9 SPICE Simulator Interface 9-1
9.1 Getting Started....9-1
9.1.1 SPICE Parts....9-2
9.1.2 Running *Sch2spi*....9-2
9.1.3 *Sch2spi* Output files....9-2
9.1.3.1 SPICE netlist files....9-2
9.1.3.2 Two CapFast SPICE Alias files....9-2
9.1.4 Device Name Assignment 9-3
9.1.5 Node Name Assignment....9-3
9.2 *Sch2spi* Options....9-4
9.3 Properties and Qualifiers....9-5
9.4 Element Cards 9-6
9.4.1 Resistors 9-6
9.4.2 Capacitors and Inductors....9-7
9.4.3 Coupled (Mutual) Inductors 9-8
9.4.4 Transmission Lines (Lossless)....9-9
9.5 Linear Dependent Sources....9-10
9.5.1 Linear Voltage-Controlled Current Sources 9-10
9.5.2 Linear Voltage-Controlled Voltage Sources....9-10
9.5.3 Linear Current-Controlled Current Sources....9-11
9.5.4 Linear Current-Controlled Voltage Sources 9-11
9.5.5 Independent Sources....9-12
9.5.5.1 Pulse 9-13
9.5.5.2 Sinusoidal 9-14
9.5.5.3 Exponential....9-14
9.5.5.4 Piece-Wise Linear 9-15
9.5.5.5 Single-Frequency FM 9-15
9.6 Semiconductor Devices 9-16
9.6.1 Junction Diodes 9-17
9.6.2 Bipolar Junction Transistors (BJTs)....9-17
9.6.3 Junction Field-Effect Transistors (JFETs)....9-18
9.6.4 MOSFETs....9-19
9.7 Models and Subcircuits 9-20
9.7.1 Creating Model Files 9-20
9.7.1.1 Example File....9-21
9.7.1.2 Including Models in your SPICE netlist 9-21
9.7.2 Creating a Subcircuit File 9-21
9.7.2.1 Using Sch2spi to create a subcircuit definition....9-21
9.7.3 Including Subcircuits....9-22
9.7.4 Passing Parameters to Subcircuits 9-22
9.7.5 Device and Subcircuit Libraries 9-23
9.8 Analysis and Output 9-23
9.8.1 Using SPICE Command Files....9-23
9.8.2 Using SPICE Control Symbols....9-24
9.8.3 Analysis 9-24
9.8.3.1 DC Analysis....9-24
9.8.3.2 AC Analysis....9-25
9.8.3.3 Transient Analysis 9-26
9.8.3.4 DC Operating Point Analysis 9-26
9.8.3.4.1 The .OP Card....9-26
9.8.3.5 Temperature Analysis....9-26
9.8.3.5.1 The .TEMP Card 9-26
9.8.3.6 Meters....9-27
9.8.3.7 Other SPICE Control Symbols 9-28
9.8.3.7.1 The library Symbol....9-28

9.8.3.7.2 The options Symbol 9-28
9.8.3.7.3 The include Symbol 9-28
9.8.3.7.4 The define Symbol 9-28
9.8.3.7.5 The Distribution Card 9-28

Chapter 9 SPICE Simulator Interface

The CapFast SPICE netlister is called *SPICE Simulator Interface*, which we will refer to subsequently as *Sch2spi*. This chapter shows you how to use *Sch2spi* to produce netlists for input to PSPICE, HSPICE, ISSPICE, and SPICE 2G6. In many ways, we have made this chapter an introduction for designers who are doing simulation for the first time, but also have structured it in a way that allows experienced SPICE users to quickly glean relevant information.

We've divided this chapter into five sections. Each section deals with an important aspect of the SPICE netlister, providing examples of usage and the definitions of important terms. Here's a summary of each section.

Getting Started with *Sch2spi*:
Describes the input and output files for *Sch2spi* and how you can use *Sch2spi* to produce a netlist for a specific type of SPICE.

***Sch2spi* Options:**
Shows you the options you can use with *Sch2spi*.

Properties and Qualifiers:
Shows you how to use properties and qualifiers with *Sch2spi* and lists the properties required for each SPICE device.

Models and Subcircuits:
Shows you how to include device models and subcircuit definitions in your SPICE netlist. Also shows you how to use *Sch2spi* to create a subcircuit.

Analysis and Output:
Shows you how to perform analysis on your circuit and how to use the CapFast SPICE post-processing tools to see the results of your simulation.

9.1 Getting Started

Running *Sch2spi* requires that you use your schematic (`.sch`) or multi-page schematic design (`.dsn`) files as input. *Sch2spi* accepts other input files but they are optional. These optional files may supply model definitions, subcircuit definitions, and command scripts. You'll learn how to use model and subcircuit files, as well as command scripts later in this chapter. Running *Sch2spi* produces an input netlist for PSPICE, HSPICE, ISSPICE or SPICE 2G6.

9.1.1 SPICE Parts

You can find all the symbols representing SPICE devices in the SPICE parts library. To get parts from the SPICE library, first configure the `Part` menu to use the SPICE library with `Configure→Select Parts Library→SPICE Library`. Then select the `Part` menu, and a list of SPICE sub-menus appears. Choose the appropriate menu for the device you want.

9.1.2 Running *Sch2spi*

To run *Sch2spi*, double-click on the `Sch2spi` icon in the CapFast for Windows group, or, on UNIX systems, type **`sch2spi`** at the system prompt. To produce a netlist for a specific SPICE format, use the following options:

```
    -Q SPICE_TYPE BASENAME
```

where *`SPICE_TYPE`* may be `PSPICE`, `HSPICE`, `ISSPICE`, or `2G6`; where `-Q` defines the type of SPICE extraction, as well as enabling the properties qualified with *`SPICE_TYPE`* (the `-Q` option and property qualifiers are discussed later in this chapter in *Options* and *Properties and Qualifiers*); and where *`BASENAME`* is the name of your schematic or design file without the `.sch` or `.dsn` extension. Using the extension is optional.

9.1.3 *Sch2spi* Output files

Sch2spi outputs a SPICE netlist file and two alias files. The alias files are simply text files that list names of devices and nodes and their corresponding names in the SPICE netlist.

The format and file extension of the SPICE netlist file depends on what kind of SPICE you are extracting for. Here's a list of some of the files and their extensions that *Sch2spi* can produce.

