Chapter 5 Analog Design and Simulation

This chapter provides an overview of SPICE. It also illustrates the various types of Analog Analyses.

5.1 Overview of SPICE

SPICE (Simulation Program with Integrated Circuit Emphasis) was first released in 1972. It is product of a research project that begun at the University of California at Berkely in the late 1960s. SPICE is a general purpose circuit analysis tool originally intended for the modelling of transistor-level circuits which has grown into a simulation system that can operate at virtually any level of circuit abstraction.

Commercialisation of SPICE began in the late 1970s and throughout the 1980s, as computing platforms for engineering became prevalent in the industry. It is demonstrated that SPICE, with graphical design tools such as schematic entry and waveform editors is an integral part of system level design.

SPICE is an Analog Simulation standard. Virtually all analog simulators available today are based on the SPICE technology.

Analog Simulation allows you to model the electrical characteristics of a circuit and analyse its behaviour under various conditions before you build any actual hardware. An analogue simulator applies mathematical modelling and numerical analysis to the know characteristics of circuits and semiconductors to produce an approximation of signal values at given points in time.

The difference between digital and analogue simulation lies primarily in how the values of the various signals are modelled in the computer. An analog simulator is capable of calculating and storing actual voltages and currents for selected signals in the circuit, using complex numerical algorithms and formulas. On the hand, digital simulator only represents signal values using a small number of possible values.

There are many version of SPICE available in the market e.g. HSPICE (Meta-Software), PSpice (MicroSim), Spectre (Cadence), AccuSim (Mentor Graphics), ViewSpice (View Logic), etc.

There are many types of analysis that you can perform using SPICE:

- *DC Operating Point Analysis* or *DC Analysis* is used for circuits with time-invariant sources e.g. steady state dc source. It calculates all node voltages and branch currents over a range of values, and their quiescent (dc) values are the outputs.

- *DC Sweep Analysis* performs a succession of DC operating point analyses while sweeping the value of an input source over a specified range.

- *Transient Analysis* is used for circuits with time-variant sources (e.g. ac sources and switched dc sources). It calculates all node voltages and node branch currents over a time interval, and their instantaneous values are the outputs.

- *AC Analysis* is used for small-signal analysis of circuits with sources of variable frequencies. It calculates all node voltages and branch currents over a range of frequencies, and their magnitudes and phase angles are the outputs.

- *Noise Analysis* is done in conjunction with AC analysis or Transient analysis. It is used to determine the noise level generated by the element in the circuits and their contributions to the output nodes.

- *DC Sensitivity Analysis* is used to determine how sensitive the operating point of a circuit is to changes in component values.

- *Small Signal DC Transfer Function Analysis* is used to determines the input resistance, gain and output resistance for a given input source and output variable of a circuit.

In this course, we will focus on the first four type of analysis.

Circuit Description

Not until recently, circuit is described to a computer by using a file called the *circuit file*, which is normally typed in from a keyboard. However as computer acquire more and more processing power, circuit is input to the computer through schematic capture program. These programs convert the circuit that you have drawn to SPICE circuit file so that the computer can process further. Therefore it is still important to know how to read circuit file. SPICE_NET is the standard use for analog simulator.

A circuit file contains the circuit details of components and elements, the information about the sources, and the commands for what to calculate and what to provide as output.
The circuit file used to be the input file to the SPICE program, which after executing the commands, produces the results in another file called the *output file*.

The circuit must be specified in terms of element names, element values, nodes, variable parameters, and sources. Consider the circuit in Figure 5.1 that is to be simulated for calculating all node voltages and currents through R2 and R3. We shall describe this circuit using SPICE_NET.

