29 February 2020

وَمَا أُوتِيتُمْ مِنَ الْعِلْمِ إِلَّا قَلِيلًا

5 رجب 1441

Computer Aided Circuit Design

Lecture 02 – Part I
Introduction to SPICE

Dr. Hesham A. Omran

Integrated Circuits Laboratory (ICL)
Electronics and Communications Eng. Dept.
Faculty of Engineering
Ain Shams University

Simulators History

- Circuit simulators, as we know them today, first began to appear in the late 1960’s and early 70’s.
- There was an explosive growth of the integrated circuit market in the 1970’s
  - Prototypes were expensive to build and difficult to troubleshoot.
  - This triggered the rise of importance of circuit simulation.
  - Circuit simulators were necessary to evaluate designs before they were fabricated.
  - As designs became larger and more complicated, the need to use circuit simulators increased.

[Kundert, 2003]

SPICE History

- SPICE: Simulation Program with Integrated Circuit Emphasis.
- The simulation effort at Berkeley started as a class project.
- That modest beginning culminated in the release of SPICE in 1972 and then SPICE2 in 1975.
- The SPICE group at the University of California at Berkeley developed and propagated the de facto standard simulator.
- In the late 80's, Berkeley upgraded SPICE by releasing SPICE3
  - Written in C.
  - Algorithmically it was the same as SPICE2.
  - Architecturally a big step forward from SPICE2: considerably easier to add new component models.

[Kundert, 2003]

Classical SPICE References

- L. W. Nagel and R. A. Rohrer. Computer analysis of nonlinear circuits, excluding radiation (CANCER). *IEEE Journal of Solid-State Circuits*, 6(4):166–182, August 1971.
- L. W. Nagel. *SPICE2: A Computer Program to Simulate Semiconductor Circuits*. PhD thesis, University of California, Berkeley, 1975. Memorandum No. ERL-M520.

Why SPICE became very important?

1. Unlike the simulators that preceded it, SPICE had all the models one needed to simulate integrated circuits built into it. As such, it was easier to use than earlier simulators.
2. SPICE was freely distributed.
3. The source code for SPICE was made available to anyone who wanted it at a nominal cost.
4. Startup companies provided enhanced and supported robust versions of SPICE
5. Berkeley graduates took SPICE with them as they went to work at universities and electronics companies.

[Kundert, 2003]

SPICE-Based Simulators

- **HSPICE (The Gold Standard for Circuit Simulation)**
  - Meta-Software → Avanti → Synopsys
- PSpice (OrCAD)
  - Microsim → Cadence
  - PC-based free limited student version
- **LTSPICE (Free)**
  - Linear Technology → Analog Devices
- Eldo
  - Mentor
- NGSPICE
  - Open-source project
- Cadence Spectre was also influenced by SPICE

Rule #1: Don't be a SPICE Monkey!

- In absence of a simple set of equations for hand analysis, many designers tend to converge toward a "SPICE monkey" design methodology:
  - No hand calculations and no real understanding of the design trade-offs.
  - Just iterate in SPICE until the circuit "somehow" meets the specifications.
  - Typically results in sub-optimal designs, uninformed design decisions, etc.
- A SPICE monkey is someone who does not use hand analysis/equations to figure out how to design a circuit.
  - He rather plugs stuff into SPICE and uses whatever value works.
- Remember!
  - Circuits are designed by people, not computers.
  - Use the computer to verify your circuit, not to design it for you.
  - You must understand the options and limitations of your CAD tool.

Rule #2: Garbage in, Garbage out!

- Simply, the output of the simulator depends on your input.
- If the models are wrong → the results are wrong.
- If your schematic (netlist) is wrong → the results are wrong.
- If the simulator options are wrong → the results are wrong.
- Example:
  - If the Total Harmonic Distortion (THD) is much better than your expectations, check if the non-linearities are modeled properly.
- Remember!
  - Circuits are designed by people, not computers.
  - Use the computer to verify your circuit, not to design it for you.
  - You must understand the options and limitations of your CAD tool.

LTspice

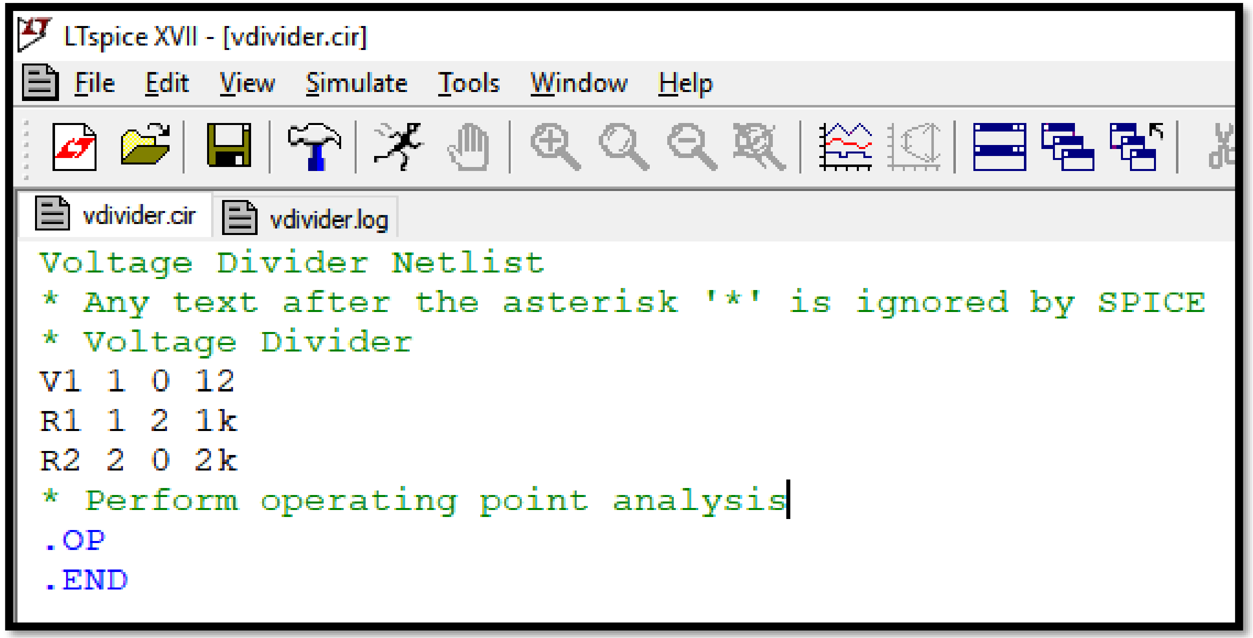
- □ The LTspice simulator was originally based on Berkeley SPICE 3F4/5.
  - ▪ The simulator has gone through a complete re-write in order to improve the performance.
- □ One of the best available free simulators.
- □ Circuits are defined by a text netlist.
  - ▪ The netlist consists of a list of circuit elements and their nodes, model definitions, and other SPICE commands.
- □ The schematic editor GUI is not the simulator.
  - ▪ The netlist information is extracted from the schematic graphical information to a file with a “.net” extension.
  - ▪ LTspice reads in and simulates this netlist.
- □ Files with the extensions ".net", ".cir", or ".sp" are recognized by LTspice as netlists.

OrCAD PSpice

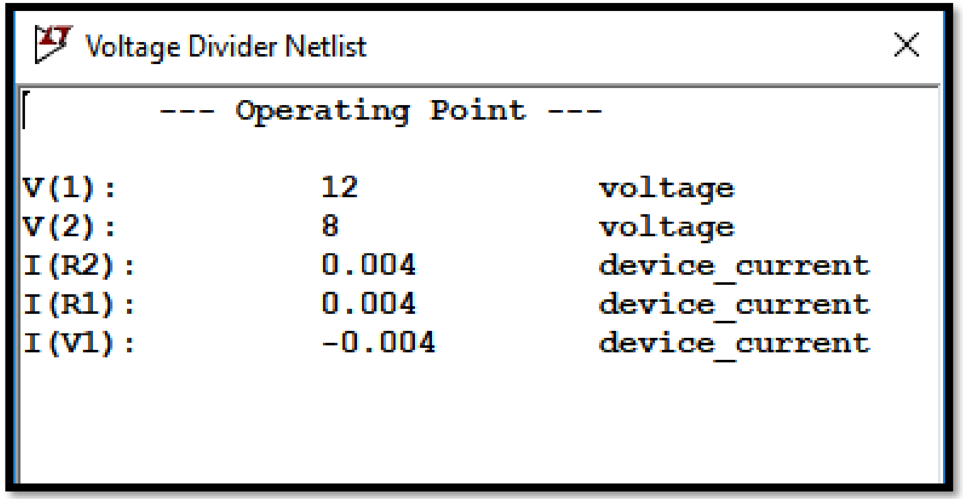
- The first PC version of SPICE (1984).
- The built-in convergence aids in PSpice are not as mature, transparent, or effective as they are in other simulators. [J. Baker]
- Student (Lite) version
  - http://www.orcad.com/products/orcad-lite-overview
- Student (Lite) version limitations
  - 75 nodes
  - 20 transistors
  - MOSFET BSIM 3.2 and BSIM 4 are not supported

LTspice Example

- ❑ The first line is the title.
  - **ALWAYS IGNORED BY SPICE**
- ❑ The first non-blank character of a line defines the type of circuit element.
- ❑ Control statements start with “.”.
- ❑ Nodes names may be arbitrary character strings.
  - Global circuit common node (ground) is “0”.
- ❑ The last line is “.END”.

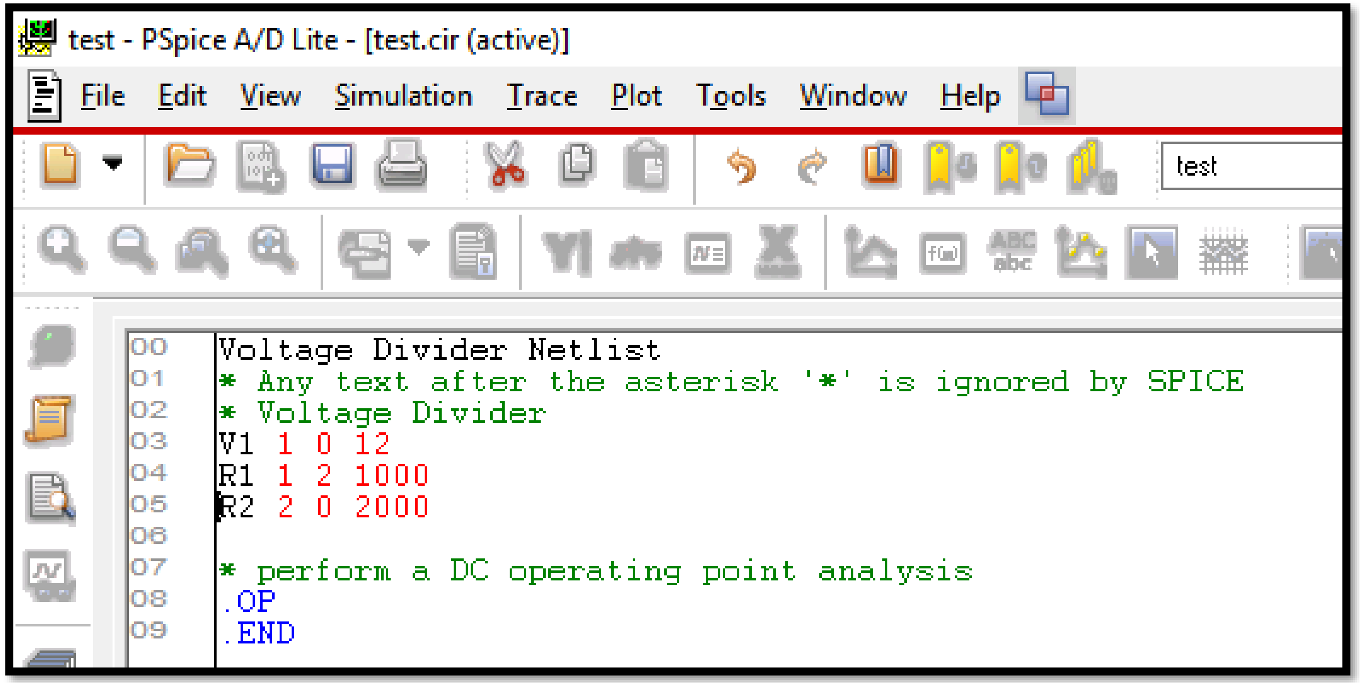

LTspice XVII - [vdivider.cir]
File Edit View Simulate Tools Window Help
vdivider.cir
vdivider.log

```
Voltage Divider Netlist
* Any text after the asterisk '*' is ignored by SPICE
* Voltage Divider
V1 1 0 12
R1 1 2 1k
R2 2 0 2k
* Perform operating point analysis
.OP
.END
```