9.1.3.1 SPICE netlist files

- The file *`BASENAME`*`.spi` for SPICE 2G6 and HSPICE
- The file *`BASENAME`*`.cir` for PSPICE and ISSPICE

9.1.3.2 Two CapFast SPICE Alias files

The file *`BASENAME`*`.nn` which lists the names of nodes and devices in the netlist and their corresponding instance names.

The file `BASENAME.met` which lists the names for meters in the SPICE netlist and the corresponding names supplied by you.

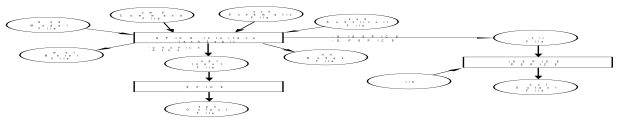

Figure 9-1. Extraction for SPICE

Figure 9-1 is a diagram that illustrates SPICE extraction. It shows the input files *Sch2spi* accepts and various kinds of output files that it can produce.

9.1.4 Device Name Assignment

SPICE recognizes certain names and rejects others. By default, *Sch2spi* uses the instance name of a symbol as the SPICE identifier, if that instance name is a legal SPICE identifier. If it is not a legal identifier, then *Sch2spi* automatically generates a unique and legal SPICE identifier for that symbol. You can change the name of a symbol or node in your schematic by selecting the symbol or node and invoking the `Text→Relabel` command.

You may have cases where you do not want *Sch2spi* to use the instance name of the device as the SPICE identifier. In these cases, you can use the CapFast packaging utility, *Component Packager*. (To use *Component Packager* with *Sch2spi*, you must use the `-ref` option). *Component Packager* automatically assigns unique reference designators to parts and has provisions for maintaining those names when you make modifications to the schematic. To learn how to use *Component Packager*, see *Chapter 8: Extracting PCB Netlists*.

9.1.5Node Name Assignment

By default, *Sch2spi* assigns a unique node number to every node in your schematic. The exception is if you name a node with a number, in which case *Sch2spi* maintains that name of the node.

When you become experienced at running *Sch2spi*, you might find situations where you want *Sch2spi* to pass alphanumeric node names into your netlist, instead of just numerical names. To do this simply use the `-alpha` option with *Sch2spi*:

```
sch2spi -Q PSPICE SPICE_TYPE -alpha BASENAME
```

9.2 *Sch2spi* Options

The following is a list of options *Sch2spi* recognizes.

`-alpha`	Tells *Sch2spi* to pass alphanumeric node names into the netlist. If *Sch2spi* encounters a non-alphanumeric node name, it always generates a unique number for that name. The default is to generate a unique number for all non-numeric node names.
`-k` *`CONTROL_FILE_PATH`*	Tells *Sch2spi* in which directory to look when searching for model files. The default path is `.+~+~p3/wcs/lib`. You can append or prepend to the default path or override it with the `-k` option. The `-k` option works the same as the `-p` option, explained further on this list.
`-l` *`LIBRARY_FILE`*	Tells *Sch2spi* to substitute *`LIBRARY_FILE`*`.sub` for `BASENAME.sub` as the subcircuit file. You should place the subcircuit file on the current data file search path. Use the `-p` option on the command line or in your `cad.rc` to define the data file search path. For more on subcircuits and the `-l` option, see the *Models and Subcircuits* section of this chapter.
`-nomod`	Tells *Sch2spi* not to search for models. You'll find this option useful if you use external device libraries included by SPICE at run-time. The default is to search for models and print warning messages if not found.
`-p` *`DATA_FILE_PATH`*	Specifies the path you want the program to use when searching for data files, such as schematic or symbol files. See *Appendix A: Customizing CapFast* for more information.
`-Q` *`QUALIFIER_LIST`*	Specifies one or more qualifiers which are in effect for the current netlist extraction. Separate qualifiers with commas if you use more than one. For example: **`sch2spi -Q PSPICE,BYPASS`** *`filename`* adds the `PSPICE` and `BYPASS` qualifiers to the list of qualifiers. The `SPICE` qualifier is automatically added by *Sch2spi*. For more on qualifiers and how they affect extraction, see the *Properties and Qualifiers* section of this chapter.
`-ref`	Use the reference designator assigned by *Component Packager* for device names. The default is to use the symbol instance name.

`-sub`	Tells *Sch2spi* to generate a subcircuit definition and to place it in the file *`BASENAME`*`.sub`. For more on generating subcircuits see the *Models and Subcircuits* section of this chapter.
`-t` *`TECHNOLOGY_FILE`*	Tells *Sch2spi* to look for model definitions in the file *`TECHNOLOGY_FILE`*`.mod`. You can specify the directory path with the `-k` option. For more on device models and the `-t` option, see the *Models and Subcircuits* section of this chapter.

9.3 Properties and Qualifiers

When you generate a netlist, you want *Sch2spi* to extract certain information from your schematic and place it in the netlist. *Sch2spi* knows which information is relevant to the netlist by reading properties. Properties are the means by which you pass information from your schematic to the netlist.

Qualifiers determine which properties are visible to the netlister during extraction. If a property has no qualifier, *Sch2spi* will see it. A qualifier appears in parentheses before the property name. For example,

```
(PSPICE)primitive:r
```

shows that the qualifier is `PSPICE`, the property is `primitive` and the property value is `r`. Qualifiers may be combined by using the logical operators `&`, which means *and*; `!` which means *not*; and `|`, which means *or*. For example:

```
((HSPICE|ISSPICE)&!PSPICE)primitive:x
```

defines a `primitive` property `x` if the `PSPICE` qualifier is not on the qualifier list AND either `HSPICE` or `ISSPICE` is on the qualifier list.

Note: No spaces are allowed within the parentheses.

You can add qualifiers to the qualifier list by using the `-Q` option when you run *Sch2spi*. Properties with no qualifier, or properties qualified with SPICE are always visible to *Sch2spi*.

You can make any recognized property visible to *Sch2spi* by qualifying the property and adding the qualifier to the list. For example, let's say you want the value of a resistor to be 100k in most cases, but for one simulation netlist you want to short the resistor. Here's how you'd define the properties:

```
(SHORT)model:null
(SHORT)wire(n1):n2
(!SHORT)value:100k
```

Now, when you want to produce a netlist with the resistor shorted, run *Sch2spi* with the following options:

```
    -Q SHORT filename
```

The `-Q` option adds the qualifier `SHORT` to the list of qualifiers. This means that all properties qualified with `SHORT` as well as those qualified with `SPICE` and those with no qualifiers are visible to *Sch2spi*.

