

T-Spice Examples — Contents

1	Circuit Analysis Examples	2
	The T-Spice Pro™ Circuit Simulation System	2
	Simulation and Analysis Examples.	3
	Example 1: DC Operating Point Analysis	4
	SPICE Simulation Setup in S-Edit	5
	Export the Netlist to T-Spice	5
	Run the Simulation In T-Spice	7
	Open the Output File	8
	Example 2: DC Transfer Analysis	9
	Example 3: Transient Analysis—Inverter	13
	Example 4: AC Analysis	16
	Example 5: Using Subcircuits	19
	Example 6: Transient Analysis—CMOS D-Latch	23
	Example 7: Transient Analysis, Powerup Mode.	26
	Example 8: Transient Analysis, Preview Mode.	29
	Example 9: Noise Analysis.	32
	Example 10: MOS Transconductance Amplifier.	34
	Design Examples.	37
	 Index	 39
	 Credits	 42

The T-Spice Pro™ Circuit Simulation System

The design cycle for the development of electronic circuits includes an important pre-fabrication verification phase. Because of the expense and time pressures associated with the fabrication step, accurate verification is crucial to efficient design. The role of T-Spice is to help design and verify a circuit's operation by numerically solving the differential equations describing the circuit. T-Spice simulation results allow circuit designers to verify and fine-tune designs before submitting them for fabrication.

T-Spice Pro is a complete circuit design and analysis system that includes:

- **DxDesigner schematic editor.** DxDesigner is a powerful design capture and analysis package that can generate netlists directly usable in T-Spice simulations.
- **T-Spice circuit simulator.** T-Spice performs fast and accurate simulation of analog and mixed analog/digital circuits. The simulator includes the latest and best device models available, as well as coupled line models and support for user-defined device models via tables or C functions.
- **W-Edit waveform viewer.** W-Edit displays T-Spice simulation output waveforms as they are being generated during simulation.

T-Spice uses an extended version of the SPICE input language that is compatible with all industry-standard SPICE simulation programs. All of SPICE's device models are incorporated, as well as resistors, capacitors, inductors, mutual inductors, single and coupled transmission lines, current sources, voltage sources, controlled sources, and a full complement of the latest advanced semiconductor device models from Berkeley and Philips Labs.

T-Spice also incorporates numerous innovations and improvements not found in other SPICE and SPICE-compatible simulators:

- **Speed.** T-Spice provides highly optimized code for evaluating device models, formulating the systems of linear equations, and solving those systems. In addition to the standard direct model evaluation, T-Spice also provides the option of table-base transistor model evaluation, in which the results of device model evaluations are stored in tables and reused. Because evaluation of device models can be computationally expensive, this technique can yield dramatic simulation speed increases.
- **Convergence.** T-Spice uses advanced mathematical methods to achieve superior numerical stability. Large circuits and feedback circuits, impossible to analyze with other SPICE products, can be simulated in T-Spice.
- **Accuracy.** T-Spice uses very accurate numerical methods and charge conservation to achieve superior simulation accuracy.
- **Macromodeling.** T-Spice simulates circuits containing "black box" macrodevices. A macrodevice can directly use experimental data as its device model. Macrodevices can also represent complex devices, such as logic gates, for which only the overall transfer characteristics are of interest.

- **Input language extensions.** The T-Spice input language is an enriched version of the standard SPICE language. It contains many enhancements, including parameters, algebraic expressions, and a powerful bit and bus input wave specification syntax.
- **External model interface.** You can develop custom device models using C or C++.
- **Runtime waveform viewing.** The W-Edit waveform viewer displays graphical results during simulation. T-Spice analysis results for voltages, currents, charges, and power can be written to single or multiple files.

Simulation and Analysis Examples

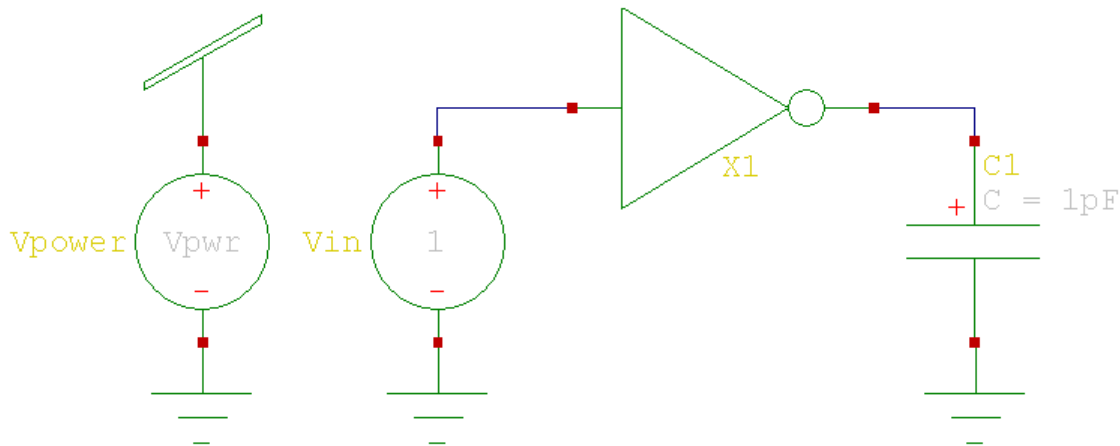
This collection of examples provides a hands-on introduction to the integrated components of the T-Spice Pro circuit analysis suite. The most common types of analysis and simulation features are demonstrated, including DC operating point computations, DC transfer sweeps, transient analysis, AC and noise analysis, and direct versus table-based model evaluation.

Example 1: DC Operating Point Analysis

DC operating point analysis finds a circuit's steady-state condition, obtained (in principle) after the input voltages have been applied for an infinite amount of time.

<i>Schematic</i>	...\\Tanner EDA\\Tanner Tools v12.6\\T-Spice\\AnalysisExamples\\ Inverter_TestBench OperatingPoint
<i>T-Spice Input</i>	...\\Tanner EDA\\Tanner Tools v12.6\\T-Spice\\SimulationResults\\ InverterOP.sp
<i>Output</i>	...\\Tanner EDA\\Tanner Tools v12.6\\T-Spice\\SimulationResults\\ InverterOP.out

Schematic



(This CMOS inverter is also used in [Example 2: DC Transfer Analysis](#) and [Example 3: Transient Analysis—Inverter](#).)

Each of the components visible in the schematic has properties associated with it. Properties are textual elements, created in DxDesigner, that are attached to an object and provide key information about its design and simulation commands in T-Spice.

If you "push in" to open a specific instance, you can see that the physical dimensions of the component **N1** in the inverter are defined by the properties:

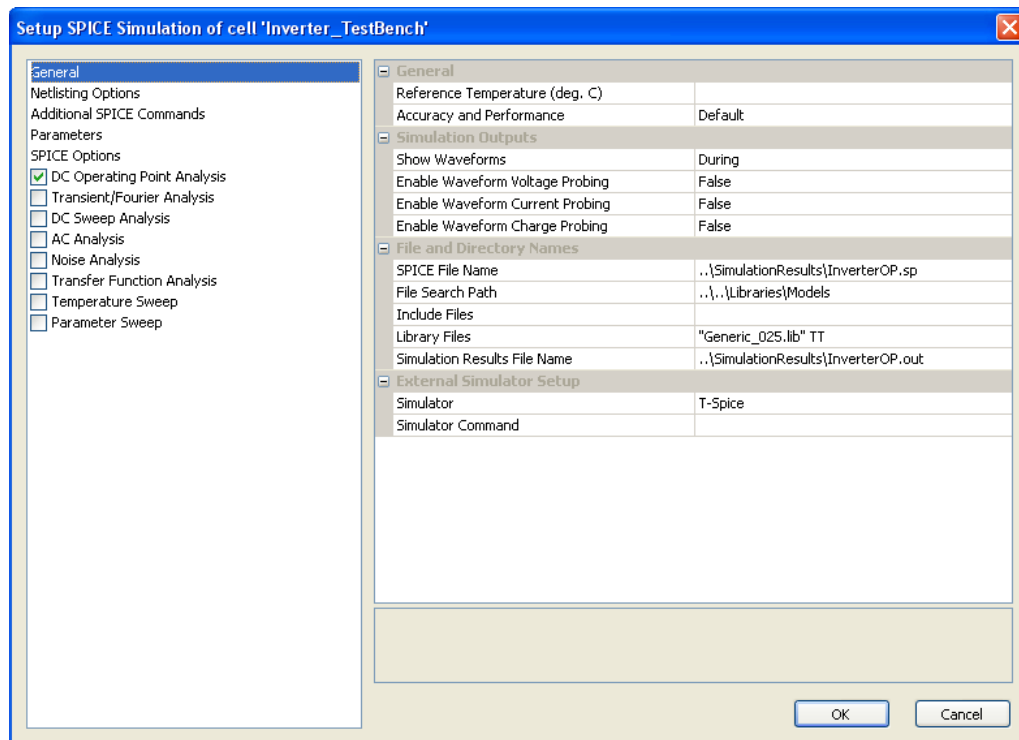
```
M = 1
W = 2.5u
L = 0.25u
```

N1 is an instance of the symbol **NMOS**, which represents an *n*-channel MOSFET transistor. Properties that describe the operation of a generic *n*-channel MOSFET are defined at the symbol level. Properties specific to component **N1**, such as length and width, are defined when **N1** is created. Property values defined at the component level take precedence over default (symbol) values.

SPICE Simulation Setup in S-Edit

Prior to running the T-Spice simulation, the analysis commands and all processing options need to be established. This is accomplished using the **Setup SPICE Simulation** dialog in DxDesigner.

- ☑ Ensure that you are viewing the top level schematic. For this example, the top level cell is named **inverter_TestBench OperatingPoint**. Right-click on **inverter_TestBench** in the **Libraries** window and use **Open View** to select the schematic **OperatingPoint**.
- ☑ Use **Setup > SPICE Simulation** to launch the **Setup SPICE Simulation** dialog. The proper simulation settings for the **Inverter_TestBench** example have already been entered for you. Note that the **DC Operating Point Analysis** box is checked. Also note the settings in the **General** options for **File Search Path** and **Include Files**.



Export the Netlist to T-Spice

- ☑ In the **inverter_Testbench Operating Point** schematic, use **Tools > Design Checks** to execute the **Design Checker**.
- ☑ The **Design Checker** will display any violation or errors in the **Command** window. There should not be any errors in the file **inverter_Testbench Operating Point**.
- ☑ Press the T-Spice icon (🔌) to export a T-Spice netlist file named **inverterOP.sp**. DxDesigner will launch T-Spice with the **inverterOP.sp** netlist open.

T-Spice Input

```
***** Simulation Settings - General section *****
.option search="C:\src\TannerToolsShippingFiles\Libraries\Models"
.lib "Generic_025.lib" TT
```

```

*----- Devices: SPICE.ORDER < 0 -----
* Design: AnalysisExamples / Cell: Inverter_TestBench / View: OperatingPoint
  / Page:
* Designed by: Tanner EDA Library Development Team
* Organization: Tanner EDA - Tanner Research, Inc.
* Info: Operating point analysis testbench of an inverter
* Date: 06/15/2007 2:56:17 PM
* Revision: 0

***** Subcircuits *****
.subckt INV A Out Gnd Vdd
*----- Devices: SPICE.ORDER < 0 -----
* Design: LogicGates / Cell: INV / View: Main / Page:
* Designed by: Tanner EDA Library Development Team
* Organization: Tanner EDA - Tanner Research, Inc.
* Info: Inverter
* Date: 06/15/2007 2:56:17 PM
* Revision: 0

*----- Devices: SPICE.ORDER == 0 -----
MP1 Out A Vdd Vdd PMOS W=2.5u L=250n M=2 AS=4.5p PS=13.6u AD=3.125p PD=7.5u
MN1 Out A Gnd 0 NMOS W=2.5u L=250n AS=2.25p PS=6.8u AD=2.25p PD=6.8u
.ends

***** Simulation Settings - Parameters and SPICE Options *****
.param Vpwr = 3.3v

*----- Devices: SPICE.ORDER == 0 -----
VVin N_2 Gnd DC 1
XX1 N_2 N_1 Gnd Vdd INV
VVpower Vdd Gnd DC Vpwr
CC1 N_1 Gnd 1p

***** Simulation Settings - Analysis section *****
.op

***** Simulation Settings - Additional SPICE commands *****

.end

```

Two transistors, **MP1** and **MN1**, are defined in **inverterOP.sp**. These are MOSFETs, as indicated by the key letter **M** that begins their names. Following each transistor name are the names of its terminals in the required order: drain–gate–source–bulk. Then the model name (**PMOS** or **NMOS** in this example) and physical characteristics, such as length and width, are specified.

A capacitor **CC1** (signified by the key letter **C**) connects nodes **N 1** and **GND** with a capacitance of 1p. (Strictly speaking, the capacitor could be omitted from the circuit for this example, since it does not affect the DC operation of the inverter.)

Two DC voltage sources are defined: **VVin**, which sets node **N2** to 1.0 volt relative to ground and **VVpower**, which sets node **Vdd** to 3.3 volts as defined by the variable **Vpwr**.