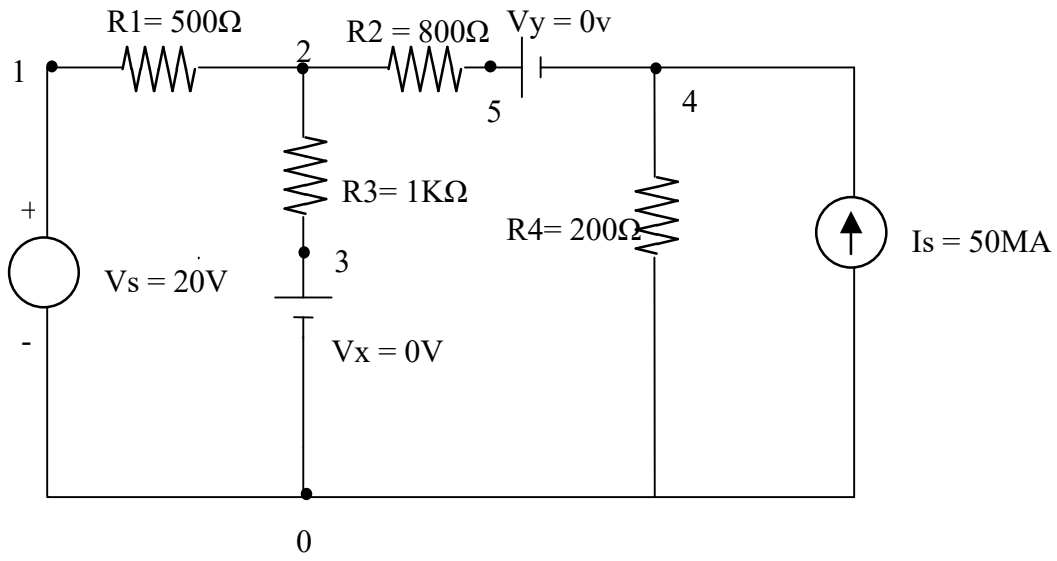

Figure 5.1 A simple DC circuit

Circuit Elements

Circuit elements are identified by names. A name must start with a letter symbol corresponding to the elements. Table 5.1 shows the first letters of elements and sources. For example the name of a resistor must start with R, an independent current source with I, and an independent voltage source with V and so on.

A circuit must be specified in terms of element names and element values,

First Letter	Circuit elements and sources
B	GaAs MEC field-effect transistor
C	Capacitor
D	Diode
E	Voltage-controlled voltage source
F	Current-controlled current source
G	Voltage-controlled current source
H	Current-controlled voltage source
I	Independent current source
J	Junction field effect transistor
K	Mutual inductors (transformer)
L	Inductor
M	MOS field effect transistor
Q	Bipolar junction transistor
R	Resistor
S	Voltage-controlled switch
T	Transmission line
V	Independent voltage source
W	Current controlled switch

Table 5.1 First letter of elements

Element Values

There are two types of suffixes, scale suffix and units suffix. The scale suffixes that are normally used are:

F = 1E-15, P = 1E-12, N = 1E-9, U = 1E-6, M = 1E-3,
K = 1E3, MEG = 1E6, G = 1E9, T = 1E12

Using the scale suffix you may express the same value in many ways:
1050000 = 1.05E6 or 1.05MEG or .00105G

The unit suffixes that are normally used are:
V = volt, A = amp, HZ = hertz, OHM = Ω, H = henry, F =farad

Nodes

The node number identifies the location of an element. Each element is connected between two nodes. First, let us assign node numbers to the circuit in Figure 5.1. Node 0 is pre-defined as the ground.

A circuit must be specified in terms of element names & element values. The passive elements in Figure 4.1 are described as follows:

- R1 has a value of 500 Ω and is connected between nodes 1 and 2,
  R1 1 2 500
- R2 has a value of 800 Ω and is connected between nodes 2 and 5,
  R2 2 5 800
- R3 has a value of 1 kΩ and is connected between nodes 2 and 3,
  R3 2 3 1K
- R4 has a value of 200 Ω and is connected between nodes 4 and 0,
  R4 4 0 200
- Vs has a value of 20V, node 1 is a higher potential with respect to node 0,

Vs 1 0 DC 20V

- Is has a value of 50 mA, Is flows from node 0 to node 4,
  Is 0 4 DC 50MA

Types of Analysis and Output statement

For the older spice version, there is a need to indicate in the circuit file, the type of analysis and output desired. The statements for different type of analysis are as follows:

.OP	DC operating-point analysis
.DC	DC sweep analysis
.AC	AC analysis
.NOISE	Noise analysis
.SENS	DC sensitivity analysis
.TF	Small signal DC transfer function
.TRAN	Transient Analysis

Each analysis is invoked by including its command statement. Figure 5.1 is a dc circuit, we are concerned with dc analysis only. Whenever a circuit file is run, SPICE always calculates the dc bias point, which consists of all node voltages and the current through all voltage sources. The statement for Figure 5.1 is therefore ***.OP***.

In the original version of SPICE, output is obtained as tabulated values or as printer or plots. For example, if we have a program that performs a DC sweep analysis of the value of the source, we can obtain a tabulation of the voltages at a given node, say node 4, with respect to ground by using the statement

.PRINT DC V(4)

Similarly, the statement

.PLOT DC V(4)

results in a printer plot of the voltage at node 4 versus the source values.

Commercial versions of SPICE have improved the capability to obtain and display results. It allows data to be displayed graphically after completion of a SPICE analysis. It is not necessary to decide in advance what data are of interest. Output for the Probe program is requested by including the statement

.PROBE

in the SPICE program. After the analysis is completed, the Probe program is executed and the results can be observed using menu commands.

For the circuit in Figure 5.1, to print the required voltage, the statement is as follows:

.PRINT DC V(1) V(2) V(3) V(4) I(VX) I(VY)

Format of Circuit Files

A circuit file that can be read by SPICE may be divided into five parts:

1. Title which describes the type of circuit or any comments.
2. Circuit description that defines the circuit elements and the set of model parameters.
3. Analysis description that defines the type of analysis that you want to perform.
4. Output description that defines the way the output is to be presented.
5. End of program (the .END command)

We have discussed all the details of describing the circuit of Figure 5.1 as a SPICE input file. The circuit of Figure 4.1 is to be simulated on SPICE to calculate and print all node voltages and the current and power of all voltage sources (Vs, Vx and Vy). The name of this file is EX4-1.CIR. The description file is as follow:

Circuit description for Figure 5.1

```
Title
VS  1  0  DC  20V          ;  DC voltage source of 10V
IS  0  4  DC  50MA         ;  DC current source of 50mA
R1  1   2   500            ;  Resistance of 500Ω
R2  2   5   800            ;  Resistance of 800Ω
R3  2   3   1KOHM          ;  Resistance of 1kΩ
R4  4   0   200            ;  Resistance of 200Ω
VX  3  0  DC  0V           ;  Measure current through R3
VY  5  4  DC  0V           ;  Measure current through R2
.OP                        ;  Perform DC operating point analysis
.PRINT DC V(1) V(2) V(3) V(4) I(VX) I(VY)    ;  Prints the results  of dc analysis
.END                       ;  End of circuit file
```

Schematic capture using Cadence Design Systems' Virtuoso Schematic Editor

Figure 5.2 shows the schematic of a op-amp circuit captured using Virtuoso Schematic Editor.

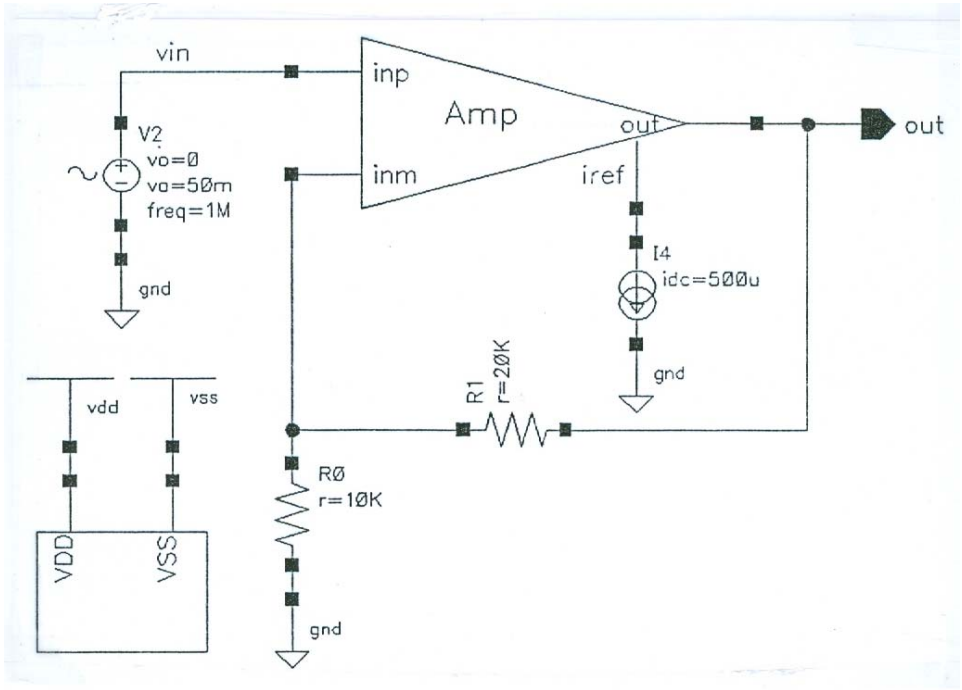

Figure 5.2 Circuit captured using Virtuoso Schematic Editor

After entering the design into the system, Virtuoso software would generate a netlist for the circuit when you run the simulation.

Modeling of Elements

The models are necessary to take into account the parameter variations. A model that specifies a set of parameters for an element is specified by the .MODEL command. The same model can be used by one or more elements in the same circuit.

The general form of the model statement is:

.MODEL MNAME TYPE (P1=A1 P2=A2 …..PN=AN), where

P1, P2, ... are element parameters and A1, A2, ... are their values, respectively.
MNAME is the name of the model and must start with a letter.
TYPE is the type name of the elements and must have the correct type, as shown in Table 4.2.
An element must have the correct model type name. Example, a resistor must have the type name RES, not the type IND or CAP.

Type	Elements
RES	Resistor
R	RC wire
CAP	Capacitor
IND	Inductor
NPN	NPN bipolar junction transistor
PNP	PNP bipolar junction transistor
LPNP	Lateral PNP bipolar junction transistor
D	Diode
NMOS	N-channel metal oxide field effect transistor
PMOS	P-channel metal oxide field effect transistor
NJF	N-channel junction field effect transistor
PJF	P-channel junction field effect transistor
NSW	N-type switch (SC)
PSW	N-type switch (SC)
OPA	Operational amplifier (SC)
MODFAS	Analogue macromodel
LOGIC	Digital Gate

Table 5.2 Type name of the elements

Some model statements as shown below:

```
.MODEL  RMOD   RES  (R=1.1  TCE=0.001)
.MODEL  RLOAD RES  (R=1 TC1=0.02  TC2=0.005)
.MODEL  CPASS  CAP (C=1 VC1=0.01  VC2=0.005  TC1=0.02 TC2=0.005)
.MODEL  LFILTER  IND  (L=1 IL1=0.1   IL2=0.002  TC1=0.02  TC2=0.005)
.MODEL  QOUT  NPN  (BF=50  IS=1E-9)
```

Note: R is resistance multiplier, TC1 is linear temperature coefficient, TC2 is quadratic temperature coefficient and TCE is exponential temperature coefficient. VC1 & VC2 are the linear voltage coefficient and IL1 & IL2 are the linear current coefficient.

