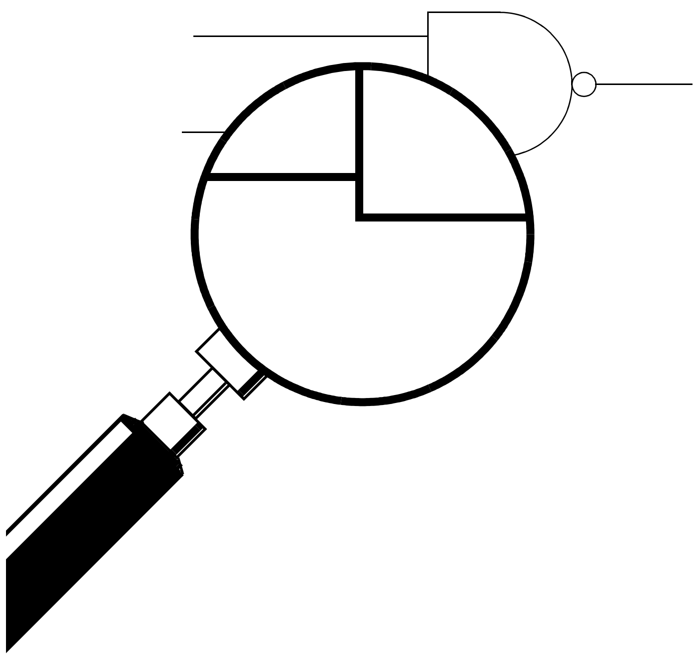
# Gendel User's Guide



Genashor Corp.
17 Dogwood Drive
Hillsborough, New Jersey 08844
Phone: (908) 369-8554

Genashor Corp
17 Dogwood Drive
Hillsborough, NJ 08844-2517
Telephone: (908) 369-8554

# Gendel Users Guide

## Overview

One of the most complicated and time-consuming tasks in generating simulation models for a library of cells is timing characterization. A cell may be asked to operate with a wide range of output loading, and input slew-rate conditions in a design. It requires a large number of simulations for each cell in the library to capture this information. Typically, each cell is simulated using a circuit level simulator a number of times, each with changes in the operating point for the cell. The resultant timing is measured from these simulations and then fit to an equation or curve(s) that represents the timing of the cell for a variety of differing operating conditions.The final accuracy of timing, or delays used in logic simulation depends on the quantity and accuracy of the simulations, the resultant equation accuracy, and the ability to handle all aspects of these equations when calculating the final delays. Realize, that a typical library consisting of an excess of 150 cells, this process can be a formidable task. **Gendel** provides a means of reducing the complexity and number of simulations required to model delays a the library without sacrificing accuracy.

**Gendel** is part of the Genashor solution for simplifying cell modeling. **Gendel**, coupled with **Sp2log** and **Simic** provides an automated path for library development. The following is the recommended procedure for modeling using **Gendel**:

(1) Generate a list of basic building block configurations, with the specific transistor sizes used in the library. These building block configurations each consist of a single level, single output inverter functions, with a varying number of N and P devices in series.

(2) Simulate each of the basic configurations with varying input slew rates, and output loading conditions. Generate a table of the results of these simulations.

(3) Using **Gendel**, generate curves that fit the measured data for Sp2log/Simic

(4) Using **Sp2log**, starting fro the Spice descriptions of each cell in the library, generates logic models that includes references the respective delay curves generated by **Gendel**.

Libraries created by this technique can match circuit level simulation results with a very high correlation, even with diverse operating conditions. Figure 1 demonstrates the promoted methodology for generating a **Simic** cell library.