- ☒ Notice that the simulation settings which were entered in the **SPICE Simulation Setup** dialog resulted in **.option**, **.lib**, and **.op** commands being written to the T-Spice input file. The **.lib** command causes T-Spice to read the contents of the **Generic_025.lib** library file for the evaluation of transistors **MP1** and **MN1**, and the **search** option identifies the path to the library files. In this case, the library file contains two device **.model** commands, describing MOSFET models **PMOS** and **NMOS**, as shown below for **PMOS**:

```


.MODEL PMOS PMOS (
+VERSION = 3.1          TNOM    = 27          TOX    = 5.6E-9
+XJ      = 1E-7          NCH     = 4.1589E17    VTH0   = -0.4935548
+K1      = 0.6143278     K2      = 6.804492E-4    K3     = 0
+K3B     = 5.8844074     W0      = 1E-6          NLX    = 6.938169E-9
+DVT0W   = 0            DVT1W   = 0            DVT2W  = 0
+DVT0    = 2.3578746     DVT1    = 0.7014778      DVT2   = -0.1881376
+U0      = 100           UA      = 9.119231E-10   UB     = 1E-21
+UC      = -1E-10        VSAT    = 1.782051E5     A0     = 0.9704347
+AGS     = 0.1073973     B0      = 2.773991E-7      B1     = 8.423987E-7
+KETA    = 0.0104811     A1      = 0.0193128      A2     = 0.3
+RDSW    = 694.5830247   PRWG    = 0.3169639      PRWB   = -0.1958978
+WR      = 1            WINT     = 0            LINT   = 2.971337E-8
+XL      = 0            XW       = -4E-8          DWG    = -2.967296E-8
+DWB     = -2.31786E-10  VOFF    = -0.1152095      NFACTOR = 1.1064678
+CIT      = 0           CDSC     = 2.4E-4          CDSCD  = 0
+CDSCB    = 0           ETA0     = 0.3676411      ETAB   = -0.0915241
+DSUB     = 1.1089801    PCLM     = 1.3226289      PDIBLC1 = 9.913816E-3
+PDIBLC2  = -1.499968E-6 PDIBLCB  = -1E-3          DROUT  = 0.1276027
+PSCBE1   = 8E10        PSCBE2   = 5.772776E-10    PVAG   = 0.0135936
+DELTA    = 0.01        RSH      = 3            MOBMOD  = 1
+PRT      = 0           UTE      = -1.5          KT1     = -0.11
+KT1L     = 0           KT2      = 0.022         UA1     = 4.31E-9
+UB1      = -7.61E-18   UC1      = -5.6E-11      AT      = 3.3E4
+WL       = 0           WLN      = 1            WW     = 0
+WWN      = 1           WWL      = 0            LL     = 0
+LLN      = 1           LW       = 0            LWN     = 1
+LWL      = 0           CAPMOD   = 2            XPART   = 0.5
+CGDO     = 5.59E-10    CGSO     = 5.59E-10      CGBO   = 5E-10
+CJ       = 1.857995E-3  PB      = 0.9771691      MJ      = 0.4686434
+CJSW     = 3.426642E-10 PBSW    = 0.871788      MJSW   = 0.3314778
+CJSWG    = 2.5E-10     PBSWG   = 0.871788      MJSWG  = 0.3314778
+CF       = 0           PVTH0    = 4.137981E-3     PRDSW  = 7.2931065
+PK2      = 2.600307E-3 WKETA    = 0.0192532     LKETA   = -5.972879E-3
)

```

Generic_025.lib assigns values to various Level 49 MOSFET model parameters for both n - and p -channel devices. T-Spice uses these parameters to evaluate Level 49 MOSFET model equations.

The **.op** command performs a DC operating point calculation and writes the results to the file specified in the **Simulation > Run Simulation** dialog.

Run the Simulation In T-Spice

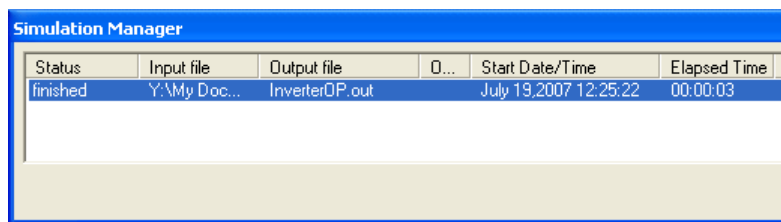
- ☒ With **inverterOP.sp** open in T-Spice, use **File > Save** to save the file.
- ☒ Click the **Run Simulation** button () in the T-Spice simulation toolbar.
- ☒ In the **Run Simulation** dialog, click **Start Simulation**.
- ☒ T-Spice will open a new window displaying the simulation log.

Output

The output file lists the DC operating point information for the circuit. You can read this file in T-Spice or any text editor.

Open the Output File

- ☑ If not already displayed, select **View > Simulation Manager** from the T-Spice menu to open the **Simulation Manager**:



- ☑ Select the **InverterOP.out** display line in the window, then click the **Show Output** button to open the output file **InverterOP.out** in a new T-Spice window.
- ☑ If you prefer to view the output in a text editor, simply open **InverterOP.out** as a text file. (It is located in the same directory as the input file.)
- ☑ The output file contains the following DC operating point information (in addition to comments of various kinds, not shown here. (You can also view DC operating voltages, currents and small-signal parameters in S-Edit.)

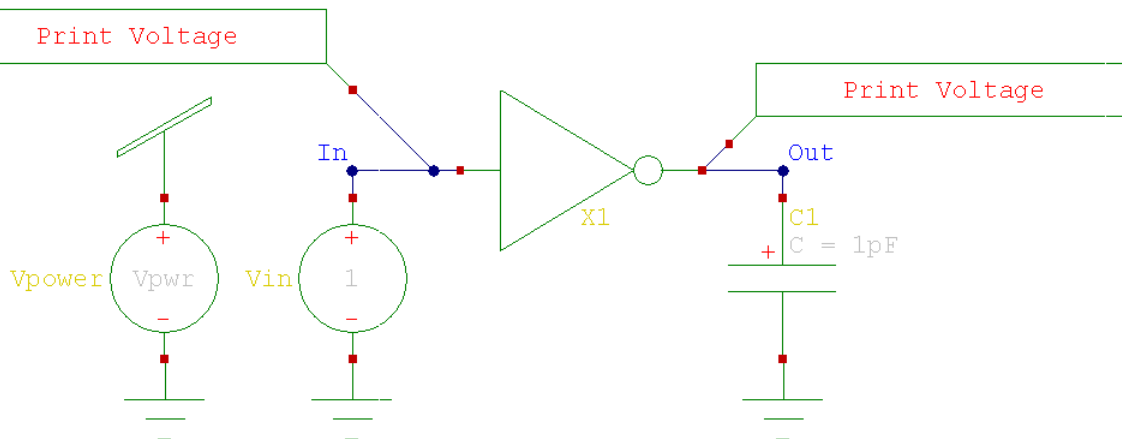
```
DC ANALYSIS - temperature=25.0
v(N_1) = 3.0633e+000
v(N_2) = 1.0000e+000
v(Vdd) = 3.3000e+000
i1(VVin) = 0.0000e+000
i2(VVin) = 0.0000e+000
i1(VVpower) = -3.1508e-004
i2(VVpower) = 3.1508e-004
```


Example 2: DC Transfer Analysis

DC transfer analysis is used to study the voltage or current at one set of points in a circuit as a function of the voltage or current at another set of points. This is done by *sweeping* the source variables over specified ranges and recording the output.

<i>Schematic</i>	...\\Tanner EDA\\Tanner Tools v12.6\\T-Spice\\AnalysisExamples\\ Inverter_Testbench DCAnalysis
<i>T-Spice Input</i>	...\\Tanner EDA\\Tanner Tools v12.6\\T-Spice\\SimulationResults\\ inverterDC.sp
<i>Output</i>	...\\Tanner EDA\\Tanner Tools v12.6\\T-Spice\\SimulationResults\\ inverterDC.out

Schematic



This schematic includes a **.print** command, which measures and records voltages at the input and output nodes of the circuit. The command is contained within the **DC analysis output** cell.

T-Spice Input

If T-Spice is open when you run the simulation in S-Edit, the SPICE file **inverterDC.sp** will open and run automatically.

```
***** Simulation Settings - General section *****
.option search="...\\Tanner EDA\\Tanner Tools v12.6\\Libraries\\Models"
.probe
.option probev
.lib "Generic_025.lib" TT

*----- Devices: SPICE.ORDER < 0 -----
* Design: AnalysisExamples / Cell: Inverter_TestBench / View: DCAnalysis /
  Page:
* Designed by: Tanner EDA Library Development Team
* Organization: Tanner EDA - Tanner Research, Inc.
* Info: DC analysis testbench of an inverter
* Date: 06/15/2007 2:56:17 PM
```

```

* Revision: 0

***** Subcircuits *****
.subckt INV A Out Gnd Vdd
*----- Devices: SPICE.ORDER < 0 -----
* Design: LogicGates / Cell: INV / View: Main / Page:
* Designed by: Tanner EDA Library Development Team
* Organization: Tanner EDA - Tanner Research, Inc.
* Info: Inverter
* Date: 06/15/2007 2:56:17 PM
* Revision: 0

*----- Devices: SPICE.ORDER == 0 -----
MP1 Out A Vdd Vdd PMOS W=2.5u L=250n M=2 AS=4.5p PS=13.6u AD=3.125p PD=7.5u
MN1 Out A Gnd 0 NMOS W=2.5u L=250n AS=2.25p PS=6.8u AD=2.25p PD=6.8u
.ends
\

***** Simulation Settings - Parameters and SPICE Options *****
.param Vpwr = 3.3v

*----- Devices: SPICE.ORDER == 0 -----
VVin In Gnd DC 1
XX1 In Out Gnd Vdd INV
VVpower Vdd Gnd DC Vpwr
CC1 Out Gnd 1p
*----- Devices: SPICE.ORDER > 0 -----
.PRINT DC V(Out)
.PRINT DC V(In)

***** Simulation Settings - Analysis section *****
.dc lin VVin 0.0 3.3 0.02 lin VVpower 2.3 4.3 0.5

***** Simulation Settings - Additional SPICE commands *****

.end

```

The **.DC** command, indicating transfer analysis, is followed by the parameter **lin**, which specifies a linear sweep. Next is a list of sources to be swept, and the voltage ranges across which the sweeps are to take place.

In this example, **VVin** will be swept from 0 to 3.3 volts in 0.02 volt increments, and **VVpower** will be swept from 2.3 to 4.3 volts in 0.5 volt increments.

The transfer analysis will be performed as follows: **VVpower** will be set at 2.3 volts and **V1** will be swept over its specified range; **VVpower** will then be incremented to 2.5 volts and **V1** will be reswept over its range; and so on, until **VVpower** reaches the upper limit of its range.

The **.DC** command ignores the values assigned to the voltage sources **VVpower** and **VVin** in the voltage source statements; however, they must be declared in those statements.

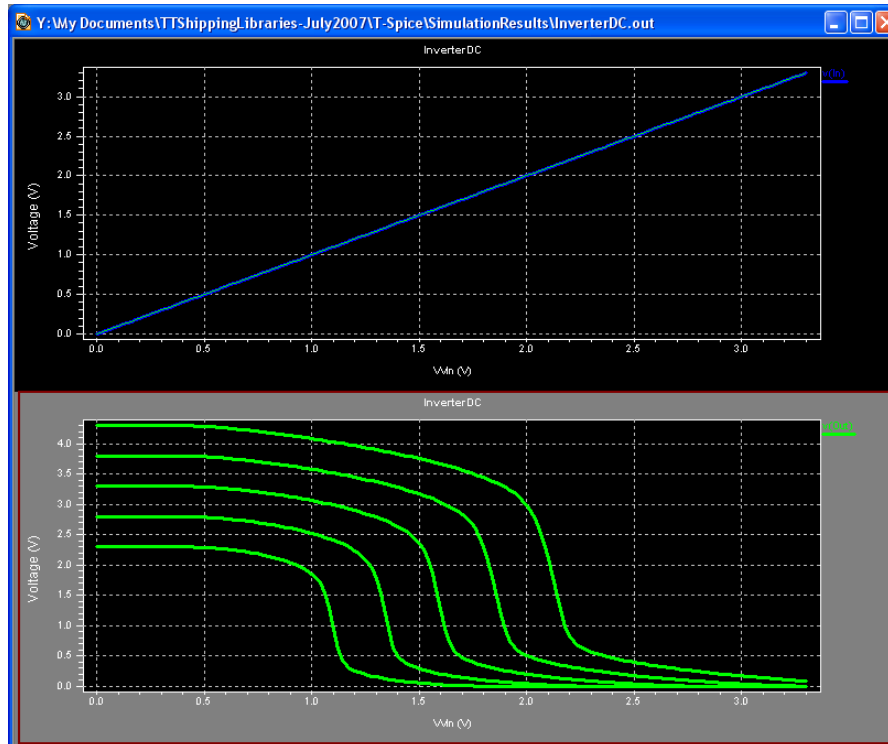
The resulting voltages for nodes **IN** and **OUT** are reported by the **.PRINT DC** command to the specified destination.

Output

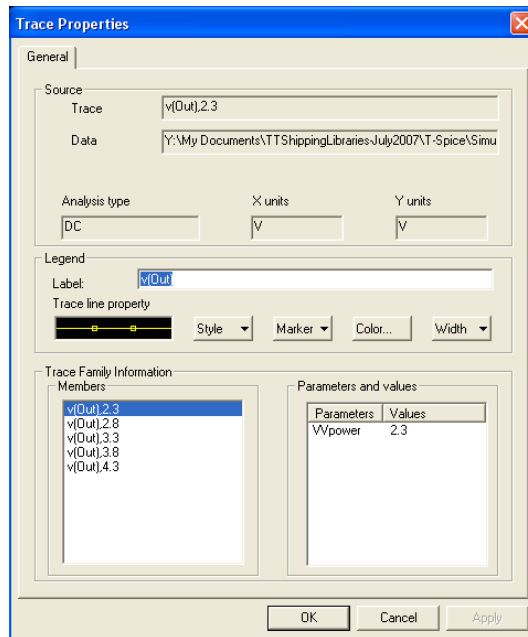
When W-Edit launches, simulation results of the same data type, which in this case is voltage, are automatically plotted on a single chart. In this example, traces were separated into different charts and

reorganized (according to data type) using the commands in **Chart > Expand Chart** (page 109) of the W-Edit menu.