A word of caution: Only one value of a given property is qualified for a given extraction. If a property with the same name is qualified more than once, the value chosen is unpredictable. Use the logical operators `!`, `&`, and | to make sure only one value of a given property is qualified. If the property is a default property, that is, a property defined on the symbol definition, you will need to edit the symbol using *Symbol Editor* to change the qualifiers. For a more detailed description of properties and qualifiers, see *Chapter 4: Creating Electrical Designs*.

Different SPICE devices have different information which characterizes their behavior. For this reason, the symbols that represent SPICE devices have different properties which are recognized by *Sch2spi*. There is one property required on every non-hierarchical device symbol: the `primitive` property. The `primitive` property tells *Sch2spi* which type of device the symbol represents, and thus which properties to look for on the symbol. All unrecognized primitives are considered to be subcircuits and assigned a `primitive` of `x` in the netlist.

Following is a list of SPICE devices and the properties recognized for each device. Properties whose value is `*req*` are required properties, that is, they must have a value for *Sch2spi* to generate a valid netlist. Properties with a value of `*opt*` are optional; they are recognized by *Sch2spi*, but they are not required to produce a valid netlist.

9.4 Element Cards

The element cards represent resistors, capacitors, inductors, transmission lines, and voltage and current sources.

9.4.1 Resistors

General form:

```
    RXXXXXXX N1 N2 VALUE <TC=TC1<,TC2>>
```

CapFast SPICE resistor properties:

Required	Element Card Field	Example

`primitive:r`	`R`	
`value`	`VALUE`	`value:1K`

Optional

`tc`	`TC=`	`tc:0.001,0.015`

Examples:

```
R1 1 2 100
RC1 12 17 1K TC=0.001,0.015
```

`N1` and `N2` are the two element nodes. `VALUE` is the resistance (in ohms) and may be positive or negative but not zero. `TC1` and `TC2` are the (optional) temperature coefficients; if not specified, zero is assumed for both. The value of the resistor as a function of temperature is given by:

```
value(TEMP)
    =value(TNOM)*(1+TC1*(TEMP-TNOM)+TC2*((TEMP-TNOM)**2))
```

9.4.2 Capacitors and Inductors

General form:

```
CXXXXXXX N+ N- VALUE <IC=INCOND>
LYYYYYYY N+ N- VALUE <IC=INCOND>
```

CapFast SPICE capacitor and inductor properties:

Required	Element Card Field	Example
`primitive:c`	`C`	
or		
`primitive:l`	`L`	
`value`	`VALUE`	`value:10U`
	or	
	`POLY  C0 C1 C2`	`value:POLY  1 2 3`
	or	
	`POLY  L0 L1 L2`	`value:POLY  1 2`

Optional

`ic`	`IC=`	`ic:15.7MA`

Examples:

```
CBYP 13 0 1UF
COSC 17 23 10U IC=3V
LLINK 42 69 1UH
```

```
LSHUNT 23 51 10U IC=15.7MA
```

`N+` and `N-` are the positive and negative element nodes, respectively. `VALUE` is the capacitance in Farads or the inductance in Henries.

For the capacitor, the (optional) initial condition is the initial (time-zero) value of capacitor voltage (in Volts). For the inductor, the (optional) initial condition is the initial (time-zero) value of inductor current (in Amps) that flows from N+, through the inductor, to N-. Note that the initial conditions (if any) apply only if the `UIC` option is specified on the `.TRAN` card.

Nonlinear capacitors and inductors can be described.

General form:

```
CXXXXXXX N+ N- POLY C0 C1 C2 ... <IC=INCOND>
LYYYYYYY N+ N- POLY L0 L1 L2 ... <IC=INCOND>
```

`C0 C1 C2 ...` (and `L0 L1 L2 ...`) are the coefficients of a polynomial describing the element value. The capacitance is expressed as a function of the voltage across the element while the inductance is a function of the current through the inductor. The value is computed as

```
value=C0+C1*V+C2*V**2+...
value=L0+L1*I+L2*I**2+...
```

where `V` is the voltage across the capacitor and `I` the current flowing in the inductor.

9.4.3 Coupled (Mutual) Inductors

General form:

```
KXXXXXXX LYYYYYYY LZZZZZZZ VALUE
```

CapFast SPICE coupled inductor properties:

Required	Element Card Field	Example
primitive:k	K	
mutind	VALUE	mutind:0.999
ind1	LYYYYYYY	ind1:LAA
ind2	LZZZZZZZ	ind2:LBB

Examples:

```
K43 LAA LBB 0.999KXFRMR L1 L2 0.87
```

LYYYYYYY and LZZZZZZZ are the names of the two coupled inductors, and VALUE is the coefficient of coupling, K, which must be greater than 0 and less than or equal to 1. Using the "dot" convention for inductors, place a "dot" on the first node of each inductor.

9.4.4 Transmission Lines (Lossless)

General form:

```
    TXXXXXXX N1 N2 N3 N4 Z0=VALUE <TD=VALUE><F=FREQ <NL=NRMLEN>>
        + <IC=V1,I1,V2,I2>
```

CapFast SPICE transmission line properties:

Required	**Element Card Field**	**Example**
primitive:t	T	
z0	Z0	z0:50
<u>Optional</u>		
td	TD=	td:10NS
f	F=	f:50KHZ
nl	NL=	nl:.25
ic	IC=	ic:.5V,.01A,.2V,.02A

Note that one of td or f must be specified.

Examples:

```
    T1 1 0 2 0 Z0=50 TD=10NS
```

N1 and N2 are the nodes at port 1; N3 and N4 are the nodes at port 2. Z0 is the characteristic impedance. The length of the line may be expressed in either of two forms. The transmission delay, TD, may be specified directly (as TD=10ns, for example). Alternatively, a frequency F may be given, together with NL, the normalized electrical length of the transmission line with respect to the wavelength in the line at the frequency F. If a frequency is specified but NL is omitted, 0.25 is assumed (that is, the frequency is assumed to be the quarter-wave frequency).

Note that although both forms for expressing the line length are indicated as optional, one of the two must be specified.

Also note that this element models only one propagating mode. If all four nodes are distinct in the actual circuit, then two modes may be excited. To simulate such a situation, two transmission line elements are required.

The (optional) initial condition specification consists of the voltage and current at each of the transmission line ports. Note that the initial conditions (if any) apply *only* if the UIC option is specified on the .TRAN card.