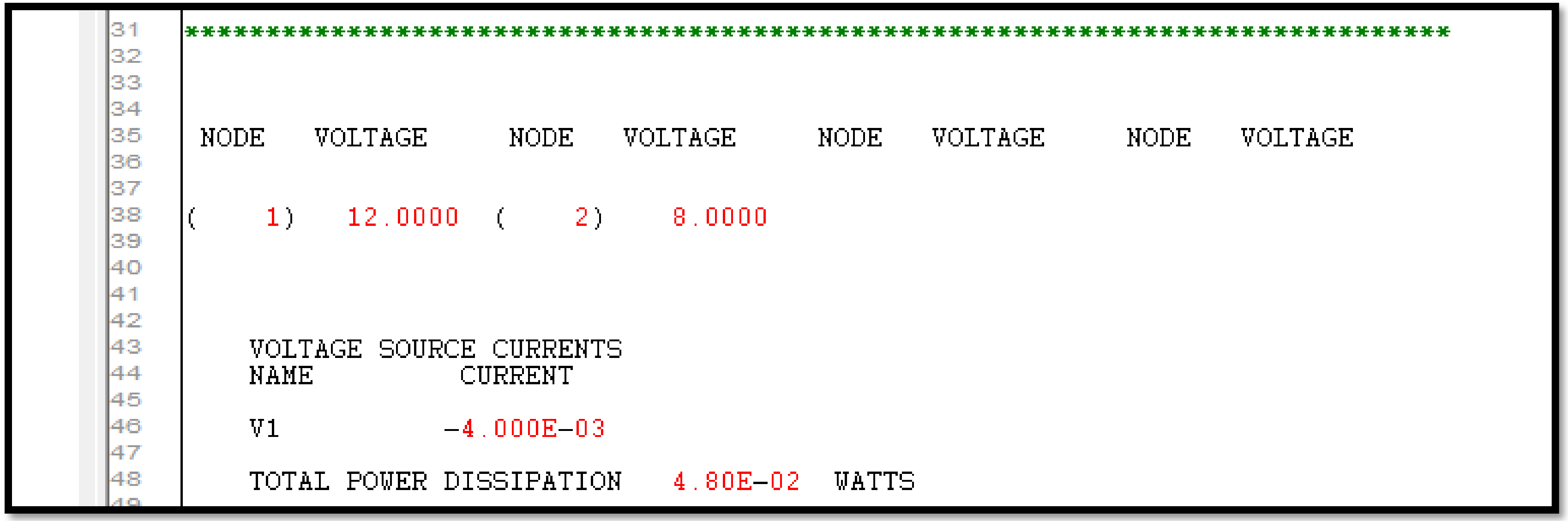


```
       --- Operating Point ---

V(1):            12            voltage
V(2):            8             voltage
I(R2):           0.004         device_current
I(R1):           0.004         device_current
I(V1):           -0.004        device_current
```

PSpice Example



```
Voltage Divider Netlist
* Any text after the asterisk '*' is ignored by SPICE
* Voltage Divider
V1 1 0 12
R1 1 2 1000
R2 2 0 2000

* perform a DC operating point analysis
.OP
.END
```



```
******************************************************************************



 NODE   VOLTAGE     NODE   VOLTAGE     NODE   VOLTAGE     NODE   VOLTAGE


(    1)   12.0000  (    2)    8.0000




    VOLTAGE SOURCE CURRENTS
    NAME         CURRENT

    V1          -4.000E-03

    TOTAL POWER DISSIPATION   4.80E-02  WATTS
```

SPICE Netlist / Deck / Input File Format

```
Title statement
     Model Statements
     Parameter Definitions
     Circuit Description
          Power supplies /signal sources
          Element Descriptions
     Analysis Requests
     Output Requests
.END
```

Why Do We Learn Netlists?

Scale Factors

- SPICE is NOT case sensitive.
- Note the difference between mega and milli!

Scale	Symbol	Name
10^{-15}	F	femto-
10^{-12}	P	pico-
10^{-9}	N	nano-
10^{-6}	U	micro-
$25.4*10^{-6}$	MIL	--
10^{-3}	M	milli-
10^{+3}	K	kilo-
10^{+6}	MEG	mega-
10^{+9}	G	giga-
10^{+12}	T	tera-

Selected Circuit Elements

- ❑ The first non-blank character of a line defines the type of circuit element.

Letter	Device type	Letter	Device type
C	Capacitor	J	Junction FET
D	Diode	L	Inductor
E	Voltage-controlled voltage source	M	MOSFET
F	Current-controlled current source	Q	Bipolar transistor
G	Voltage-controlled current source	R	Resistor
H	Current-controlled voltage source	V	Independent voltage source
I	Independent current source	X	Subcircuit instantiation

Selected Function Expressions

- ❑ See LTspice/PSpice help for a full list of functions.

Function	Meaning	Comments
ABS(x)	|x|	
COS(x)	cos(x)	x in radians
DDT(x)	time derivative of x	transient analysis only
EXP(x)	e^x	
LIMIT(x,min,max)	Clipping (limiter)	result is min if x < min, max if x > max, and x otherwise
LOG(x)	ln(x)	log base e
LOG10(x)	log(x)	log base 10
MAG(x)	magnitude of x	this produces the same result as ABS(x)
MAX(x,y)	maximum of x and y	
MIN(x,y)	minimum of x and y	
PH(x)	phase of x	returns 0.0 for real numbers
PWR(x,y)	$\|x\|^y$or, {x**y}	the binary operator ** is interchangeable with PWR(x,y).
SIN(x)	sin(x)	x in radians
SQRT(x)	$x^{1/2}$	
TAN(x)	tan(x)	x in radians

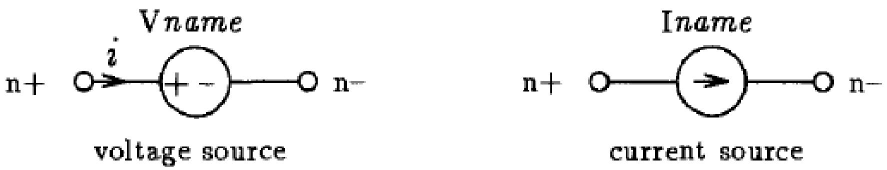
[PSpice Documentation]

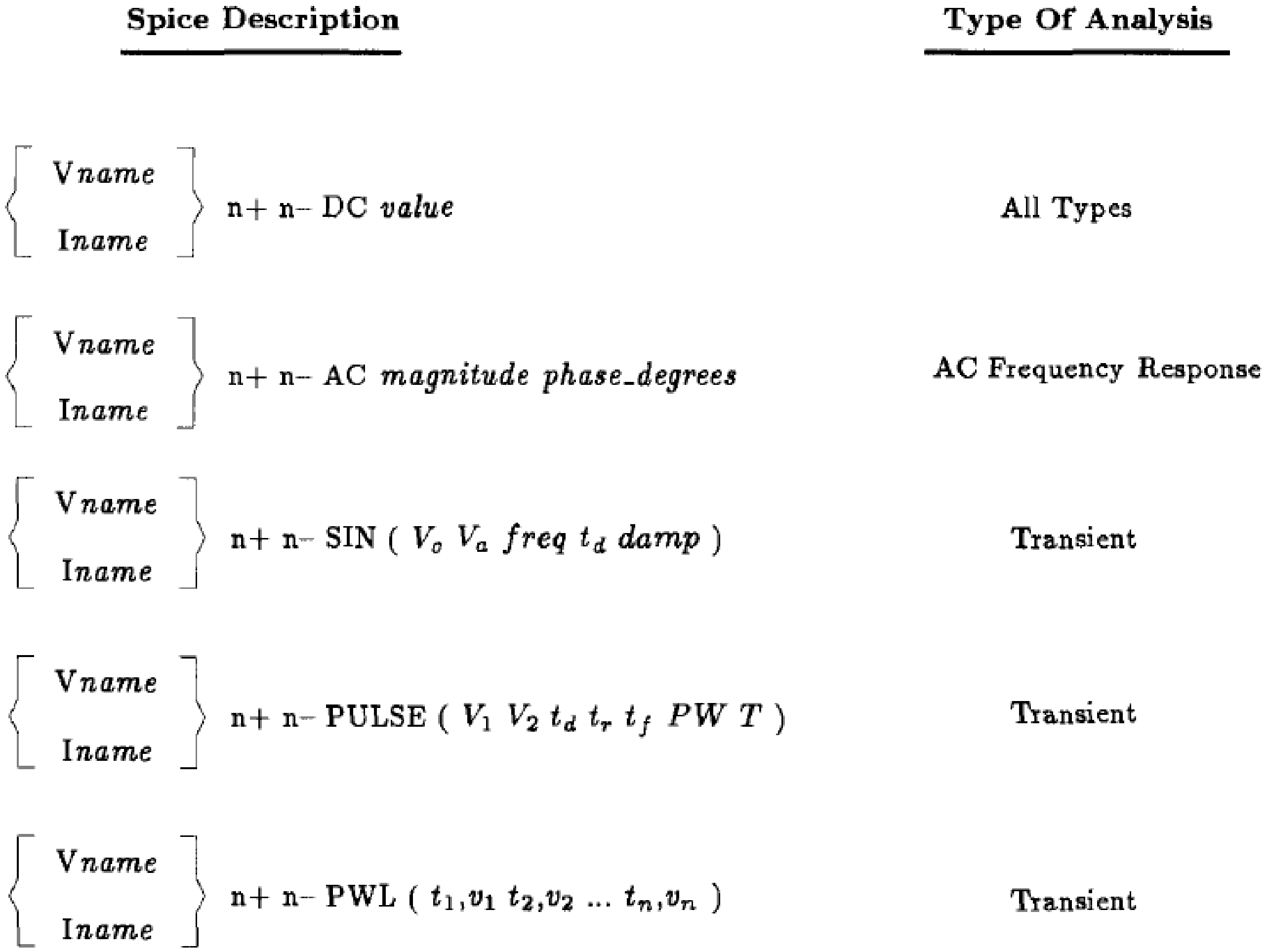
Selected Operators

Operators	Meaning
arithmetic	
+	addition (or string concatenation)
-	subtraction
*	multiplication
/	division
**	exponentiation
logical	
~	unary NOT
|	boolean OR
^	boolean XOR
&	boolean AND

[PSpice Documentation]

Independent Sources





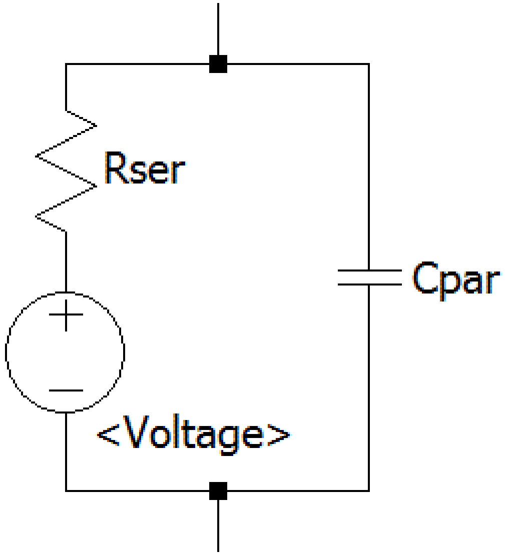
Spice Description	Type Of Analysis
$\left\{\begin{array}{c} Vname \\ Iname \end{array}\right\}$ n+ n− DC *value*	All Types
$\left\{\begin{array}{c} Vname \\ Iname \end{array}\right\}$ n+ n− AC *magnitude phase_degrees*	AC Frequency Response
$\left\{\begin{array}{c} Vname \\ Iname \end{array}\right\}$ n+ n− SIN (V_o V_a $freq$ t_d $damp$)	Transient
$\left\{\begin{array}{c} Vname \\ Iname \end{array}\right\}$ n+ n− PULSE (V_1 V_2 t_d t_r t_f PW T)	Transient
$\left\{\begin{array}{c} Vname \\ Iname \end{array}\right\}$ n+ n− PWL (t_1,v_1 t_2,v_2 ... t_n,v_n)	Transient

[Sedra, 1997]

V. Independent Voltage Source

```
Vxxx n+ n- <voltage> [AC=<amplitude>] [Rser=<value>] [Cpar=<value>]
```

- ❑ This element sources a constant voltage between nodes n+ and n-.
- ❑ For AC analysis, the value of AC is used as the amplitude of the source at the analysis frequency.
- ❑ A series resistance and parallel capacitance can be defined.
- ❑ Voltage sources have historically been used as the current meters in SPICE and are used as current sensors for current-controlled elements.
- ❑ In LTspice, the current of any circuit element, including the voltage source, can be plotted.