The charts below show input and output voltages to the circuit, with separate traces for each sweep of **v(Out)**. To view detailed information about a trace, double-click on the trace or on the trace label located in the upper right corner of the chart.



The **Trace Properties** dialog displays the value of parameter **v(Out)** corresponding to each trace, as well as labels and line properties. For more information on trace properties, see ["Properties" on page 100](#) of the *W-Edit User Guide*.



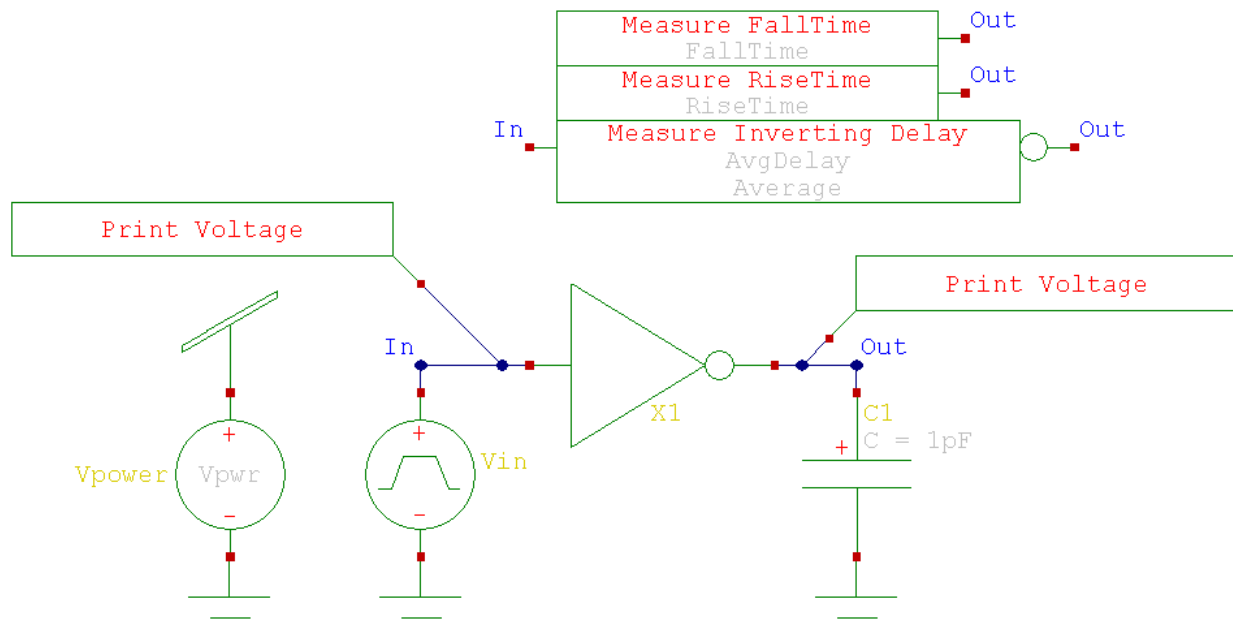
Example 3: Transient Analysis—Inverter

Transient analysis provides information on how circuit elements vary with time.

The basic T-Spice command for transient analysis has three modes. In the *op* mode (default), the DC operating point is computed, and T-Spice uses this as the starting point for the transient simulation. Example 3 illustrates this option. (The other startup modes, powerup and preview, are shown in "Example 7: Transient Analysis, Powerup Mode" on page 26 and "Example 8: Transient Analysis, Preview Mode" on page 29.)

<i>Schematic</i>\Tanner EDA\Tanner Tools v12.6\T-Spice\AnalysisExamples\ InverterTRAN
<i>T-Spice Input</i>	...\\Tanner EDA\Tanner Tools v12.6\T-Spice\SimulationResults\ InverterTRAN.sp
<i>Output</i>	...\\Tanner EDA\Tanner Tools v12.6\T-Spice\SimulationResults\ InverterTRAN.out

Schematic



T-Spice Input

```
***** Simulation Settings - General section *****
.option search="...\Tanner EDA\Tanner Tools v12.6\Libraries\Models"
.probe
.option probev
.option probei
.lib "Generic_025.lib" TT

*----- Devices: SPICE.ORDER < 0 -----
```

```

* Design: AnalysisExamples / Cell: Inverter_TestBench / View:
  TransientAnalysis / Page:
* Designed by: Tanner EDA Library Development Team
* Organization: Tanner EDA - Tanner Research, Inc.
* Info: Transient analysis testbench of an inverter
* Date: 12/18/2005 7:28:14 PM
* Revision: 5

***** Subcircuits *****
.subckt INV A Out Gnd Vdd
*----- Devices: SPICE.ORDER < 0 -----
* Design: LogicGates / Cell: INV / View: Main / Page:
* Designed by: Tanner EDA Library Development Team
* Organization: Tanner EDA - Tanner Research, Inc.
* Info: Inverter
* Date: 12/18/2005 7:28:14 PM
* Revision: 5

*----- Devices: SPICE.ORDER == 0 -----
MP1 Out A Vdd Vdd PMOS W=2.5u L=250n M=2 AS=4.5p PS=13.6u AD=3.125p PD=7.5u
MN1 Out A Gnd 0 NMOS W=2.5u L=250n AS=2.25p PS=6.8u AD=2.25p PD=6.8u
.ends

***** Simulation Settings - Parameters and SPICE Options *****
.param Vpwr = 3.3v

*----- Devices: SPICE.ORDER == 0 -----
VVin In Gnd PULSE(0 Vpwr 0 1n 1n 49n 100n)
XX1 In Out Gnd Vdd INV
VVpower Vdd Gnd DC Vpwr
CC1 Out Gnd 1p
*----- Devices: SPICE.ORDER > 0 -----
.PRINT TRAN V(Out)
.PRINT TRAN V(In)
.MEASURE TRAN RiseDelay_MeasureDelay_1 TRIG v(In) VAL='(Vpwr-0)*50/100+0'
  TD='0' RISE=1 TARG v(Out) VAL='(Vpwr-0)*50/100+0' TD='0' FALL=1 OFF
.MEASURE TRAN FallDelay_MeasureDelay_1 TRIG v(In) VAL='(Vpwr-0)*50/100+0'
  TD='0' FALL=1 TARG v(Out) VAL='(Vpwr-0)*50/100+0' TD='0' RISE=1 OFF
.MEASURE TRAN AvgDelay
  PARAM='(RiseDelay_MeasureDelay_1+FallDelay_MeasureDelay_1)/2.0' ON
.MEASURE TRAN FallTime TRIG v(Out) VAL='(Vpwr-0)*90/100+0' TD=0 Fall=1 TARG
  v(Out) VAL='(Vpwr-0)*10/100+0' TD=0 FALL=1 ON
.MEASURE TRAN RiseTime TRIG v(Out) VAL='(Vpwr-0)*10/100+0' TD=0 RISE=1 TARG
  v(Out) VAL='(Vpwr-0)*90/100+0' TD=0 RISE=1 ON

***** Simulation Settings - Analysis section *****
.tran 250p 300n

***** Simulation Settings - Additional SPICE commands *****

.end

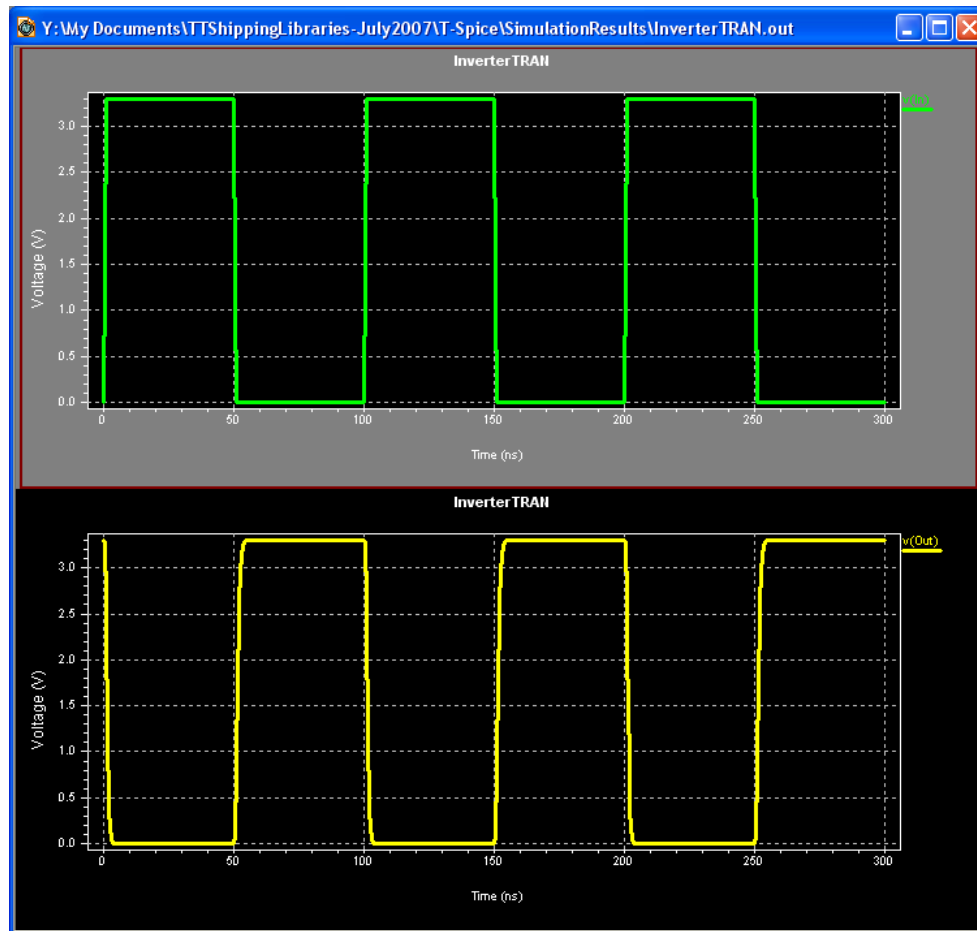
```

This circuit is similar to that of [Example 1](#), except that voltage source **VVin** here generates a pulse (indicated by the keyword **pulse**) to **In**, rather than setting a constant value.

The times and voltages that define the “legs” of the waveform are specified in the arguments to **pulse**. The initial current is zero amperes and the peak current is **Vpwr**, with an initial delay of zero seconds. The rise and fall times are one nanosecond, with a pulse width of 49 nanoseconds and a pulse period of 100 nanoseconds.

The **.tran** command specifies the characteristics of the transient analysis to be performed; in this example the maximum time step allowed is 250 pico with a total duration of 300 nanoseconds.

Output



Example 4: AC Analysis

AC analysis characterizes the circuit's behavior dependence on small-signal input frequency. It involves three steps: (1) calculating the DC operating point; (2) linearizing the circuit; and (3) solving the linearized circuit for each frequency.

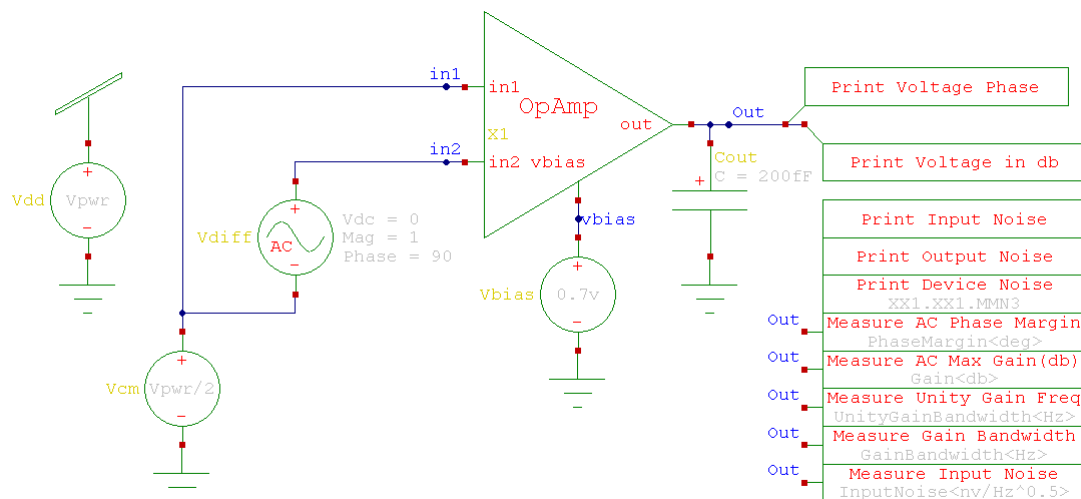
This example involves a standard operational amplifier, consisting of one PMOS, one NMOS, a transconductance amplifier and one capacitor.

