Sampling frequency (Hz): fs`
- `Oversampling ratio: M`
- `Input Frequency (Hz): fi`
- `Window: Kaiser`
- `N. of Points: N`
- `Beta: 20`
- `Kind of Spectrum: LP`
- `Figure of merit: SNDR`

After setting up the aforementioned parameters, click on `Compute` and then `Plot` to obtain the curve given in Figure 1.19.

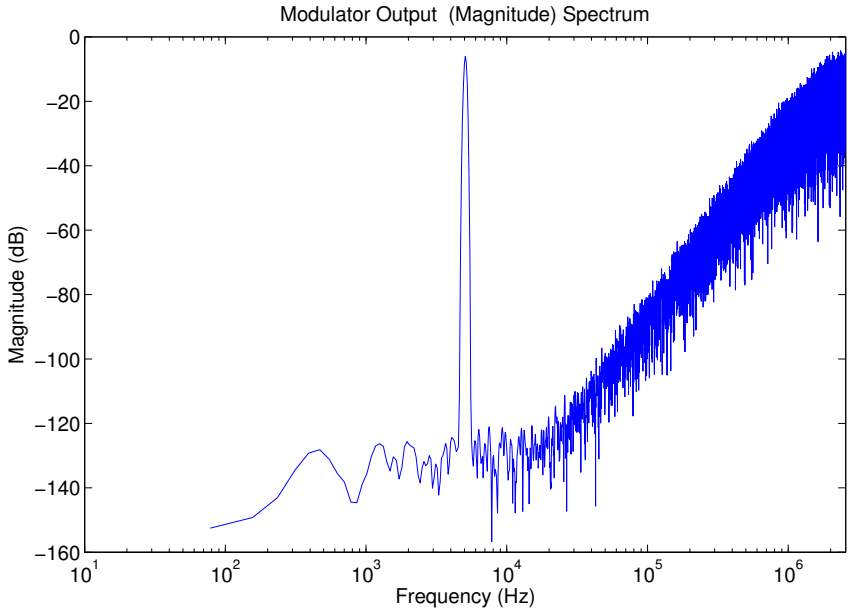


Figure 1.18 Output spectrum (magnitude) of the ΣΔM in Figure 1.16b.

Parametric Analysis Considering Only One Parameter

The `Parametric Analysis` menu can be used for studying the effect of a given model parameter on the modulator performance. For instance, let us consider the effect of the OTA transconductance g_m of the front-end integrator in Figure 1.16b. In order to analyze the impact of this parameter on the effective resolution of the modulator, go to `Parametric Analysis` menu and set up the following parameters:

- `Parameter name: gm1`, which stands for g_m of the front-end integrator block in Figure 1.16b.
- `Range [vi,vf]: [1e-5,1e-3]`
- `N. of points: 50`
- `Scale: Linear`
- `Analysis: SNR/SNDR`
- `Second Parameter: Off`

Once these parameters are defined, click on `Continue` and proceed in a similar way to previous examples in order to compute the SNDR. Figure 1.20 shows the results of this analysis, by depicting the SNDR versus `gm1`.

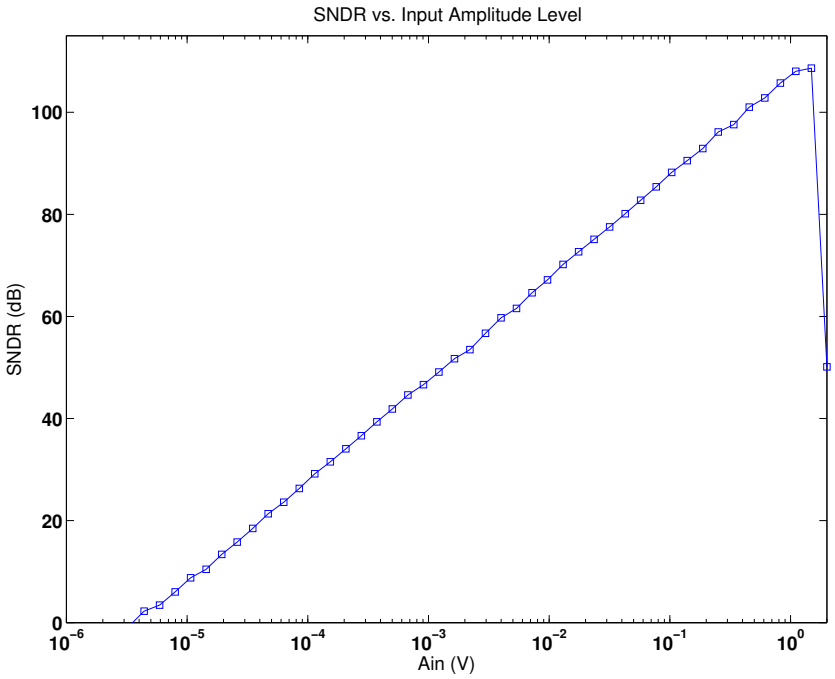


Figure 1.19 SNDR versus input amplitude level of the ΣΔM in Figure 1.16b.

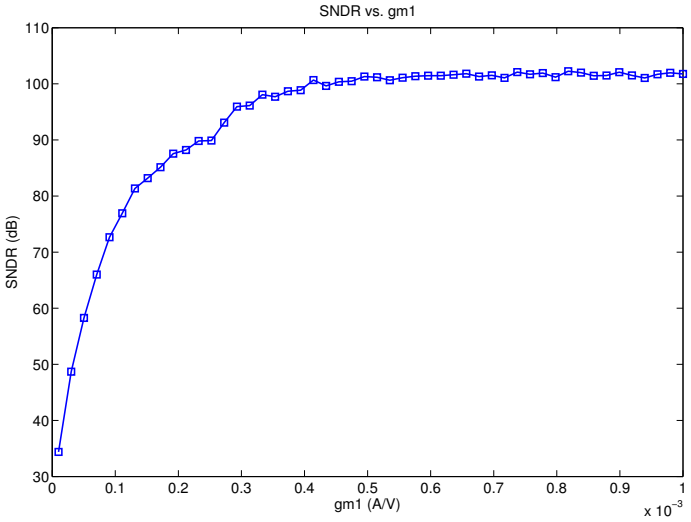


Figure 1.20 Using parametric analysis to study the effect of a single model parameter: SNDR versus transconductance of the front-end amplifier for the ΣΔM in Figure 1.16b.

Parametric Analysis Considering Two Parameters

The `Parametric Analysis` menu can be used also for implementing parametric analyses considering the variation of two different parameters. As an example, Figure 1.21 shows the effect of both the OTA transconductance g_{m1} and the maximum output current I_{o1} of the front-end amplifier on the SNDR of the ΣΔM in Figure 1.16b.

In order to obtain the graph in Figure 1.21, the following parameters are set up in the `Parametric Analysis` menu:

- `Parameter name: io1`, which stands for the maximum output current I_o of the front-end integrator.
- `Range [vi,vf]: [1e-4,1e-3]`
- `N. of points: 10`

Computing Histograms

Finally, to conclude this example, Figure 1.22 illustrates the histograms of the integrators outputs in the front-end stage of the modulator in Figure 1.16b.

These histograms have been obtained by using the `Analysis -> Histograms` menu from SIMSIDES and setting up the following model parameters:

- `Name of the signal(s) to process: y1,y2`, which are the names given to the output of the integrators saved into the MATLAB workspace by using `To Workspace` blocks from the SIMULINK elementary library.
- `Number of bins: 100`

1.6 Getting Help

SIMSIDES includes a help menu (illustrated in Figure 1.23) from which this user guide can be opened by selecting `Help -> User Manual` in the SIMSIDES main window. In addition, a complete list of all behavioral models (and their corresponding parameters) included in SIMSIDES–described in Chapter 2–can be also obtained from this menu by selecting `Help -> Libraries and Models`.

References

[1] J. Ruiz-Amaya *et al.*, "High-Level Synthesis of Switched-Capacitor, Switched-Current and Continuous-Time $\Sigma\Delta$ Modulators Using SIMULINK-based Time-Domain Behavioral Models," *IEEE Trans. on Circuits and Systems – I: Regular Papers*, pp. 1795–1810, Sep. 2005.

[2] J. M. de la Rosa, *Sigma-Delta Converters: Practical Design Guide (2nd Edition).* Wiley-IEEE Press, 2018.

[3] B. Cortes-Delgadillo *et al.*, "Embedding MATLAB Optimizers in SIMSIDES for the High-Level Design of Sigma-Delta Modulators," *IEEE Transactions on Circuits and Systems - II: Express Briefs*, vol. 65, pp. 547–551, May 2018.

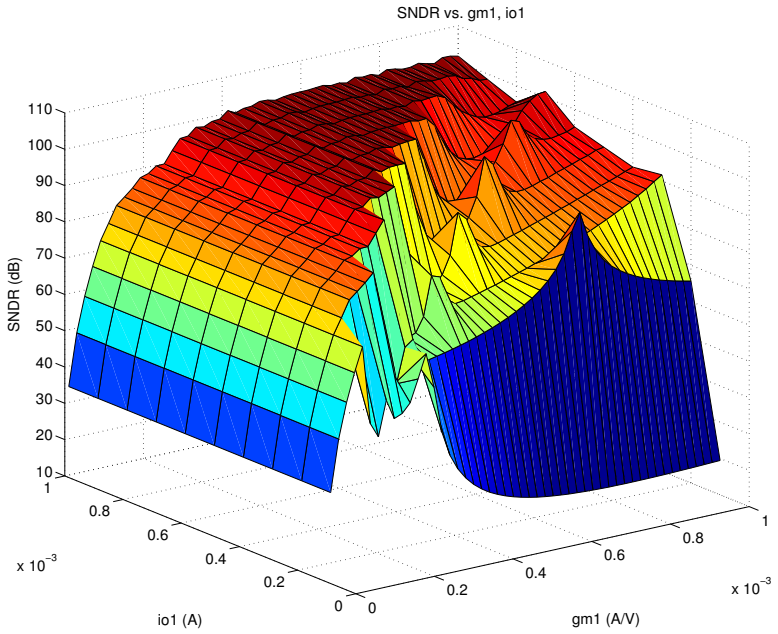


Figure 1.21 Parametric analysis considering the effect of two parameters (g_{m1} and I_{o1}) on the SNDR.

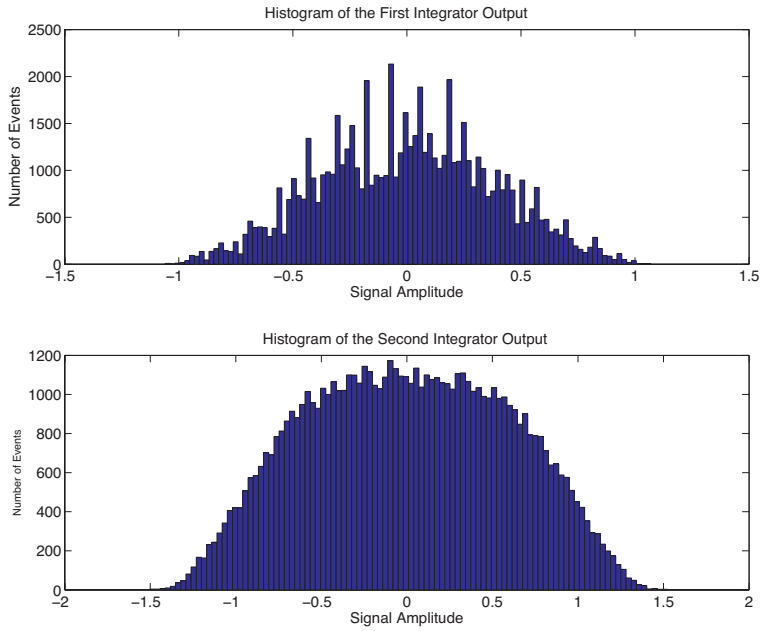


Figure 1.22 Illustrating the use of histograms of the modulator in Figure 1.16b.

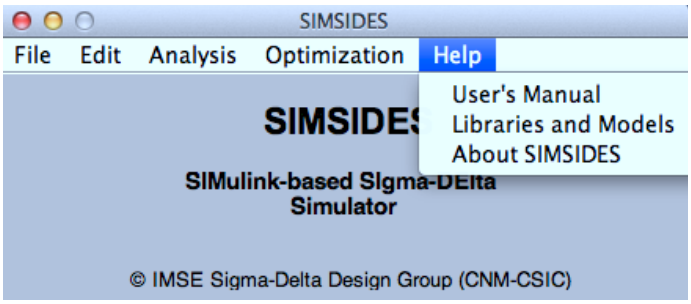


Figure 1.23 Help menu.