[LTspice Help]

I. Independent Current Source

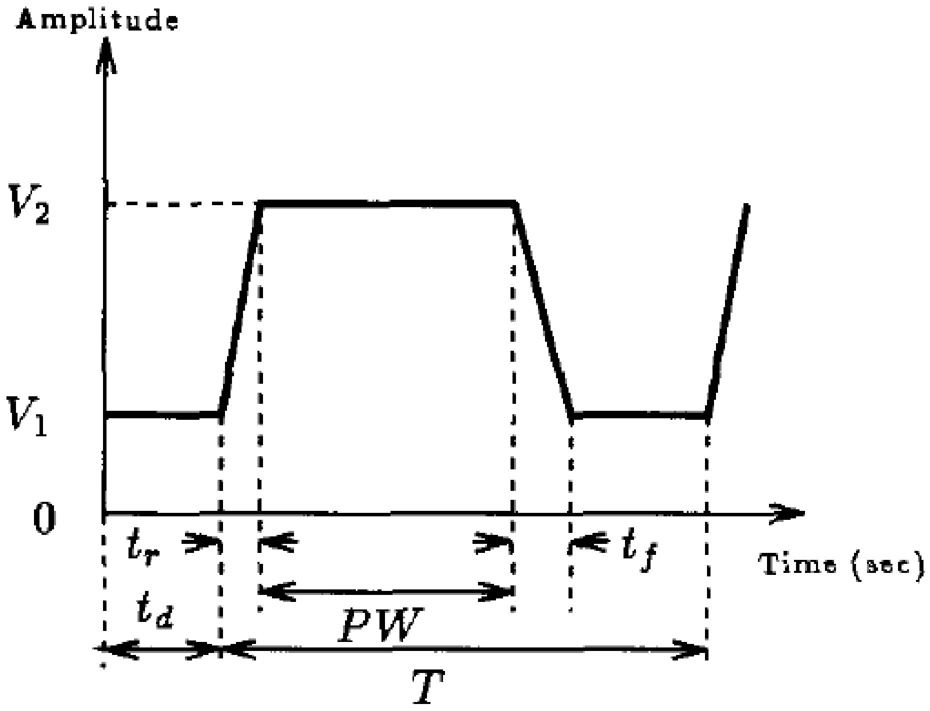
```
Ixxx n+ n- <current> [AC=<amplitude>] [load]
```

- ❑ This circuit element sources a constant current between nodes n+ and n-.
- ❑ If the source is flagged as a load, the source is forced to be dissipative.
  - The current goes to zero if the voltage between nodes n+ and n- goes to zero or a negative value.
  - The purpose of this option is to model a current load on a power supply that doesn't draw current if the output voltage is zero.
- ❑ For AC analysis, the value of AC is used as the amplitude of the source at the analysis frequency.

[LTspice Help]

Independent Source: PULSE

$\left\{\begin{matrix} \mathrm{V}name \\ \mathrm{I}name \end{matrix}\right\}$ n+ n− PULSE (V_1 V_2 t_d t_r t_f PW T)   Transient



$\mathrm{V}name$ n + n − PULSE (V_1 V_2 t_d t_r t_f PW T)

[Sedra, 1997]

Independent Source: PULSE

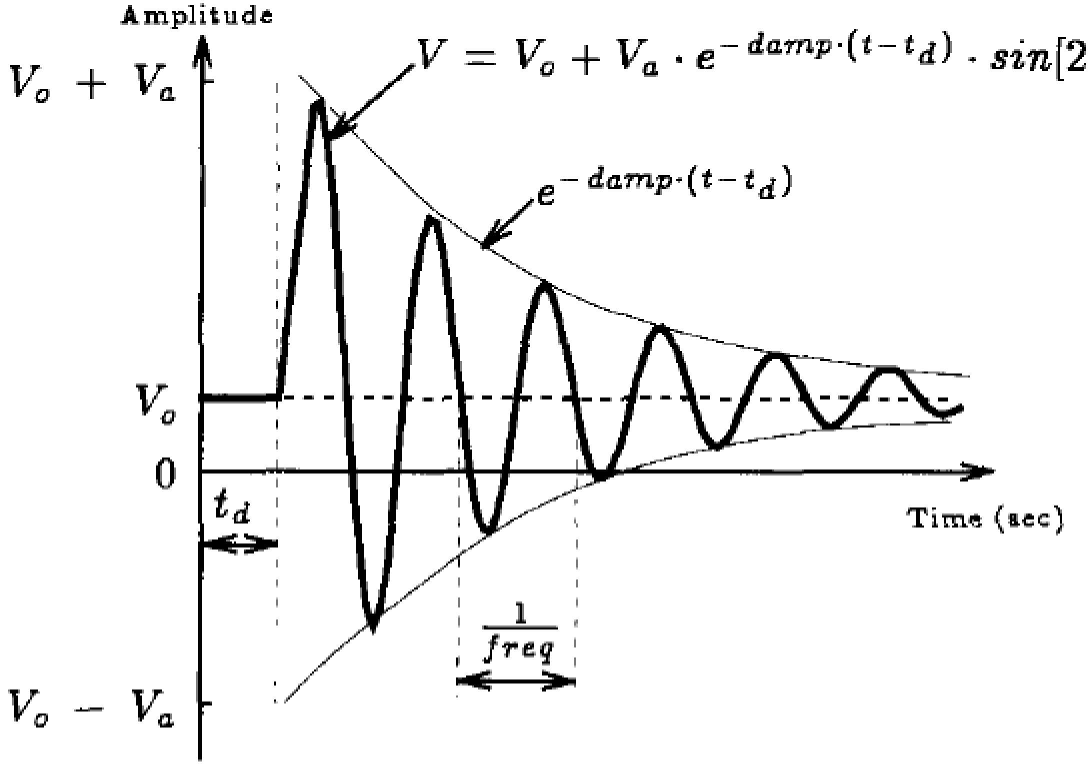
```
Vxxx n+ n- PULSE(V1 V2 Tdelay Trise Tfall Ton Tperiod Ncycles)
```

	Description	Units
Voff	Initial value	V
Von	Pulsed value	V
Tdelay	Delay	sec
Tr	Rise time	sec
Tf	Fall time	sec
Ton	On time	sec
Tperiod	Period	sec
Ncycles	Number of cycles (omit for free-running pulse function)	cycles

[LTspice Help]

Independent Source: SIN

$\left\{\begin{matrix} \mathrm{V}\mathit{name} \\ \mathrm{I}\mathit{name} \end{matrix}\right\}$ n+ n– SIN (V_o V_a freq t_d damp)   Transient



$$V = V_o + V_a \cdot e^{-\mathit{damp}\cdot(t-t_d)} \cdot \mathit{sin}[2\pi \cdot \mathit{freq} \cdot (t-t_d)], \qquad t \geq t_d.$$

Vname n + n − SIN (V_o V_a freq t_d damp)

[Sedra, 1997]

Independent Source: SIN

```
Vxxx n+ n- SINE(Voffset Vamp Freq Td Theta Phi Ncycles)
```

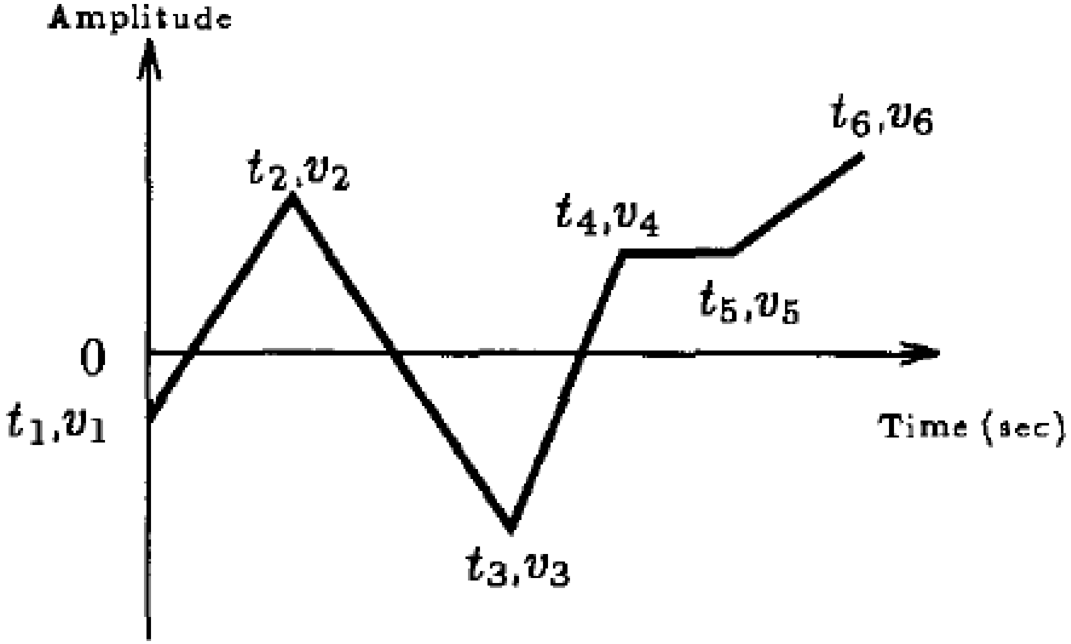
Name	Description	Units
Voffset	DC offset	V
Vamp	Amplitude	V
Freq	Frequency	Hz
Td	Delay	sec
Theta	Damping factor	1/sec
Phi	Phase of sine wave	degrees
Ncycles	Number of cycles (omit for free-running sine function)	cycles

```
V = Voffset + Vamp*exp(-(time-Td)*Theta)*sin(2*p*Freq*(time-Td)+p*Phi/180)
```

[LTspice Help]

Independent Source: PWL

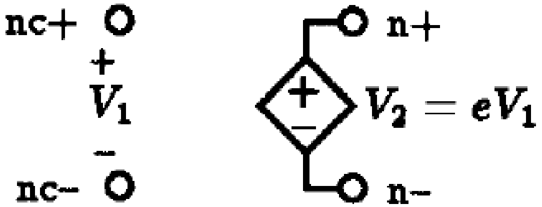
$\left\{\begin{array}{c} V\mathit{name} \\ I\mathit{name} \end{array}\right\}$ n+ n- PWL ($t_1,v_1\ t_2,v_2\ \ldots\ t_n,v_n$) Transient



$V\mathit{name}$ n + n − PWL $(t_1, v_1\ t_2, v_2\ \ldots\ t_n, v_n)$

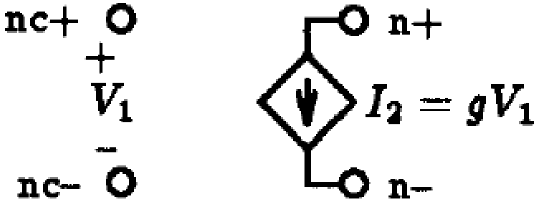
[Sedra, 1997]

Dependent Sources



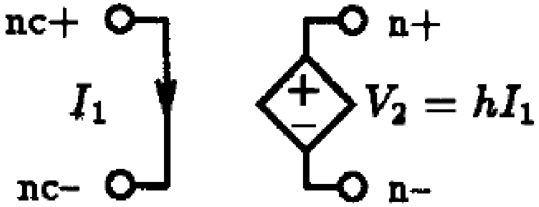
voltage-controlled voltage source

E*name* n+ n– nc+ nc– *e_value*



voltage-controlled voltage source

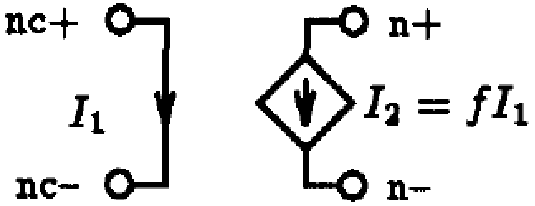
G*name* n+ n– nc+ nc– *g_value*



current-controlled voltage source

H*name* n+ n– V*name* *h_value*

V*name* nc+ nc– 0



current-controlled voltage source

F*name* n+ n– V*name* *f_value*

V*name* nc+ nc– 0

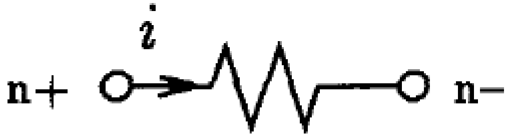
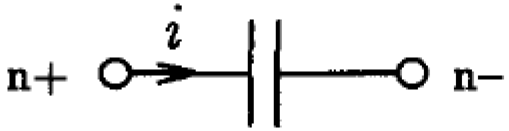
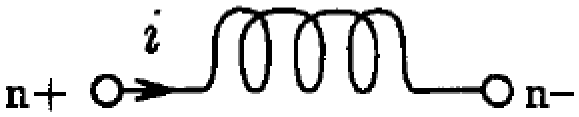
[Sedra, 1997]

.MODEL (Define a SPICE Model)

```
.model <modname> <type>[(<parameter list>)]
```

- The equations describing the model are built into SPICE itself.
  - The user cannot change the model equations.
  - But the user can change the model parameters.
- Some circuit elements (e.g., transistors) have many parameters.
  - Transistors of the same type are grouped by model name and have some common parameters (common model parameters, e.g., VTH).
  - The transistors of the same model can have different parameters from one instance to another (instance specific parameters, e.g., W and L).
- The parameter list depends on the type of model.
- Examples of model types (<type>): RES, CAP, IND, NPN, PNP, NMOS, PMOS
- <modname> is arbitrary