Schematic ...\\Tanner EDA\\Tanner Tools v12.6\\T-Spice\\AnalysisExamples\\
OpAmp_TestBench AC_Noise_Analysis

T-Spice Input ...\\Tanner EDA\\Tanner Tools v12.6\\T-Spice\\SimulationResults\\
OpAmpAC.sp

Output ...\\Tanner EDA\\Tanner Tools v12.6\\T-Spice\\SimulationResults\\
OpAmpAC.out

Schematic



T-Spice Input

```
***** Simulation Settings - General section *****
.option Accurate
.option search="...\\Tanner EDA\\Tanner Tools v12.6\\Libraries\\Models"
.probe
.option probev
.option probei
.lib "Generic_025.lib" TT

***** Subcircuits *****
.subckt TransAmp in1 in2 out vbias Gnd Vdd
----- Devices: SPICE.ORDER < 0 -----
* Design: AnalysisExamples / Cell: TransAmp / View: Main / Page:
* Designed by: Tanner EDA Library Development Team
* Organization: Tanner EDA - Tanner Research, Inc.
```



```

* Info: Transconductance Amplifier
* Date: 06/15/2007 2:56:17 PM
* Revision: 0

*----- Devices: SPICE.ORDER == 0 -----
MMP1 vm1 vm1 Vdd Vdd PMOS W=2u L=2u AS=1.8p PS=5.8u AD=1.8p PD=5.8u
MMP2 out vm1 Vdd Vdd PMOS W=2u L=2u AS=1.8p PS=5.8u AD=1.8p PD=5.8u
MMN1 vm1 in1 vn1 0 NMOS W=2u L=2u AS=1.8p PS=5.8u AD=1.8p PD=5.8u
MMN2 out in2 vn1 0 NMOS W=2u L=2u AS=1.8p PS=5.8u AD=1.8p PD=5.8u
MMN3 vn1 vbias Gnd 0 NMOS W=2u L=3u AS=1.8p PS=5.8u AD=1.8p PD=5.8u
.ends
...
*----- Devices: SPICE.ORDER == 0 -----
XX1 in1 in2 vf1 vbias Gnd Vdd TransAmp
MMP1 Out vf1 Vdd Vdd PMOS W=6u L=2u AS=5.4p PS=13.8u AD=5.4p PD=13.8u
CComp vf1 Out 200f
MMN1 Out vbias Gnd 0 NMOS W=3u L=2u AS=2.7p PS=7.8u AD=2.7p PD=7.8u
.ends

***** Simulation Settings - Parameters and SPICE Options *****
.param Vpwr = 3.3v

*----- Devices: SPICE.ORDER == 0 -----
VVdd Vdd Gnd DC Vpwr
VVdiff in2 in1 DC 0 AC 1 90
VVcm in1 Gnd DC Vpwr/2
VVbias vbias Gnd DC 700m
CCout Out Gnd 200f
XX1 Out in1 in2 vbias Gnd Vdd OpAmp
*----- Devices: SPICE.ORDER > 0 -----
.PRINT AC Vdb(Out)
.PRINT AC Vp(Out)
.PRINT NOISE INOISE
.PRINT NOISE ONOISE
.PRINT NOISE TRANSFER dn(XX1.XX1.MMN3)
.MEASURE NOISE InputNoise<nv/Hz^0.5> FIND 'inoise/1E-9' WHEN Vdb(Out)=0 ON
.MEASURE AC UnityGainBandwidth<Hz> WHEN Vdb(Out)=0 ON
.MEASURE AC MeasureGainBandwidthProduct_1_Gain MAX vdb(Out) OFF
.MEASURE AC MeasureGainBandwidthProduct_1_UGFreq WHEN Vdb(Out)=0 OFF
.MEASURE AC GainBandwidth<Hz>
    PARAM='MeasureGainBandwidthProduct_1_Gain*MeasureGainBandwidthProduct_1_U
    GFreq' ON
.MEASURE AC Gain<db> MAX vdb(Out) ON
.MEASURE AC PhaseMargin<deg> FIND '90+vp(Out)' WHEN vdb(Out)=0 ON

***** Simulation Settings - Analysis section *****
.op
.ac dec 10 1 100Meg
.noise v(Out) VVdiff 5

```

Three voltage sources (in addition to **Vdd**) are defined.

- **Vdiff** sets the DC voltage difference between nodes **IN2** and **IN1** to 0 volts; the AC magnitude is 1 volt and its AC phase is 90 degrees.
- **Vcm** sets node **IN1** to 2 volts, relative to **GND**.
- **Vbias** sets node **vbias** to 700 millivolts, relative to **GND**.

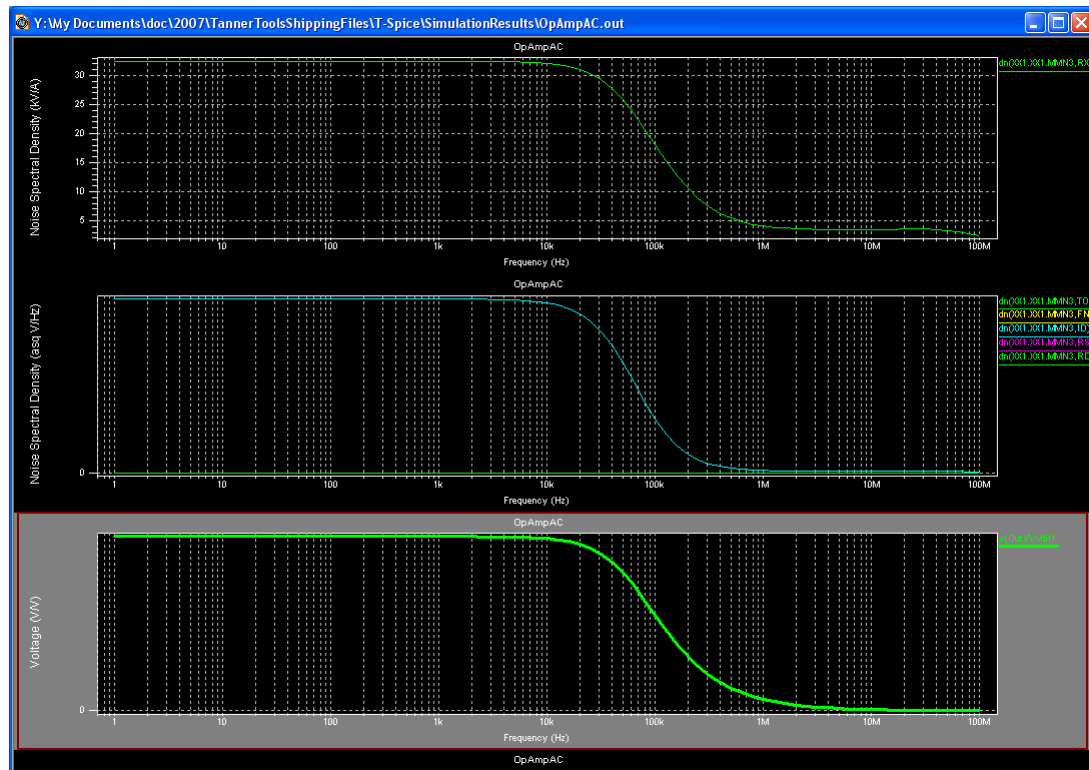
The **.ac** command performs an AC analysis. Following the **.ac** keyword is information concerning the frequencies to be swept during the analysis. In this case, the frequency is swept logarithmically, by

decades (**DEC**); 10 data points are to be included per decade; the starting frequency is 1 Hz and the ending frequency is 100 MHz.

The **.PRINT** command writes the voltage magnitude (in decibels) and phase (in degrees), respectively, for the node **OUT** to the specified file. (The other print and measurement commands are discussed in later examples.)

Output

The AC simulation will result in **AC** small-signal model parameters being written to the output file, in addition to all output generated from the **.print** statements.



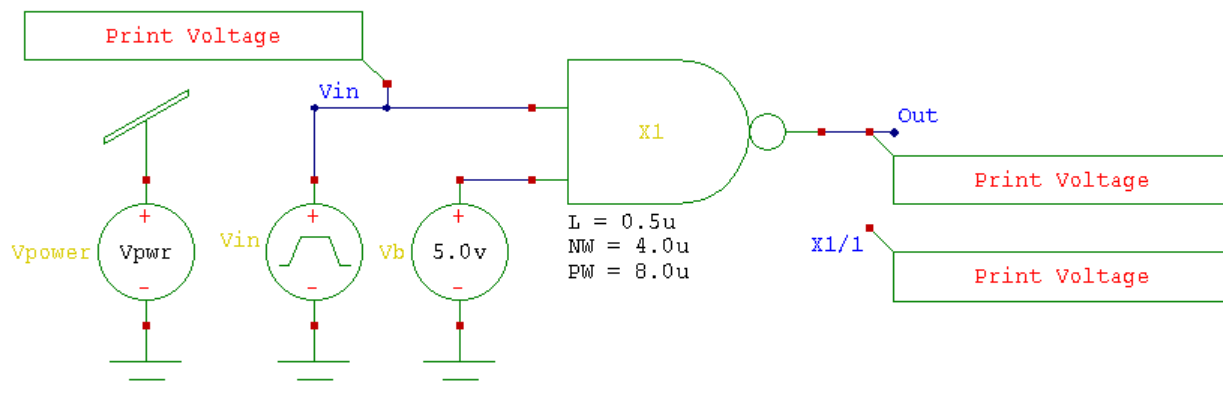
Example 5: Using Subcircuits

Subcircuit definitions allow arbitrarily complex arrangements of nodes and devices to be easily reused multiple times in a circuit. A subcircuit definition in DxDesigner is contained within a cell definition, and is comprised of both a schematic view and a symbol view. Each instance of the symbol encapsulates the subcircuit schematic, allowing a simple but complete representation of subcircuit dynamics.

Example 5 uses a NAND gate to illustrate the use of subcircuit definitions and subcircuit parameters.

<i>Schematic</i>	...\Tanner EDA\Tanner Tools v12.6\T-Spice\AnalysisExamples\ Subcircuit_Testbench
<i>T-Spice Input</i>	...\Tanner EDA\Tanner Tools v12.6\T-Spice\SimulationResults\ SubcircuitTRAN.sp
<i>Output</i>	...\Tanner EDA\Tanner Tools v12.6\T-Spice\SimulationResults\ SubcircuitTRAN.out

Schematic



An instance of the subcircuit **NAND** is created in the schematic and labeled **X1**. (To access **NAND** from the main schematic, double-click on the **NAND** item in the Libraries list.)

As discussed in Example 1, symbol properties are used to define component properties such as length and width. This example introduces a new symbol property, **SPICE.PARAMETER**, which allows parameters to be passed through a hierarchical netlist.

The symbol that represents NAND has the SPICE parameter property **L=NW=PW=** which specifies that the cell properties **L**, **NW**, and **PW** are subcircuit parameters of **NAND**. The cell also contains the three property definitions **L = 0.5u**, **NW = 4.0u**, and **PW = 8.0u**.

These parameters define properties of all *n*-channel and *p*-channel MOSFETS within the subcircuit such that **L** represents the length property of both *n*- and *p*-channel MOSFETS, **NW** represents *n*-channel width and **PW** represents *p*-channel width.

Attaching these parameters to **NAND** allows component properties within the subcircuit definition to be controlled in the subcircuit call.

T-Spice Input

```

***** Simulation Settings - General section *****
.option search="...\Tanner EDA\Tanner Tools v12.6\Libraries\Models"
.probe
.option probev
.option probei
.lib "Generic_025.lib" TT

*----- Devices: SPICE.ORDER < 0 -----
* Design: AnalysisExamples / Cell: Subcircuit_TestBench / View: Main / Page:
* Designed by: Tanner EDA Library Development Team
* Organization: Tanner EDA - Tanner Research, Inc.
* Info: Testbench for subcircuit example
* Date: 7/13/2007 7:59:21 AM
* Revision: 19

***** Subcircuits *****
.subckt NAND A B Out Gnd Vdd L=500n NW=4u PW=8u
*----- Devices: SPICE.ORDER < 0 -----
* Design: AnalysisExamples / Cell: NAND / View: Main / Page:
* Designed by: Tanner EDA Library Development Team
* Organization: Tanner EDA - Tanner Research, Inc.
* Info: 2 input NAND gate
* Date: 7/13/2007 7:59:21 AM
* Revision: 95

*----- Devices: SPICE.ORDER == 0 -----
MP1 Out A Vdd Vdd PMOS W=PW L=L M=2 AS='if(0, (900n*PW+floor(2/2)*1.25u*PW),
(2*900n*PW+(floor(2/2)-1)*1.25u*PW))' PS='if(0, (2*900n+PW+PW*1+floor(2/
2)*2*(1.25u+PW*1)), (2*2*900n+PW+PW*1+(floor(2/2)-1)*2*(1.25u+PW*1)))'
AD='if(0, (900n*PW+floor(2/2)*1.25u*PW), floor(2/2)*1.25u*PW)' PD='if(0,
(2*900n+PW+PW*1+floor(2/2)*2*(1.25u+PW*1)), floor(2/2)*2*(1.25u+PW*1))'

MP2 Out B Vdd Vdd PMOS W=PW L=L M=2 AS='if(0, (900n*PW+floor(2/2)*1.25u*PW),
(2*900n*PW+(floor(2/2)-1)*1.25u*PW))' PS='if(0, (2*900n+PW+PW*1+floor(2/
2)*2*(1.25u+PW*1)), (2*2*900n+PW+PW*1+(floor(2/2)-1)*2*(1.25u+PW*1)))'
AD='if(0, (900n*PW+floor(2/2)*1.25u*PW), floor(2/2)*1.25u*PW)' PD='if(0,
(2*900n+PW+PW*1+floor(2/2)*2*(1.25u+PW*1)), floor(2/2)*2*(1.25u+PW*1))'

MN1 Out A 1 0 NMOS W=NW L=L AS='if(1, (900n*NW+floor(1/2)*1.25u*NW),
(2*900n*NW+(floor(1/2)-1)*1.25u*NW))' PS='if(1, (2*900n+NW+NW*1+floor(1/
2)*2*(1.25u+NW*1)), (2*2*900n+NW+NW*1+(floor(1/2)-1)*2*(1.25u+NW*1)))'
AD='if(1, (900n*NW+floor(1/2)*1.25u*NW), floor(1/2)*1.25u*NW)' PD='if(1,
(2*900n+NW+NW*1+floor(1/2)*2*(1.25u+NW*1)), floor(1/2)*2*(1.25u+NW*1))'

MN2 1 B Gnd 0 NMOS W=NW L=L AS='if(1, (900n*NW+floor(1/2)*1.25u*NW),
(2*900n*NW+(floor(1/2)-1)*1.25u*NW))' PS='if(1, (2*900n+NW+NW*1+floor(1/
2)*2*(1.25u+NW*1)), (2*2*900n+NW+NW*1+(floor(1/2)-1)*2*(1.25u+NW*1)))'
AD='if(1, (900n*NW+floor(1/2)*1.25u*NW), floor(1/2)*1.25u*NW)' PD='if(1,
(2*900n+NW+NW*1+floor(1/2)*2*(1.25u+NW*1)), floor(1/2)*2*(1.25u+NW*1))'
.ends

*----- Devices: SPICE.ORDER == 0 -----
VVin Vin Gnd PULSE(0 Vpwr 0 1n 1n 49n 100n)
XX1 Vin N_1 Out Gnd Vdd NAND L=500n NW=4u PW=8u
VVb N_1 Gnd DC 5
VVpower Vdd Gnd DC Vpwr
*----- Devices: SPICE.ORDER > 0 -----
.PRINT TRAN V(Out)
.PRINT TRAN V(Vin)

```

```
.PRINT TRAN V(X1/1)

***** Simulation Settings - Analysis section *****
.tran 250p 300n

***** Simulation Settings - Additional SPICE commands *****

.end
```

Subcircuits are defined by blocks of device statements bracketed with the **.SUBCKT** and **.ENDS** commands, and *instanced* by statements beginning with the key letter **X**.