The user should be aware that SPICE will use a transient time-step which does not exceed ½ the minimum transmission line delay. Therefore very short transmission lines (compared with the analysis time frame) will cause long run times.

9.5 Linear Dependent Sources

SPICE allows circuits to contain linear dependent sources characterized by any of the four equations:

i=g*v
v=e*v
i=f*i
v=h*i

where g, e, f, and h are constants representing transconductance, voltage gain, current gain, and transresistance, respectively.

9.5.1 Linear Voltage-Controlled Current Sources

General form:

```
GXXXXXXX N+ N- NC+ NC- VALUE
```

CapFast SPICE linear voltage-controlled current source properties:

Required	Element Card Field	Example
primitive:g	G	
value	VALUE	value:0.1MMHO

Examples:

```
G1 2 0 5 0 0.1MMHO
```

`N+` and `N-` are the positive and negative nodes, respectively. Current flow is from the positive node, through the source, to the negative node. `NC+` and `NC-` are the positive and negative controlling nodes, respectively. `VALUE` is the transconductance (in mhos).

9.5.2 Linear Voltage-Controlled Voltage Sources

General form:

```
EXXXXXXX N+ N- NC+ NC- VALUE
```

CapFast SPICE linear voltage-controlled current source properties:

Required	Element Card Field	Example
`primitive:e`	`E`	
`value`	`VALUE`	`value:2.0`

Examples:

```
    E1 2 3 14 1 2.0
```

`N+` is the positive node, and `N-` is the negative node. `NC+` and `NC-` are the positive and negative controlling nodes, respectively. `VALUE` is the voltage gain.

9.5.3 Linear Current-Controlled Current Sources

General form:

```
    FXXXXXXX N+ N- VNAM VALUE
```

CapFast SPICE linear current-controlled current source properties:

Required	Element Card Field	Example
`primitive:f`	`F`	
`value`	`VALUE`	`value:5`

Examples:

```
    F1 13 5 VSENS 5
```

`N+` and `N-` are the positive and negative nodes, respectively. Current flow is from the positive node, through the source, to the negative node. `VNAM` is the name of a voltage source through which the controlling current flows. The direction of positive controlling current flow is from the positive node, through the source, to the negative node of `VNAM`. `VALUE` is the current gain.

9.5.4 Linear Current-Controlled Voltage Sources

General form:

```
    HXXXXXXX N+ N- VNAM VALUE
```

CapFast SPICE linear current-controlled voltage source properties:

Required	Element Card Field	Example

primitive:h	H	
value	VALUE	value:0.5K

Examples:

```
    HX 5 17 VZ 0.5K
```

N+ and N- are the positive and negative nodes, respectively. VNAM is the name of a voltage source through which the controlling current flows. The direction of positive controlling current flow is from the positive node, through the source, to the negative node of VNAM. VALUE is the transresistance (in ohms).

9.5.5 Independent Sources

General form:

```
    VXXXXXXX N+ N- <<DC> DC/TRAN VALUE> <AC <ACMAG <ACPHASE>>>
    IYYYYYYY N+ N- <<DC> DC/TRAN VALUE> <AC <ACMAG <ACPHASE>>>
```

CapFast SPICE independent source properties:

Required	Element Card Field	Example
primitive:v	V	
primitive:i	I	
<u>Optional</u>		
dctran	DC/TRAN VALUE	dctran:5V
	or	
	PULSE(V1 V2 TD TR TF PW PER)	dctran:PULSE(-1 1 2NS 2NS 2NS 50NS 100NS)
	or	
	SIN(VO VA FREQ TD THETA)	dctran:SIN(0 1 100MEG 1NS 1E10)
	or	
	EXP(V1 V2 TD1 TAU1 TD2 TAU2)	dctran:EXP(-4 -1 2NS 30NS 60NS 40NS)
	or	
	PWL(T1 V1 <T2 V2 T3 V3 ...>)	dctran:PWL(0 -7 10NS -7 11NS -3 17NS -3 18NS -7 50NS -7)
	or	
	SFFM(VO VA FC MDI FS)	dctran:SFFM(0 1 10K 5 1K)
acmag	ACMAG	acmag: 10V
acphase	ACPHASE	acphase: 90

Examples:

```
    VCC 10 0 DC 6
    VIN 13 2 0.001 AC 1 SIN(0 1 1MEG)
    ISRC 23 21 AC 0.333 45.0 SFFM(0 1 10K 5 1K)
    VMEAS 12 9
```

`N+` and `N-` are the positive and negative nodes, respectively. Note that voltage sources need not be grounded. Positive current is assumed to flow from the positive node, through the source, to the negative node. A current source of positive value, will force current to flow out of the `N+` node, through the source, and into the `N-` node. Voltage sources, in addition to being used for circuit excitation, are the "ammeters" for SPICE, that is, zero valued voltage sources may be inserted into the circuit for the purpose of measuring current. They will, of course, have no effect on circuit operation since they represent short-circuits.

`DC/TRAN` is the dc and transient analysis value of the source. If the source value is zero both for dc and transient analyses, this value may be omitted. If the source value is time-invariant (e.g., a power supply), then the value may optionally be preceded by the letters `DC`.

`ACMAG` is the ac magnitude and `ACPHASE` is the ac phase. The source is set to this value in the ac analysis. If `ACMAG` is omitted following the keyword `AC`, a value of unity is assumed. If `ACPHASE` is omitted, a value of zero is assumed. If the source is not an ac small-signal input, the keyword `AC` and the ac values are omitted.

Any independent source can be assigned a time-dependent value for transient analysis. If a source is assigned a time-dependent value, the time-zero value is used for dc analysis. There are five independent source functions: pulse, exponential, sinusoidal, piece-wise linear, and single-frequency FM. If parameters other than source values are omitted or set to zero, the default values shown will be assumed. (`TSTEP` is the printing increment and `TSTOP` is the final time (See the `.TRAN` card for explanation)).

9.5.5.1 Pulse

General form:

```
    PULSE(V1 V2 TD TR TF PW PER)
```

Examples:

```
    VIN 3 0 PULSE(-1 1 2NS 2NS 2NS 50NS 100NS)
```

parameters	default values	units
`V1` (initial value)		Volts or Amps
`V2` (pulsed value)		Volts or Amps
`TD` (delay time)	0.0	seconds
`TR` (rise time)	`TSTEP`	seconds

TF (fall time)	TSTEP	seconds
PW (pulse width)	TSTOP	seconds
PER (period)	TSTOP	seconds

A single pulse so specified is described by the following table:

time	value
0	V1
TD	V1
TD+TR	V2
TD+TR+PW	V2
TD+TR+PW+TF	V1
TSTOP	V1

Intermediate points are determined by linear interpolation.