Passive Elements

Element	Spice Description
	R*name* n+ n− *value*
	C*name* n+ n− *value* [IC=*initial_voltage_condition*]
	L*name* n+ n− *value* [IC=*initial_current_condition*]

[Sedra, 1997]

Resistor

General form

```
R<name> <(+) node> <(-) node> [model name] <value>
+ [TC = <TC1> [,<TC2>]]
```

Examples

```
RLOAD 15 0 2K
R2 1 2 2.4E4 TC=.015,-.003
RFDBCK 3 33 RMOD 10K
```

Model form

```
.MODEL <model name> RES [model parameters]
```

[PSpice Documentation]

Resistor Model Parameters

$$\text{<value>}\times R \times [1 + TC1(T - T_{nom}) + TC2(T - T_{nom})^2]$$

Model parameters[1]	Description	Units	Default
R	resistance multiplier		1.0
TC1	linear temperature coefficient	$°C^{-1}$	0.0
TC2	quadratic temperature coefficient	$°C^{-2}$	0.0

[PSpice Documentation]

Capacitor

General form

```
C<name> <(+) node> <(-) node> [model name] <value>
+ [IC=<initial value>]
```

Examples

```
CLOAD 15 0 20pF
C2 1 2 .2E-12 IC=1.5V
CFDBCK  3 33 CMOD 10pF
```

Model form

```
.MODEL <model name> CAP [model parameters]
```

[PSpice Documentation]

Capacitor Model Parameters

$$\text{<value>} \times C \times (1 + VC1 \times V + VC2 \times V^2)$$
$$\times [1 + TC1(T - T_{nom}) + TC2(T - T_{nom})^2]$$

Model parameters[1]	Description	Units	Default
C	capacitance multiplier		1.0
TC1	linear temperature coefficient	$°C^{-1}$	0.0
TC2	quadratic temperature coefficient	$°C^{-2}$	0.0
VC1	linear voltage coefficient	volt^{-1}	0.0
VC2	quadratic voltage coefficient	volt^{-2}	0.0

[PSpice Documentation]

Inductor

General form

```
L<name> <(+) node> <(-) node> [model name] <value>
+ [IC=<initial value>]
```

Examples

```
LLOAD 15 0 20mH
L2 1 2 .2E-6
LCHOKE 3 42 LMOD .03
LSENSE 5 12 2UH IC=2mA
```

Model form

```
.MODEL <model name> IND [model parameters]
```

[PSpice Documentation]

Inductor Model Parameters

$$\boxed{\begin{gathered}\text{<value>}\times L \times (1 + IL1 \times I + IL2 \times I^2) \\ \times [1 + TC1(T - T_{nom}) + TC2(T - T_{nom})^2]\end{gathered}}$$

Model parameters[1]	Description	Units	Default
L	Inductance multiplier		1.0
IL1	Linear current coefficient	amp^{-1}	0.0
IL2	Quadratic current coefficient	amp^{-2}	0.0
TC1	Linear temperature coefficient	$°\text{C}^{-1}$	0.0
TC2	Quadratic temperature coefficient	$°\text{C}^{-2}$	0.0

[PSpice Documentation]

29 February 2020

وَمَا أُوتِيتُم مِّنَ الْعِلْمِ إِلَّا قَلِيلًا

5 رجب 1441

Computer Aided Circuit Design

Lecture 02 – Part II
Introduction to SPICE

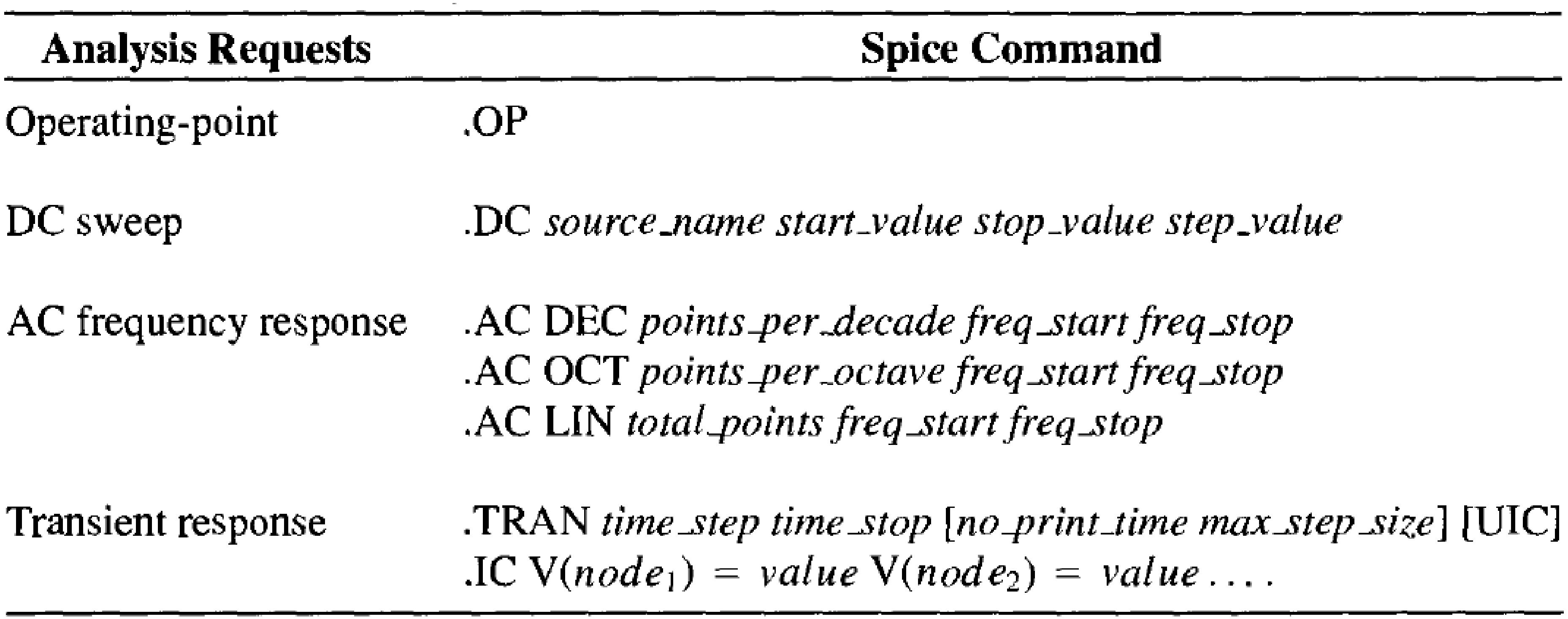
Dr. Hesham A. Omran

Integrated Circuits Laboratory (ICL)
Electronics and Communications Eng. Dept.
Faculty of Engineering
Ain Shams University

Analyses Commands

- ❑ .OP (bias point)
- ❑ .DC (DC analysis/sweeps)
- ❑ .AC (AC analysis)
- ❑ .TRAN (transient analysis)
- ❑ .TF (DC small-signal transfer function)
- ❑ .PARAM (define parameters for parametric sweep)
- ❑ .STEP (parametric sweep)
- ❑ .TEMP (temperature parametric sweep)

Basic Analysis Commands



Analysis Requests	Spice Command
Operating-point	.OP
DC sweep	.DC *source_name start_value stop_value step_value*
AC frequency response	.AC DEC *points_per_decade freq_start freq_stop* .AC OCT *points_per_octave freq_start freq_stop* .AC LIN *total_points freq_start freq_stop*
Transient response	.TRAN *time_step time_stop* [*no_print_time max_step_size*] [UIC] .IC V($node_1$) = *value* V($node_2$) = *value* . . . .

[Sedra, 1997]

.OP (Bias Point)

Purpose The .OP command causes detailed information about the bias point to be printed.

General form `.OP`

Examples `.OP`

Comments Without the .OP command, the only information about the bias point in the output is a list of the node voltages, voltage source currents, and total power dissipation.

Using a .OP command can cause the small-signal (linearized) parameters of all the nonlinear controlled sources and all the semiconductor devices to be printed in the output file.

[PSpice Documentation]

.DC (DC Analysis)

General form

```
.DC [LIN] <sweep variable name>
+ <start value> <end value> <increment value>
+ [nested sweep specification]
```

Examples

```
.DC VIN -.25  .25  .05
.DC LIN I2  5mA  -2mA  0.1mA
.DC VCE 0V 10V .5V  IB 0mA 1mA 50uA
.DC RES RMOD(R)  0.9  1.1  .001
.DC DEC NPN QFAST(IS) 1E-18 1E-14  5
```

Purpose The .DC command performs a linear, logarithmic, or nested DC sweep analysis on the circuit. The DC sweep analysis calculates the circuit's bias point over a range of values for <sweep variable name>.

Sweep type The sweep can be linear, logarithmic, or a list of values.

[PSpice Documentation]

DC Sweep Type

General form

```
.DC [LIN] <sweep variable name> <value>*
+[nested sweep specification]
```

Examples

```
.DC TEMP LIST 0 20 27 50 80 100 PARAM Vsupply 7.5 15 .5
```

Parameter	Description	Meaning
LIN	linear sweep	The sweep variable is swept linearly from the starting to the ending value.
OCT	sweep by octaves	Sweep by octaves. The sweep variable is swept logarithmically by octaves.
DEC	sweep by decades	Sweep by decades. The sweep variable is swept logarithmically by decades.
LIST	List of values	Use a list of values.

[PSpice Documentation]

DC Sweep Variable

General form

```
.DC [LIN] <sweep variable name> <value>*
+[nested sweep specification]
```

Examples

```
.DC TEMP LIST 0 20 27 50 80 100 PARAM Vsupply 7.5 15 .5
```

Parameter	Description	Meaning
Source	A name of an independent voltage or current source.	During the sweep, the source's voltage or current is set to the sweep value.
Model Parameter	A model type and model name followed by a model parameter name in parenthesis.	The parameter in the model is set to the sweep value. The following model parameters cannot be (usefully) swept: L and W for the MOSFET device (use LD and WD as a work around), and any temperature parameters, such as TC1 and TC2 for the resistor.
Temperature	Use the keyword TEMP for <sweep variable name>.	Set the temperature to the sweep value. For each value in the sweep, all the circuit components have their model parameters updated to that temperature.
Global Parameter	Use the keyword PARAM, followed by the parameter name, for <sweep variable name>.	During the sweep, the global parameter's value is set to the sweep value and all expressions are reevaluated.

[PSpice Documentation]

Nested DC Sweep

General form

```
.DC [LIN] <sweep variable name> <value>*
+[nested sweep specification]
```

Examples

```
.DC TEMP LIST 0 20 27 50 80 100 PARAM Vsupply 7.5 15 .5
```

Comments

For a nested sweep, a second sweep variable, sweep type, start, end, and increment values can be placed after the first sweep. In the nested sweep example, the first sweep is the inner loop: the entire first sweep is performed for each value of the second sweep.

When using a list of values, there are no start and end values. Instead, the numbers that follow the keyword LIST are the values that the sweep variable is set to.

The rules for the values in the second sweep are the same as for the first.

[PSpice Documentation]

.AC (AC Analysis)

Purpose

The .AC command calculates the frequency response of a circuit over a range of frequencies.

General form

```
.AC <sweep type> <points value>
+ <start frequency value> <end frequency value>
```

Examples

```
.AC LIN 101 100Hz  200Hz
.AC OCT  10  1kHz  16kHz
.AC DEC  20  1MEG 100MEG
```

Arguments and options

<sweep type>

Must be LIN, OCT, or DEC.

.TRAN (transient analysis)

Purpose

The .TRAN command causes a transient analysis to be performed on the circuit and specifies the time period for the analysis.

General form

```
.TRAN[/OP] <print step value> <final time value>
+[no-print value [step ceiling value]][SKIPBP]
```

Examples

```
.TRAN 1ns 100ns
.TRAN/OP 1ns 100ns 20ns SKIPBP
.TRAN 1ns 100ns 0ns .1ns
.TRAN 1ns 100ns 0ns {SCHEDULE(0,1ns,25ns,.1ns)}
.Tran {2*4ns+1ns} {5/param1+0.1m} {param2} 0.1ns
```

where param1, param2 are parameters defined using .param

[PSpice Documentation]

.TF (DC Transfer Function)

Purpose

The .TF command/statement causes the small-signal DC gain to be calculated by linearizing the circuit around the bias point.
Voltage gain, current gain, transconductance, or transresistance.
It also computes input and output resistances.
Simply, it calculates Thevenin or Norton equivalent.

General form

```
.TF <output variable> <input source name>
```

Examples

```
.TF V(5) VIN
.TF I(VDRIV) ICNTRL
```

Comments

The gain from *<input source name>* to *<output variable>* and the input and output resistances are evaluated and written to the output file.
When <output variable> is a current, it is restricted to be the current through a voltage source.
Note: The results of the .TF command are only available in the output file.