2

SIMSIDES Block Libraries and Models

This chapter contains a compilation of most of the ΣΔM building blocks and libraries included in SIMSIDES. A brief description of their purpose and functionality, as well as their main model parameters, is provided. Revised versions of this chapter are periodically updated at SIMSIDES website and provided together with this user manual included in SIMSIDES. In addition, a list of ΣΔM models available in SIMSIDES can be downloaded from the SIMSIDES web site at: `http://www2.imse-cnm.csic.es/simsides/docs/SIMSIDES_Examples.pdf`.

2.1 Overview of SIMSIDES Libraries

Table 2.1 compiles all libraries included in SIMSIDES together with a brief description of their contents. These libraries are divided into two main categories: ideal libraries and real libraries. The former contains ideal building blocks, whereas the latter includes behavioral models that incorporate circuit-level nonidealities. The libraries containing integrators and resonators are subdivided into several specific sublibraries, which include in turn building-block models corresponding to different circuit-level implementations. For instance, SC integrators are subdivided into FE and LD integrators; CT integrators are subdivided into Gm-C, active-RC, etc.

2.2 Ideal Libraries

As shown in Table 2.1, SIMSIDES includes four ideal libraries, namely: integrators, resonators, quantizers, and DACs. The building blocks contained in these libraries are described in the following sections.

2.2.1 Ideal Integrators

There are three kinds of ideal integrators in this library, namely: `Ideal_CT_Integrator`, `Ideal_FE_Integrator`, and `Ideal_LD_Integrator`.

Table 2.1 Overview of SIMSIDES libraries.

Ideal Libraries	**Sublibraries**	**Building Blocks**
Integrators	–	Ideal DT/CT integrators
Resonators	–	Ideal resonators
Quantizers & Comparators	–	Ideal quantizers
D/A Converters	–	Ideal DACs
Real Libraries	**Sublibraries**	**Building Blocks**
Integrators	SC FE integrators	Forward-Euler SC integrators
	SC LD integrators	Lossless-Direct SC integrators
	SI FE integrators	Forward-Euler SI integrators
	SI LD integrators	Lossless-Direct SI integrators
	gm-C integrators	Gm-C integrators
	gm-MC integrators	Miller OTA integrators
	RC integrators	Active-RC integrators
	MOSFET-C integrators	MOSFET-C integrators
Resonators	SC FE resonators	Resonators based on FE SC integrators
	SC LD resonators	Resonators based on LD SC integratorss
	SI FE resonators	Resonators based on FE SI integrators
	SI LD resonators	Resonators based on LD SI integrators
	gm-C resonators	Resonators based on Gm-C integrators
	gm-LC resonators	Resonators based on Gm-LC integrators
Quantizers & Comparators	–	Nonideal single-bit & multibit quantizers
D/A Converters	–	Nonideal single-bit and multibit DACs
Auxiliary Blocks	–	Adders, latches, DEM blocks, etc

Building-Block Model Purpose and Description

`Ideal_CT_Integrator` block models the ideal S-domain transfer function of a CT integrator, given by,

$$\mathrm{ITF}(s) = g \cdot \frac{1}{s} \tag{2.1}$$

where g stands for the integrator gain (also referred to as integrator weight).
`Ideal_FE_Integrator` and `Ideal_LD_Integrator` blocks model the ideal Z-domain transfer functions of DT FE and LD integrators, respectively given by:

$$\begin{aligned} \mathrm{ITF}_{\mathrm{FE}}(z) &= g \cdot \frac{z^{-1}}{1 - z^{-1}} \\ \mathrm{ITF}_{\mathrm{LD}}(z) &= g \cdot \frac{z^{-1/2}}{1 - z^{-1}} \end{aligned} \tag{2.2}$$

Model Parameters

The following model parameters are included in the dialog box of the models above:

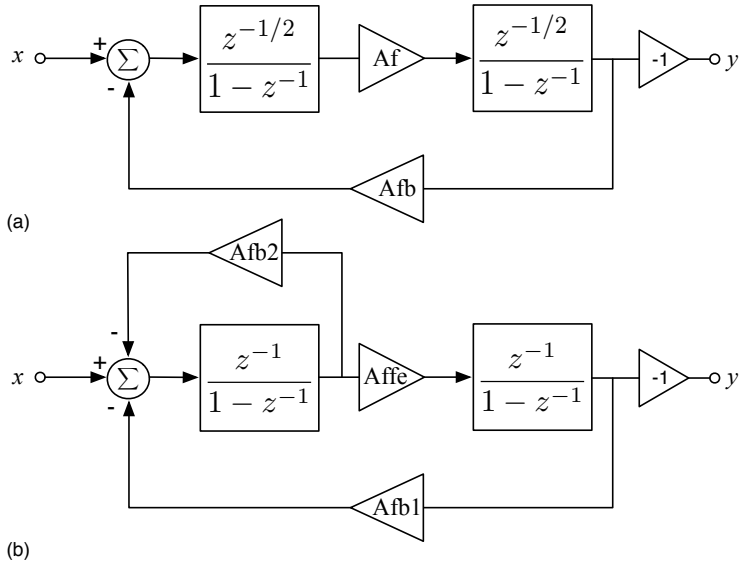


Figure 2.1 Z-domain block diagram of: (a) Ideal_LD_Resonator. (b) Ideal_FE_Resonator.

- `Gain`, which defines the integrator weight g.
- `Sampling Time`; i.e., the sampling period of both the `Ideal_FE_Integrator` and `Ideal_LD_Integrator` blocks.

2.2.2 Ideal Resonators

This library includes diverse ideal resonator blocks described below.

Ideal_LD_Resonator

This block consists of a DT resonator made up of two LD integrators connected in a feedback loop as illustrated in Figure 2.1a. Its model parameters are the following:

- `Af`; i.e., the forward loop gain.
- `Afb`, which stands for the feedback loop gain.
- `Sampling Time`.

Ideal_FE_Resonator

This block models a DT resonator made up of two FE integrators connected in a feedback loop as shown in Figure 2.1b. In this case, the model parameters are the following:

- `Affe`, which stands for the feed-forward loop gain.
- `Afb1`; i.e., the global feedback loop gain.

- `Afb2`; i.e., the local feedback loop gain.
- `Sampling Time`.

Ideal_CT_Resonator

This model corresponds to a CT (biquad) resonator with an S-domain transfer function given by:

$$\text{RTF}(s) = \frac{(\pi/2) \cdot s}{s^2 + (\pi/2)^2} \tag{2.3}$$

The above transfer function is implemented by using the `Transfer Fcn` block from the `Continuous` SIMULINK library.

2.2.3 *Ideal Quantizers*

This library includes several building blocks that model single-bit and multibit/multilevel quantizers intended for both voltage-mode (SC/CT) and current-mode (SI) $\Sigma\Delta$Ms.

Ideal_Comparator

This block models the input/output DC characteristic of an ideal comparator, given by a sign function as,

$$v_o = \begin{cases} V_{\text{high}} & v_i \geq 0 \\ V_{\text{low}} & v_i < 0 \end{cases} \tag{2.4}$$

where v_i and v_o stand for the input and output voltages, respectively, and V_{high} and V_{low} represent the analog values of the logic one and logic zero, respectively.

The model parameters of the `Ideal_Comparator` block are the following:

- `Vhigh,Vlow`, which stand respectively for V_{high} and V_{low}
- `Phase`, which models the clock phase in which the input signal is sampled. Two nonoverlapping clock phases, denoted as `phi1,2` are assumed.
- `Sampling Time`; i.e., the clock signal period.
- `Identifier for this Quantizer`, which defines an identification name for the block (used by some dynamic errors like the incomplete settling error in SC circuits).

Ideal_Comparator_for_SI

This block models a current-mode comparator used in SI-$\Sigma\Delta$Ms. The behavioral model is exactly the same as that used in the `Ideal_Comparator` block, except that in this case the input signal is a current-mode signal that is modeled as a matrix made up of two vector elements, namely: the current signal itself and the output conductance of the current-mode building block (i.e, integrator, resonator, etc) connected at the input of the comparator.

This way, the information provided to the model at each sampling time is a vector of two elements as illustrated in Figure 2.2, where i_i is the input current, v_o is the output voltage, and g_{oi} is the output conductance of the building block connected at the comparator input.

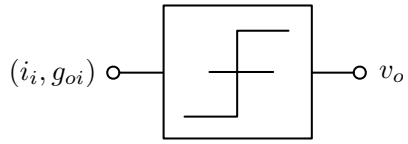


Figure 2.2 Input and output signals in an Ideal_Comparator_for_SI model.

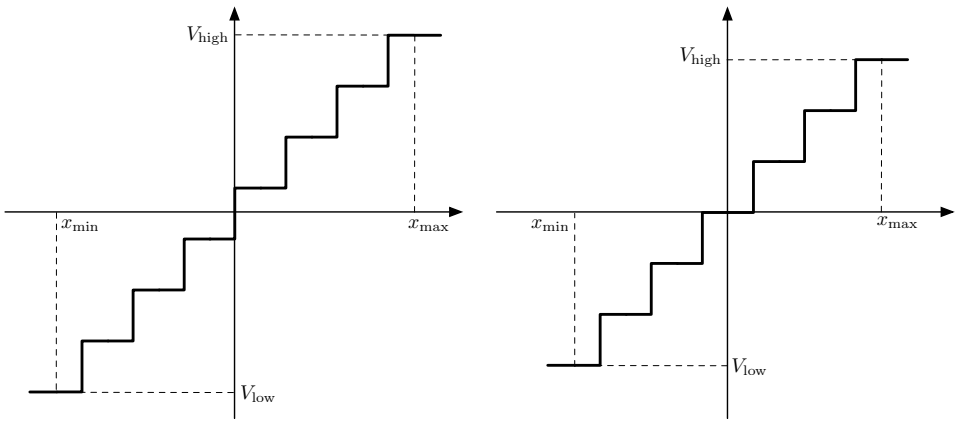


Figure 2.3 Illustrating the input/output DC characteristic of: (a) Multibit (3-bit) *midrise* quantizer. (b) Multilevel (7-level) *midtreat* quantizer.

Ideal_Multibit_Quantizer

This building block includes the ideal behavioral model of a multibit quantizer with a DC input/output *midrise* characteristic as illustrated in Figure 2.3a.

The model parameters of this block are the following ones:

- `Vhigh,Vlow`, which stand respectively for the maximum and minimum values of the quantizer output FS range–see Figure 2.3a.
- `Number of bits` of the quantizer.
- `Xmax-Xmin`, that defines the input FS range of the quantizer.
- `Phase ON`, which is the clock phase in which the input signal is sampled, considering a two-phase clock signal generator.
- `Sampling Time`, that denotes the clock period.

Ideal_Multibit_Quantizer_for_SI

This block models a current-mode multibit quantizer with the same input/output characteristic as an `Ideal_Multibit_Quantizer`, but considering that the input signal has two components–i_i and g_{oi}–in the same way as in Figure 2.2.

Ideal_Multibit_Quantizer_levels

This building block models a multilevel quantizer in which the input/output characteristic is defined as a function of the number of levels, instead of the number of bits as in the `Ideal_Multibit_Quantizer` block. Thus, the same parameters are used, except for the number of bits, which is replaced by `Number of levels`. If this parameter is even, a *midrise* quantization characteristic like that shown in Figure 2.3a is implemented. Otherwise, a *midtreat* characteristic like that depicted in Figure 2.3b is provided.

Ideal_Multibit_Quantizer_levels_SD2

This building block includes the same behavioral model as the previous one, but the output is a thermometric-coded bit array. This is used in combination with multilevel DACs with DEM techniques, as will be illustrated in Section 2.8.

Ideal_Sampler

This building block models an ideal S&H circuit which can be used in CT-$\Sigma\Delta$Ms, in which the signal is sampled at the input of internal quantizers.

Their model parameters are the following:

- `Sampling Time`, that denotes the clock period.
- `Input clock phase`, which defines the sampling clock phase.

2.2.4 Ideal D/A Converters

The blocks included in this library model different kinds of ideal DACs, namely:

- `Ideal_DAC`, which is used for modeling ideal single-bit DACs for either SC- or CT-$\Sigma\Delta$Ms. They are simply modeled as a voltage gain, named `Gain` in the model.
- `Ideal_DAC_for_SI`, which models a single-bit DAC for SI-$\Sigma\Delta$Ms.
- `Ideal_DAC_dig_level_SD2`, which consists of a multilevel ideal DAC.