9.5.5.2 Sinusoidal

General form:

```
    SIN(VO VA FREQ TD THETA)
```

Examples:

```
    VIN 3 0 SIN(0 1 100MEG 1NS 1E10)
```

parameters	default value	units
VO (offset)		Volts or Amps
VA (amplitude)		Volts or Amps
FREQ (frequency)	1/TSTOP	Hz
TD (delay)	0.0	seconds
THETA (damping factor)	0.0	1/seconds

The shape of the waveform is described by the following table:

time	value
0 to TD	VO
TD to TSTOP	VO + VA*exp(-(time-TD)*THETA)*sine(2*pi*FREQ*(time+TD))

9.5.5.3 Exponential

General form:

```
EXP(V1 V2 TD1 TAU1 TD2 TAU2)
```

Examples:

```
VIN 3 0 EXP(-4 -1 2NS 30NS 60NS 40NS)
```

parameters	default values	units
V1 (initial value)		Volts or Amps
V2 (pulsed value)		Volts or Amps
TD1 (rise delay time)	0.0	seconds
TAU1 (rise time constant)	TSTEP	seconds
TD2 (fall delay time)	TD1+TSTEP	seconds
TAU2 (fall time constant)	TSTEP	seconds

The shape of the waveform is described by the following table:

time	value
0 to TD1	V1
TD1 to TD2	V1+(V2-V1)*(1-exp(-(time-TD1)/TAU1))
TD2 to TSTOP	V1+(V2-V1)*(1-exp(-(time-TD1)/TAU1)) +(V1-V2)*(1-exp(-(time-TD2)/TAU2))

9.5.5.4 Piece-Wise Linear

General form:

```
PWL(T1 V1 <T2 V2 T3 V3 T4 V4 ...>)
```

Examples:

```
VCLOCK 7 5 PWL(0 -7 10NS -7 11NS -3 17NS -3 18NS -7 50NS -7)
```

Each pair of values (Ti, Vi) specifies that the value of the source is Vi (in Volts or Amps) at time=Ti. The value of the source at intermediate values of time is determined by using linear interpolation on the input values.

9.5.5.5 Single-Frequency FM

General form:

```
SFFM(VO VA FC MDI FS)
```

Examples:

```
    V1 12 0 SFFM(0 1M 20K 5 1K)
```

parameters	default values	units
VO (offset)		Volts or Amps
VA (amplitude)		Volts or Amps
FC (carrier frequency)	1/TSTOP	Hz
MDI (modulation index)		
FS (signal frequency)	1/TSTOP	Hz

The shape of the waveform is described by the following equation:

VALUE = VO + VA*sine((2*pi*FC*time)+MDI*sine(2*pi*FS*time))

9.6 Semiconductor Devices

The elements that have been described to this point typically require only a few parameter values to specify completely the electrical characteristics of the element. However, the models for the four semiconductor devices that are included in the SPICE program require many parameter values. Moreover, many devices in a circuit often are defined by the same set of device model parameters. For these reasons, a set of device model parameters is defined on a separate .MODEL card and assigned a unique model name. The device element cards in SPICE then reference the model name. This scheme alleviates the need to specify all of the model parameters on each device element card.

Each device element card contains the device name, the nodes to which the device is connected, and the device model name. In addition, other optional parameters may be specified for each device: geometric factors and an initial condition.

The area factor used on the diode, BJT and JFET device card determines the number of equivalent parallel devices of a specified model. The affected parameters are marked with an asterisk under the heading "area" in the model descriptions below. Several geometric factors associated with the channel and the drain and source diffusions can be specified on the MOSFET device card.

Two different forms of initial conditions may be specified for devices. The first form is included to improve the dc convergence for circuits that contain more than one stable state. If a device is specified OFF, the dc operating point is determined with the terminal voltages for that device set to zero. After convergence is obtained, the program continues to iterate to obtain the exact value for the terminal voltages. If a circuit has more than one dc stable state, the OFF option can be used to force the solution to correspond to a desired state. If a device is specified OFF when in reality the device is conducting, the program will still obtain the correct solution (assuming the solutions converge) but more iterations will be required since the program must independently converge to two separate solutions. The `.NODESET` card serves a similar purpose as the OFF option. The `.NODESET` option is easier to apply and is the preferred means to aid convergence.

The second form of initial conditions are specified for use with the transient analysis. These are true "initial conditions" as opposed to the convergence aids above. See the description of the `.IC` card and the `.TRAN` card for a detailed explanation of initial conditions.

9.6.1 Junction Diodes

General form:

```
    DXXXXXXX N+ N- MNAME <AREA> <OFF> <IC=VD>
```

CapFast SPICE junction diode properties:

Required	**Element Card Field**	**Example**
primitive:d	D	
mname	MNAME	mname:DIODE1
<u>Optional</u>		
area	AREA	area:3.0
off	OFF	off:0
ic	IC=	ic:0.2

Examples:

```
    DBRIDGE 2 10 DIODE1
    DCLMP 3 7 DMOD 3.0 IC=0.2
```

`N+` and `N-` are the positive and negative nodes, respectively. `MNAME` is the model name, `AREA` is the area factor, and `off` indicates an (optional) starting condition on the device for dc analysis. If the area factor is omitted, a value of 1.0 is assumed. The (optional) initial condition specification using `IC=VD` is intended for use with the UIC option on the `.TRAN` card, when a transient analysis is desired starting from other than the quiescent operating point.

9.6.2 Bipolar Junction Transistors (BJTs)

General form:

```
    QXXXXXXX NC NB NE <NS> MNAME <AREA> <OFF> <IC=VBE,VCE>
```

CapFast SPICE bipolar junction transistor properties:

Required	Element Card Field	Example
`primitive:q`	`Q`	
`mname`	`MNAME`	`mname:NBJT1`
Optional		
`area`	`AREA`	`area:1.0`
`off`	`OFF`	`off:0`
`ic`	`IC=`	`ic:0.6,5.0`

Examples:

```
Q23 10 24 13 QMOD IC=0.6,5.0
Q50A 11 26 4 20 MOD1
```

`NC`, `NB`, and `NE` are the collector, base, and emitter nodes, respectively. `NS` is the (optional) substrate node. If unspecified, ground is used. `MNAME` is the model name, `AREA` is the area factor, and `OFF` indicates an (optional) initial condition on the device for the dc analysis. If the area factor is omitted, a value of 1.0 is assumed. The (optional) initial condition specification using `IC=VBE,VCE` is intended for use with the UIC option on the `.TRAN` card, when a transient analysis is desired starting from other than the quiescent operating point. See the `.IC` card description for a better way to set transient initial conditions.