5.2 DC Operating Point Analysis (DC Analysis)

DC Operating Point Analysis or *DC Analysis* is used for circuits with time-invariant sources e.g. steady state dc source. It calculates all node voltages and branch currents over a range of values, and their quiescent (dc) values are the outputs.

Sources in dc circuits are constant voltage or current sources that are referred to as *direct current* or *dc* sources. In the electrical circuit module, you have been introduced the basic circuit laws and techniques such as, Kirchhoff's voltage and current laws, node-voltage and mesh-current methods, and Thevenin and Norton equivalents.

In dc analysis, all the independent and dependent sources are dc types. If inductors and capacitors are present in a circuit, they are considered as short circuits and open circuits, respectively, because at zero frequency, the impedance represented by an inductor is zero and that of capacitor is infinite.

The commands that are commonly used for dc analysis are:

```
.OP                 ; DC operating point
.TF   VOUT VIN ; Small-signal transfer function
.SENS               ; DC sensitivity analysis
```

DCOP Analysis using Virtuoso Analog Design Environment

Prior to performing any analysis, Virtuoso software must determine the bias-point (DC operating point) of a circuit with inductors shorted and capacitors opened. DC operating Point (DCOP) is the default analysis when you do not specify an analysis type. The DCOP analysis can be omitted for the transient analysis when using initial conditions to preset bias levels.

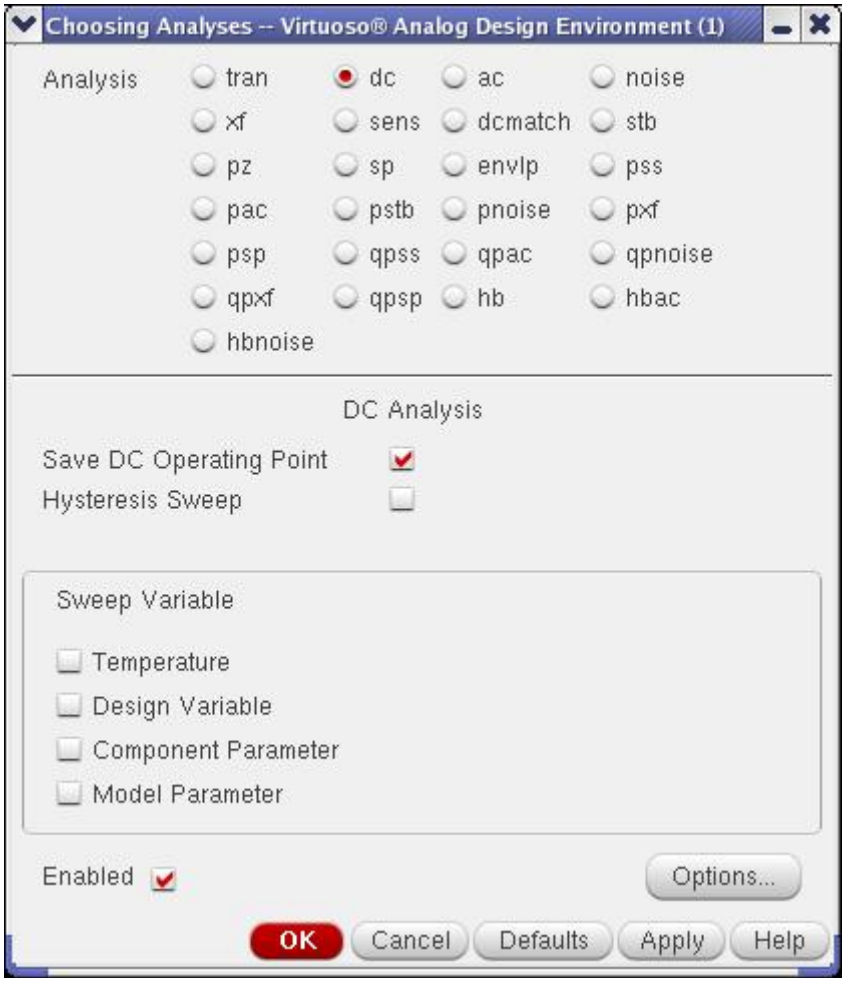

Figure 5.3 DCOP Analysis

5.3 DC Sweep Analysis

DC Sweep Analysis performs a succession of DC operating point analyses while sweeping the value of an input source over a specified range.

The statement for DC sweep analysis is as follows:

.DC LIN SWNAME SSTART SEND SINC

LIN stand for linear (you can have OCT and DEC which stands for octave and decade respectively), SWNAME stand for the sweep variable, SSTART stand for the starting point, SEND stand for the ending point and SINC stand for the increment.