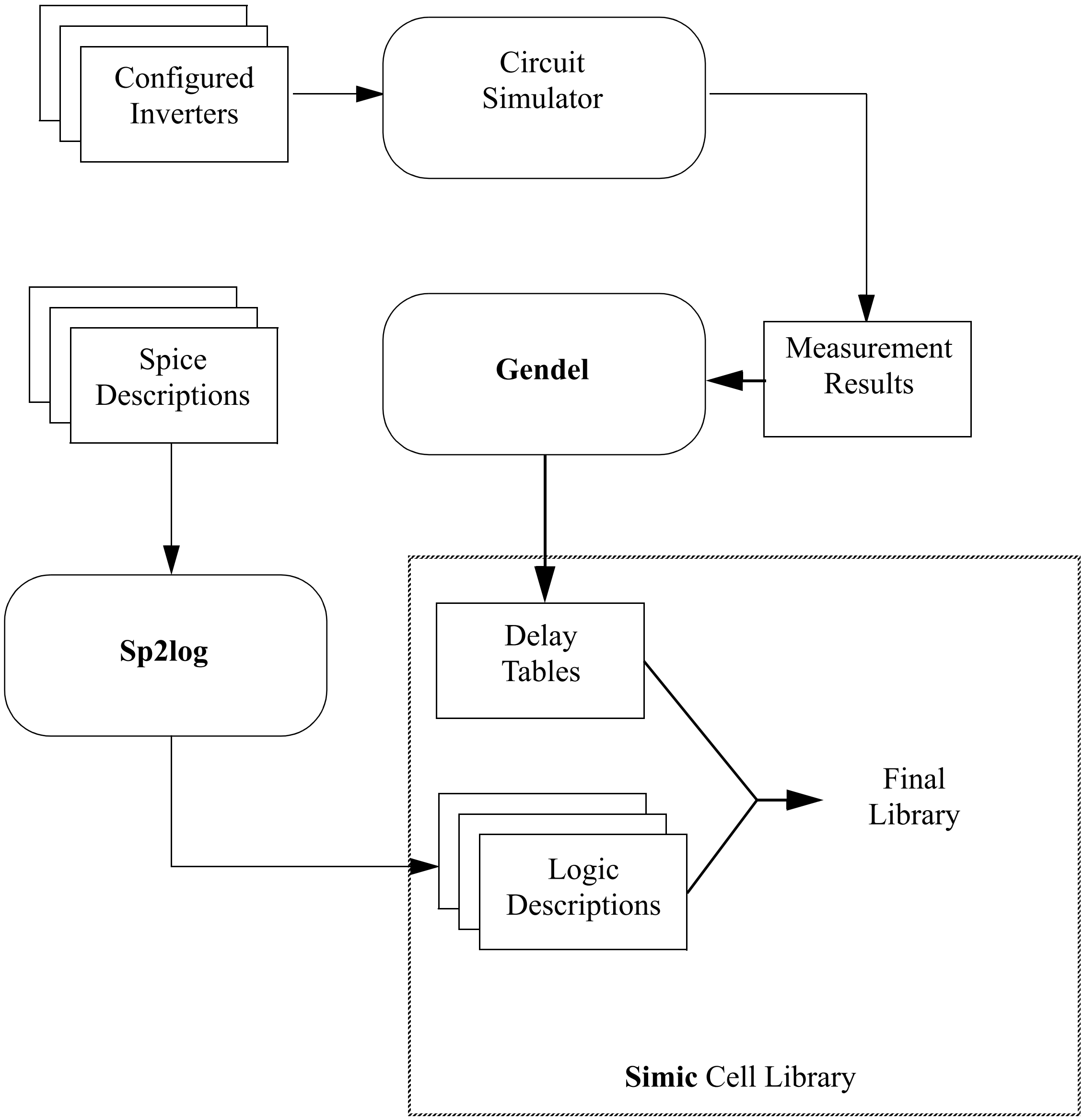



**Figure 1 Creation of a Simic Cell Library**

## Cell Modeling

If we look at any cell in a typical library, we find that it consists of functional nodes that are comprised of a number of transistors in series and parallel. In CMOS, there will be one or more pullup paths, and one or more pulldown paths. When determining the delay time for a switching function, we can neglect all non-contributing paths in the analysis. This typically distills down to the equivalent to an inverter function, with a number of N devices in series, and a number of P devices in series. All other paths are opened by an off transistor. Using this property, **Gendel** is used to fit a delay curve to each of these potential inverter function configurations. In this way, a small number of configurations represent what is found in the library cells. For those cells that do not conform to this scenario, characterization can be performed on the entire function instead.