9.6.3 Junction Field-Effect Transistors (JFETs)

General form:

```
JXXXXXXX ND NG NS MNAME <AREA> <OFF> <IC=VDS,VGS>
```

CapFast SPICE junction field-effect transistor properties:

Required	Element Card Field	Example
`primitive:j`	`J`	
`mname`	`MNAME`	`mname:JFET1`
Optional		
`area`	`AREA`	`area:1.0`
`off`	`OFF`	`off:0`
`ic`	`IC=`	`ic:0.6,5.0`

Examples:

```
J1 7 2 3 JM1 OFF
```

ND, NG, and NS are the drain, gate, and source nodes, respectively. MNAME is the model name, AREA is the area factor, and OFF indicates an (optional) initial condition on the device for dc analysis. If the area factor is omitted, a value of 1.0 is assumed. The (optional) initial condition specification, using IC=VDS,VGS is intended for use with the UIC option on the .TRAN card, when a transient analysis is desired starting from other than the quiescent operating point (see the .IC card for a better way to set initial conditions).

9.6.4 MOSFETs

General form:

```
MXXXXXXX ND NG NSNB MNAME <L=VAL> <W=VAL> <AD=VAL> <AS=VAL> +
    <PD=VAL><PS=VAL> <NRD=VAL> <NRS=VAL><OFF>
    <IC=VDS,VGS,VBS>
```

CapFast SPICE junction field-effect transistor properties:

Required	Element Card Field	Example
primitive:m	M	
mname	MNAME	mname:NMOS1
Optional		
l	L	l:10
uw	W	w:5U
ad	AD=	ad:100P
as	AS=	as:100P
pd	PD=	pd:40U
ps	PS=	ps:40U
nrd	NRD=	nrd:1.0n
rs	NRS=	nrs:1.0
off	OFF	off:0
ic	IC=	ic:0.6,5.0,-1.5

Examples:

```
M1 24 2 0 20 TYPE1
M31 2 17 6 10 MODM L=5U W=2U
M31 2 16 6 10 MODM 5U 2U
M1 2 9 3 0 MOD1 L=10U W=5U AD=100P AS=100P PD=40U PS=40U
M1 2 9 3 0 MOD1 10U 5U 2P 2P
```

ND, NG, NS, and NB are the drain, gate, source, and bulk (substrate) nodes, respectively. MNAME is the model name. L and W are the channel length and width, in meters. AD and AS are the areas of the drain and source diffusions, in sq-meters. Note that the suffix U specifies microns (1E-6 m) and `P` sq-microns (1E-12 sq-m). If any of `L`, `W`, `AD`, or `AS` are not specified, default values are used. The user may specify the values to be used for these default parameters on the `.OPTIONS` card. The use of defaults simplifies input deck preparation, as well as the editing required if device geometries are to be changed. `PD` and `PS` are the perimeters of the drain and source junctions, in meters. `NRD` and `NRS` designate the equivalent number of squares of the drain and source diffusions; these values multiply the sheet resistance `RSH` specified on the `.MODEL` card for an accurate representation of the parasitic series drain and source resistance of each transistor. `PD` and `PS` default to 0.0 while `NRD` and `NRS` to 1.0. `OFF` indicates an (optional) initial condition on the device for dc analysis. The (optional) initial condition specification using `IC=VDS,VGS,VBS` is intended for use with the UIC option on the `.TRAN` card, when a transient analysis is desired starting from other than the quiescent operating point. See the `.IC` card for a better and more convenient way to specify transient initial conditions.

9.7 Models and Subcircuits

Certain SPICE primitive devices require you to have model definitions to characterize their behavior. There are several ways to include model definitions in your netlist, and most current versions of SPICE, such as HSPICE, PSPICE and ISSPICE, provide libraries with model and subcircuit definitions in them. This section will:

- Show you how to create and include user-defined models and subcircuits
- Discuss the topic of device libraries.

9.7.1 Creating Model Files

You use model files to specify models for one or more devices. You create these files with a text editor of your choice. Each line specifies a device model unless the line begins with a `+`, in which case it is a continuation of the previous line. The format for each entry in a model file is:

```
.MODEL MODEL_NAME MODEL_TYPE(PARAMETERS)
```

where *`MODEL_NAME`* corresponds to the value assigned to the `mname` property on the device you are modeling. Any valid identifier is acceptable. The *`MODEL_TYPE`* is the SPICE model type such as `nmos`, `pmos`, `npn`, `pnp`, `d`, etc. *`PARAMETERS`* is a string containing the parameters that characterize the model. See your SPICE user guide for valid model parameters. If the parameters take more than one line, insert a `+` at the beginning of each continuation line.

9.7.1.1 Example File

The following is an example of a `.mod` file that describes the n-channel enhancement and n-channel depletion transistors:

```
.MODEL NENH NMOS(LEVEL=2 VTO=1.0 KP=2.1E-5 GAMMA=1.4
+RD=5.0 RS=5.0 CGSO=2.3E-10 CGDO=2.3E-10 RSH=30
+CJ=3.5E-4 CJSW=3.0E-10
+TOX=5.5E-8 NSUB=2.6E16 TPG=1 XJ=0.6U
+LD=0.65U UO=490)
.model PENH PMOS(LEVEL=2 VTO=-1.0 KP=8.0E-6 GAMMA=0.6
+RD=5.0 RS=5.0
+CGSO=1.4E-10 CGDO=1.4E-10 RSH=30
+CJ=2.2E-4 CJSW=2.0E-10
+TOX=5.5E-8 NSUB=3.7E15 TPG=-1 XJ=0.6U
+LD=0.65U UO=150)
```

9.7.1.2 Including Models in your SPICE netlist

You can include models in your netlist in several ways. You can use the `-t` *`TECHNOLOGY_FILE`* option for specifying a file which includes many model definitions. If you specify *`TECHNOLOGY_FILE`*, *Sch2spi* looks there first for models. Only those models referenced by devices will be included. If you do not specify *`TECHNOLOGY_FILE`*, *Sch2spi* looks for model files with the name `MNAME.mod`, where `MNAME` is any value of any `mname` property you have placed on devices in your schematic. The default search path for all model files, including *`TECHNOLOGY_FILE`* is first, the current directory (`.`), then `~`, and lastly `~p3/wcs/lib`. You can change this path with the `-k` option. Options are discussed earlier in this chapter. If the models are not found, *Sch2spi* prints a warning message unless you have disabled model searching with the `-nomod` option.