An example of the above statement is :

.DC LIN IIN 50MA 150MA 1MA; Sweep the current linearly with 1mA increment

DC Sweep Analysis using Virtuoso Analog Design Environment

The Spectre simulator interface offers a powerful, yet simple way to sweep a parameter. You can sweep frequency, temperature, control parameters, and model parameters. You also specify the starting voltage or current, the corresponding ending value, and the increment amount between analysis points. Virtuoso software allows DC sweeps in both positive and negative increments in case your circuit has multi-stable solution points.

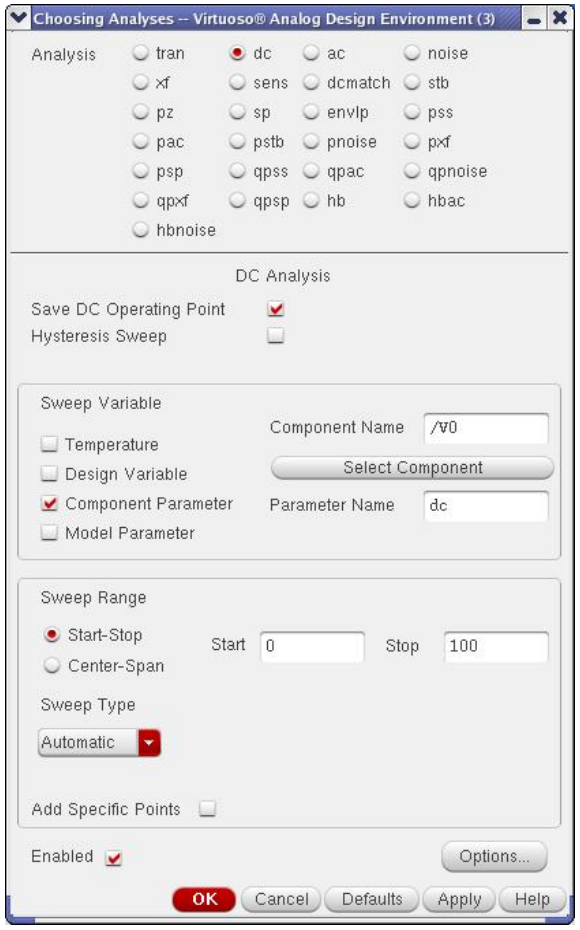

Figure 5.4 DC Sweep Analysis

5.4 Transient Analysis

A transient analysis deals with the behaviour of an electric circuit as a function of time. If a circuit contains an energy storage element(s), a transient can also occur in a dc circuit after a sudden change due to switches opening or closing.

Transient Sources

SPICE allows the generation of dependent (or independent) voltage and current sources. Independent sources can be time-variant. A non-linear sources can be simulated by a polynomial. The independent voltage and current sources such as EXPONENTIAL, PULSE, PIECEWISE LINEAR and SINUSOIDAL can be modeled by SPICE.