The last two models and their associated parameters are described below.

Ideal_DAC_for_SI

Figure 2.4 shows the equivalent circuit of the `Ideal_DAC_for_SI` block. It consists of a voltage-controlled current source in parallel with a finite output conductance g_o. The current source i_{DAC} is a sign function of the input voltage v_i given by,

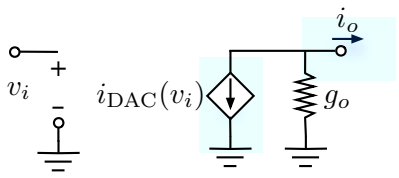


Figure 2.4 Equivalent circuit of Ideal_DAC_for_SI model.

$$i_{\mathrm{DAC}}(v_i) = \begin{cases} +I_{\mathrm{ref}} & v_i = +v_{\mathrm{ref}} \\ -I_{\mathrm{ref}} & v_i = -v_{\mathrm{ref}} \end{cases} \tag{2.5}$$

where I_{ref} and v_{ref} stand for the modulator FS reference current and voltage, respectively. The model parameters included in the `Ideal_DAC_for_SI` model are the following:

- `Gain`, which models the DAC gain; i.e., $I_{\mathrm{ref}}/v_{\mathrm{ref}}$.
- `Gout`, which stands for g_o.
- `Sampling Time`, which models the clock period.
- `Input Clock Phase`; i.e., the clock phase at which the DAC input is sampled.

Ideal_DAC_dig_level_SD2

This block transforms a thermometric-coded digital input into its corresponding analog level. The model parameters used in this behavioral model are:

- `Vhigh`, which defines the upper limit of the quantization FS range.
- `Vlow`, which corresponds to the lower limit of the quantization FS range.
- `Number of levels` of the embedded quantizer.

2.3 Real SC Building-Block Libraries

SIMSIDES includes two libraries of SC integrators and two libraries of SC resonators. These libraries are described below.

2.3.1 Real SC Integrators

There are two SC integrator model libraries in SIMSIDES: one including FE SC integrator models and the other one including LD SC integrators. In both cases, integrator models are classified according to the nonideal effects that are included in the model and the number of SC branches connected at the integrator input. This way, for each model there are four building blocks using the same behavioral model except for the number of input SC branches.

As an illustration, Figure 2.5 shows the symbol used in SIMSIDES for one-branch SC FE integrators (Figure 2.5a) and two-branch SC FE integrators (Figure 2.5b), together with their

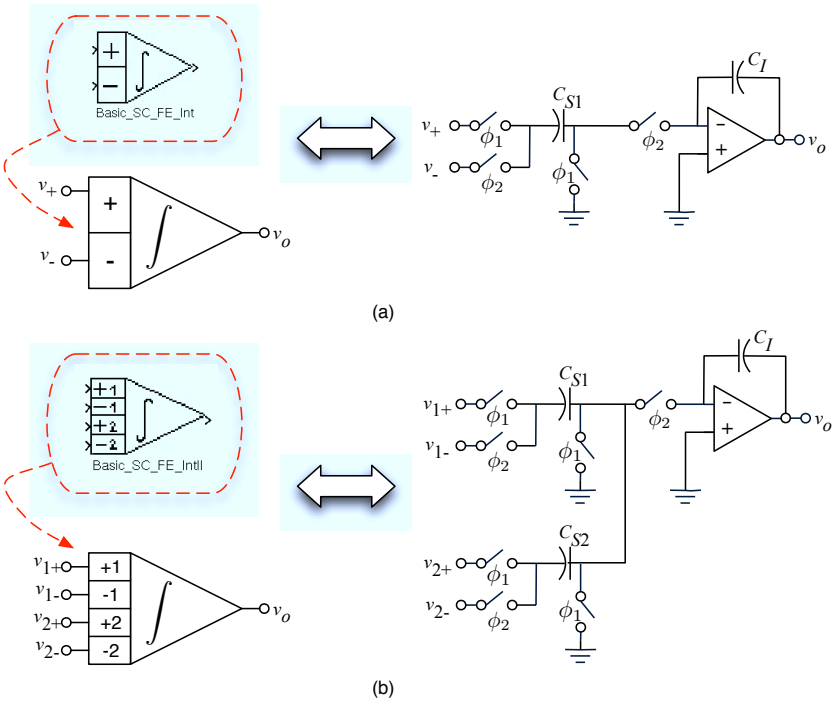


Figure 2.5 SC integrator symbol in SIMSIDES: (a) One-branch integrator. (b) Two-branch integrator.

equivalent SC circuits. Note that, although single-ended conceptual schematics are shown in this figure, fully-differential circuits are assumed in the behavioral models.

Both integrators in Figure 2.5 use the same behavioral model, which consists of an ideal SC FE integrator with output swing limitation. The behavioral model corresponding to a one-branch SC FE integrator is named `Basic_SC_FE_Int`, while the model of the two-branch SC FE integrator is named `Basic_SC_FE_IntII`. Following this nomenclature, `Basic_SC_FE_IntIII` and `Basic_SC_FE_IntIV` models are used for three- and four-branch SC FE integrators, respectively.

Table 2.2 lists all SC integrator models available in SIMSIDES including a brief description of the nonidealities included in each of them. Note that the model names included in Table 2.2 correspond to one-branch integrators. The same models are available for integrators with up to four input branches.

The model named `SC_FE_Int_1b_SD2` represents one-branch SC FE integrators with all circuit nonidealideal effects, including the degradation caused by the switch on-resistance on the integrator GB and SR. In this model, notation `1b` is used for denoting one input SC branch. Similarly, other models whose names include `nb` denote *n* input SC branches.

The model named `SC_FE_Int_1b_DEM_SD2` includes the same nonideal effects as `SC_FE_Int_1b_SD2` model but it also allows to model the sampling capacitor as an array of unit capacitors. This array is used in combination with multilevel quantizers and DACs which incorporate DWA/DEM algorithms as described in [1]. This way, the number

of unit capacitors in the array must be exactly the same as the number of DAC levels. Moreover, this array should include the DAC mismatch error, which can be modeled as a Gaussian distribution. Both `SC_FE_Int_1b_SD2` and `SC_FE_Int_1b_DEM_SD2` models provide detailed information about the integrator equivalent input-referred thermal noise, equivalent load capacitances, and transient response model parameters. All these pieces of information are displayed in the MATLAB command window after simulation.

Table 2.3 lists the most important parameters used by the SC integrator behavioral models in SIMSIDES, as well as a brief description of all of them.

2.3.2 Real SC Resonators

SIMSIDES has two SC resonator model libraries corresponding to FEI-based resonators and LDI-based resonators. All building-block models correspond to the Z-domain block diagrams shown in Figure 2.1 but they are implemented by using the SC integrator models described in the previous section. As an illustration, Figure 2.6 shows the SIMSIDES block diagram of an SC LDI-based resonator (Figure 2.6a) and an SC FEI-based resonator (Figure 2.6b), which correspond to Figures 2.1a and 2.1b, respectively.

Following the same philosophy as that used in SC integrators, the behavioral models of SC resonators in SIMSIDES are classified attending to the number of input SC branches and the circuit nonideal effects included in the models. As an illustration, Figure 2.7 shows an excerpt of both SC resonators libraries. Note that each row of blocks include the same circuit nonideal effects, with the only difference among them being the number of input branches.

Table 2.4 lists all SC resonator models available in SIMSIDES, including a brief description of the nonidealities considered in each of them. The parameters used in these models are the same as those included in SC integrator models–listed in Table 2.3. In addition to these parameters, the resonator gain can also be defined by the user by setting a parameter named `Gain` (see Figure 2.6), which can be defined in the model dialogue box.

2.4 Real SI Building-Block Libraries

SIMSIDES includes all necessary building blocks for the simulation of SI-$\Sigma\Delta$Ms. This section describes SI integrators and resonators, as well as their main model parameters.

2.4.1 Real SI Integrators

Following the same classification criteria as that followed for SC building-block models, there are two libraries of SI integrators in SIMSIDES: one including FE SI integrators and the other one including LD SI integrators. Figure 2.8 shows the symbol used in SIMSIDES for SI integrators together with a conceptual schematic of a FE SI integrator and a LD SI integrator. In both cases, different models are included in SIMSIDES, which are classified attending to the number of nonideal effects that are taken into account, as detailed in Table 2.5.

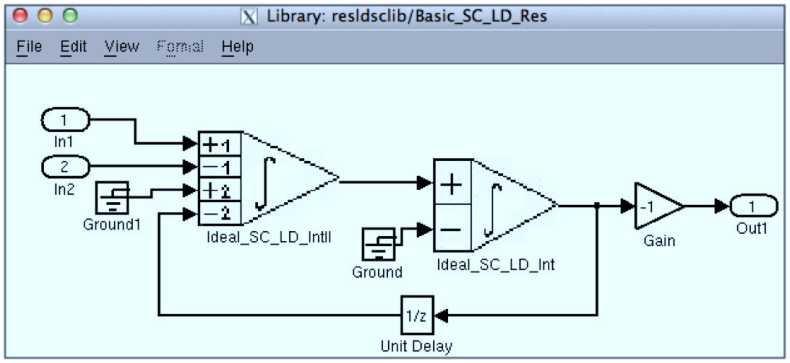
Apart from the building blocks in Table 2.5, a current-mode *buffer* block, named `Buffer`, is included in all SIMSIDES SI libraries. This block transforms an input current vector into an output matrix made up of two vector elements: the input current signal itself in parallel with an output conductance, named `Gout of the source` in the model. Figure 2.9 illustrates

Table 2.2 Library of SC (FE/LD) integrators included in SIMSIDES.

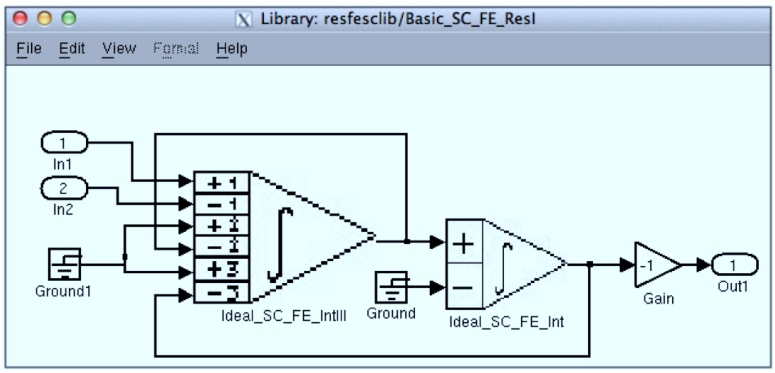
Model name	Circuit effects included
`Basic_SC_FE_Int` `Basic_SC_LD_Int`	Output swing limitation
`SC_FE_Int_Non_linear_C` `SC_LD_Int_Non_linear_C`	Output swing limitation, capacitor nonlinearity.
`SC_FE_Int_Weight_Mismatch` `SC_LD_Int_Weight_Mismatch`	Output swing limitation, capacitor mismatch.
`SC_FE_Int_Non_Linear_Sampling` `SC_LD_Int_Non_Linear_Sampling`	Output swing limitation, nonlinear switch on-resistance.
`SC_FE_Int_FiniteDCgain` `SC_LD_Int_FiniteDCgain`	Finite OTA DC gain, output swing limitation, parasitic OTA caps.
`SC_FE_Int_Finite&Non_LinearDCGain` `SC_LD_Int_Finite&Non_LinearDCGain`	Finite nonlinear OTA DC gain, output swing limitation parasitic OTA caps.
`SC_FE_Int_Noise` `SC_LD_Int_Noise`	OTA thermal noise, output swing limitation, parasitic/load OTA caps.
`SC_FE_Int_Settling` `SC_LD_Int_Settling`	Incomplete settling error, output swing limitation, parasitic/load OTA caps.
`SC_FE_Integrator_All_Effects` `SC_LD_Integrator_All_Effects`	switch on-resistance, capacitor nonlinearity and mismatch, settling error, finite (nonlinear) DC gain, thermal noise, parasitic/load capacitors, output swing limitation.
`SC_FE_Integrator_All_Effects&NonLinSamp` `SC_LD_Integrator_All_Effects&NonLinSamp`	switch nonlinear on-resistance, capacitor nonlinearity and mismatch, settling error, finite (nonlinear) DC gain, thermal noise, parasitic/load capacitors, output swing limitation.
`SC_FE_Int_1b_SD2`	switch on-resistance and its effect on GB and SR, capacitor nonlinearity and mismatch, settling error, finite (nonlinear) DC gain, thermal noise, parasitic/load capacitors, output swing limitation.
`SC_FE_Int_1b_DEM_SD2`	switch on-resistance and its effect on GB and SR, array of unit sampling capacitors, capacitor nonlinearity and mismatch, settling error, finite (nonlinear) DC gain, thermal noise, parasitic/load capacitors, output swing limitation.