## Configuration Simulation

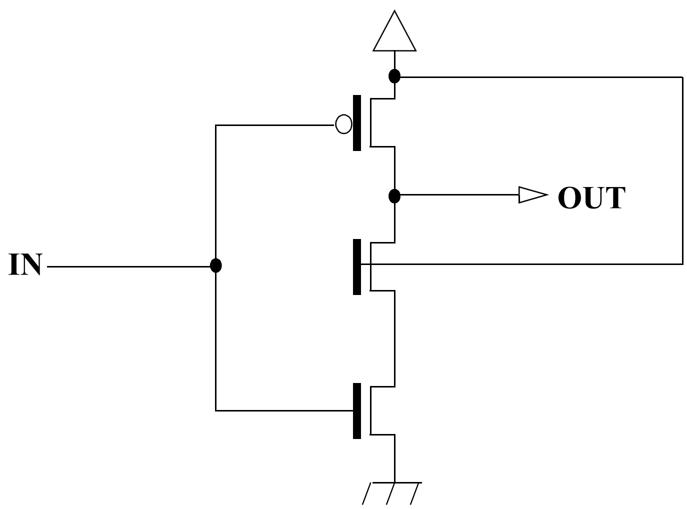



**Figure 2 Typical Simulation Configuration**

One possible circuit configuration for characterization is shown in Figure 2. All configurations function as an inverter, and have a number of N and P devices in series with the output. In figure 2, there are two N devices and one P device. Typical libraries range from one to four devices in series. Note that one N and P gate in the configurations should be tied together, choosing the gates closest to the rails will help the simulation converge easily. The remaining gates should be tied to the power rail that keeps the transistor in the ON state. The input stimuli should be a ramp, and the output measurements are made in the linear portion of the output waveform. If there are a variety of transistor sizes used in the design, then all measurement results of configurations with like device sizes should be grouped together and placed in the same file. Other such groups should be placed together in other files.

## Data Extraction

Once the basic configurations used in the cell library are determined, each one must be simulated using a circuit simulator such as Spice. Since each of the configurations are an inverter function, the same input stimulus and output loading can be used on each configurationl. This provides a means of automatically generating all the models, performing the simulations, capturing the results, and generating the final tables, with a small amount of programming in your favorite language. For example, If 10 loading points, and 10 input slew-rates are to be used for data extraction, 200 simulations will be required for each configuration (100 for rise times, and 100 for fall times). If there are 16 unique inverter configurations, then 3200 simulations are required. Since these are all very simple circuits, the simulations can be performed rapidly. If we were to characterize each cell of a 100 cell family, this would require at least 20,000 simulations, more if the library contains multi-level and/or multi-output cells.

## Input File Format

For each configuration, the output delay and output ramp time is measured. The circuit is stimulated by a ramp input. The starting ramp time, t0, is used as the starting point for each time measurement. Measurements are made by selecting two voltage points within the linear portion of the output curve. For example, at the 35% and 65% output voltage points. The file must declare the measurement points prior to listing the measured values. This declaration is in the form:

```
thresholds first=<p> second=<p>
```

where <p> is a percentage (with the optional percent sign suffix). For the 35% and 65% points, this would be:

```
thresholds first=35 second=65
```

A file containing the table of results is created with each measurement described in the following format:

```
<type> <parameter list>
```

where:

<type> is the string “fall” for falling output measurements or “rise” for rising output measurements. The <parameter list> is a series of floating point values assigned to the following properties:

```
first=<f>      which describes the amount of simulated time from
               t0, at which the output reaches the first
               threshold point.
second=<f>     which describes the amount of simulated time from
               t0, at which the output reaches the second
               threshold point.
ramp=<f>       which describes the time the input ramp takes to
               go from rail to rail.
load=<f>       which describes the amount of external loading on
               the output node.
nn=<f>         which describes the number of N transistors in
               series.
np=<f>         which describes the number of P transistors in
               series.
```

For example, a possible measurement record might be:

```
fall first=5e-10 second=7.9e-10 ramp=3e-10 load=3e-13 nn=1 np=1
```

Comments are permitted in the file. They are prefixed by a number sign (#) and terminate at the end of the line or end of the file.

## Running Gendel

Once the data files(s) are created, **Gendel** can be used to fit the data and output curves in **Simic**'s Delay Table format. The command line syntax for **Gendel** is:

```
gendel <infile> [<outfile>] [<switches>]
```

where <infile> is the name of the input data file, <outfile> is optional, and specifies the name of the file to output to. If <outfile> is omitted, then the output goes to stdout (the terminal). The following is a list of the optional <switches>:

| | |
|---|---|
| -a <s> | This specifies a formatting string used to construct a symbolic name for each generated curve. A %n in the string is replaced by the number of N devices in series for this curve. A %p in the string is replaced by the number of P devices in series for this curve. A %% is replaced by a single percent character (%). All other characters remain the same. The default value is "N%nP%p" |
| -c <f> | Specifies the scaling factor for capacitance values. The default is 1.0. |
| -d | This specifies to allow Gendel to use disjoint segments. This can result in lower errors within each segment, but can result in higher errors near the end points of each segment. If this option is used, be sure that the segments endpoints do not fall in a region that the cell will be operated in. |
| -e <p> | Sets the targeted fitting error below <p> percent. Setting the value small will result in more segments in each curve. Note that this is the fitting error at the measured points, sparse data can result in larger errors in areas far from measured data. The default value is 10 (percent). |
| -f | This forces Gendel to fill in configurations that where skipped in the data file. |
| -n <i> | Specifies the minimum number of series N and P transistors in to generate curves for. Note that -n implies -f. This switch is used to generate curves outside of the measured configurations in |

| | |
|---|---|
| | the data file. |
| -s <i> | Specifies the maximum number of segments allowed in each curve. |
| -t <f> | This specifies the scaling factor for time. The default is 1e10. |