The **.SUBCKT** command includes the name of the subcircuit being defined (**NAND**), a list of terminals, and three subcircuit parameters. The *terminals* do not have a predefined order, but whatever order is used in the definition must be used in instances. *Parameters* can be written in any order in both the definition and the instances. If a parameter value is not specified in the instance the value in the definition is used as the default.

Within the subcircuit definition, four MOSFETs are defined in the usual manner—and in these statements the order of terminals *is* important: drain–gate–source–bulk. Node **1** is the source of transistor **MN1** and the drain of transistor **MN2**. Subcircuit parameters, enclosed by single quotes, are used in place of numerical values.

After the subcircuit is defined, you can create an instance of the subcircuit. The instance statement begins with the key letter **X**. The name of the instance, by which it is to be identified in the rest of the input file, is **X1** (not "XX1.")

The list of terminals in the instance statement must have the same order as on the first line of the subcircuit definition so that **A B Out Gnd** in the definition corresponds to **Vin N_1 OUT Gnd** in the instance. The next argument of the instance statement is the original subcircuit name **NAND**.

The default subcircuit parameter values, as specified by the definition, are overridden by instance-specific value assignments, which can appear in any order. Any parameters omitted from the instance statement retains its default value.

A standard DC operating point calculation (.OP) analysis is carried out on this circuit, with a duration of 300 nanoseconds and a maximum timestep of 250 picoseconds. The **.param** command sets the initial node voltages to 3.3 volts. The **.PRINT** command reports simulation results for the voltages at nodes **Vin**, **OUT**, and **X1/N_1**.

Output



Example 6: Transient Analysis—CMOS D-Latch

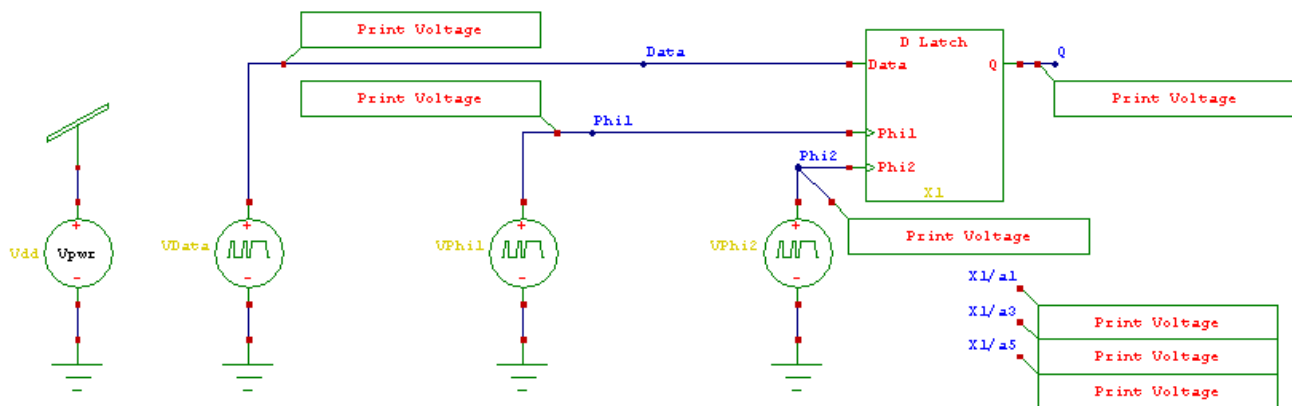
Transient analysis on a CMOS static D-latch demonstrates the analog D-latch characteristics of a digital circuit. The circuit has four inverters and four transmission gates.

Schematic ...\\Tanner EDA\\Tanner Tools v12.6\\T-Spice\\AnalysisExamples\\
DLatch_Testbench

T-Spice Input ...\\Tanner EDA\\Tanner Tools v12.6\\T-Spice\\SimulationResults\\
DLatchTRAN.sp

Output ...\\Tanner EDA\\Tanner Tools v12.6\\T-Spice\\SimulationResults\\
DLatchTRAN.out

Schematic



T-Spice Input

```
***** Simulation Settings - Parameters and SPICE Options *****
.param Vpwr = 3.3v
.option NUMDGT = 2

*----- Devices: SPICE.ORDER == 0 -----
VVdd Vdd Gnd DC Vpwr
VVDData Data Gnd BIT({1000} PW=16n ON=Vpwr RT=500p FT=500p LT=15.5n
      HT=15.5n)
VVPhi1 Phi1 Gnd BIT({0011} PW=8n ON=Vpwr RT=500p FT=500p LT=7.5n HT=7.5n)
VVPhi2 Phi2 Gnd BIT({1100} PW=8n ON=Vpwr RT=500p FT=500p LT=7.5n HT=7.5n)
XX1 Q Data Phi1 Phi2 Gnd Vdd DLatch
*----- Devices: SPICE.ORDER > 0 -----
.PRINT TRAN V(Q)
.PRINT TRAN V(X1/a1)
.PRINT TRAN V(X1/a3)
.PRINT TRAN V(X1/a5)
.PRINT TRAN V(Data)
.PRINT TRAN V(Phi1)
.PRINT TRAN V(Phi2)

***** Simulation Settings - Analysis section *****
.op
```

```
.tran 75ps 300ns
```

```
***** Simulation Settings - Additional SPICE commands *****
```

```
.end
```

Voltage source **VDD** sets the voltage between power and ground to the **Vpwr** parameter value of 3.3 volts.

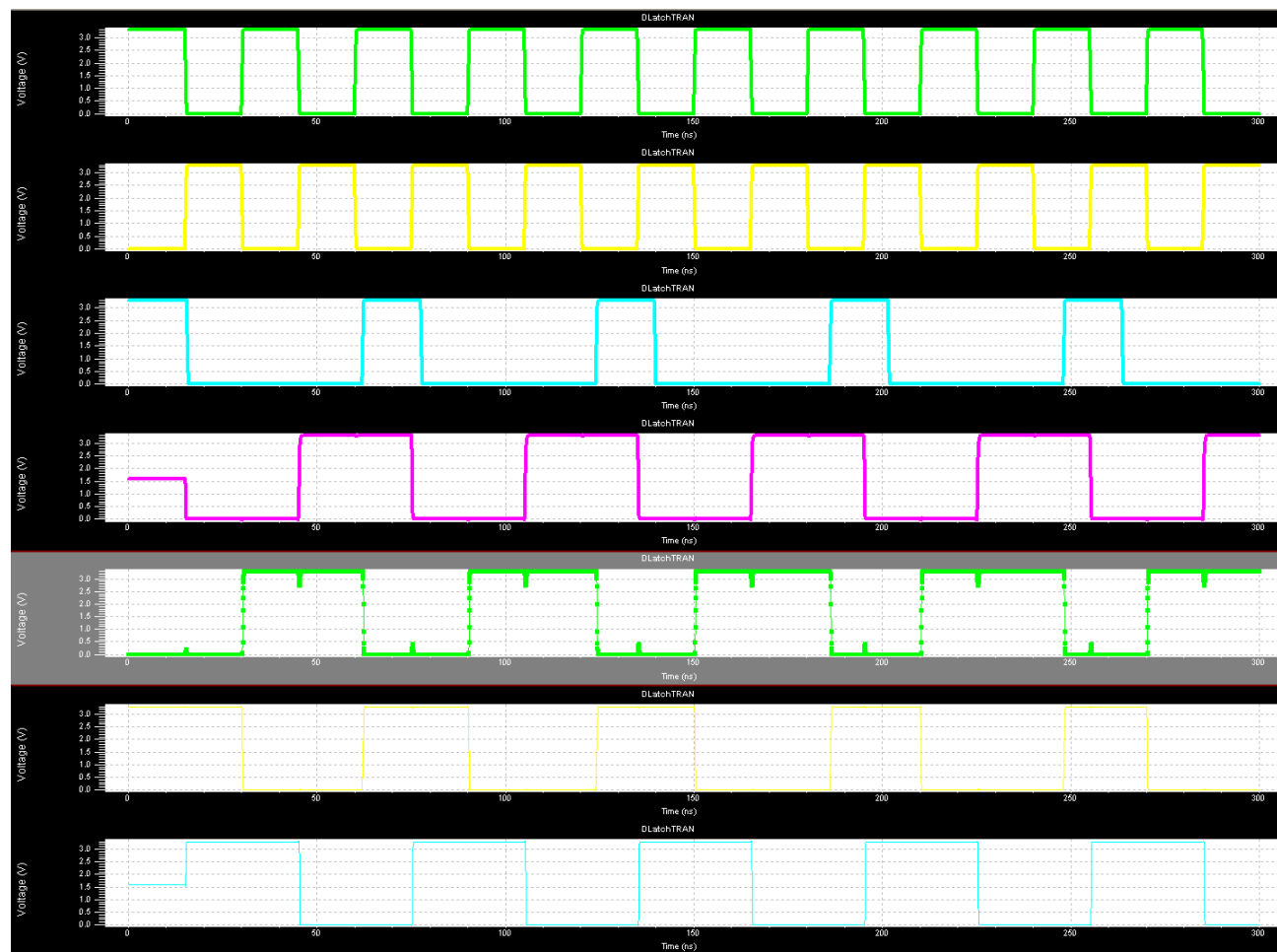
The next three statements beginning with **V** define voltage sources for custom input waveforms. Following each voltage source name are the names of the input nodes (Data, Phi1, Phi2) and the type of waveform. Here, however, not piecewise linear but rather *bit* waveforms are used. Following the keyword **BIT** in parentheses are parameters specifying the waveform characteristics. The four-digit sequence in braces { } specifies the sequence of the wave's states (either **1**, “on,” or **0**, “off”). This sequence will be repeated until the simulation is complete. The pulse widths (**PW**) are 16, 8 and 8 nanoseconds, respectively. The **OFF** voltage is zero, the **ON** voltage is 3.3 volts, the rise (**RT**) and fall (**FT**) times are each 500 picoseconds, and the low time (**LT**) and high time (**HT**) are both 7.5 nanoseconds.

The **.TRAN** command instructs T-Spice to perform a 300-nanosecond simulation while printing node voltages at least every 75 picoseconds.

The **.PRINT** commands write simulation results for the voltages at each of seven nodes to the specified file.

Output

The following chart shows the seven output voltages measured during simulation, plotted across time. The chart is expanded and the traces are displayed in the same order in which they were loaded. Traces can be loaded or unloaded on a chart using *Chart > Options—Format* in W-Edit. Note that when a trace is added to an existing chart, scaling of the axes is not automatically updated. To rescale the axes for optimal display, you can use the **HOME** key. Select *Chart > Options—Axes* to choose appropriate spacing of the major and minor tick marks.