[PSpice Documentation]

.PARAM (Parametric Sweep / Analysis)

Purpose

The .PARAM statement defines the value of a parameter. A parameter name can be used in place of most numeric values in the circuit description. Parameters can be constants, or expressions involving constants, or a combination of these, and they can include other parameters.
Note that expressions are placed between braces: `{<expression>}`

General form

```
.PARAM <name> = <value>
.PARAM <name> = {<expression>}
```

Examples

```
.PARAM VSUPPLY = 5V
.PARAM VCC = 12V, VEE = -12V
.PARAM BANDWIDTH = {100kHz/3}
.PARAM PI = 3.14159, TWO_PI = {2*3.14159}
.PARAM VNUM = {2*TWO_PI}
```

[PSpice Documentation]

.STEP (Parametric Sweep / Analysis)

Purpose

The .STEP command performs a parametric sweep for all of the analyses of the circuit.
The .STEP command is similar to the .TEMP (temperature) command in that all of the typical analyses--such as .DC (DC analysis), .AC (AC analysis), and .TRAN (transient analysis)-- are performed for each step.
Probe displays nested sweeps as a family of curves.

General form

```
.STEP [LIN] <sweep variable name> <start value> <end value>
<increment value>
.STEP [DEC|OCT] <sweep variable name> <start value> <end value>
<points value>
.STEP <sweep variable name> LIST <value>
```

*The first general form is for doing a linear sweep. The second form is for doing a logarithmic sweep. The third form is for using a list of values for the sweep variable.

Examples

```
.STEP VCE 0V 10V .5V
.STEP LIN I2 5mA -2mA 0.1mA
.STEP RES RMOD(R) 0.9 1.1 .001
.STEP DEC NPN QFAST(IS) 1E-18 1E-14 5
.STEP TEMP LIST 0 20 27 50 80 100
.STEP PARAM CenterFreq 9.5kHz 10.5kHz 50Hz
```

[PSpice Documentation]

.TEMP (Temperature Sweep)

Purpose

The .TEMP command sets the temperature at which all analyses are done.

General form

```
.TEMP <temperature value>*
```

Examples

```
.TEMP 125
.TEMP 0 27 125
```

Comments

The temperatures are in degrees Centigrade. If more than one temperature is given, then all analyses are performed for each temperature.

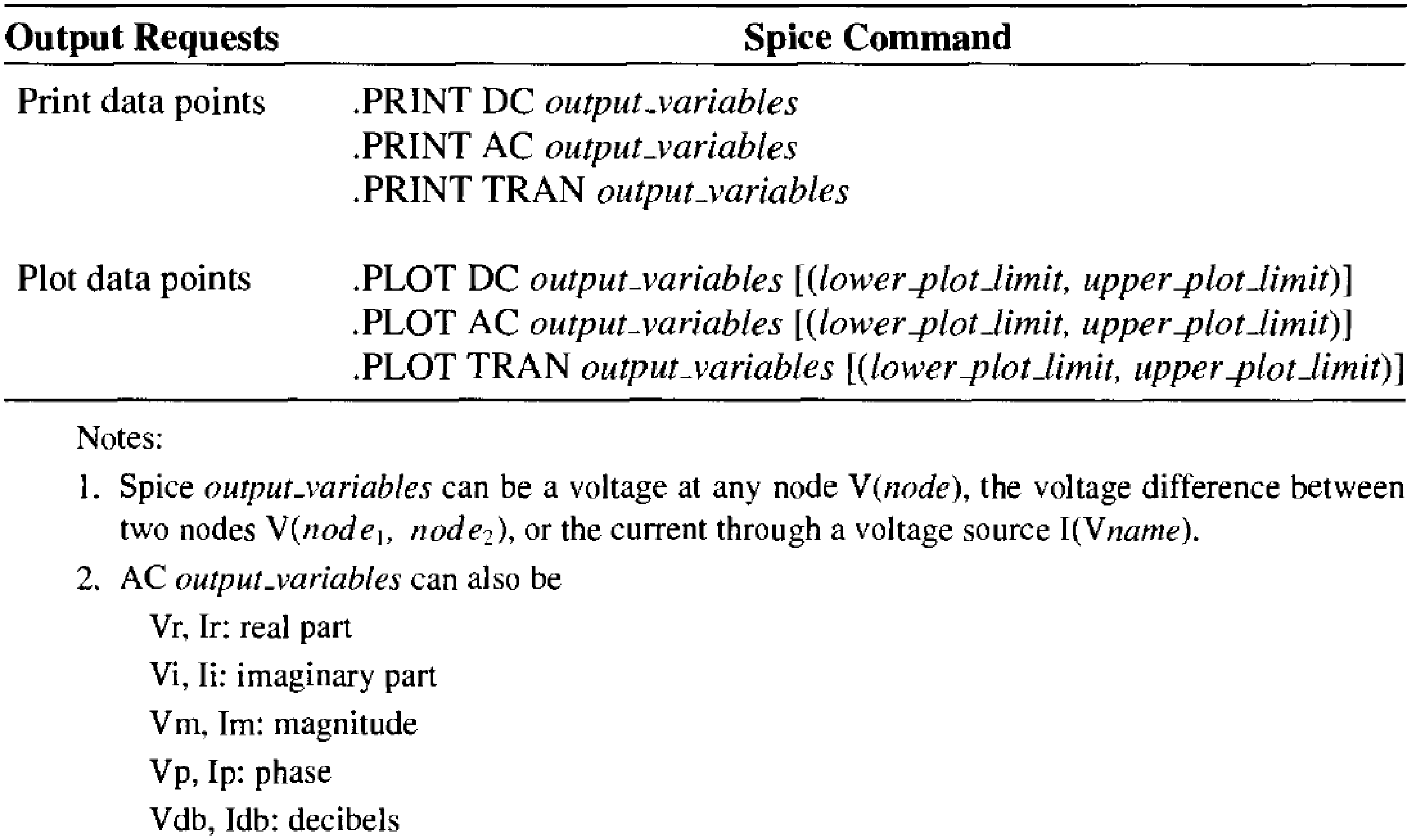
It is assumed that the model parameters were measured or derived at the nominal temperature, TNOM (27°C by default).

See the .OPTIONS (analysis options) command for setting TNOM.

.TEMP behaves similarly to the list variant of the .STEP (parametric analysis) statement, with the stepped variable being the temperature.

[PSpice Documentation]

Output Commands



Output Requests	Spice Command
Print data points	.PRINT DC *output_variables* .PRINT AC *output_variables* .PRINT TRAN *output_variables*
Plot data points	.PLOT DC *output_variables* [(*lower_plot_limit, upper_plot_limit*)] .PLOT AC *output_variables* [(*lower_plot_limit, upper_plot_limit*)] .PLOT TRAN *output_variables* [(*lower_plot_limit, upper_plot_limit*)]

Notes:

1. Spice *output_variables* can be a voltage at any node V(*node*), the voltage difference between two nodes V($node_1$, $node_2$), or the current through a voltage source I(V*name*).
2. AC *output_variables* can also be

   Vr, Ir: real part

   Vi, Ii: imaginary part

   Vm, Im: magnitude

   Vp, Ip: phase

   Vdb, Idb: decibels

[Sedra, 1997]

.PLOT (Plot)

Purpose

The .PLOT command causes results from DC, AC, noise, and transient analyses to be line printer plots in the output file.

General form

```
.PLOT <analysis type> [output variable]*
+ ( [<lower limit value> , <upper limit value>] )*
```

Examples

```
.PLOT DC  V(3)  V(2,3) V(R1)  I(VIN) I(R2) IB(Q13) VBE(Q13)
.PLOT AC  VM(2) VP(2)  VM(3,4) VG(5) VDB(5) IR(D4)
.PLOT NOISE INOISE ONOISE DB(INOISE) DB(ONOISE)
.PLOT TRAN V(3) V(2,3) (0,5V)   ID(M2) I(VCC) (-50mA,50mA)I
.PLOT TRAN D(QA) D(QB) V(3) V(2,3)
.PLOT TRAN V(3) V(R1) V([RESET])
```

[PSpice Documentation]

.PRINT (Print)

Purpose

The .PRINT command allows results from DC, AC, noise, and transient analyses to be an output in the form of tables, referred to as print tables in the output file.

General form

```
.PRINT[/DGTLCHG] <analysis type> [output variable]*
```

Examples

```
.PRINT DC  V[3]  V[2],[3] V(R1) I(VIN) I(R2) IB(Q13) VBE(Q13)
.PRINT AC  VM[2] VP[2] VM[3],[4] VG[5] VDB[5] IR[6] II[7]
.PRINT NOISE INOISE ONOISE DB(INOISE) DB(ONOISE)
.PRINT TRAN V[3] V([2],[3]) ID[M2] I[VCC]
.PRINT TRAN D(QA) D(QB) V[3] V([2],[3])
.PRINT/DGTLCHG TRAN QA QB RESET
.PRINT TRAN V[3] V(R1) V([RESET])
```

The last example illustrates how to print a node that has a name, rather than a number. The first item to print is a node voltage, the second item is the voltage across a resistor, and the third item to print is another node voltage, even though the second and third items both begin with the letter R. The square brackets force the names to be interpreted as node names.

[PSpice Documentation]

.PROBE (Probe: PSpice ONLY)

Purpose

The .PROBE command writes the results from DC, AC, and transient analyses to a data file used by Probe.

General form

```
.PROBE    [output variable]*
```

Examples

```
.PROBE
```

Writes **all the node voltages and all the device currents** to the data file. The list of device currents written is the same as the device currents allowed as output variables.

```
.PROBE V[3] V([2],[3]) V(R1) I(VIN) I(R2) IB(Q13) VBE(Q13)
```

Writes only those output variables specified to the data file, to restrict the size of the data file.

```
.PROBE V[3] V(R1) V([RESET])
```

The square brackets force the interpretation of names to mean node names

```
.PROBE D(QBAR)
```

Writes only the output at digital node QBAR to the data file, to restrict the size of the data file.

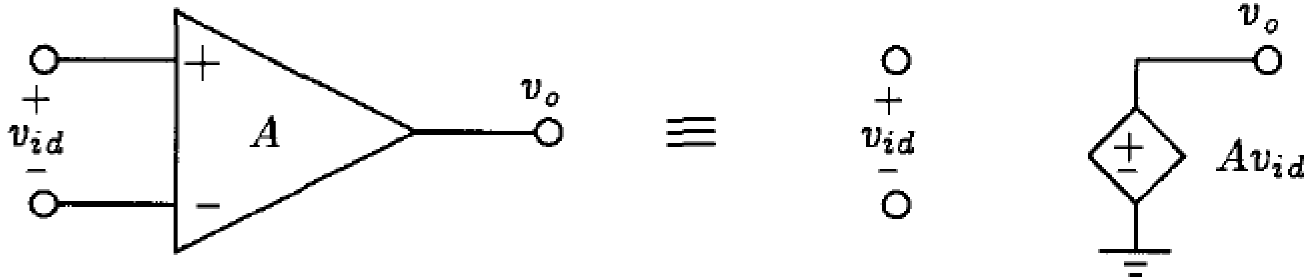
```
.PROBE P(FREQ)
```

Writes the parameter FREQ defined using the .PARAM command.

[PSpice Documentation]

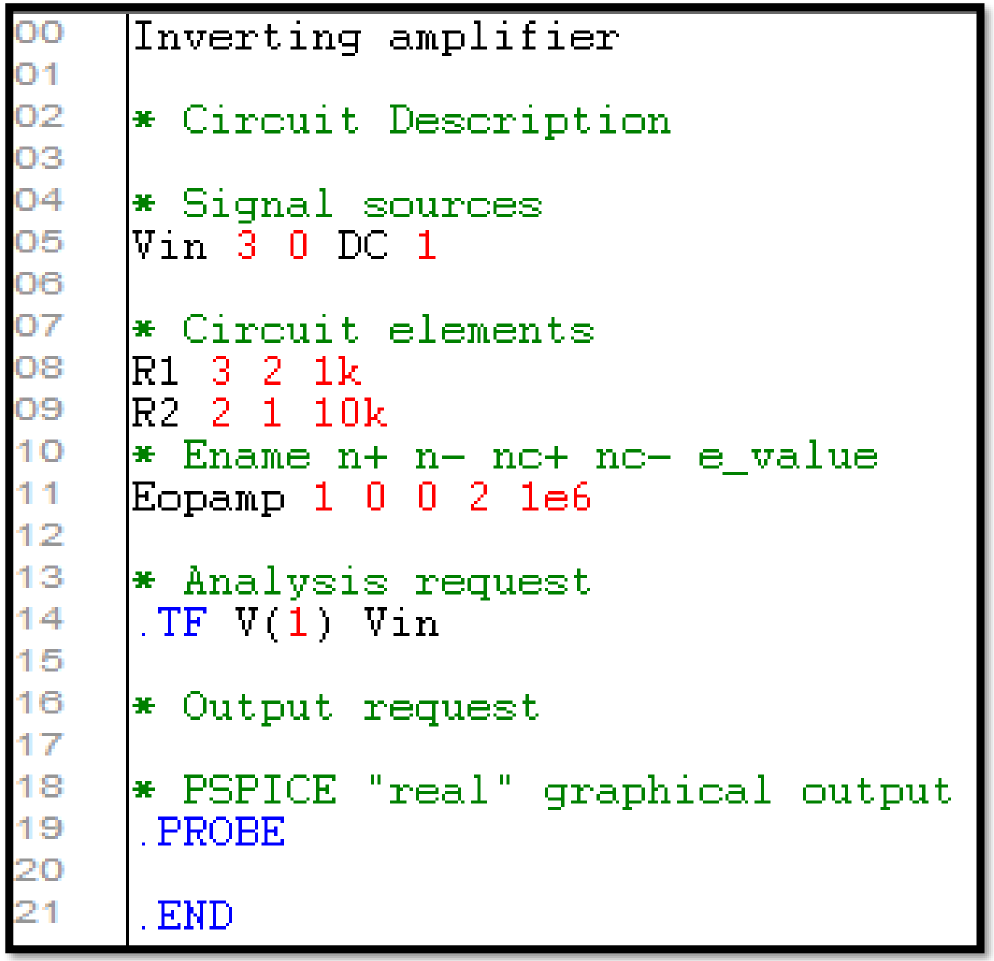
Example: Modeling Ideal Op-Amp

- Use VCVS
- Specify large value for the gain, e.g., 1e5
  - Ename n+ n- nc+ nc- e_value
  - Eopamp 1 0 0 2 1e5