9.7.2 Creating a Subcircuit File

There are two ways to create a subcircuit for incorporation into your netlist. You can create a subcircuit file with the text editor of your choice. You can also use *Sch2spi* to create a subcircuit from a schematic you created with *Schematic Editor*.

9.7.2.1 Using **Sch2spi** *to create a subcircuit definition*

To generate a subcircuit definition from a schematic you have created using *Schematic Editor*, you need to identify the nodes which will be inputs and outputs to the subcircuit. To identify the input and output nodes on your circuit, connect a subcircuit connector to the node. The subcircuit connector is on the `Parts` menu after it has been configure to use the SPICE library. The name of the connector is used to order the nodes on the `.SUBCKT` line from lexically least. For example, if you have a subcircuit with four ports named `vin`, `v2`, `v12`, and `vin2`, the order will be `v2`, `v12`, `vin`, `vin2`. You need to be sure that the order of the nodes on the symbol that represents the subcircuit is the same as the order on the `.subckt` line. You may want to use numbers to ensure that the nodes are properly ordered. You can always change the name of the subcircuit connector by using *Schematic Editor*'s `Text→Relabel` command.

To create a subcircuit file using *Sch2spi*, invoke *Sch2spi* with the `-sub` option. This will generate a file named *`BASENAME`*`.sub` which you can include in your schematic.

9.7.3 Including Subcircuits

To include a subcircuit in your schematic, you need to have a symbol that represents the subcircuit. You can create a symbol yourself using *Symbol Editor*, or use one of the predefined subcircuit symbols on the SPICE parts menus. Subcircuit symbols must have the following properties:

```
primitive:X
order:PORT1,PORT2,...,PORTN
subnam:SUBCIRCUIT_NAME
```

The `order` property identifies the order in which the nodes should appear on the subcircuit invocation. These must match the order of the corresponding ports on the subcircuit definition.

Warning

Sch2spi does not check the correspondence of ports between subcircuit definition and invocation. If you generated the subcircuit using *Sch2spi*, the ports for the subcircuit definition are listed in order from lexically least. To ensure correspondence between subcircuit definition and invocation, every port in the order list on a given symbol should be listed in order from lexically least, and should have a corresponding node connected to a subcircuit connector.

The symbol merely references (invokes) the subcircuit; we must also include the subcircuit definition. *Sch2spi* automatically includes *`BASENAME`*`.sub`, or if you use the `-l` *`SUBCKT_FILE`* option, *Sch2spi* includes *`SUBCKT_FILE`*. In addition, *Sch2spi* searches the data file search path for a file named *`SUBNAM`*`.sub`, where *`SUBNAM`* is the value of the `subname` property on the symbol that represents the circuit. You can change the data file search path by using the `-p` option.

9.7.4 Passing Parameters to Subcircuits

Some versions of SPICE support parameter passing to subcircuit definitions. To pass parameters from a subcircuit symbol to a subcircuit definition, the following properties must be added to the subcircuit symbol:

```
porder:PARAMETER-1,PARAMETER-2,...,PARAMETER-N
PARAMETER-1:value
PARAMETER-2:value
     .
     .
     .
PARAMETER-N:value
```

For example, to pass the parameters `FT` and `DVDT` to a subcircuit, you'd add the following properties to the subcircuit symbol:

```
porder:FT,DVDT
FT:8.0MEG
DVDT:2.8MEG
```

9.7.5 Device and Subcircuit Libraries

Some versions of SPICE supply libraries with device models and subcircuit definitions. The references to models and subcircuits are resolved when you run SPICE. To use these libraries, you will need to include the `LIBRARY` symbol in your schematic. It is found on the `Part→Meters/Control→SPICE Control Symbols` menu. On the `LIBRARY` symbol, the `value` property lists the names of the libraries employed in the schematic. For example, to include all of the default libraries supplied with PSPICE, the value on the library symbol would be `NOM.LIB`. If you are using vendor-supplied libraries, you'll probably want to run *Sch2spi* with the `-nomod` option. See the *Options* section of this chapter for more information.

9.8 Analysis and Output

There are two ways to place SPICE analysis commands in your netlist:

- Place the SPICE command and control symbols on your schematic using *Schematic Editor*.
- Write a command file using a text editor. *Sch2spi* automatically processes and includes the *`BASENAME`*`.cmd` file if found on the data file search path.

You can use either or both of these methods. The command file offers more versatility.

9.8.1 Using SPICE Command Files

The SPICE command file provides a way to have *Sch2spi* automatically include the statements and commands your version of SPICE needs to perform circuit analysis. If the name of your schematic is `BASENAME.sch` or `BASENAME.dsn` then create a file using a text editor and name the file BASENAME.cmd. This file may contain any command or statements that your version of SPICE recognizes, including model and subcircuit definitions. The only adjustment you need to make is to precede the node names by a $, if you want *Sch2spi* to substitute the assigned node number for the name. When you invoke *Sch2spi*, it will look for a the command file on the current data file search path as defined in your cad.rc file or with the -p option.

9.8.2 Using SPICE Control Symbols

The Meters/Control menu in the SPICE Part menu includes symbols for controlling the analysis and output of SPICE. Included are symbols for various types of analyses and other common SPICE control commands.

9.8.3 Analysis

Symbols are provided to perform dc, ac, transient, temperature and dc operating point analysis. These symbols are found on the Part→Meters/Control menu. A summary of the function and properties required for each of these symbols follows.

9.8.3.1 DC Analysis

Sch2spi treats the dc analysis (dc) element as two pairs of ports. It uses each pair to insert into the circuit the source that the dc element controls. This implementation makes it unnecessary for the dc element to have the name of the voltage or current source.

The dc analysis control symbol defines sources and control for a dc transfer curve analysis of the circuit. The sources are defined in terms of the range of a voltage or current sweep. The form of the .DC card this symbol represents is as follows:

```
.DC SRCNAM VSTART VSTOP VINCR [SRC2 START2 STOP2 INCR2]
```

The symbol properties correspond to the element card fields as follows:

```
primitive NAME1 START1 STOP1 INCR1 [NAME2 START2 STOP2 INCR2]
```

TYPE1 and TYPE2 are also CapFast SPICE properties on the symbol. The properties ending in 1 are for the required first source, and the rest are for the optional second source. Both TYPE1 and TYPE2 are either v or i. Both sources can be either voltage or current. START, STOP, and INCR are the start, stop, and increment of the voltage or current sweep. NAME1 and NAME2 let you optionally specify a name to show up on the output graphs.