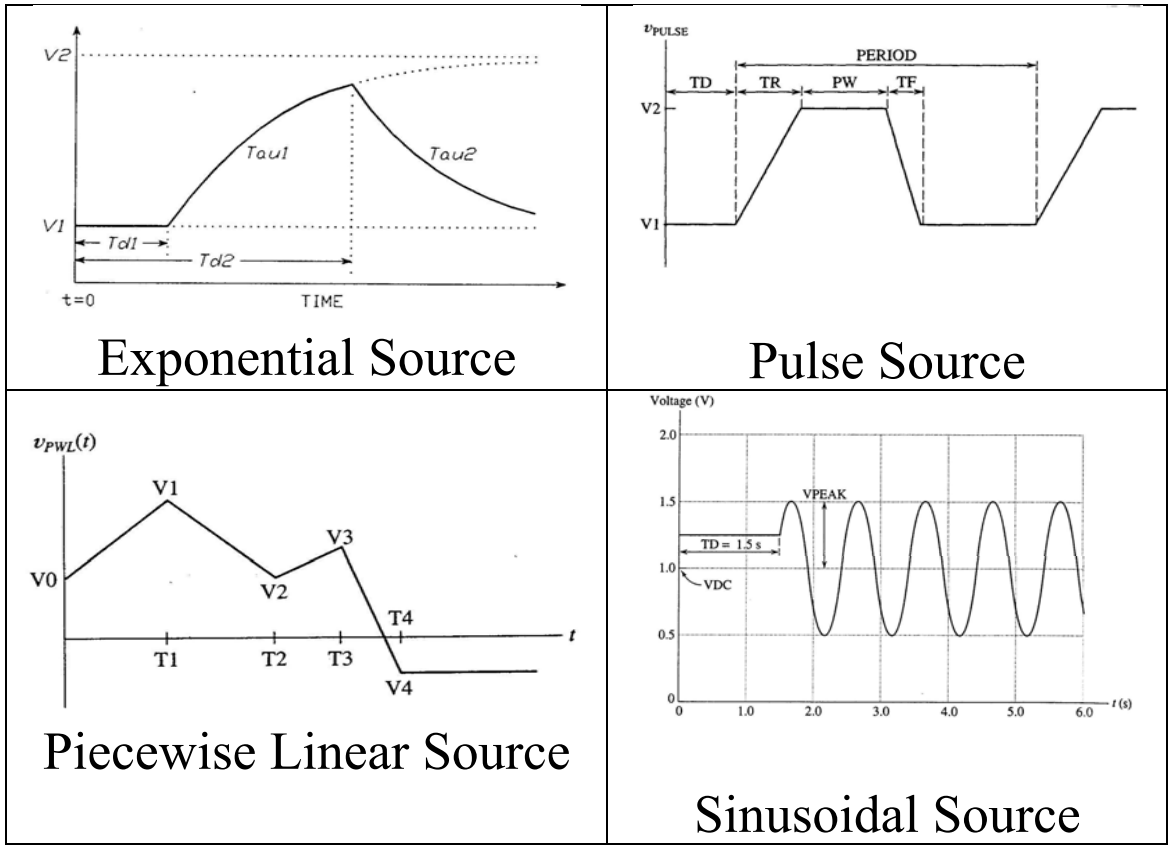

Figure 4.5 Transient Sources

Initial Condition

The various nodes can be assigned to initial voltages during transient analysis, and the general form for assigning initial values is

.IC V(1) = A1 V(2) = A2 V(3) = A3 ... V(N) = AN

where A1, A2, A3, ... are the initial values for node voltages V(1), V(2), V(3), ... respectively. These values are used to calculate the transient bias point and linearised parameters of non-linear devices.

Example of statement for Initial Transient Conditions

.IC V(1) = 2.5 V(5) = 1.7V V(7) = 0.5V

Transient Analysis Parameters

Transient analysis can be performed by the .TRAN command, which has the general form of :

.TRAN TSTEP TSTOP [TSTART TMAX] [UIC]

where TSTEP is the printing increment, TSTOP is the final time (or stop time), TMAX is the maximum size of the internal time step, and UIC means use initial condition.

Example of statement for Transient Analysis

.TRAN 5US 1MS 200US 0.1NS UIC

Transient Analysis using Virtuoso Analog Design Environment

Figure 4.6 shows the interface for Virtuoso software.

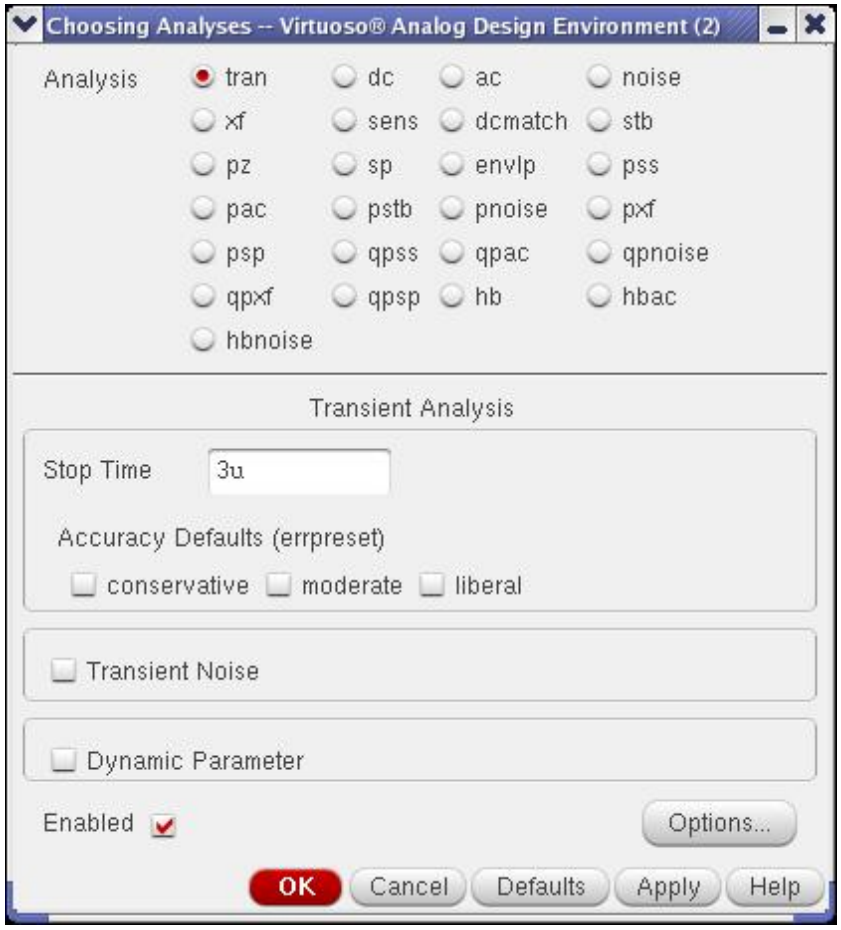

Figure 5.6 Transient Analysis

When Virtuoso software performs a transient analysis, the circuit's transient output responses are mapped as a function of time. It automatically performs a DC operating point analysis prior to a transient analysis, except when the Use Initial Conditions option is applied. The Setup Transient Analysis form specifies a transient analysis, defines the duration of the analysis, and the minimum number of analysis points returned by the analog simulation server. Making the time step shorter generally improves the resolution of displayed waveforms and accuracy on a chart, but also increases the run time.
Options for the Setup Transient Analysis form are:

time_step	Value that specifies the time increment between points.
stop_time	Value that specifies the length of the transient analysis.
max_timestep	Optional number that specifies the maximum time step the simulator takes.