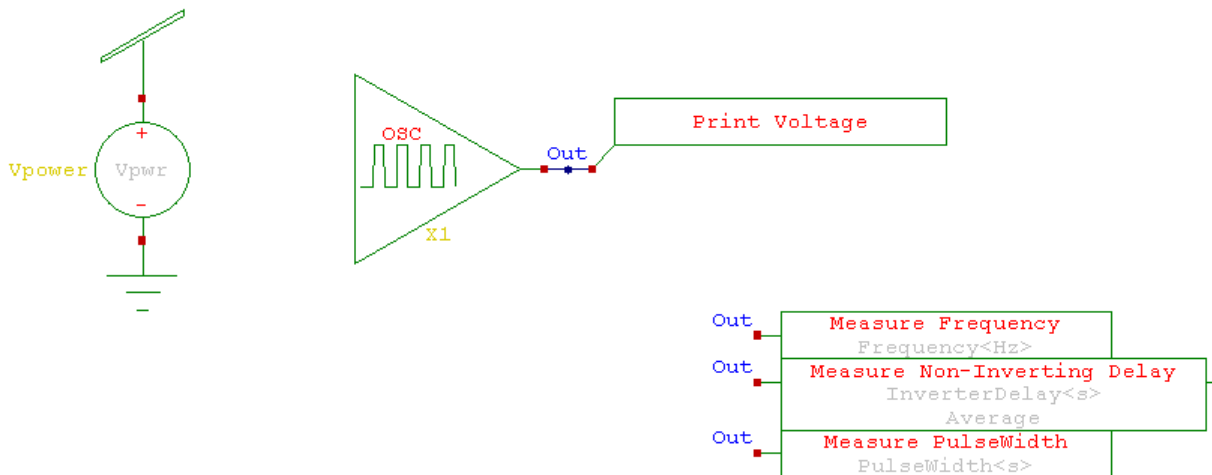
Example 7: Transient Analysis, Powerup Mode

Some circuits do not have a DC steady state or “quiescent” point. Because such circuits are constantly fluctuating with time, finding the starting point for their simulation is difficult. More precisely, the challenge is to define the initial state of a circuit which has no definite DC steady-state condition. This is done in T-Spice with the **powerup** option of the **.tran** command. The **powerup** option essentially sets the entire circuit to zero for time equal to zero. As the simulation proceeds, the voltage sources are allowed to ramp up to their specified values.

The *ring oscillator* is an example of such a time-dependent circuit.

<i>Schematic</i>	...\\Tanner EDA\\Tanner Tools v12.6\\T-Spice\\AnalysisExamples\\ RingOscillator_TestBench
<i>T-Spice Input</i>	...\\Tanner EDA\\Tanner Tools v12.6\\T-Spice\\SimulationResults\\ RingOscillatorTRAN.sp
<i>Output</i>	...\\Tanner EDA\\Tanner Tools v12.6\\T-Spice\\SimulationResults\\ RingOscillatorTRAN.out

Schematic



T-Spice Input

```
***** Subcircuits *****
.subckt INV A Out Gnd Vdd
----- Devices: SPICE.ORDER < 0 -----
* Design: LogicGates / Cell: INV / View: Main / Page:
* Designed by: Tanner EDA Library Development Team
* Organization: Tanner EDA - Tanner Research, Inc.
* Info: Inverter
* Date: 6/13/2007 2:17:11 PM
* Revision: 55
----- Devices: SPICE.ORDER == 0 -----
MP1 Out A Vdd Vdd PMOS W=2.5u L=250n M=2 AS=4.5p PS=13.6u AD=3.125p PD=7.5u
MN1 Out A Gnd 0 NMOS W=2.5u L=250n AS=2.25p PS=6.8u AD=2.25p PD=6.8u
.ends
```

```

.subckt RingOscillator a7 Gnd Vdd
*----- Devices: SPICE.ORDER < 0 -----
* Design: AnalysisExamples / Cell: RingOscillator / View: Flat / Page:
* Designed by:
* Organization:
* Info:
* Date: 7/13/2007 7:59:21 AM
* Revision: 24
*----- Devices: SPICE.ORDER == 0 -----
CC1 a1 Gnd Cap
CC2 a2 Gnd Cap
CC3 a3 Gnd Cap
CC4 a4 Gnd Cap
CC5 a7 Gnd Cap
CC6 a6 Gnd Cap
CC7 a5 Gnd Cap
XX1 a7 a1 Gnd Vdd INV
XX2 a1 a2 Gnd Vdd INV
XX3 a2 a3 Gnd Vdd INV
XX4 a3 a4 Gnd Vdd INV
XX5 a4 a5 Gnd Vdd INV
XX6 a5 a6 Gnd Vdd INV
XX7 a6 a7 Gnd Vdd INV
.ends
***** Simulation Settings - Parameters and SPICE Options *****
.param Vpwr = 3.3v
.param Cap = 0.8pF
*----- Devices: SPICE.ORDER == 0 -----
XX1 Out Gnd Vdd RingOscillator
VVpower Vdd Gnd DC Vpwr
*----- Devices: SPICE.ORDER > 0 -----
.PRINT TRAN V(Out)
.MEASURE TRAN PulseWidth<s> TRIG V(Out) VAL='Vpwr/2.0' TD='102n' RISE='1'
    TARG V(Out) VAL='Vpwr/2.0' TD='102n+2n*Cap/0.2p' FALL='1' ON
.MEASURE TRAN MeasureFrequency_1_Period TRIG V(Out) VAL='Vpwr/2.0' TD='100n'
    RISE='1' TARG V(Out) VAL='Vpwr/2.0' TD='100n' RISE='1+1' OFF
.MEASURE TRAN Frequency<Hz> PARAM='1.0/MeasureFrequency_1_Period*1' ON
.MEASURE TRAN RiseDelay_MeasureDelay_1 TRIG v(Out) VAL='(Vpwr-0)*50/100+0'
    TD='103n' RISE=1 TARG v(XX1.a1) VAL='(Vpwr-0)*50/100+0' TD='103n' FALL=1
    OFF
.MEASURE TRAN FallDelay_MeasureDelay_1 TRIG v(Out) VAL='(Vpwr-0)*50/100+0'
    TD='103n' FALL=1 TARG v(XX1.a1) VAL='(Vpwr-0)*50/100+0' TD='103n' RISE=1
    OFF
.MEASURE TRAN InverterDelay<s>
    PARAM='(RiseDelay_MeasureDelay_1+FallDelay_MeasureDelay_1)/2.0' ON
***** Simulation Settings - Analysis section *****
.tran/Powerup 100ps 400ns
.step lin Cap 0.2pF 1.0pF 0.2pF
***** Simulation Settings - Additional SPICE commands *****
.OPTIONS POWERUPLEN=10ns

.end

```

A subcircuit named **INVERT** is defined with two terminals. (This inverter is structurally identical to the one used in [Example 1](#) and [Example 4](#).)

Seven instances of the subcircuit, labeled **INV1** through **INV7**, are defined next. The output of each inverter is connected to the input of the next in the ring.

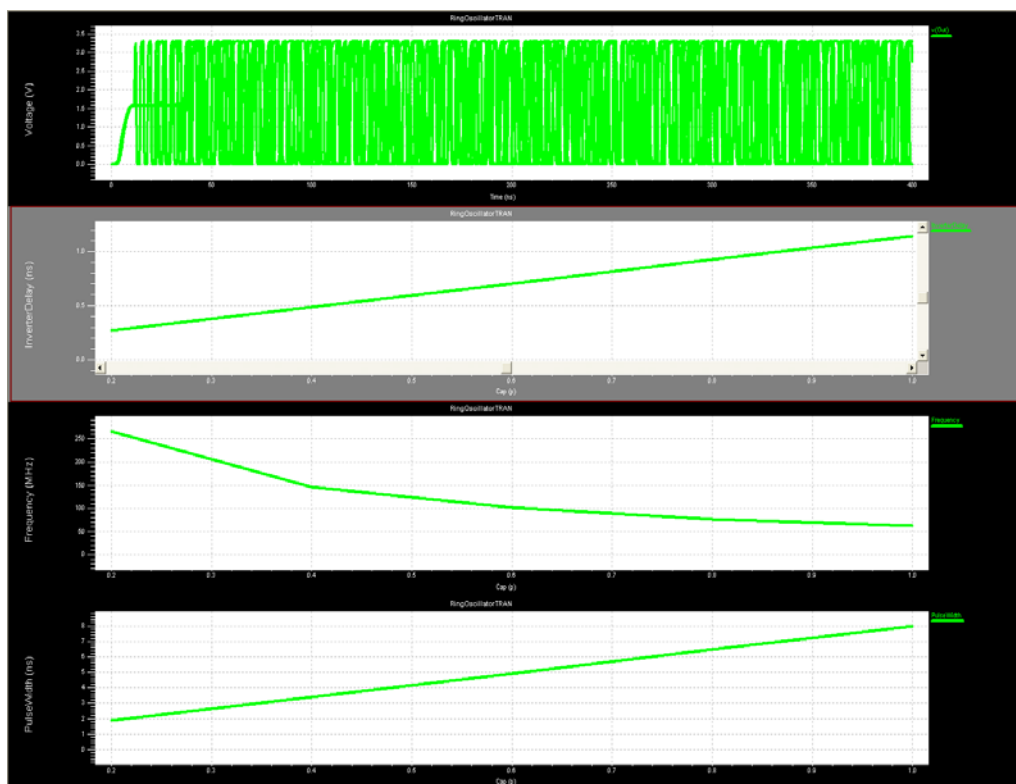
The **POWERUP** option of the **.TRAN** command eliminates the DC convergence problem for unstable circuits. If the **POWERUP** option were not specified, then T-Spice would try to calculate a DC operating point, which would lead to problems for this oscillator.

This simulation example also introduces us to two of the more powerful advanced features of T-Spice: parameter sweeping and the **.measure** command.

In the T-Spice Simulation Setup, a parameter sweep is defined, in which the capacitor value is varied in a linear sweep from 200 femto-Farads to 1000 femto-Farads in increments of 200. This results in 5 sets of simulation results being generated, one for each capacitor value.

The **.measure** command is a tool for capturing and summarizing the electrical behavior of a circuit. Information such as delay, rise and fall times, minimum and maximum signal values, and a host of other computed results can be acquired with these measurements. In this example, 3 measurements are computed, period, pulsewidth, and delay. Since the simulation also includes a parameter sweep, a separate set of measurement values will be computed for each parameter value (capacitance), and displayed in a summary table at the end of the Simulation log.

Output



Example 8: Transient Analysis, Preview Mode

Before you run a lengthy transient simulation on a large circuit, you can examine the input waveforms using the **preview** option of the **.tran** command. This option instructs T-Spice to report only specified input stimuli, and to forego simulation of the remainder of the circuit.

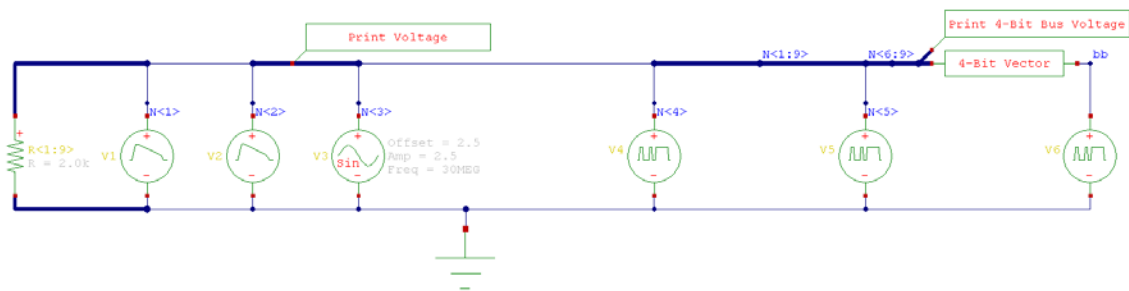
In addition to illustrating the preview mode, this simple example circuit showcases the variety and flexibility of input patterns available to current and voltage sources in T-Spice.

Schematic ...\\Tanner EDA\\Tanner Tools v12.6\\T-Spice\\AnalysisExamples\\
PreviewMode_TestBench

T-Spice Input ...\\Tanner EDA\\Tanner Tools v12.6\\T-Spice\\SimulationResults\\
PreviewMode.sp

Output ...\\Tanner EDA\\Tanner Tools v12.6\\T-Spice\\SimulationResults\\
PreviewMode.out

Schematic



T-Spice Input

```
***** Simulation Settings - Parameters and SPICE Options *****

*----- Devices: SPICE.ORDER == 0 -----
VV1 N<1> Gnd PWL(0ns 0v 100ns 0v 101ns 5v 300ns 5v 301ns 0v 500ns 0v 680ns
      5v 700ns 0v 880ns 5v 900ns 0v)
.VECTOR bb { N<6> N<7> N<8> N<9> }
VV2 N<2> Gnd PWL(0ns 0v 100ns 0v 101ns 1v 200ns 1v 201ns 2v 300ns 2v 301ns
      3v 400ns 3v 401ns 4v 500ns 4v 501ns 5v 600ns 5v 601ns 4v 700ns 4v 701ns
      3v 800ns 3v 801ns 2v 900ns 2v 901ns 1v)
RR<5> N<5> Gnd R=2k
VV3 N<3> Gnd SIN(2.5 2.5 30meg 100n 0 0)
VV4 N<4> Gnd BIT({01010 11011} PW=50n ON=5 OFF=0 RT=10n FT=30n )
VV5 N<5> Gnd BIT({5(01010 5(1))} PW=10n ON=5 OFF=0 )
VV6 bb Gnd BUS({50(Ah) 30(7d4) 20(1000)} PW=5n ON=5 OFF=0 )
RR<4> N<4> Gnd R=2k
RR<3> N<3> Gnd R=2k
RR<2> N<2> Gnd R=2k
RR<1> N<1> Gnd R=2k
RR<9> N<9> Gnd R=2k
RR<8> N<8> Gnd R=2k
RR<7> N<7> Gnd R=2k
RR<6> N<6> Gnd R=2k
*----- Devices: SPICE.ORDER > 0 -----
```

```
.PRINT TRAN V(N<6>)
.PRINT TRAN V(N<5>)
.PRINT TRAN V(N<4>)
.PRINT TRAN V(N<3>)
.PRINT TRAN V(N<2>)
.PRINT TRAN V(N<1>)
.PRINT TRAN Bus_N6-9<V>='STP(V(N<6>)-5/2)*2^0+STP(V(N<7>)-5/
    2)*2^1+STP(V(N<8>)-5/2)*2^2+STP(V(N<9>)-5/2)*2^3'
.PRINT TRAN V(N<9>)
.PRINT TRAN V(N<8>)
.PRINT TRAN V(N<7>)

***** Simulation Settings - Analysis section *****
.tran/Preview 1ns 1us

***** Simulation Settings - Additional SPICE commands *****

.end
```

Nine resistor/node/voltage source combinations, numbered **1** through **9**, are defined. Each resistor has a resistance of 2 kilohms; each voltage source, connected across the corresponding resistor to ground, supplies its characteristic waveform to the corresponding node.