Table 2.3 Model parameters used in SIMSIDES SC (FE/LD) integrators.

Parameter name (in alphabetical order)	**Brief description**
`Array of sampling capacitors for DEM branch`	Array of unit capacitors used with multilevel DACs with DEM
`B (switch parameters)`	MOS large-signal transconductance (analytic model)
`Bandwidth (BW)`	Input signal bandwidth
`Capacitor (first/second)-order nonlinearity`	Capacitor (first/second) order nonlinearity
`Finite and Linear Ron`	switch on-resistance, linear model
`Finite DC Gain of the AO`	Finite OTA DC gain
`g (switch parameters)`	Finite switch on-conductance (analytic model)
`Identifier for this integrator`	Identifier used for settling error model
`Input Equivalent Thermal Noise`	OTA input-referred thermal noise
`Input parameters [A,fi,ph] (switch)`	Amplitude, frequency, and phase of the sinewave input (table look-up model)
`Integration/Sampling Capacitor`	Integration/sampling capacitors
`Integration additional load`	Additional load capacitance at the integration phase
`Load Capacitor (cload)`	Integrator load capacitance
`Maximum output current (Io)`	OTA maximum output current
`Nonlinearity of the DC Gain`	OTA DC gain nonlinear coefficients
`Output Swing Up/Down`	Maximum/minimum output swing limits
`Parasitic Capacitor before the AO (Cp)`	Parasitic capacitance at the OTA input
`pcoef (switch parameters)`	nonlinear coefficients of the switch on-resistance (table look-up model)
`Positive Input is Sampled in...`	Input-switch clock phase
`Ron`	switch on-resistance
`Sampling additional load`	Additional load capacitance at the sampling phase
`Sampling Time`	Clock signal period
`Switch on-resistance (Ron)`	Switch on-resistance
`Temp`	Temperature (K)
`Transconductance of the AO (gm)`	OTA transconductance
`Variance`	Variance of the capacitor mismatch error



(a)



(b)

Figure 2.6 SIMSIDES diagram of SC resonators: (a) LDI-based resonator. (b) FEI-based resonator.

the operation of the `Buffer` block, by showing the SIMSIDES block symbol and its equivalent circuit.

2.4.2 Real SI Resonators

As in the case of SC circuits, two different types of SI resonators are modeled in SIMSIDES, namely: SI FEI-based resonators and LDI-based resonators. Both block diagrams are depicted in Figure 2.10. In both cases, integrator gain parameters–denoted as `Affe, Afb1,` and `Afb2` –can be defined by the user in the block dialogue window, as well as their associated gain errors–respectively denoted in the model as `MU1, MU2, MU3`.

Table 2.6 lists all SI resonator blocks included in SIMSIDES together with a brief description of the error mechanisms taken into account in each model. These errors and their associated model parameters, which are the same as those used in SI integrators, are described below.

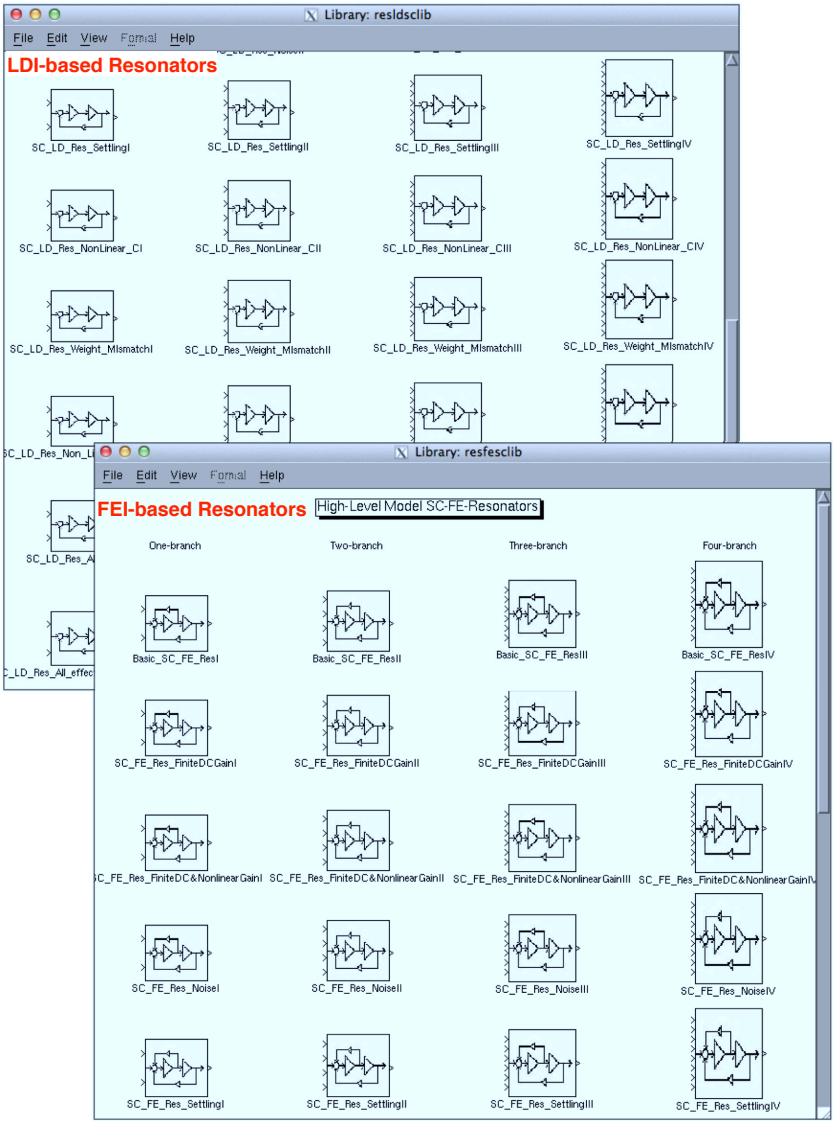


Figure 2.7 Excerpt of SC resonator libraries in SIMSIDES.

2.4.3 SI Errors and Model Parameters

This section gives a brief description of the main errors and model parameters included in SI model libraries of SIMSIDES. For the sake of clarity, these errors are linked to some of the building blocks in which they are included.

Table 2.4 Library of SC (FE/LD) resonators included in SIMSIDES.

Model name	Circuit effects included
`Basic_SC_FE_Res` `Basic_SC_LD_Res`	Output swing limitation
`SC_FE_Res_NonLinear_C` `SC_LD_Res_NonLinear_C`	Output swing limitation, capacitor nonlinearity.
`SC_FE_Res_Weight_Mismatch` `SC_LD_Res_Weight_Mismatch`	Output swing limitation, capacitor mismatch.
`SC_FE_Res_Non_Linear_Sampling` `SC_LD_Res_Non_Linear_Sampling`	Output swing limitation, nonlinear switch on-resistance.
`SC_FE_Res_FiniteDCgain` `SC_LD_Res_FiniteDCgain`	Finite OTA DC gain, output swing limitation, parasitic OTA caps.
`SC_FE_Res_FiniteDC&NonLinearGain` `SC_LD_Res_FiniteDC&NonLinearGain`	Finite nonlinear OTA DC gain, output swing limitation, parasitic OTA caps.
`SC_FE_Res_Noise` `SC_LD_Res_Noise`	OTA thermal noise, output swing limitation, parasitic/load OTA caps.
`SC_FE_Res_Settling` `SC_LD_Res_Settling`	Incomplete settling error, output swing limitation, parasitic/load OTA caps.
`SC_FE_Res_All_effects` `SC_LD_Res_All_effects`	switch on-resistance, capacitor nonlinearity and mismatch, settling error, finite (nonlinear) DC gain, thermal noise, parasitic/load capacitors.
`SC_FE_Res_All_effects&NonLinSamp` `SC_LD_Res_All_effects&NonLinSamp`	switch nonlinear on-resistance, capacitor nonlinearity and mismatch, settling error, finite (nonlinear) DC gain, thermal noise, parasitic/load capacitors, output swing limitation.

Basic_SI_FE(LD)_Integrator and Basic_SI_FE(LD)_Resonator

The model parameters included in these blocks are:

- `Integrator Gain`: integrator weight (see Figure 2.8).
- `Iomax(=-Iomin)`: maximum/minimum integrator output current.
- `MU1,MU3`: first-order and third-order coefficients of the gain nonlinearity.

The Z-domain transfer function of this integrator block is given by the following expression,

$$i_o(z) = (1 - \text{MU1}) \cdot i_{oi}(z) + \text{MU3} \cdot i_{oi}^3(z) \tag{2.6}$$

where $i_{oi}(z)$ is the Z-transform of the ideal output current.

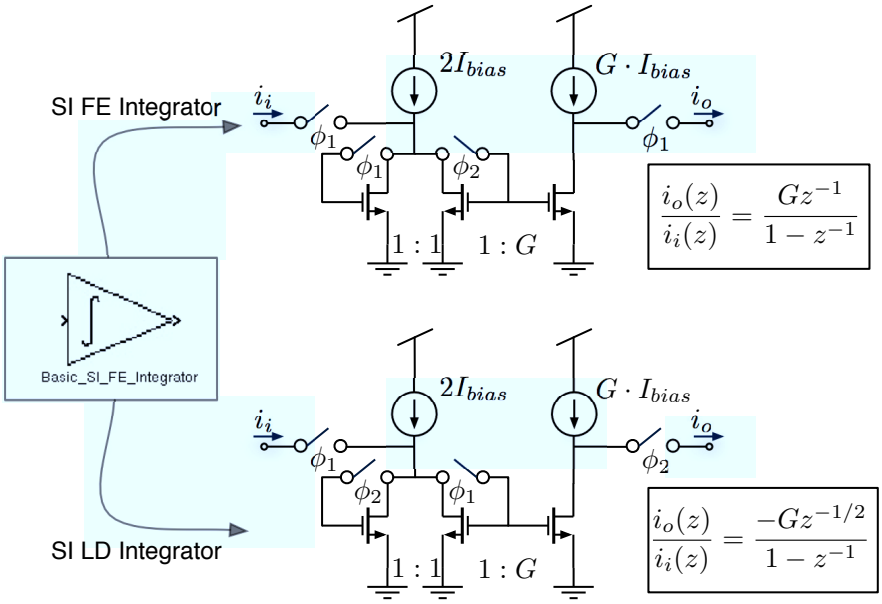


Figure 2.8 SI integrator symbol used in SIMSIDES and its corresponding conceptual schematics.

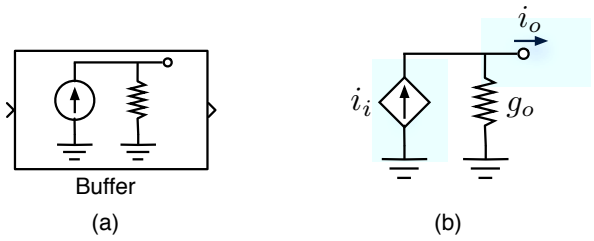


Figure 2.9 SI buffer used in SIMSIDES: (a) Symbol. (b) Equivalent circuit.

The above expression is also used for both LDI- and FEI-based SI resonators, although in this case the nonlinear coefficients are named `SHI1, SHI3` instead of `MU1, MU3`.

SI_FE(LD)_Int_Finite_Conductance

These blocks include the effect of finite input-output conductance ratio error. To this purpose, the equivalent circuit shown in Figure 2.11 is solved during sampling phase. This circuit corresponds to the SI LD integrator[1] of Figure 2.8 on clock phase ϕ_1. In this circuit, the input signal is modeled as an ideal current in parallel with a finite conductance, denoted as g_{oi}. Memory-cell transistor 1 (M1) is configured in the hold phase and is modeled by its drain

[1] A similar circuit is used for SI FE integrators.

Table 2.5 Library of SI (FE/LD) integrators included in SIMSIDES.