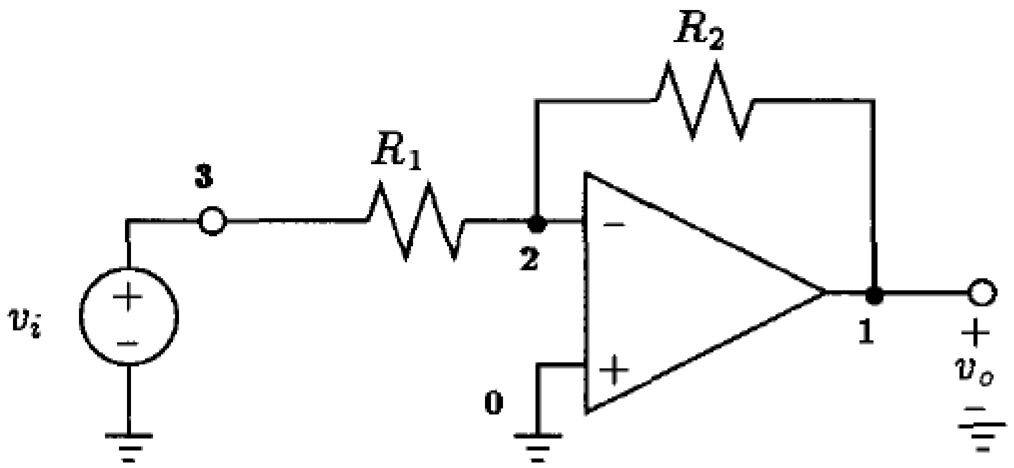


[Sedra, 1997]

Example: Inverting Amplifier



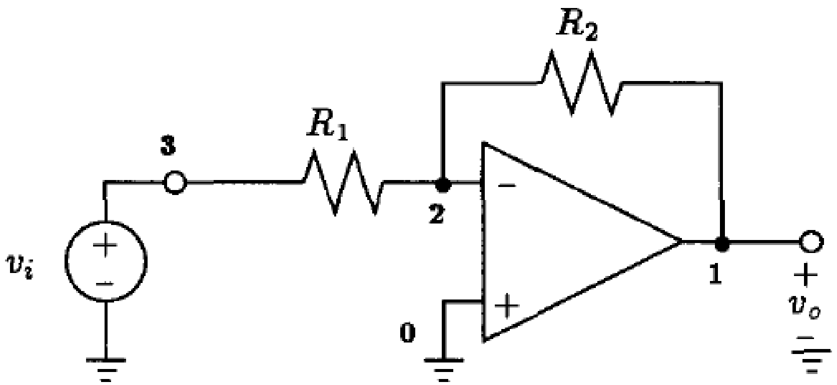
```
Inverting amplifier

* Circuit Description

* Signal sources
Vin 3 0 DC 1

* Circuit elements
R1 3 2 1k
R2 2 1 10k
* Ename n+ n- nc+ nc- e_value
Eopamp 1 0 0 2 1e6

* Analysis request
.TF V(1) Vin

* Output request

* PSPICE "real" graphical output
.PROBE

.END
```

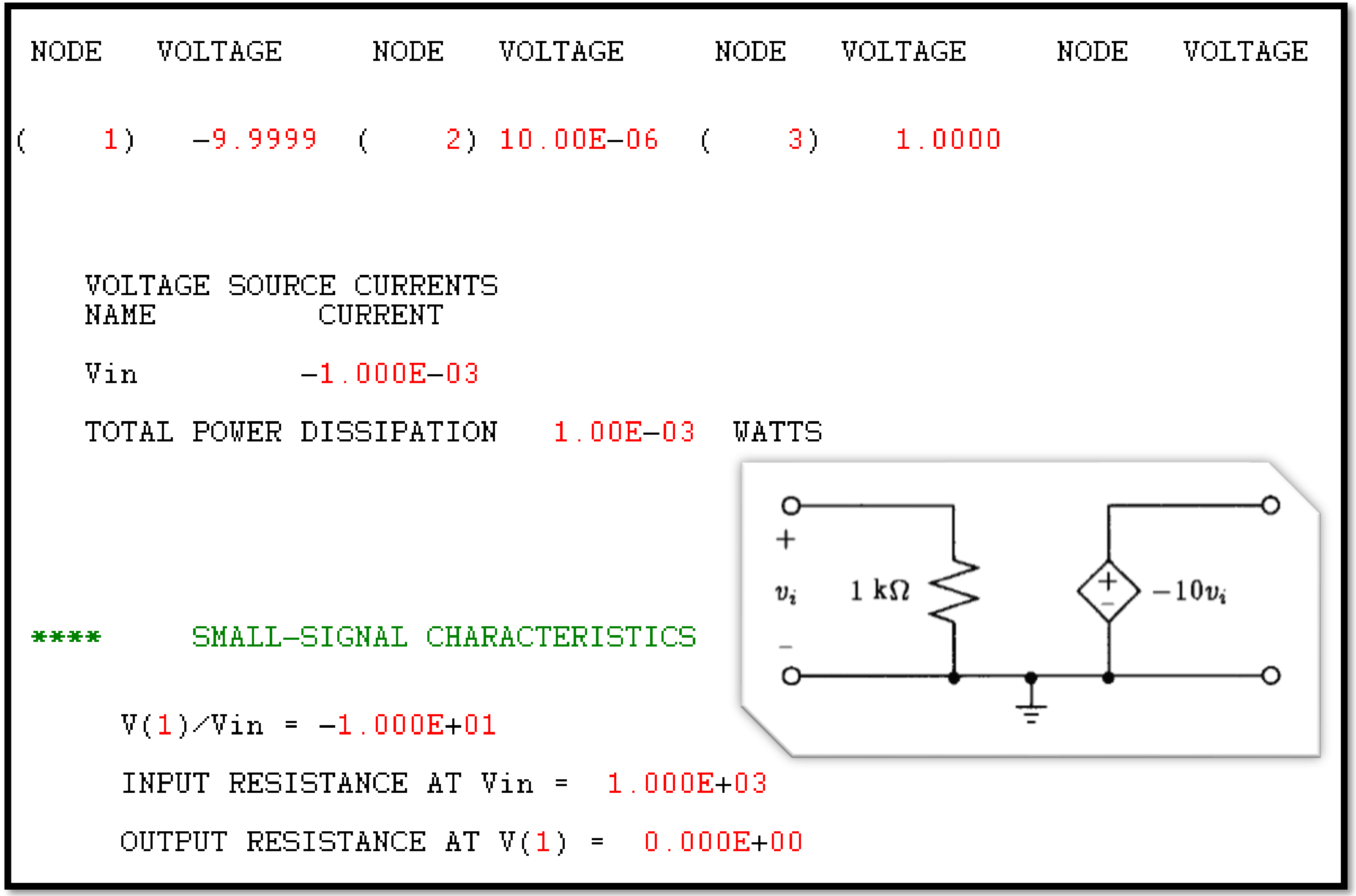


[Sedra, 1997]

Inverting Amplifier: .TF Analysis

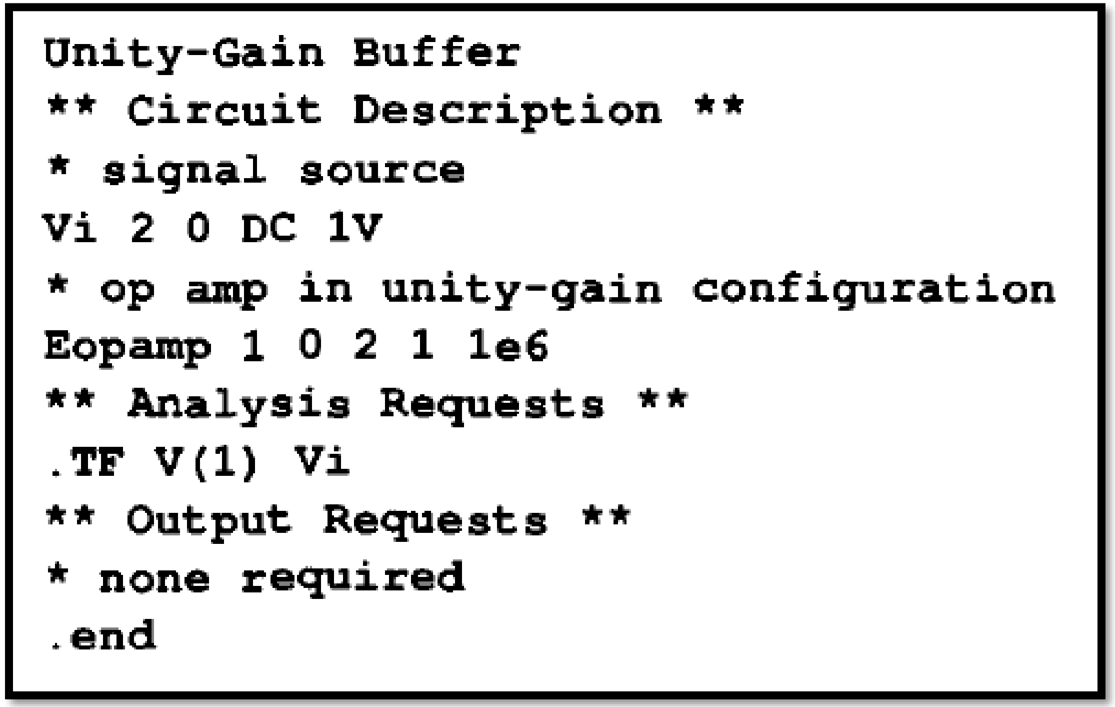
- ❑ Transfer function command computes the DC small signal gain from a specified signal source to a specified output
- ❑ Simply, it calculates Thevenin or Norton equivalent



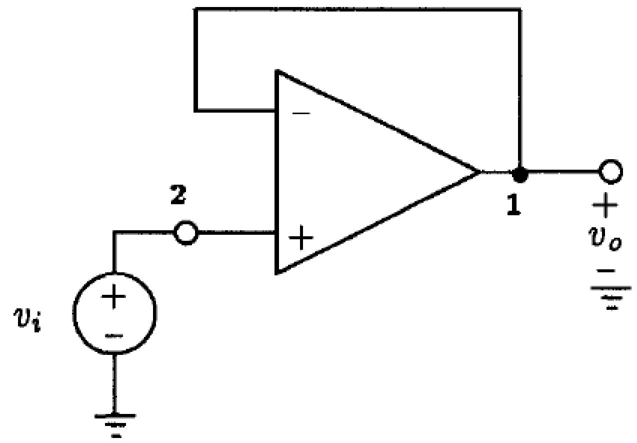


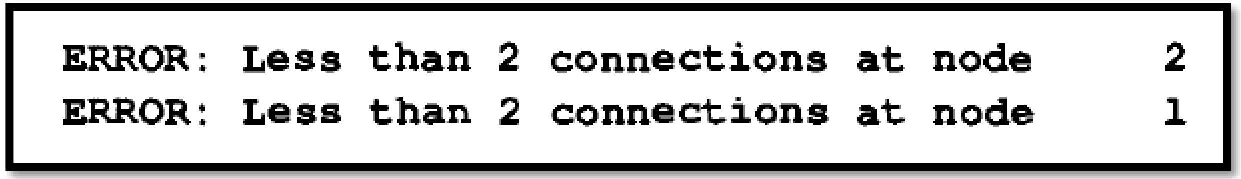
```
NODE   VOLTAGE     NODE   VOLTAGE     NODE   VOLTAGE     NODE   VOLTAGE

(    1)   -9.9999  (    2) 10.00E-06  (    3)    1.0000



    VOLTAGE SOURCE CURRENTS
    NAME          CURRENT

    Vin          -1.000E-03

    TOTAL POWER DISSIPATION   1.00E-03  WATTS



****     SMALL-SIGNAL CHARACTERISTICS

      V(1)/Vin = -1.000E+01

      INPUT RESISTANCE AT Vin =   1.000E+03

      OUTPUT RESISTANCE AT V(1) =   0.000E+00
```