Two voltage sources, **V1** and **V2**, generate piecewise linear (**PWL**) inputs. **V1** produces a single pulse followed by a pair of sawtooth cycles, and **V2** produces a staircase waveform which takes 1-volt steps from zero up to 5 volts and back down to 1 volt.

Voltage source **V3** generates a sinusoidal (**SIN**) input. It has an amplitude of 2.5 volts, a frequency of 30 MHz, an offset of 2.5 volts from system ground, and a time delay of 100 nanoseconds after the start of the simulation before the wave begins.

Voltage source **V4** generates a **BIT** input. Enclosed in braces { } are two binary-valued five-bit patterns specifying the waveform. The two patterns alternate in time. The **ON** voltage value is 5.0 volts; the **OFF** voltage value is zero. The pulse width (**PW**) of 50 nanoseconds is the time the wave is either ramping up and on, or dropping down and off. The rise time (**RT**) of 10 nanoseconds is the time given for the wave to ramp from off to on and the fall time (**FT**), 30 nanoseconds, is the time given for the wave to drop from on to off.

Voltage source **V5** generates a repeating **BIT** input. Two distinct patterns are given again, but now *multiplier factors* are included. The wave consists of two alternating patterns: the first pattern contains five bits, the second is a single bit. The five-bit pattern is followed by five successive repetitions of the single-bit pattern, and this sequence is repeated five times. (The same pattern could be described by {5(3(01) 4(1))}.) The pulse width and on and off voltages are again specified, but the rise and fall times take default values.

The **.VECTOR** command defines the bus waveform generated by voltage source **vb**. The command assigns the bus a name (**bb**) and specifies by name the number of bits the bus waveform will have (four: **N6** through **N9**). The voltage source statement, which contains the **BUS** keyword, specifies waveforms with one or more patterns, along with pulse width and level information. The patterns can be in binary, hexadecimal, octal, or decimal notation. (For decimal patterns the number of lower-order bits to be collected is also given.)

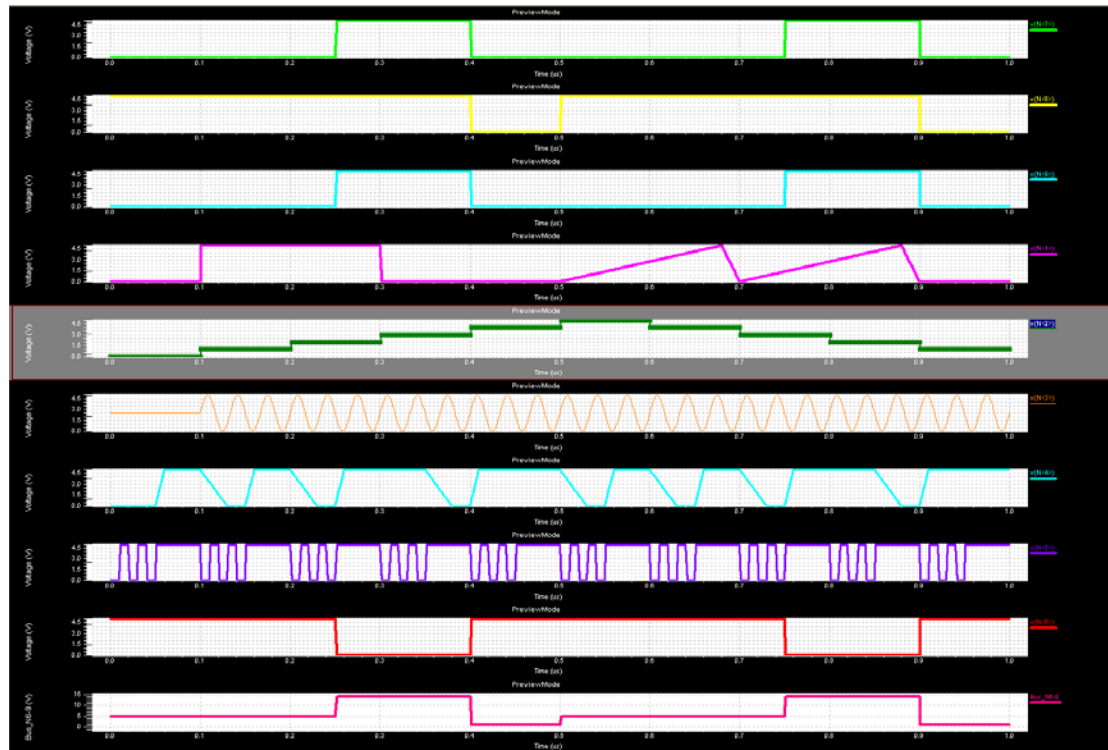
- The first pattern is **Ah** (hex) = 1010 (binary). Thus, using the pattern given in the **.vector** command, **N6**=1, **N7**=0, **N8**=1 and **N9**=0. The pattern is repeated 50 times (that is, maintained for a time period equal to the pulse width multiplied by 50).
- The next pattern is **7d4** — that is, 7 (decimal) = 111 (binary), or, to four lower-order bits, 0111. So **n6**=0, **n7**=1, **n8**=1, and **n9**=1. The pattern is repeated 30 times.

- The last pattern is **1000** (binary), so **n6=1**, **n7=0**, **n8=0**, and **n9=0**. The pattern is repeated 20 times.

The **.print** command writes the results at the output nodes of all nine voltage sources.

The **.tran/preview** command reports the input waveforms in place of running the simulation.

Output



Example 9: Noise Analysis

Real circuits are never immune from small random fluctuations in voltage and current levels. In T-Spice, the influence of noise in a circuit can be simulated and reported in conjunction with AC analysis. The purpose of noise analysis is to compute the effect of the noise associated with various circuit devices on an output voltage or voltages as a function of frequency.

<i>Schematic</i>	...\\Tanner EDA\\Tanner Tools v12.6\\T-Spice\\AnalysisExamples\\ OpAmp_TestBench AC_Noise_Analysis
<i>T-Spice Input</i>	...\\Tanner EDA\\Tanner Tools v12.6\\T-Spice\\SimulationResults\\ OpAmpAC.sp
<i>Output</i>	...\\Tanner EDA\\Tanner Tools v12.6\\T-Spice\\SimulationResults\\ OpAmpAC.out

Schematic

This circuit is shown in [Example 4: AC Analysis](#).

T-Spice Input

```
***** Simulation Settings - Parameters and SPICE Options *****
.param Vpwr = 3.3v

*----- Devices: SPICE.ORDER == 0 -----
VVdd Vdd Gnd DC Vpwr
VVdiff in2 in1 DC 0 AC 1 90
VVcm in1 Gnd DC Vpwr/2
VVbias vbias Gnd DC 700m
CCout Out Gnd 200f
XX1 Out in1 in2 vbias Gnd Vdd OpAmp
*----- Devices: SPICE.ORDER > 0 -----
.PRINT AC Vdb(Out)
.PRINT AC Vp(Out)
.PRINT NOISE INOISE
.PRINT NOISE ONOISE
.PRINT NOISE TRANSFER dn(XX1.XX1.MMN3)
.MEASURE NOISE InputNoise<nv/Hz^0.5> FIND 'inoise/1E-9' WHEN Vdb(Out)=0 ON
.MEASURE AC UnityGainBandwidth<Hz> WHEN Vdb(Out)=0 ON
.MEASURE AC MeasureGainBandwidthProduct_1_Gain MAX vdb(Out) OFF
.MEASURE AC MeasureGainBandwidthProduct_1_UGFreq WHEN Vdb(Out)=0 OFF
.MEASURE AC GainBandwidth<Hz>
    PARAM='MeasureGainBandwidthProduct_1_Gain*MeasureGainBandwidthProduct_1_U
    GFreq' ON
.MEASURE AC Gain<db> MAX vdb(Out) ON
.MEASURE AC PhaseMargin<deg> FIND '90+vp(Out)' WHEN vdb(Out)=0 ON

***** Simulation Settings - Analysis section *****
.op
.ac dec 10 1 100Meg
.noise v(Out) VVdiff 5

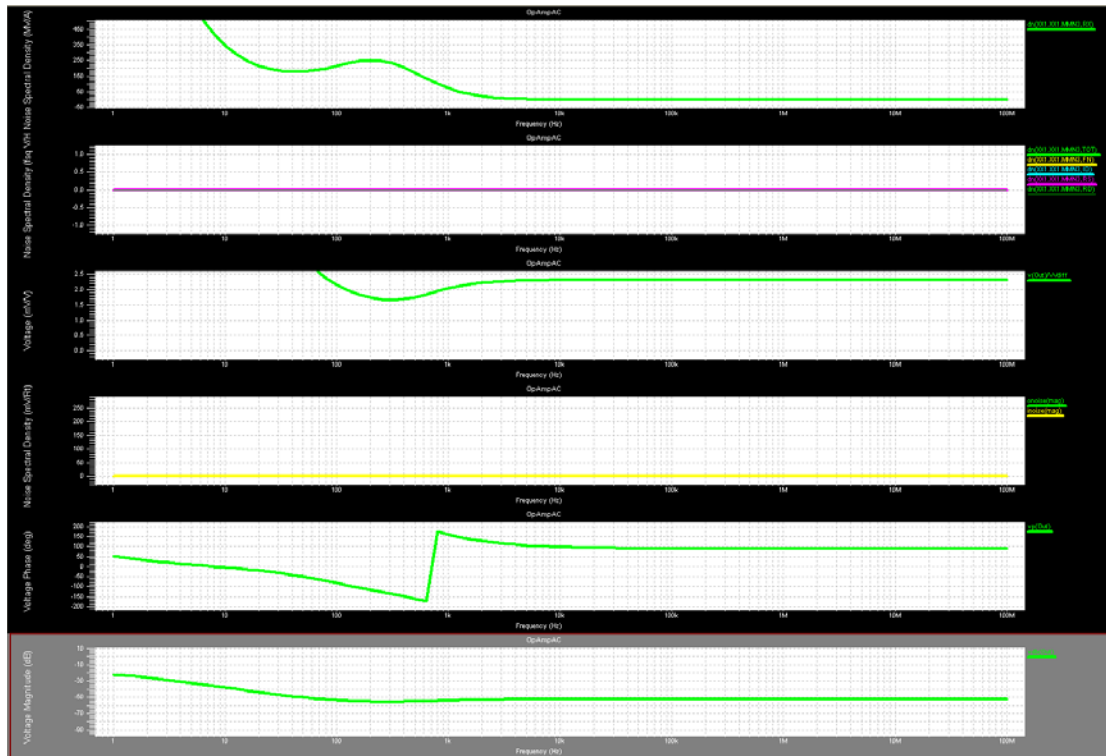
***** Simulation Settings - Additional SPICE commands *****
.end
```


Noise analysis is performed in *conjunction* with AC analysis; if the **.AC** command is missing, then the **.NOISE** command is ignored. With the **.AC** command present, the **.NOISE** command causes noise analysis to be performed at the same frequencies; in this case starting at 1 Hz and ending at 100 MHz, with 10 data points per decade.

The **.NOISE** command takes two required arguments: the *output* at which the effects of noise are to be computed, and the *input* at which the noise can be considered to be concentrated for the purpose of estimating equivalent noise spectral density. Additionally, a third optional argument specifies the frequency interval at which a noise report will be printed to the simulation log, listing every device with noise contribution.

The **.PRINT NOISE** argument **INOISE** specifies the equivalent input noise spectral density magnitude, **ONoise** outputs the noise spectral density magnitude, and **.PRINT NOISE TRANSFER** is the AC transfer function between input and output.

Output



In W-Edit, the traces show voltage magnitude, voltage phase, the output noise spectral density magnitude in several measures, voltage out over Wdiff, and (second from the top) measurements for the total device output noise, flicker noise, thermal noise due to channel, thermal noise due to source resistance and thermal noise due to drain resistance.

Example 10: MOS Transconductance Amplifier

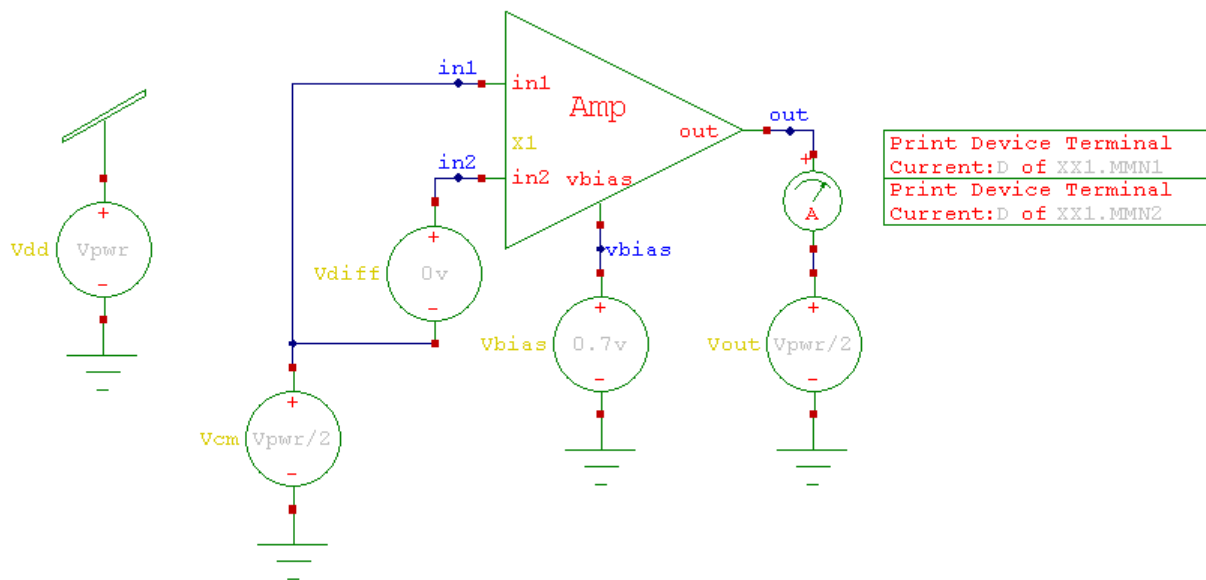
The basic transconductance amplifier produces an output current proportional in magnitude to the difference between the input voltages. Proper modeling of the transconductance amplifier's output current behavior, over a broad range of differential voltages, requires accurate subthreshold modeling of its transistors. Otherwise, the predicted transconductance characteristics will be inaccurate. This circuit can be simulated correctly in subthreshold by T-Spice.