Model name	Circuit effects included
`Ideal_SI_FE_Integrator` `Ideal_SI_LD_Integrator`	Ideal SI (FE/LD) integrator.
`Basic_SI_FE_Integrator` `Basic_SI_LD_Integrator`	Output current limits, nonlinear gain.
`SI_FE_Int_Finite_Conductance` `SI_LDI_Finite_Conductance`	Output current limits, finite nonlinear output conductance, input voltage limits, thermal noise.
`SI_FE_Int_Finite_Conductance&Settling` `SI_LDI_Finite_Conductance&Settling`	Output current limits, finite nonlinear input/output conductance, input voltage limits, settling error, thermal noise.
`SI_FE_Int_Finite_...&Settling&Charge_Injection` `SI_LDI_Finite_...&Settling&Charge_Injection`	Output current limits, finite nonlinear input/output conductance, input voltage limits, settling error, thermal noise, charge injection error.

current i_{d1} in parallel with an output conductance g_o. In contrast, memory-cell transistor 2 (M2) is operating in sampling phase and it is modeled by the parallel connection of its output conductance with an input resistor. This resistor is a nonlinear function of the drain current i_{d2} of M2, given by,

$$v_i(i_{d2}) \simeq \text{A1} \cdot i_{d2} + \text{A3} \cdot i_{d2}^3 \tag{2.7}$$

where `A1, A3` stand for the first-order and third-order nonlinear coefficients of the input resistance. These parameters are defined in the models together with the following ones:

- `Gout`: output conductance g_o of memory cells.
- `Vmax/Vmin`: maximum/minimum values of v_i in Equation (2.7).
- `Ibias`: integrator bias current (see Figure 2.8).

SI_FE(LD)_Int_Finite_Conductance&Settling&ChargeInjection

Those SI building blocks including nonlinear incomplete settling and charge injection require the following additional model parameters:

- `Gmo`: operating-point small-signal transconductance of memory transistors.
- `Cgs`: gate-to-source capacitance of memory transistors.

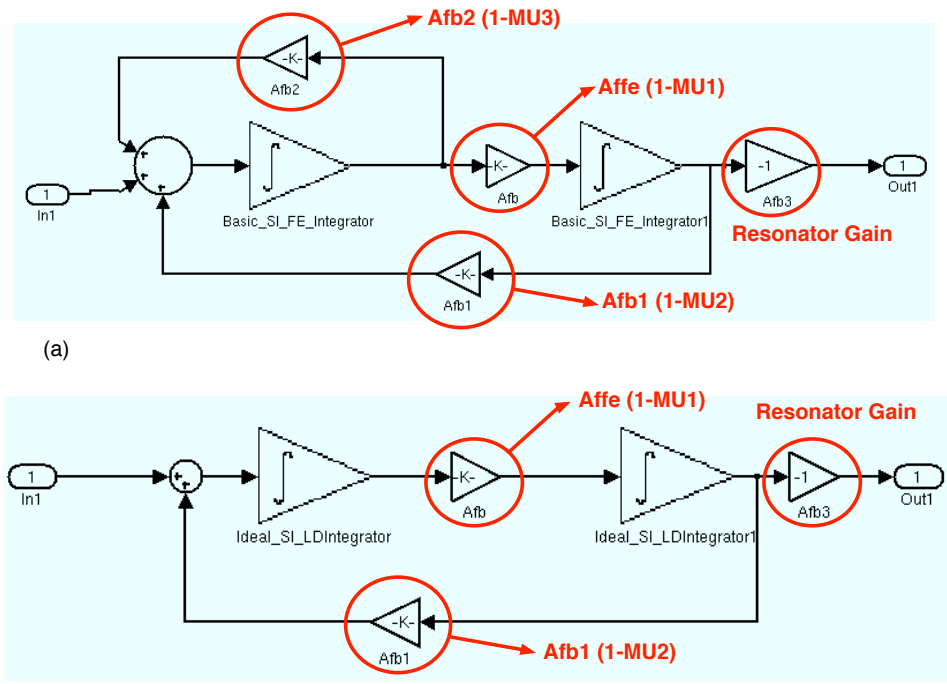


Figure 2.10 SIMSIDES diagram of SI resonators: (a) FEI-loop resonator. (b) LDI-loop resonator.

- `Eq`: charge injection error.

The charge injection error `Eq` is defined as a relative error on the voltage stored in the gate-to-source capacitance, given by:

$$v_{gs,\text{nonideal}} = (1 - \text{Eq}) \cdot v_{gs,\text{ideal}} \tag{2.8}$$

2.5 Real CT Building-Block Libraries

Figure 2.12 shows the CT building-block model libraries included in SIMSIDES. There are four libraries of CT integrators and two libraries of CT resonators, which are classified attending to the circuit nature of the building blocks, namely: Gm-C, Gm-MC, Gm-LC, active-RC, and MOSFET-C.

2.5.1 Real CT Integrators

Tables 2.7–2.10 list all models[2] included in CT integrator libraries shown in Figure 2.12, together with a brief description of the nonideal effects included. An explanation of the

[2]RC_Int_1,2,3in models allow to set up transistor-level parameters such as channel-length modulation, gate-to-source overdrive voltage, saturation voltage, supply voltage, etc.

Table 2.6 Library of SI (FE/LD) resonators included in SIMSIDES.

Model name	Circuit effects included
`Ideal_SI_FE_Resonator` `Ideal_SI_LD_Resonator`	Ideal SI (FE/LD) resonator.
`Basic_SI_FE_Resonator` `Basic_SI_LD_Resonator`	Output current limits, nonlinear gain.
`SI_FE_Res_Finite_Conductance` `SI_LD_Resonator_Finite_Conductance`	Output current limits, finite nonlinear output conductance, input voltage limits, thermal noise.
`SI_FE_Res_Finite_Conductance&Settling` `SI_LD_Res_Finite_Conductance&Settling`	Output current limits, finite nonlinear input/output conductance, input voltage limits, settling error, thermal noise.
`SI_FE_Res_Finite_...&Settling&Charge_Injection` `SI_LD_Res_Finite_...&Settling&Charge_Injection`	Output current limits, finite nonlinear input/output conductance, input voltage limits, settling error, thermal noise, charge injection error.

most representative blocks and their associated model parameters is given in the following subsections.

Model Parameters used in Transconductors and Gm-C Integrator Building Blocks

Building blocks listed in Table 2.7 are used in SIMSIDES to model transconductors and Gm-C integrators, considering the effect of different circuit-level nonideal effects. This section gives a description of the most significant model parameters included in these blocks.

`Transconductance of the OTA` and `Integration Capacitor`, define respectively the transconductance g_m and the integration capacitor C of the Gm-C integrator, whose ITF is obtained by replacing $g = g_m/C$ in Equation (2.1).

`[Upper,Lower] bound saturation voltage`, defines the maximum and minimum values of the output voltage v_o, as illustrated in Figure 2.13.

`Input Voltage [Upper, Lower] saturation limit`, defines the maximum and minimum values of the input voltage, respectively denoted as $v_{i\text{max}}$ and $v_{i\text{min}}$ in Figure 2.13.

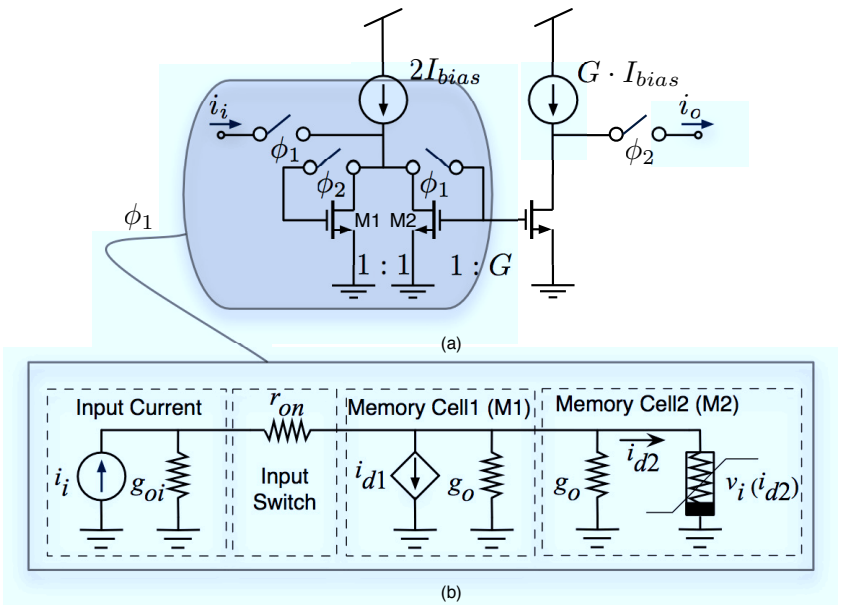


Figure 2.11 Modeling finite input-output conductance ratio error in SIMSIDES: (a) SI LD integrator. (b) Equivalent circuit during sampling phase (ϕ_1).

Table 2.7 Gm-C integrator library models in SIMSIDES.

Model name	Circuit effects included
`Ideal_OTA_C_CTint`	Ideal Gm-C integrator.
`Transconductor`	Input saturation voltage, nonlinear transconductance.
`gm_no_noise_new`	Output saturation voltage, third-order intercept point.
`1pole_gm`	Gm-C output impedance.
`OTA_C_CT_1pole`	Input/output saturation voltage, finite OTA DC gain, nonlinear transconductance, one-pole dynamic, time-constant error, nonlinear transconductance, thermal noise.
`OTA_C_CT_2poles` `OTA_C_CT_2polesb`	Input/output saturation voltage, Finite OTA DC gain, nonlinear transconductance, two-pole dynamic, time-constant error, nonlinear transconductance, thermal noise.

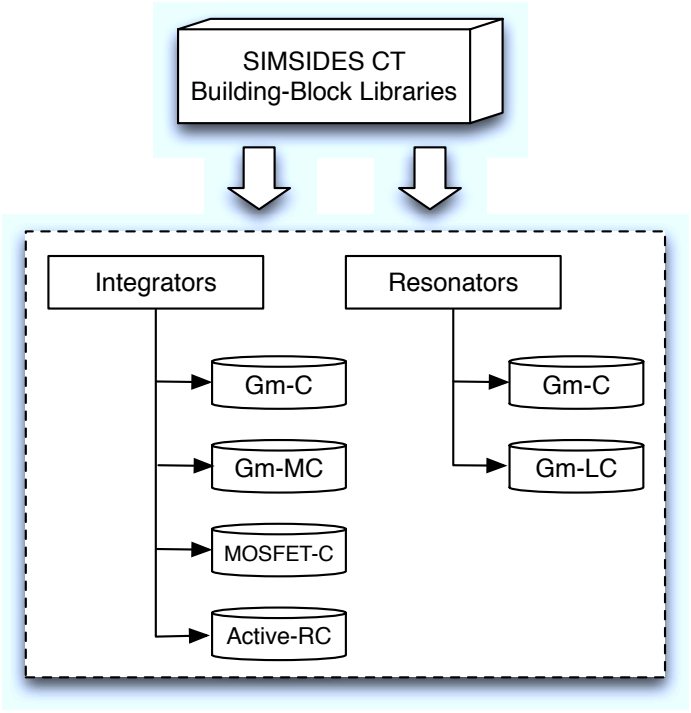


Figure 2.12 Classification of SIMSIDES CT model libraries.

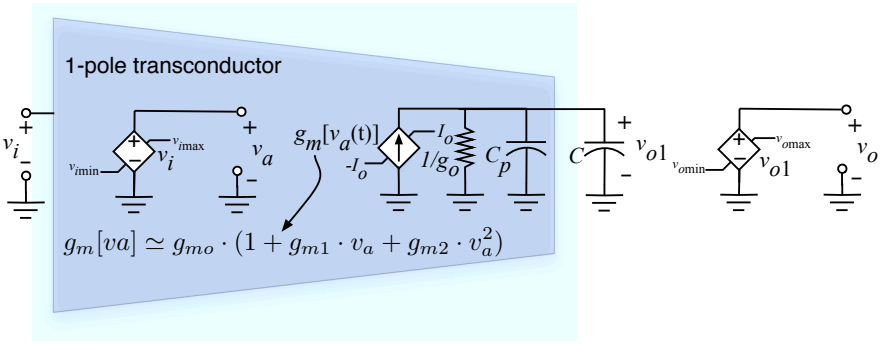


Figure 2.13 One-pole Gm-C integrator model used in SIMSIDES.

Table 2.8 Gm-MC integrator library models in SIMSIDES.