Use initial condition	Switch that specifies to use user-defined initial conditions rather than computing a DC operating point solution before starting the simulation. These initial conditions are defined by initial_condition properties, IC=xx statements in the object properties for some devices.

Forces

A force is an individual stimulus that is applied to a net. Apply AC, DC, exponential, pulse, piece-wise linear, single-frequency FM, and sinusoidal stimulus to the design using Setup>Stimuli commands. In the setup analog stimuli form, make sure the stimulus type is set to Inputs. The menu is show in Figure 5.7.

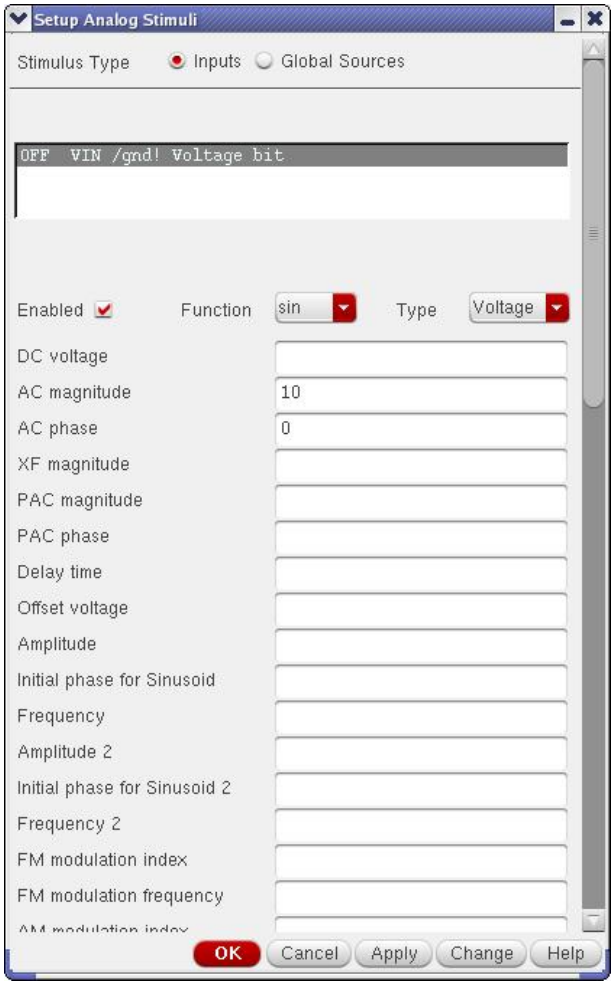

Figure 5.7 Setup Stimulus

5.5 AC Analysis

Sources in ac circuits are time-variant. They are alternating current (AC) and have both magnitude and phase displacement. The behaviour of a circuit is normally evaluated with a sinusoidal source.

In ac analysis, the output variables are sinusoidal quantities and are represented by complex numbers. An output variable can have magnitude in decibels, phase, group delay, real part, and imaginary part.

AC Analysis statement

The ac analysis calculates the frequency response of a circuit over a range of frequencies.

The command for performing frequency response takes one of the following forms:

```
.AC   LIN   NP   FSTART   FSTOP
.AC   OCT   NP   FSTART   FSTOP
.AC   DEC   NP   FSTART   FSTOP
```

where NP is the number of points in a frequency sweep, FSTART is the starting frequency, and FSTOP is the ending frequency. Only one of the LIN, OCT, or DEC must be specified in the statement. LIN, OCT, or DEC specify the type of sweep as follows:

LIN (Linear sweep): The frequency is sweep linearly from the FSTART to FSTOP. LIN is used if the frequency range is narrow.

OCT (Sweep by Octave): The frequency is swept logarithmically by octave. OCT is used if the frequency range is wide.
DEC (Sweep by Decade): The frequency is swept logarithmically by decade. DEC is used if the frequency range is the widest.

Typical statement for AC Analysis:

```
.AC   LIN  200  100HZ   300HZ
.AC   OCT  20   100HZ   10KHZ
.AC   DEC  100  1KHZ    1MEGHZ
```

AC Analysis using Virtuoso Analog Design Environment

When Virtuoso software performs AC analysis, the circuit's output responses are mapped as a function of frequency. It automatically performs a DC operating point analysis prior to AC analysis and all components are linearized about the bias point.

For an AC analysis, you define the type of frequency sweep that the simulator performs, the beginning frequency, the ending frequency and the number of analysis points per logarithmic interval or analysis period.