Schematic ...\\Tanner EDA\\Tanner Tools v12.6\\T-Spice\\AnalysisExamples\\
OpAmp_TestBench Tranconductance

T-Spice Input ...\\Tanner EDA\\Tanner Tools v12.6\\T-Spice\\SimulationResults\\
OpAmpTranconductance.sp

Output ...\\Tanner EDA\\Tanner Tools v12.6\\T-Spice\\SimulationResults\\
OpAmpTranconductance.out

Schematic



T-Spice Input

```
***** Subcircuits *****
.subckt TransAmp in1 in2 out vbias Gnd Vdd
----- Devices: SPICE.ORDER < 0 -----
* Design: AnalysisExamples / Cell: TransAmp / View: Main / Page:
* Designed by: Tanner EDA Library Development Team
* Organization: Tanner EDA - Tanner Research, Inc.
* Info: Transconductance Amplifier
* Date: 7/13/2007 7:45:59 AM
* Revision: 80

----- Devices: SPICE.ORDER == 0 -----
MMP1 vm1 vm1 Vdd Vdd PMOS W=2u L=2u AS=1.8p PS=5.8u AD=1.8p PD=5.8u
MMP2 out vm1 Vdd Vdd PMOS W=2u L=2u AS=1.8p PS=5.8u AD=1.8p PD=5.8u
```

```

MMN1 vm1 in1 vn1 0 NMOS W=2u L=2u AS=1.8p PS=5.8u AD=1.8p PD=5.8u
MMN2 out in2 vn1 0 NMOS W=2u L=2u AS=1.8p PS=5.8u AD=1.8p PD=5.8u
MMN3 vn1 vbias Gnd 0 NMOS W=2u L=3u AS=1.8p PS=5.8u AD=1.8p PD=5.8u
.ends

***** Simulation Settings - Parameters and SPICE Options *****
.param Vpwr = 3.3v

*----- Devices: SPICE.ORDER == 0 -----
VVdd Vdd Gnd DC Vpwr
VVdiff in2 in1 DC 0
VVout N_1 Gnd DC Vpwr/2
VVbias vbias Gnd DC 700m
VVcm in1 Gnd DC Vpwr/2
XX1 in1 in2 out vbias Gnd Vdd TransAmp
*----- Devices: SPICE.ORDER > 0 -----
.PRINT DC ID(XX1.MMN1)
.PRINT DC ID(XX1.MMN2)
VA_Out out N_1 0v
.PRINT DC I(VA_Out)

***** Simulation Settings - Analysis section *****
.op
.dc lin VVdiff -1.00 1.00 0.01

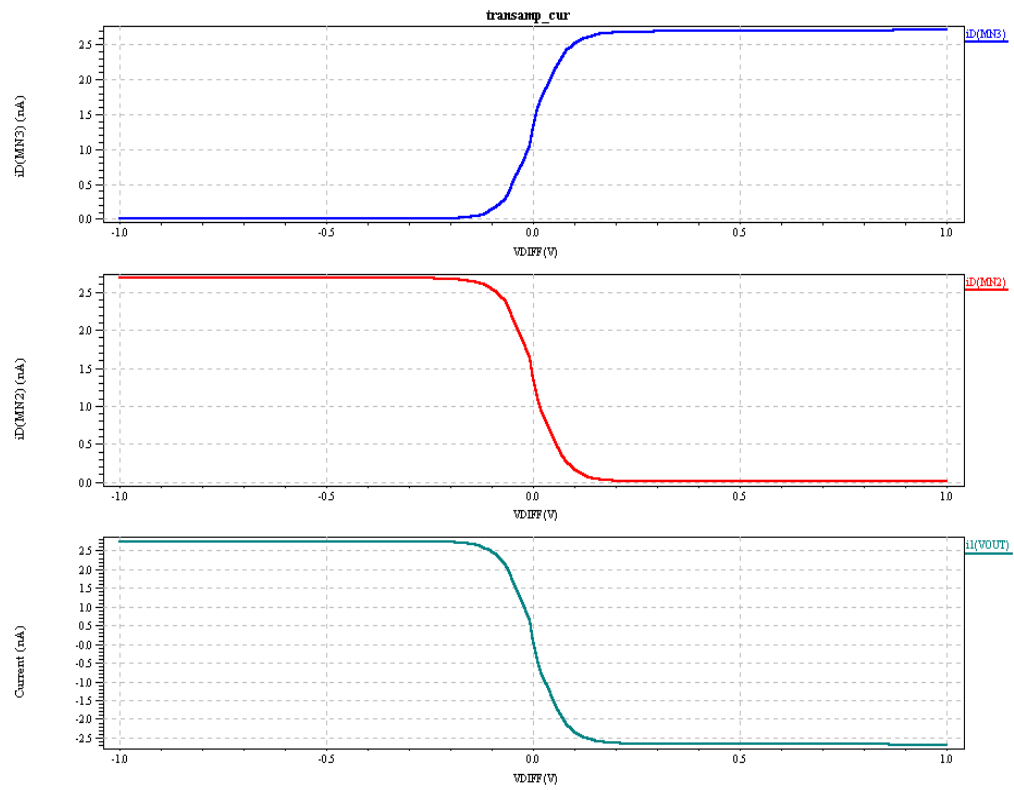
***** Simulation Settings - Additional SPICE commands *****

.end

```

This circuit is identical to the one described in [Example 4](#), except that the output stage (inverter) has been removed, and a voltage source has been connected to the output so that the transconductance characteristics of the amplifier can be measured.

Output



Design Examples

These design examples created and simulated by T-Spice Pro are intended to provide a practical synthesis of the concepts presented in the preceding examples, as well as to present some more advanced features of T-Spice. There are two tutorial projects:

...\TannerToolsShippingFiles\T-Spice\GaAsAmp\GaAsAmp.tanner	Files needed for simulation of a sample wide-band GaAs IC amplifier.
...\TannerToolsShippingFiles\S-Edit\Tutorial\RingVCO\RingVCO.tanner	Files needed for simulation of a sample voltage-controlled ring oscillator.

Wide-Band GaAs IC Amplifier

This example project describes the design and performance of a wide-band GaAs IC amplifier. The amplifier consists of two cascaded stages. It features a high-voltage gain (>20 dB), wide bandwidth, and very low input VSWR. This amplifier is useful for many signal processing and instrument/measurement applications.

The **GaAsAmp_Testbench** in **GaAsAmp.tanner** contains four examples for characterizing the AC, DC and transient performance and measuring S-parameters of the circuit:

ACAnalysis	The amplifier schematic, containing commands for AC, noise, and small-signal parameter analysis.
DCAnalysis	The amplifier schematic, containing commands for DC transfer analysis.
SParameterMeasurements	Schematic of a circuit design appropriate for measurement of the amplifier s-parameters.
TransientAnalysis	The amplifier schematic, containing commands for transient analysis.

Each of the schematics contains components to generate T-Spice analysis functions that were explored in previous examples. The transient analysis example **TransientAnalysis**, contains a polynomial voltage controlled voltage source. This provides a ramp-up sinusoidal wave as an input source.

The circuit shown in the schematic **SParameterMeasurements** is wired in the same way that s-parameters are measured. The components labeled **Amp** are instances of the basic GaAs two stage amplifier schematic **GaAsAmp**. You can view the analysis settings in the Properties window.

Voltage Controlled Ring Oscillator

This example describes the design and performance of a voltage-controlled ring oscillator. The ring VCO consists of a control stage that provides bias and tuning voltage and seven differential cells that form a ring oscillator. The tuning voltage from the control stage controls the frequency of the ring

oscillator. The applications of VCO include frequency modulators, tone generators, A/D converters, and digital voltmeters.

ringvco	The ring oscillator schematic, containing commands for transient analysis.
control	Subcircuit schematic for the bias control stage in the ring VCO.
diffcell	Subcircuit schematic for a differential amplifier.

Index

A

- AC analysis, 16
- .ac** command, 16–??, 18
- accuracy, 2
- Advanced Model Package, 2
- amplifiers
 - GaAs IC, 37
 - transconductance, 34
- analysis
 - AC, 16
 - DC operating point, 4–8
 - DC transfer, 9
 - noise, 32, 33
 - transient, 13, 26, 29
- attributes
 - defined, 4
 - PARNAM, 19
- axes
 - formatting, 25

B

- binary notation, 30
- bit** keyword, 24
- bitwise input, 24
- bus** keyword, 30

C

- C**, key letter, 6
- capacitor, 6
- charts
 - expand/collapse operations, 11
 - new, 24
- collapsing charts, 11
- convergence, 2, 28

D

- .dc** command, 10
- DC operating point analysis, 7
- dec** keyword, 18
- decimal notation, 30
- dialogs
 - Simulation Manager, 8
 - Trace Properties, 11, 12
- digital circuit example, 23
- DxDesigner
 - attributes, 4

- subcircuits, 19

E

- .ends** command, 21
- examples
 - AC Analysis, 16–18
 - DC Operating Point Analysis, 4–8
 - DC Transfer Analysis, 9, 9–12
 - MOS Transconductance Amplifier, 34–36
 - Noise Analysis, 32–33
 - Subcircuits, 19–22
 - Transient Analysis, 13–15, 23–25
 - Transient Analysis, Powerup Mode, 26–28
 - Transient Analysis, Preview Mode, 29–31
 - Voltage Controlled Ring Oscillator, 37–??
 - Wide-Band GaAs IC Amplifier, 37
- expanding charts, 11
- export netlist, 5

F

- frequency
 - sweeping, 18

G

- GaAs amplifier, 37

H

- hexadecimal notation, 30

I

- .ic** command, 21
- .include** command, 6
- initial conditions, 21, 26
- input, previewing, 29
- instance
 - subcircuit, 21

K

- key letter, 6
 - C**, 6
 - M**, 6
 - X**, 21

L

lin keyword, 10
 logarithmic sweep, 18
 low-current simulation, 35

M

M, key letter, 6
 macromodels, 2
.model command, 6–7
 model files, 6–7
 MOSFET
 T-Spice definition, 6

N

netlist, exporting, 5
 noise analysis, 32, 33
.noise command, 33

O

octal notation, 30
.op command, 7
.options command, 35
 oscillator, voltage-controlled, 37
 output files
 opening, 8

P

parameters
 passing through hierarchy, 19
 subcircuit, 21
 sweeping, 9–10
 PARNAM, 19
powerup keyword, 26, 28
preview keyword, 29
.print command, 9, 18
.print noise command, 33
pulse keyword, 14
 pulse waveform, 14

R

ring VCO, 37
 run simulation, 7

S

simulation commands
 .ac, 18
 .dc, 10
 .ends, 21

.ic, 21
 .include, 6
 .model, 6–7
 .noise, 33
 .op, 7
 .options, 35
 .print, 9, 18
 .print noise, 33
 .subckt, 21
 .tran, ??–15
 .tran/powerup, 26, ??–28
 .tran/preview, 29, 31
 .vector, 30
 simulation, running, 7
sin keyword, 30
 s-parameters, 37
 speed, 2
 stability, problems with, 28
 subcircuit, 19–22
 definition, 21
 described, 19
 DxDesigner, 19
 instance, 21
 parameters, 21
.subckt command, 21
 subthreshold
 behavior
 transconductance amplifier, 34
 modeling, 34
 sweeps, 9
 frequency, 18
 linear, 10
 logarithmic, 18
 symbols
 attributes, 4, 19

T

tables, 2, 7
 traces
 loading, 24
 order of, 24
 properties, 11, 12
.tran command, ??–15
.tran/powerup command, 26, ??–28
.tran/preview command, 29, 31
 transconductance amplifier, 34
 transfer analysis, 9
 transient analysis, 23
 default mode, 13
 op mode, 13
 powerup mode, 26, 28
 preview mode, 29
 T-Spice
 accuracy, 2
 convergence, 2

speed, 2

U

unstable circuits, 28

V

variables

sweeping, 9–10

.vector command, 30

voltage source

bit, 24, 30

bus, 30

DC, 6

piecewise linear, 30

polynomial, 37

pulse waveform, 14

repeating bit, 30

sinusoidal, 30

voltage-controlled, 37

W

W-Edit

add chart, 24

axes, formatting, 24

traces, 11, 12, 24

X

X key letter, 21

Credits

Software Development

Ken Van de Houten

Quality Assurance

Luba Gromova
Ken Van de Houten

Lena Neo

Documentation

Judy Bergstresser

Ken Van de Houten