Model name	Circuit effects included
`Gm_MC_CTInt_1pole`	Input/output saturation voltage, finite OTA DC gain, parasitic capacitances, one-pole dynamic, thermal noise.
`Gm_MC_CTInt_2poles`	Input/output saturation voltage, finite OTA DC gain, parasitic capacitances, two-pole dynamic, thermal noise.
`Gm...1pole&Large_signal_distortion`	Input/output saturation voltage, output current limit, finite OTA DC gain, parasitic capacitances, one-pole dynamic.
`Gm...2poles&Large_signal_distortion`	Input/output saturation voltage, output current limit, finite OTA DC gain, parasitic capacitances, two-pole dynamic.
`Gm...1pole&Small_Signal_Distortion`	Input/output saturation voltage, output current limit, finite OTA DC gain, nonlinear transconductance, parasitic capacitances, one-pole dynamic.
`Gm...2poles&Small_Signal_Distortion`	Input/output saturation voltage, output current limit, finite OTA DC gain, nonlinear transconductance, parasitic capacitances, two-pole dynamic.

`[Second, Third] order distortion coefficient`, stand for the second- and third-order nonlinear transconductance coefficients $g_{m(1,2)}$, where it is assumed that the transconductance depends on the Gm-C integrator input voltage v_i as,

$$g_m \simeq g_{mo} \cdot (1 + g_{m1} \cdot v_i + g_{m2} \cdot v_i^2) \tag{2.9}$$

with g_{mo} being the nominal value of the transconductance. The maximum current provided by the transconductor is defined as I_o.

`DC voltage gain`, stands for the finite OTA DC gain, defined as g_m/g_o, with g_o being the output conductance of the Gm-C integrator.

`Integration constant time error`, defined as C_p/C, with C_p being the parasitic capacitance at the output of the Gm-C integrator.

`High Frequency pole`, which defines the value of the high-frequency pole when a two-pole dynamic model is considered.

Table 2.9 Active-RC integrator library models in SIMSIDES.

Model name	Circuit effects included
`RC_CTInt_1pole`	OTA output swing limitation, finite OTA DC gain, parasitic capacitances, capacitance voltage coefficient, one-pole dynamic, thermal noise.
`RC_CTInt_2poles`	OTA output swing limitation, finite OTA DC gain, parasitic capacitances, capacitance voltage coefficient, two-pole dynamic, thermal noise.
`RC...1pole&Large_signal_distortion`	OTA output swing limitation, output current limit, finite OTA DC gain, parasitic capacitances, capacitance voltage coefficient, one-pole dynamic, thermal noise.
`RC...2poles&Large_signal_distortion`	OTA output swing limitation, output current limit, finite OTA DC gain, parasitic capacitances, capacitance voltage coefficient, two-pole dynamic, thermal noise.

`RC_Int_1in`
`RC_Int_2in`
`RC_Int_3in`
	OTA output swing limitation, finite OTA DC gain, nonlinear trans., slew rate, parasitic capacitances, one-pole dynamic, thermal noise.

Table 2.10 MOSFET-C integrator library models in SIMSIDES.

Model name	Circuit effects included
`MOSFET_C_CTInt_1pole`	OTA output swing limitation, finite OTA DC gain, parasitic capacitances, capacitance voltage coefficient, one-pole dynamic, thermal noise.
`MOSFET_C_CTInt_2poles`	OTA output swing limitation, finite OTA DC gain, parasitic capacitances, capacitance voltage coefficient, two-pole dynamic, thermal noise.
`MOS...1pole&Large_signal_distortion`	OTA output swing limitation, output current limit, finite OTA DC gain, parasitic capacitances one-pole dynamic, thermal noise.
`MOS...2poles&Large_signal_distortion`	OTA output swing limitation, output current limit, finite OTA DC gain, parasitic capacitances two-pole dynamic, thermal noise.

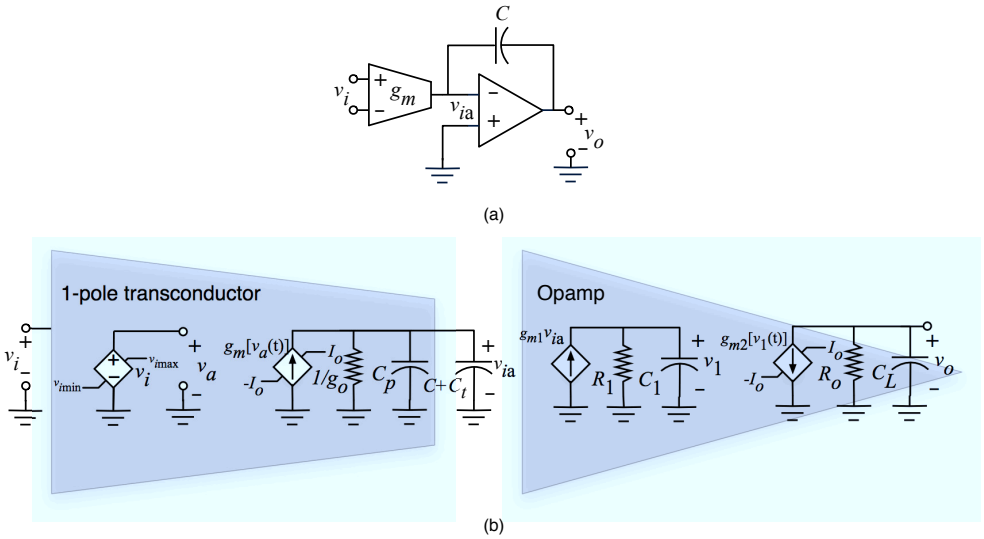


Figure 2.14 Two-pole Gm-MC integrator model used in SIMSIDES: (a) Conceptual schematic. (b) Equivalent circuit of the transconductor and the opamp.

Gm-MC Integrators

SIMSIDES contains a library of Gm-C integrators based on the connection of a transconductance element and a Miller capacitance, also referred to as Gm-MC integrators. Figure 2.14a shows the conceptual schematic of the Gm-MC integrators modeled in SIMSIDES. Several models which account the effect of different circuit nonidealities are included in this library. The most accurate one, named `Gm_MC_CTInt_1pole&Small_Signal_Distortion`, is modeled by the equivalent circuit shown in Figure 2.14b, which involves a two-pole dynamic model. In addition to the model parameters included in Gm-C integrators, Gm-MC integrator models use the additional parameters described below.

`Output transconductor parasitic, Integration capacitor ratio (Cp/C)` , which stands for C_P/C (see Figure 2.14b).

`Output Operational parasitic, Integration capacitor ratio (CI/C)` ;i.e., C_L/C.

`Operational parasitic Output, OTA output capacitor ratio (Ct/C)` ; i.e., C_t/C.

`Transconductor and Op. Amplifier Unity gain frequency (Hz) [Gb1, Gb2]`, which stand for the GB of the transconductor and for the opamp in Figure 2.14b, respectively.

`High Frequency pole`, which defines the value of the high-frequency (nondominant) pole when a two-pole dynamic is considered–given by $1/(R_1C_1)$ in Figure 2.14b.

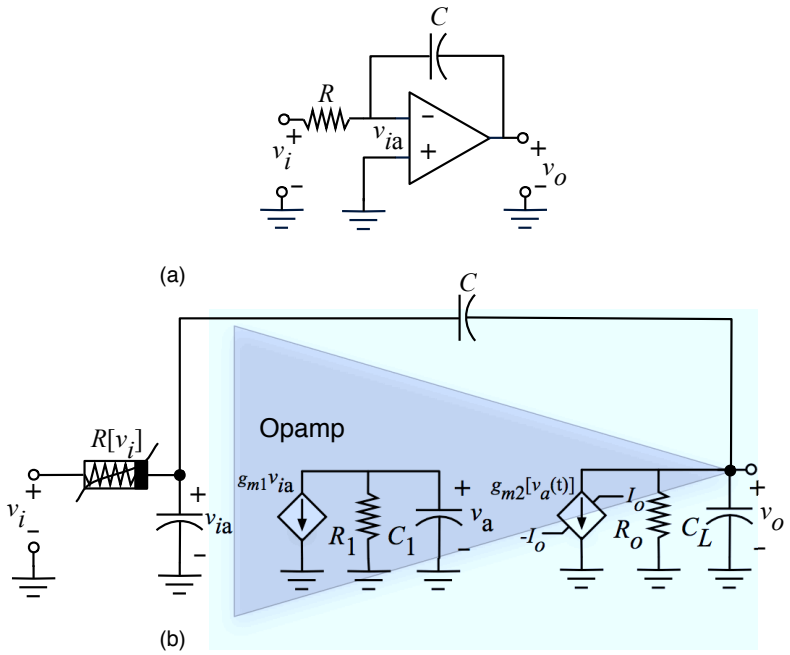


Figure 2.15 Two-pole active-RC integrator model used in SIMSIDES: (a) Conceptual schematic. (b) Equivalent circuit.

`Origin transconductance`, which refers to the operating-point transconductance g_{mo} of the nonlinear characteristic given in Equation (2.9).

Active-RC Integrators

Among all active-RC integrator models listed in Table 2.9, the most accurate and complete one is named `RC_CTInt_2poles&Large_signal_distortio`. Figure 2.15 shows the conceptual schematic (Figure 2.15a) and its corresponding equivalent model (Figure 2.15b). There are two versions of this model: one based on a linear input resistance R and another in which R is a nonlinear function of the input voltage given by,

$$R(v_i) \simeq R \cdot (1 + R_1 \cdot v_i + R_2 \cdot v_i^2) \tag{2.10}$$

where $R_{(1,2)}$ stand for the first- and second-order nonlinear coefficients.

Apart from the model parameters used by the CT building blocks described in previous sections, the following parameters are used in active-RC integrator models:

`Output resistance - integration resistance ratio`, which stands for R_o/R in Figure 2.15b.

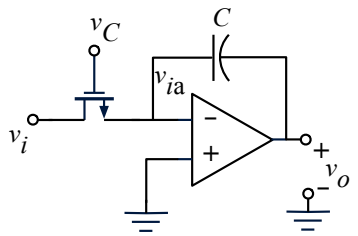


Figure 2.16 Conceptual schematic of a MOSFET-C integrator like that modeled in SIMSIDES by the building blocks listed in Table 2.10.

`Integrator Ideal Unity gain frequency (RC)`, defined as $1/(RC)$.

`Opamp ideal Unity gain frequency`, i.e. the GB of the opamp in Figure 2.15b.

`High frequency pole (Hz)`, defined as $1/(R_1C_1)$ (see Figure 2.15b).

MOSFET-C Integrators

In addition to active-RC integrator models, SIMSIDES includes also a library of MOSFET-C integrators whose conceptual schematic is shown in Figure 2.16. Essentially, these building blocks are the same as those used for modeling active-RC integrators, except that the integrator resistance R is replaced by a MOSFET transistor.

2.5.2 Real CT Resonators

Tables 2.11 and 2.12 list all models included in CT resonator libraries shown in Figure 2.12, together with a brief description of their nonideal effects. These libraries include different building blocks that are classified according to the accuracy of their models as well as to the circuit nonidealities that are taken into account. As an illustration, Figure 2.17 depicts both CT resonator libraries included in SIMSIDES, namely: Gm-C resonators and Gm-LC resonators. Figure 2.18 shows the conceptual schematic of both kinds of CT resonators. The model parameters involved in the building blocks shown in Figure 2.17 are detailed below.

Gm-C Resonators

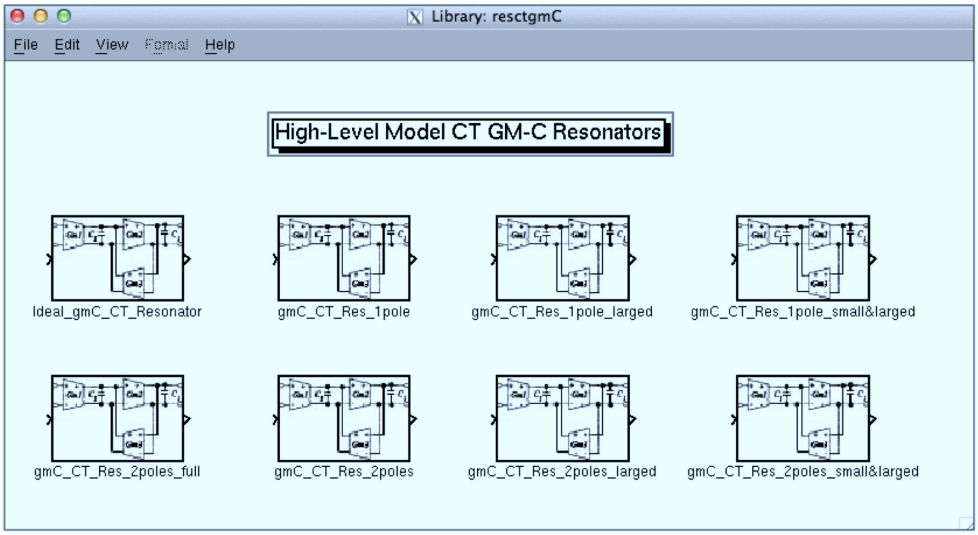
Apart from those parameters used in Gm-C integrators, the following model parameters are used in the Gm-C resonator model library:

`Transconductance of the first, second, and third OTA (gm1,gm2,gm3)`; i.e., g_{m1}, g_{m2}, and g_{m3} in Figure 2.18a.