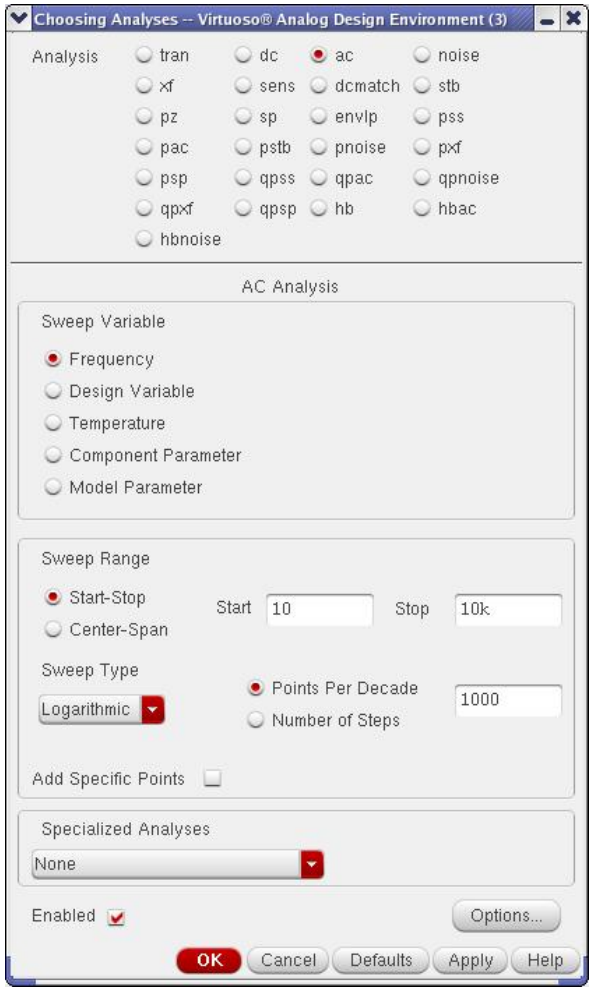

Figure 5.8 AC Analysis

· Sweep Type	Switch that specifies either a linear or logarithmic frequency sweep.
· Points per interval	User can add specific points that specifies the number of analysis points per interval or analysis period.
· Start Frequency	Value specifies the starting frequency.
· Stop Frequency	Value specifies the ending frequency.

5.6 Mixed-mode Simulation

For more than two decades, the analog IC design community has been relying on variations of the original Berkeley SPICE, introduced in the 1970s, as the simulation tool for verifying and fine-tuning analog designs. SPICE is an integration-based simulator. Although it gives good simulation accuracy, it is inherently slow, since it has to set up and solve a nonlinear system matrix for the entire circuit at every integration time point.

Analog circuits are modeled as mathematical equations which are solved by a analog simulator to analyze the circuit behavior. These equations along with the Kirchoff's current law (KCL) and Kirchoff's voltage law (KVL), form a set of differential equations**.** Simulation of analog components involves the repeated solving of these differential equations.

Today’s high-end designs require effective and fast mixed-mode simulation. The digital circuits are typically modeled in Hardware Description Languages (HDL) like VHDL and Verilog and the analog circuits are modeled using SPICE. Mixed-signal systems have both digital and analog components which exhibit discrete and continuous-time behavior respectively. Hence a mixed-signal simulator has both analog and digital modules. For such disparate modules to work in unison an interface must be created between the two systems so that they can communicate with each other.

A mixed-mode simulator has many important components including a matrix loader, solver, digital-analog synchronizer and modules to handle IO routines. One of the critical components is the digital-analog synchronizer. Since analog entities are defined by continuous-time and digital entities by discrete-time, mixed-mode simulators have two different kernels for handling the analog and digital components separately. When the two components interact they need to be synchronized if they are to work in unison. Such interface-systems are governed by protocols known as “Synchronization Protocols”. Synchronization protocols define a set of rules so that the analog and digital systems can share the available computing resources and information between them.

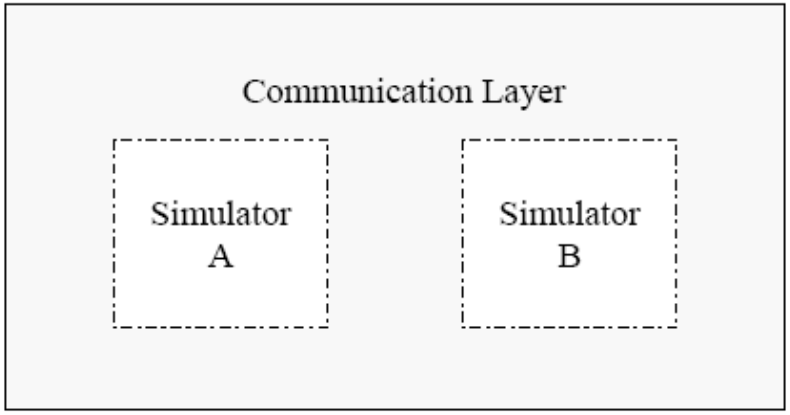

Figure 5.10 Integrated Approach

To run system-level simulations on a complex mixed-signal design and get reasonable accuracy, it is necessary to use an event-driven digital simulator to simulate the digital portion of the design, together with an integration-based analog simulator to simulate the analog circuits. The newer-generation mixed-mode simulators provide designers with both digital and analog simulation engines in one simulator. Continuous-time analog signals are computed by the analog simulator, while the digital simulator evaluates the event queue.