Topological Restrictions



```
Unity-Gain Buffer
** Circuit Description **
* signal source
Vi 2 0 DC 1V
* op amp in unity-gain configuration
Eopamp 1 0 2 1 1e6
** Analysis Requests **
.TF V(1) Vi
** Output Requests **
* none required
.end
```



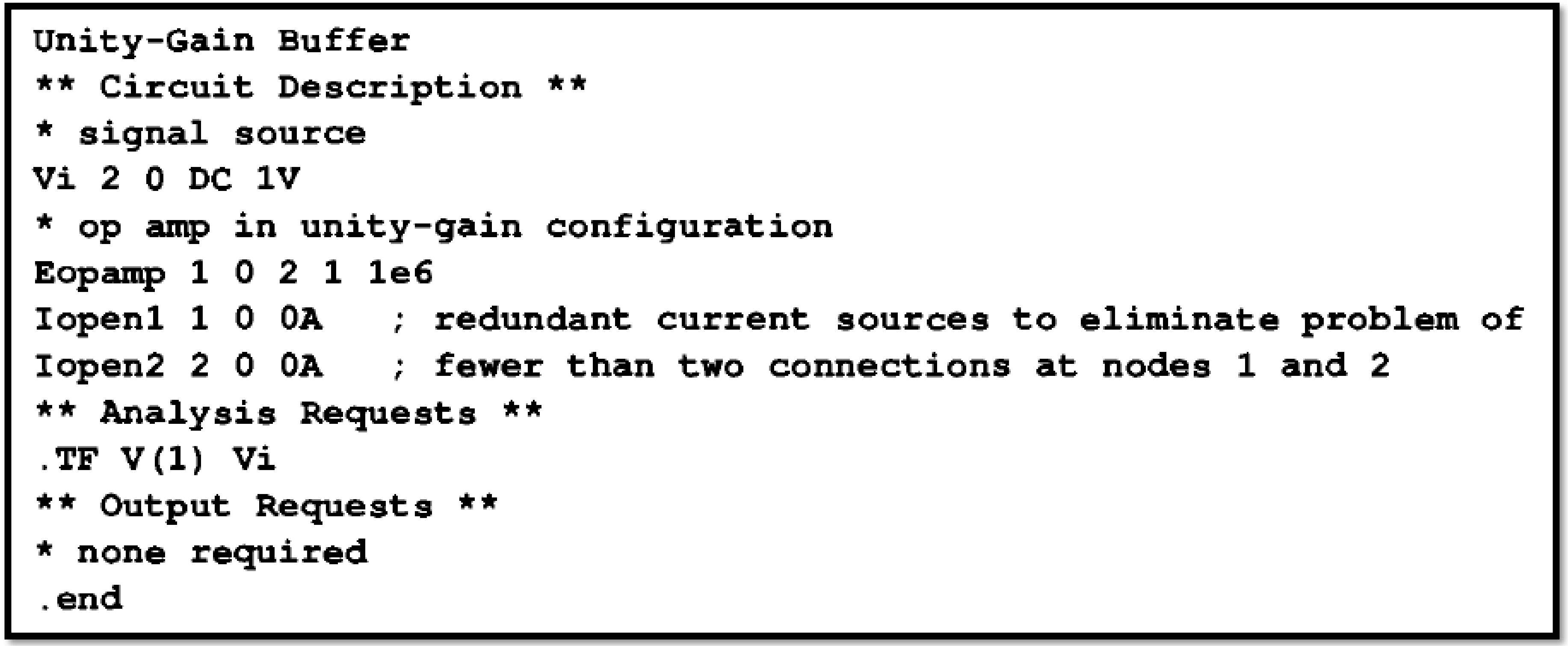


```
ERROR: Less than 2 connections at node      2
ERROR: Less than 2 connections at node      1
```

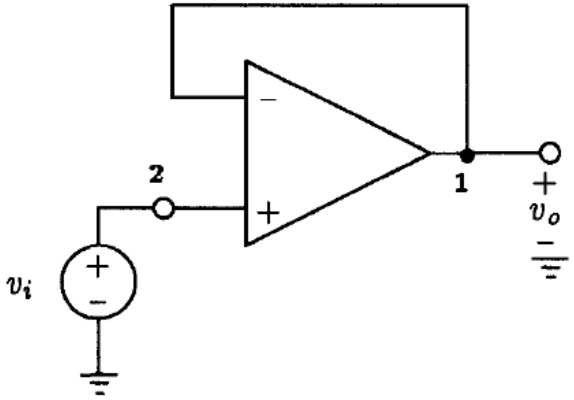
[Sedra, 1997]

Topological Restrictions Workarounds

- ❑ Use zero-current DC current source or large resistance



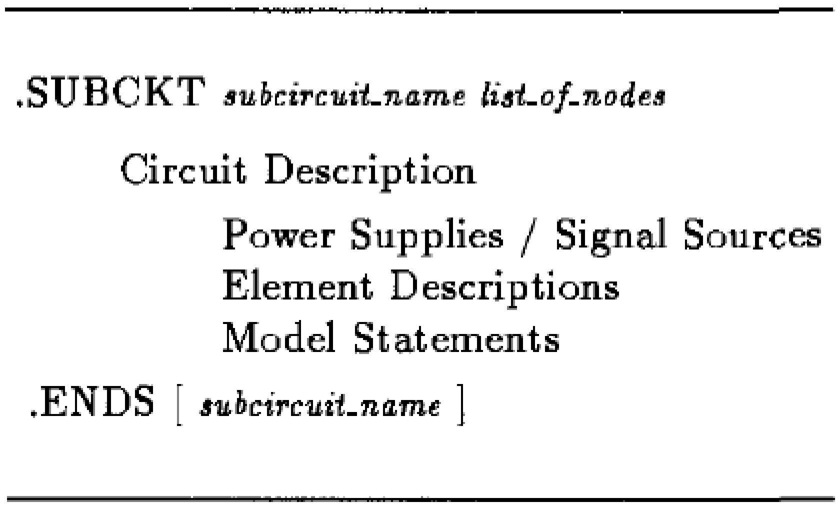
```
Unity-Gain Buffer
** Circuit Description **
* signal source
Vi 2 0 DC 1V
* op amp in unity-gain configuration
Eopamp 1 0 2 1 1e6
Iopen1 1 0 0A    ; redundant current sources to eliminate problem of
Iopen2 2 0 0A    ; fewer than two connections at nodes 1 and 2
** Analysis Requests **
.TF V(1) Vi
** Output Requests **
* none required
.end
```



[Sedra, 1997]

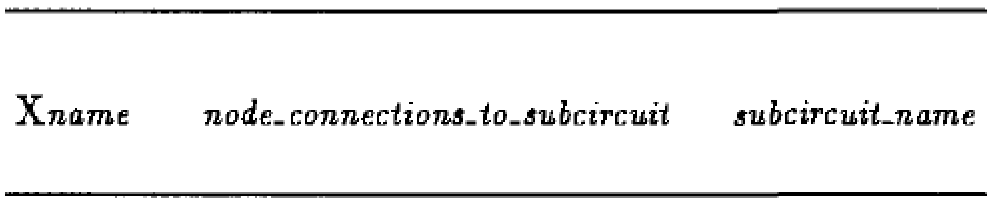
Sub-circuits

- ❑ Structured design → Hierarchy → Easier to write/read/debug
- ❑ Create a library of basic components



```
.SUBCKT subcircuit_name list_of_nodes
        Circuit Description
                Power Supplies / Signal Sources
                Element Descriptions
                Model Statements
 .ENDS [ subcircuit_name ]
```

Figure 2.16 Subcircuit format and syntax.



```
Xname     node_connections_to_subcircuit     subcircuit_name
```

Figure 2.17 Accessing a subcircuit by the main circuit.

[Sedra, 1997]

Subcircuits

Purpose

The .SUBCKT command/statement starts the subcircuit definition by specifying its name, the number and order of its terminals, and the names and default parameters that control its behavior. Subcircuits are instantiated using X (Subcircuit instantiation) devices. The .ENDS command marks the end of a subcircuit definition.

General form

```
.SUBCKT <name> [node]*
+ [PARAMS: < <name> = <value> >* ]
...
.ENDS
```

Examples

```
.SUBCKT OPAMP 1 2 101 102 17
...
.ENDS

.SUBCKT FILTER INPUT OUTPUT PARAMS: CENTER=100kHz,
+ BANDWIDTH=10kHz
...
.ENDS
```

[PSpice Documentation]

Subcircuit Instantiation

Purpose

This statement causes the referenced subcircuit to be inserted into the circuit using the given nodes to replace the argument nodes in the definition. It allows a block of circuitry to be defined once and then used in several places.

General form

```
X<name> [node]* <subcircuit name>
+ [PARAMS: <<name> = <value>>*]
```

Examples

```
X12 100 101 200 201 DIFFAMP
XBUFF 13  15  UNITAMP
XFOLLOW IN OUT VCC VEE OUT OPAMP
XFELT   1  2  FILTER PARAMS: CENTER=200kHz
XNANDI 25 28 7 MYPWR MYGND PARAMS: IO_LEVEL=2
```

[PSpice Documentation]

Subcircuit Example

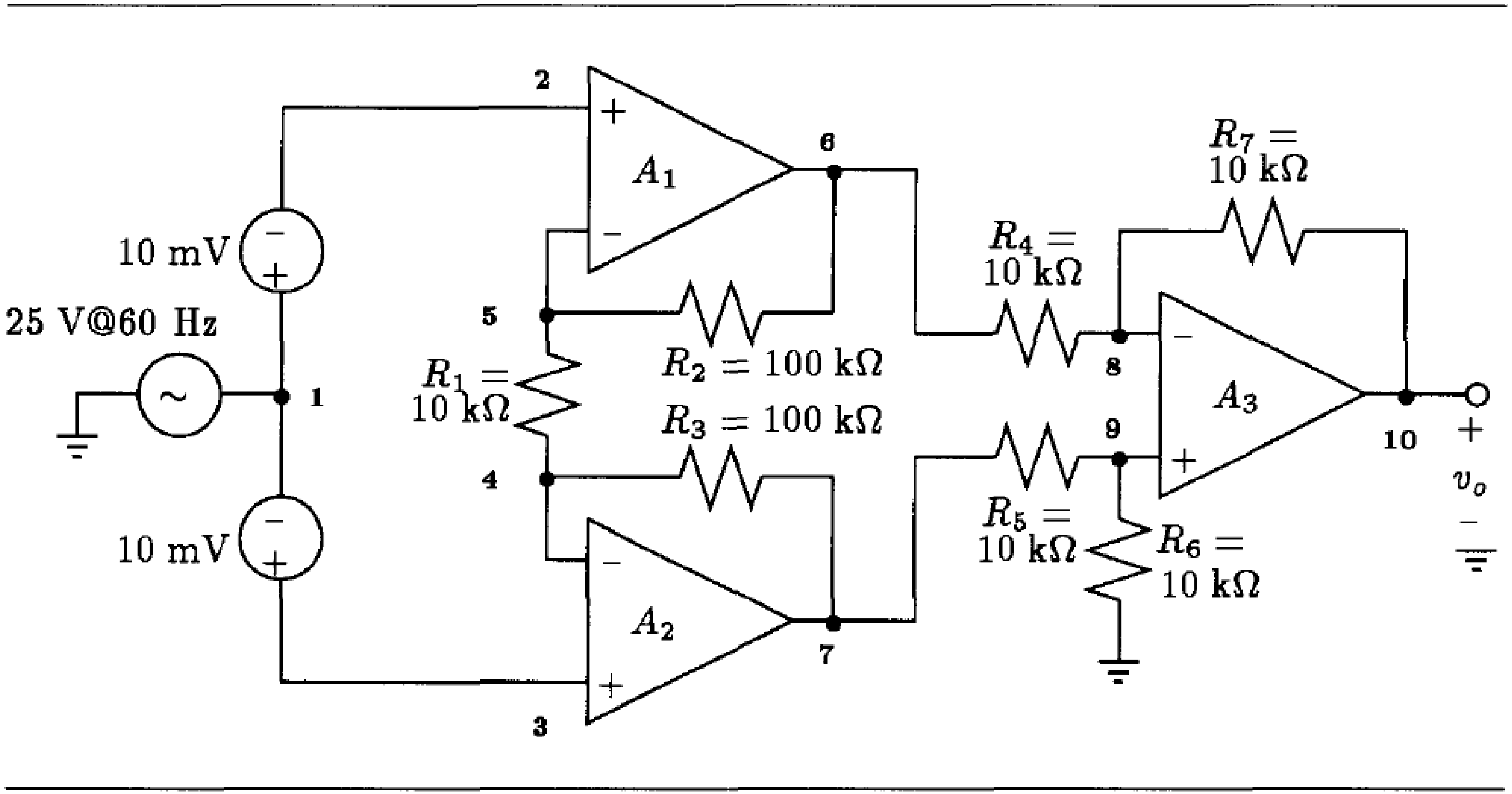
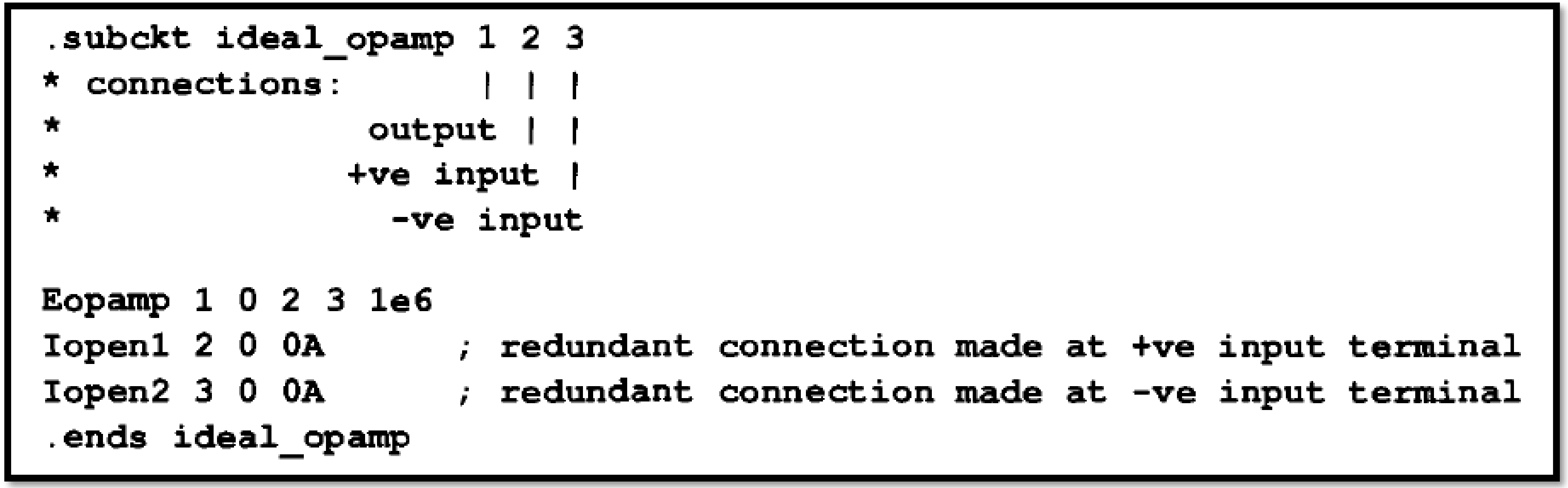