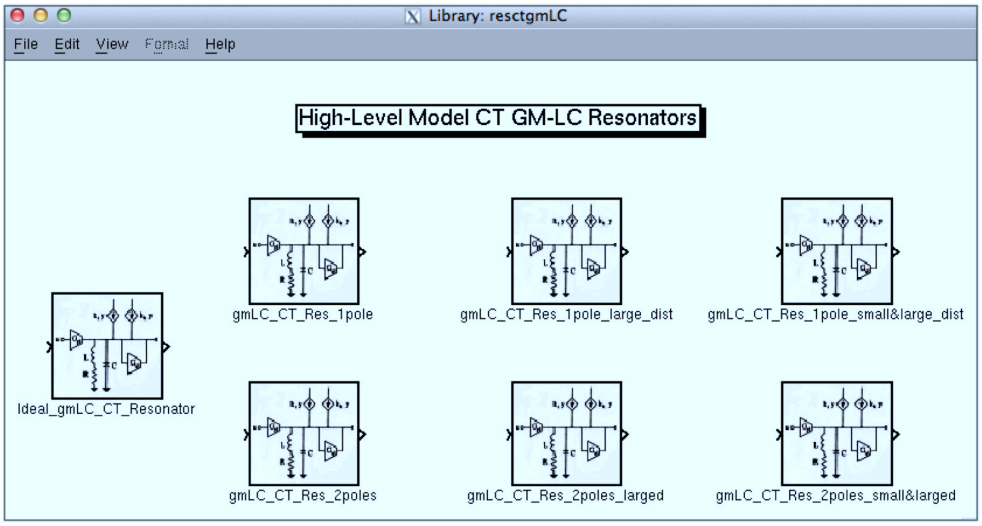
`Capacitors`; i.e., C_1, C_2 in Figure 2.18a.

Table 2.11 Gm-C resonator library models in SIMSIDES.

Model name	Circuit effects included
`Ideal_gmC_CT_Resonator`	Ideal Gm-C resonator.
`gmC_CT_Res_1pole`	Finite OTA DC gain, time-constant error, one-pole dynamic, thermal noise.
`gmC_CT_Res_2poles` `gmC_CT_Res_2polesfull`	Finite OTA DC gain, time-constant error, two-pole dynamic, thermal noise.
`gmC_CT_Res_1pole_larged`	Input/output saturation voltage, output current limit, finite OTA DC gain, time-constant error, one-pole dynamic.
`gmC_CT_Res_2poles_larged`	Input/output saturation voltage, output current limit, finite OTA DC gain, nonlinear transconductance, time-constant error, two-pole dynamic.
`gmC_CT_Res_1pole_small&larged`	Input/output saturation voltage, output current limit, finite OTA DC gain, nonlinear transconductance, time-constant error, one-pole dynamic.
`gmC_CT_Res_2poles_small&larged`	Input/output saturation voltage, output current limit, finite OTA DC gain, nonlinear transconductance, time-constant error, two-pole dynamic.



(a)



(b)

Figure 2.17 CT resonator libraries included in SIMSIDES: (a) Gm-C resonators. (b) Gm-LC resonators.

`Nonlinear transconductance coefficients [gmnl1, gmnl2]`, which stand for nonlinear coefficients $g_{m(1,2)}$ in Equation (2.9).

`DC Gain of OTAs [Av1,Av2,Av3]`, which defines the finite OTA DC gain of transconductors in Figure 2.18a.

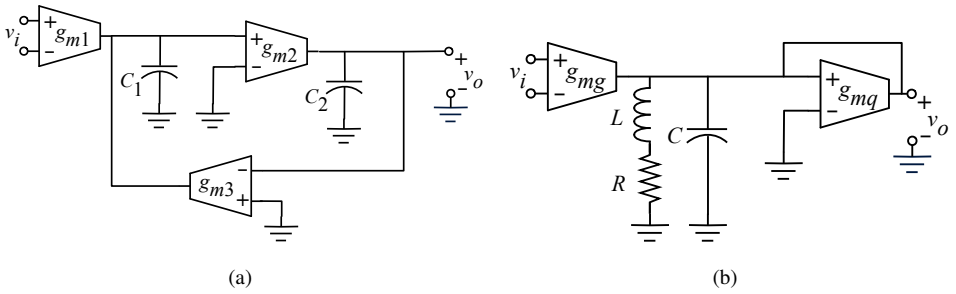


Figure 2.18 Conceptual schematics of the CT resonators modeled in SIMSIDES: (a) Gm-C resonator. (b) Gm-LC resonator.

`Percentual integration constant time error [et1, et2]`; i.e., the time constant errors associated to both feed-forward transconductances in Figure 2.18a, given by,

$$\epsilon_{t1} = \frac{C_{p1} + C_{p3}}{C_1} \quad , \quad \epsilon_{t2} = \frac{C_{p2}}{C_2} \tag{2.11}$$

where C_{pi} stands for the parasitic capacitance of the ith transconductance in Figure 2.18a.

Gm-LC Resonators

The main model parameters used by Gm-LC resonator blocks are described in this section. Essentially, these parameters deal with the inductor element in Figure 2.18b and its associated resonant frequency.

`Frequency resonance`; i.e., the resonant frequency of the Gm-LC resonator.

`Inductor Q`; i.e., the Q-factor of the inductor in Figure 2.18b.

`Series Resistance`; i.e., the parasitic resistance R of the inductor.

The rest of the model parameters used in Gm-C resonators have the same meaning as those used in Gm-C integrators and resonators.

2.6 Real Quantizers & Comparators

Table 2.13 lists the blocks included in the real `Quantizers&Comparators` SIMSIDES library, together with a brief description of their operation and main circuit nonidealities. In addition to the ideal parameters described in Section 2.2.3, additional model parameters are required to model the different circuit nonidealities. These error parameters are listed in Table 2.14.

Note that, apart from comparators and quantizers, there is a building block named `Real_Sampler`, which is used for modeling the S&H circuits that are connected at the input of embedded quantizers in CT-$\Sigma\Delta$Ms. One of the most critical errors associated to

Table 2.12 Gm-LC resonator library models in SIMSIDES.

Model name	Circuit effects included
`Ideal_gmLC_CT_Resonator`	Ideal Gm-LC resonator.
`gmLC_CT_Res_1pole`	Input/output saturation voltage, inductance quality factor and series parasitic resistance, finite OTA DC gain, time-constant error, one-pole dynamic, thermal noise.
`gmLC_CT_Res_2poles`	Input/output saturation voltage, inductance quality factor and series parasitic resistance, finite OTA DC gain, time-constant error, two-pole dynamic, thermal noise.
`gmLC...1pole_large_dist`	Input/output saturation voltage, inductance quality factor and series parasitic resistance, output current limitation, finite OTA DC gain, time-constant error, one-pole dynamic, thermal noise.
`gmLC...2poles_large_dist`	Input/output saturation voltage, inductance quality factor and series parasitic resistance, output current limitation, finite OTA DC gain, time-constant error, two-pole dynamic, thermal noise.
`gm...1pole_small&large_dist`	Input/output saturation voltage, inductance quality factor and series parasitic resistance, output current limitation, nonlinear transconductance, finite OTA DC gain, time-constant error, one-pole dynamic, thermal noise.
`gmLC...2poles_small&larged`	Input/output saturation voltage, inductance quality factor and series parasitic resistance, output current limitation, nonlinear transconductance, finite OTA DC gain, time-constant error, two-pole dynamic, thermal noise.

this building block is the clock jitter, which is modeled as an uncertainty in the sampling time δt corresponding to a stationary process with zero mean and standard deviation defined by the user (see Table 2.14).

2.7 Real D/A Converters

Table 2.15 lists the different building blocks included in the real `D/A Converters` SIMSIDES library, together with a brief description of their operation and main circuit errors.

Error parameters associated to the models listed in Table 2.15 have the same meaning as those used in multibit quantizers, except for the selectable NRZ/RZ/HRZ DAC waveform and the delay error. The latter can be chosen to be either a constant delay or a signal-dependent

delay, given by,

$$\text{delay}(v_i) = \text{d0} + \frac{\text{d1}}{\text{x1} \cdot |v_i|} < \text{dmax} \tag{2.12}$$

where `d0, d1, x1, dmax` are model parameters set by the user.

2.8 Auxiliary Blocks

In addition to the building blocks described in previous sections, SIMSIDES includes a library named `Auxiliary blocks` that contains some other blocks (like adders, DEM algorithms, and digital latches) also needed to simulate ΣΔMs. Table 2.16 lists the models included in the mentioned library together with a brief description of their operation. The most significant parameters used by these models are listed in Table 2.17.

As an illustration on the use of some of the auxiliary blocks, Figure 2.19a shows the SIMSIDES block diagram of a second-order feed-forward SC-ΣΔM, which includes an embedded 16-level quantizer and DAC with a selectable DEM algorithm.

The block diagram in Figure 2.19a includes the following building blocks:

- `Mux_SD2`. This block samples the modulator input signal in a number of unit capacitors, which corresponds to the number of unit elements used in the multilevel DAC minus one. As shown in Figure 2.19b, the only model parameter of this block is named `Number of elements`, which equals 15 in this example.
- `SC_FE_Int_All_Effects_DEM_SD2`. This block is used for modeling the front-end integrator and includes all error mechanisms with values defined in the

Table 2.13 Real Quantizers and Comparator models included in SIMSIDES.

Model name	Circuit effects included
`Real_Comparator_Offset&Hysteresis`	Voltage-mode comparator with offset, (random & deterministic) hysteresis.
`Real_Comparator_Offset&Hysteresis_for_SI`	Current-mode comparator with offset and nonlinearity (INL).
`Real_Multibit_Quantizer`	Voltage-mode multibit quantizer with gain error, offset (random & deterministic) hysteresis.
`Real_Multibit_Quantizer_for_SI`	Current-mode multibit quantizer with gain error, offset, INL, (random & deterministic) hysteresis.
`Real_Multibit_Quantizer_dig_level_SD2`	Voltage-mode multilevel quantizer with gain error, offset, INL, (random & deterministic) hysteresis.
`Real_Sampler`	Sampling & Hold circuit with clock jitter error.

Table 2.14 Error model parameters used in SIMSIDES Real Quantizers.

Parameter name (in alphabetical order)	Brief description
`Gain Error in LSB`	Gain error measured in LSB.
`Jitter typical deviation`	Standard deviation of clock jitter error.
`Kind of Hysteresis`	Comparator hysteresis. It may be either deterministic or random hysteresis.
`INL in LSB`	Integral Nonlinearity error measured in LSB.
`Number of levels`	Number of quantizer levels.
`Offset`	Offset error.
`Offset Error in LSB`	Offset error measured in LSB.
`Seed for random jitter generation`	Seed number used for generating random jitter error.

block dialogue box in Figure 2.19c. This model includes also a parameter named `Array of sampling capacitors for DEM branch`. The number of capacitors in the array must be exactly the same as the number of DAC levels; i.e., the number of quantization levels. As an illustration, Figure 2.20 shows the MATLAB code used for generating different alternative capacitor arrays together with other parameters used for simulating the block diagram in Figure 2.19a. Note that this capacitor array must include also the DAC element mismatch, defined as a Gaussian probability distribution.

- `Basic_SC_FE_Int`. This block is used for modeling the second integrator, considering only ideal values of the output swing and capacitors, as shown in Figure 2.19d.
- `Real_Multibit_Quantizer_dig_level_SD2` . This block, named `Q1` in the example, is used for modeling the quantizer considering the nonideal effects listed in Table 2.13. In this example, these nonideal effects have not been taken into account. The output of this block is a thermometric-coded bit array. This array is transformed into an analog signal for further processing by using the block named `ThermometricCode-to-Analog`.
- `DAC-DEM`. This block uses the `DAC - DEM - V04` model (see Table 2.16). The most important model parameters of these block are highlighted in Figure 2.19e.