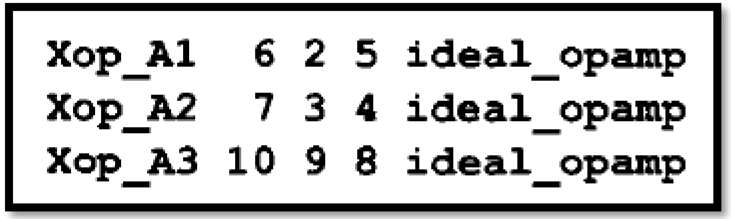
Figure 2.15 A two-stage, three–op amp instrumentation amplifier.

[Sedra, 1997]

Subcircuit Example



```
.subckt ideal_opamp 1 2 3
* connections:      | | |
*              output | |
*             +ve input |
*               -ve input

Eopamp 1 0 2 3 1e6
Iopen1 2 0 0A      ; redundant connection made at +ve input terminal
Iopen2 3 0 0A      ; redundant connection made at -ve input terminal
.ends ideal_opamp
```



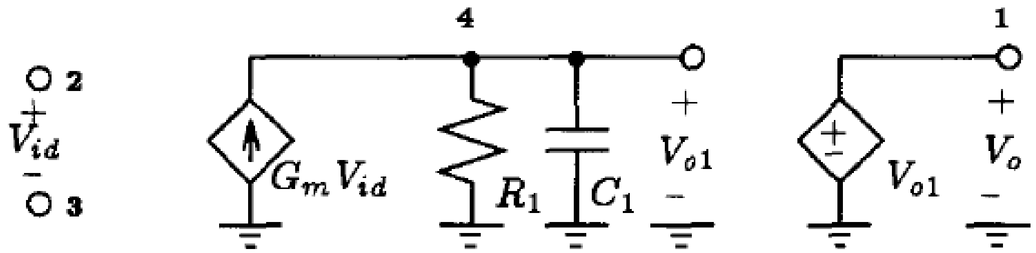
```
Xop_A1  6 2 5 ideal_opamp
Xop_A2  7 3 4 ideal_opamp
Xop_A3 10 9 8 ideal_opamp
```

[Sedra, 1997]

Modeling Non-Ideal Op-Amp (Small Signal)

- An op-amp can modeled with a single-pole transfer function (the dominant pole)
  - Note that the pole frequency is the open-loop bandwidth

$$A(s) = \frac{A_0}{1 + s/\omega_b}$$



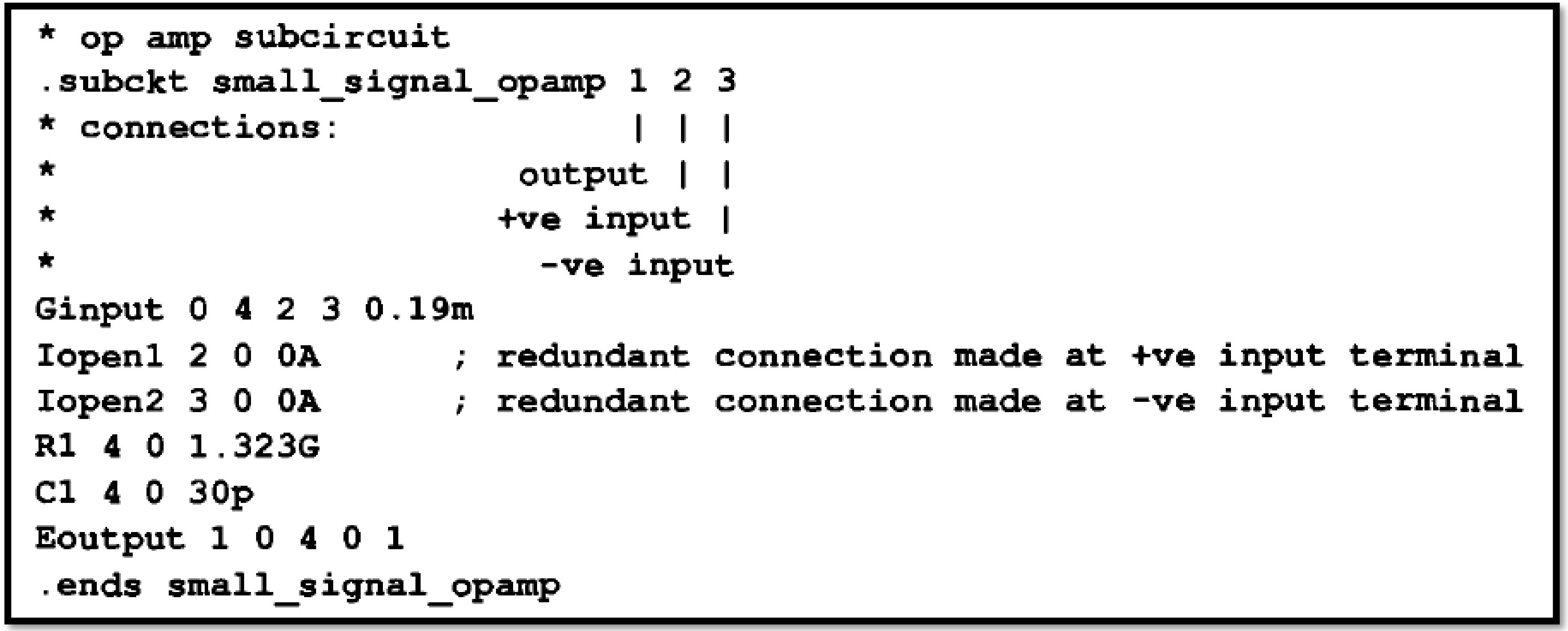
$$A_0 = G_m R_1 \qquad \omega_b = 1/R_1 C_1$$

$$\frac{V_o}{V_{id}}(s) = \frac{G_m R_1}{1 + sR_1 C_1}$$

Figure 2.21 A one-pole circuit representation of the small-signal open-loop frequency response of an internally compensated op amp.

[Sedra, 1997]

Modeling Non-Ideal Op-Amp (Small Signal)



```
* op amp subcircuit
.subckt small_signal_opamp 1 2 3
* connections:             | | |
*                     output | |
*                    +ve input |
*                      -ve input
Ginput 0 4 2 3 0.19m
Iopen1 2 0 0A      ; redundant connection made at +ve input terminal
Iopen2 3 0 0A      ; redundant connection made at -ve input terminal
R1 4 0 1.323G
C1 4 0 30p
Eoutput 1 0 4 0 1
.ends small_signal_opamp
```

[Sedra, 1997]

Other Useful SPICE Commands

- .IC (initial condition)
- .NODESET (initial guess)
- .INC (include file)
- .LIB (include library)

- See LTspice Help for full list of commands.

.IC (Initial Bias Point Condition)

Purpose

The .IC command sets initial conditions for both small-signal and transient bias points. Initial conditions can be given for some or all of the circuit's nodes. .IC sets the initial conditions for the bias point only.

General form

```
.IC < V(<node> [,<node>])=<value> >*
.IC <I(<inductor>)=<value>>*
```

Examples

```
.IC V(2)=3.4 V(102)=0 V(3)=-1V I(L1)=2uAmp
.IC V(InPlus,InMinus)=1e-3 V(100,133)=5.0V
```

Comments

The voltage between two nodes and the current through an inductor can be specified. **During bias calculations, PSpice clamps the voltages to specified values by attaching a voltage source with a 0.002 ohm series resistor between the specified nodes.** After the bias point has been calculated and the transient analysis started, the node is released.

If the circuit contains both the .IC command and .NODESET (set approximate node voltage for bias point) command for the same node or inductor, the .NODESET command is ignored (.IC overrides .NODESET).

[PSpice Documentation]

.NODESET (Initial Guess Point)

Purpose

The .NODESET command helps calculate the bias point by providing an initial best guess for some node voltages and/or inductor currents. Some or all of the circuit's node voltages and inductor currents can be given the initial guess, and in addition, the voltage between two nodes can be specified.

General form

```
.NODESET < V(<node> [,<node>])=<value> >*
.NODESET <I(<inductor>)=<value>>
```

Examples

```
.NODESET V(2)=3.4 V(102)=0 V(3)=-1V I(L1)=2uAmp
.NODESET V(InPlus,InMinus)=1e-3 V(100,133)=5.0V
```

Comments

This command is effective for the bias point (both small-signal and transient bias points) and for the first step of the DC sweep. It has no effect during the rest of the DC sweep, nor during a transient analysis. Unlike the .IC (initial bias point condition) command, **.NODESET provides only an initial guess for some initial values. It does not clamp those nodes to the specified voltages. However, by providing an initial guess, .NODESET can be used to break the tie in, for instance, a flip-flop, and make it come up in a required state.**

If both the .IC command and .NODESET command are present, the .NODESET command is ignored for the bias point calculations (.IC overrides .NODESET).

[PSpice Documentation]

.INC (Include File)

Purpose The .INC command inserts the contents of another file.

General form

```
.INC <file name>
```

Examples

```
.INC "SETUP.CIR"
.INC "C:\LIB\VCO.CIR"
```

.LIB (Library File)

Purpose The .LIB command references a model or subcircuit library in another file.

General form

```
.LIB [file_name]
```

Examples

```
.LIB
.LIB linear.lib
.LIB "C:\lib\bipolar.lib"
```

[PSpice Documentation]

SPICE Live Example

- Simple RC circuit simulation
  - AC analysis
  - Transient analysis
  - Using parameters
  - Parametric sweeps

Thank you!

SPICE Monkey Example

- ❑ You build a CMOS inverter for your boss and he asks you what the input capacitance is.
- ❑ You answer: “I'm not sure, but I will simulate it right away!”
- ❑ What you should have been able to say is:
  - ▪ “Well, boss, the first thing I did was to familiarize myself with the 12nm CMOS process and I found that the input capacitance is 0.8 fF/µm, so of course my 7µm PMOS, 3µm NMOS inverter has about 8 fF of input capacitance. ”
- ❑ But you didn't say that, because you didn't know → You are a SPICE monkey.
- ❑ Don't be so dependent on SPICE.
  - ▪ You should always be able to predict an approximate answer to simple questions like this without resorting to a SPICE simulation.