2.9 Frequency/Time-based ΣΔM Building Blocks

The last version of SIMSIDES includes also a library of basic blocks – named `FreqTimeBasedSDMBuildingBlocksLib` – that contains basic building blocks required to model and simulate ΣΔMs based on the use of frequency/time enconded subcircuits, such as Voltage Controlled Oscillators (VCOs) or Pulse-Width Modulators (PWMs). The library – illustrated in Figure 2.21 – have a number of components which are built using

Table 2.15 Real DAC models included in SIMSIDES.

Model name	Circuit effects included
`Real_DAC_Multibit`	Voltage-mode multibit DAC with offset, gain error, and INL error.
`Real_DAC_Multibit_SI`	Current-mode multibit DAC with offset, gain error, and INL error.
`Real_DAC_Multibit_delay_Jitter`	Voltage-mode multibit DAC with offset error, gain error, INL error, delay error, and clock jitter error.
`Real_DAC_Multibit_delay_Jitter_SI`	Current-mode multibit DAC with finite output conductance, offset error, gain error, INL error, delay error, and clock jitter error.
`Real_DAC_pulse_types`	Voltage-mode multibit DAC with selectable NRZ/RZ/HRZ output waveform.
`Real_DAC_Multibit_pulse_types`	Voltage-mode multibit DAC with selectable NRZ/RZ/HRZ output waveform, gain error, offset error, and INL error.
`Real_DAC..._delay_jitter`	Voltage-mode multibit DAC with selectable NRZ/RZ/HRZ output waveform, gain error, offset error, INL error, delay error, and clock jitter error.

SIMULINK standard library blocks[3], including different kinds of ring-oscillators apart from VCOs, such as Gated switched-Ring Oscillators (GROs) as well as other blocks needed to build time/frequency-based $\Sigma\Delta$Ms like signal generators, digital counters, etc. A description of all these blocks can be found in the MATLAB/SIMULINK documentation. Nevertheless, some VCO/GRO-based $\Sigma\Delta$M examples are included in SIMSIDES in order to illustrate how to use the blocks in this library.

References

[1] J. M. de la Rosa, *Sigma-Delta Converters: Practical Design Guide (2nd Edition).* Wiley-IEEE Press, 2018.

[3]It is expected that future versions of SIMSIDES will replace some of these blocks by S-functions in order to speed up the simulation time.

Table 2.16 Auxiliary building-block models used in SIMSIDES.

ANALOG ADDERS	
Model name	**Brief description**
`Analog_Adder_Ideal_SD2`	Ideal SC passive adder with parasitic input capacitance and load capacitance.
`Analog_Adder_real_SD2`	Real SC passive adder with parasitic input capacitance and load capacitance, switch on-resistance, settling error, capacitor nonlinearity, and thermal noise.
DIGITAL ADDERS	
Model name	**Brief description**

`Dig_add_generic_2outs`
`Dig_add_3L_5L_13L`
`Dig_add_3L_3L_5L_2outs`
`Dig_add_3L_3L_7L_2outs`
`Dig_add_3L_5L_9L_2outs`
`Dig_add_3L_5L_13L_2outs`
	Digital subtraction of a M_1-Level thermometric-coded signal and a M_2-Level thermometric-coded signal, which is scaled by a factor of `d`. The result is a $(M_1 + M_2/\mathrm{d})$-level thermometric-coded digital output.

DIGITAL LATCHES	
Model name	**Brief description**
`D_latch_simplest` `D_latch`	Digital "D" latches.
DAC WITH DEM ALGORITHMS	
Model name	**Brief description**
`DEM_id_SD2`	Ideal DEM algorithm.
`DAC - DEM - V04`	DAC block with a selectable DEM algorithm. There are three options: No DEM, DWA, Pseudo-DWA.
`Mux_SD2`	Building block used for sampling an input (analog) signal by a number of different branches corresponding to the number of DAC unit capacitors.

Table 2.17 Error model parameters used in SIMSIDES Auxiliary Blocks.

Parameter name (in alphabetical order)	**Brief description**
`Comparator Input Capacitor (C)`	Parasitic capacitance at the comparator/quantizer input.
`DEM type`	DEM algorithm: (1) No DEM, (2) DWA, (3) Pseudo-DWA. (Default = 1)
`Input Capacitor (C)`	Input capacitance of the analog adder.
`Nonlinearities of the capacitors`	Capacitance nonlinear coefficients in an analog adder.
`Number of elements`	Number of DAC unit elements.
`Output type`	Digital output code: (1) Binary output, (2) Trilevel output including common mode (Default = 1).
`Time interval between sampling and comparison (delta)`	Delay between the time instant when the adding operation is performed and the time instant when comparison time takes place.

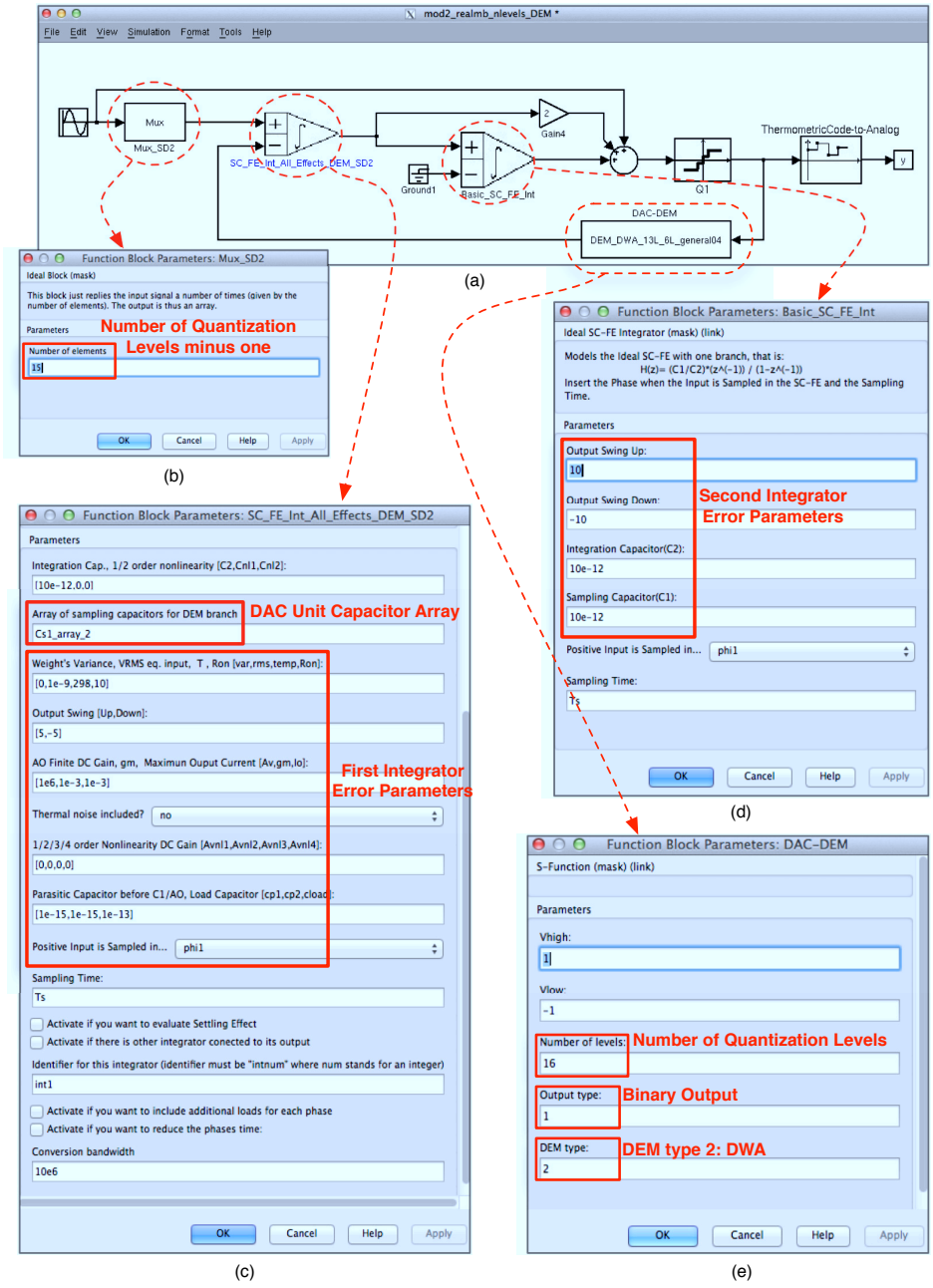


Figure 2.19 Example of a second-order feed-forward SC-ΣΔM with 16-level quantization and DEM: (a) SIMSIDES block diagram. (b) Mux_SD2 dialogue box. (c) First integrator block dialogue box. (d) Second integrator block dialogue box. (e) DAC-DEM block dialogue box.

```
%%%%%%%%%% Modulator: simulation parameters %%%%%

fs=35.2e6; % Sampling frequency
Ts=1/fs; % Sampling time
fi=137.5e3; % Input frequency
OSR=64; % OverSampling Ratio
N=65536; % Number of Points

%%%%%%%%%% DEM Parameters and Capacitor Feedback Arrays %%%%%%%%%%

nlevels=16; % Number of quantization levels
Cs1=10e-12; % Sampling Capacitor
sigma=[1/100]; % Capacitor mismatch standard deviation

%% Capacitor array

n_elements_1=(nlevels-1);
% Binary array
n_elements_2=floor((nlevels)/2);
% Tri-level array
Cs1_array_1=Cs1/n_elements_1*(ones(1,n_elements_1));
% No mismatch - binary
Cs1_array_2=Cs1/
n_elements_1*(ones(1,n_elements_1)+sigma*randn(1,n_elements_1));
% With mismatch - binary
Cs1_array_3=Cs1/
n_elements_2*(ones(1,n_elements_2)+sigma*randn(1,n_elements_2));
% With mismatch - tri_level
Cs1_array_4=Cs1/n_elements_2*(ones(1,n_elements_2));
% No mismatch - tri_level
```

Figure 2.20 MATLAB code used for defining capacitor arrays and DEM parameters of Figure 2.19e.

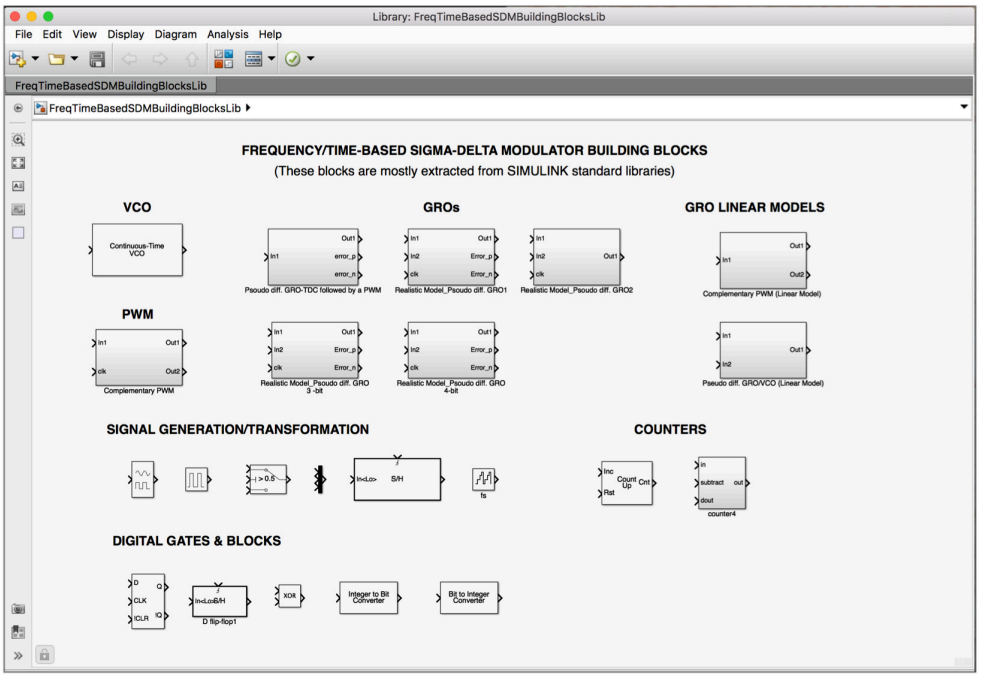


Figure 2.21 Library of frequency/time-based ΣΔM block models included in SIMSIDES.