After processing by *Sch2spi*, each dc control element gives rise to two lines in the SPICE netlist. The first line is for the control and the second is for its source. For example, suppose you fill in a dc control primitive as follows:

```
primitive:dc
type1:v
start1:.25
stop1:5.0
incr1:.25
name1:in
type2:
start2:
stop2:
incr2:
name2:
```

Using this information, *Sch2spi* might produce the following lines in the SPICE netlist.

```
DC V1  0.25 5.0 0.25
V1 8 9
```

9.8.3.2 AC Analysis

The ac analysis control symbol controls an ac frequency analysis of the circuit. The sweep frequency range controls all of the ac sources in the circuit. The form of the `.AC` card this symbol represents is as follows:

```
.AC DEC ND FSTART FSTOP
```

or

```
.AC OCT NO FSTART FSTOP
```

or

```
.AC LIN NP FSTART FSTOP
```

The symbol properties correspond to the element card files as follows:

```
primitive ND FSTART FSTOP
```

or

```
primitive NO FSTART FSTOP
```

or

```
primitive NP FSTART FSTOP
```

`FSTART` and `FSTOP` are the start and stop frequencies. One of `ND`, `NO`, or `NP` is required. They are, respectively, the number of points per decade (decade variation), the number of points per octave (octave variation), and the number of points (linear variation).

9.8.3.3 Transient Analysis

This control symbol performs a transient analysis of the circuit over the specified time interval. The form of the `.TRAN` card this symbol represents is as follows:

```
.TRAN TSTEP TSTOP <TSTART <TMAX>><UIC>
```

The symbol properties correspond to the element card fields as follows:

```
primitive TSTEP TSTOP <TSTART <TMAX>><UIC>
```

9.8.3.4 DC Operating Point Analysis

9.8.3.4.1 The `.OP` Card

The `OP` symbol includes a `.OP` card in the SPICE netlist. This symbol is found on the `Part→Meters/Control→SPICE Control Symbols` menu.

General form:

```
.OP
```

The inclusion of this card in an input deck will force SPICE to determine the dc operating point of the circuit with inductors shorted and capacitors opened.

A dc analysis is automatically performed prior to a transient analysis to determine the transient initial conditions, and prior to an ac small-signal analysis to determine the linearized, small-signal models for nonlinear devices.

SPICE performs a dc operating point analysis if no other analyses are requested.

9.8.3.5 Temperature Analysis

9.8.3.5.1 The `.TEMP` Card

The `.TEMP` card can be placed in the SPICE input deck by including the `TEMP` symbol in the schematic. This symbol is found on the `Part→Meters/Control→SPICE Control Symbols` menu. It can then be automatically incorporated into your SPICE deck.

General form:

```
.TEMP T1 <T2 <T3 ...>>
```

Examples:

```
.TEMP -55.0 25.0 125.0
```

This card specifies the temperatures at which the circuit is to be simulated. `T1`, `T2`, `...` are the different temperatures, in degrees C. Temperatures less than -223.0 deg C are ignored. Model data are specified at `TNOM` degrees (see the `.OPTIONS` card for `TNOM`); if the `.TEMP` card is omitted, the simulation will also be performed at a temperature equal to `TNOM`.

9.8.3.6 Meters

In order to graph the results of your output using a graphing program, you need to generate a `.PRINT` line for each value you wish to graph. You can have *Sch2spi* automatically generate these lines by using the SPICE voltmeter and ammeter. Following is a list of the properties and the `.PRINT` lines generated by each.

```
tran:TRACE_NAME      .PRINT TRAN V(n+,n-)
dc:TRACE_NAME        .PRINT DC V(n+,n-)
real:TRACE_NAME      .PRINT AC VR(n+,n-)
imag:TRACE_NAME      .PRINT AC VI(n+,n-)
mag:TRACE_NAME       .PRINT AC VM(n+,n-)
phase:TRACE_NAME     .PRINT AC VP(n+,n-)
db:TRACE_NAME        .PRINT AC VDB(n+,n-)
```

To select a given `.PRINT` type, change the value of the corresponding property from `*opt*` to *`TRACE_NAME`*, where *`TRACE_NAME`* is the label you want to appear on the output graph. You can supply an asterisk for *`TRACE_NAME`* in which case the label will be the default SPICE label. These label names are written to the *`BASENAME`*`.met` file. For example, let's say you have a voltmeter across two nodes `vo1` and `vo2` (hypothetically renamed `10` and `20` by *Sch2spi*) and the properties on the meter are:

```
tran:vout
mag:*
phase:*
```

then *Sch2spi* will generate the following `.PRINT` lines:

```
.PRINT TRAN V(10,20)
.PRINT AC VM(10,20)
.PRINT AC VP(10,20)
```

and the following line will appear in *`BASENAME`*`.met`:

```
TRAN V(vo1,vo2) vout
```

and the trace labels will be `vout`, `VM(vo1,vo2)`, and `VP(vo1,vo2)` respectively.

The output and properties for ammeters are the same with `I` substituted for `V`. *Sch2spi* places a zero voltage source in the circuit in place of the ammeter and then generates a `.PRINT` statement referring to this voltage source. The default label will be `I(VXX)` where `VXX` is the name *Sch2spi* generated for the voltage source.

9.8.3.7 Other SPICE Control Symbols

In addition to the control symbols used for analysis, other control symbols are provided. They are found on the `Part→Meters/Control→SPICE Control Symbols` menu. Following is a brief description of each and the properties they use.

9.8.3.7.1 The `library` Symbol

The `library` symbol, if included in your schematic, generates a `.LIB` statement. The list of libraries is specified with the `value` property which is substituted verbatim.

9.8.3.7.2 The `options` Symbol

The `options` symbol allows you to set a variety of options by appropriately setting the values of the corresponding properties on the symbol.

9.8.3.7.3 The `include` Symbol

If you place the `include` symbol in your schematic a `.INC` statement will be generated. The list of include files is specified with the `value` property which is substituted verbatim.

9.8.3.7.4 The `define` Symbol

The `define` symbol, if included in your schematic, generates a `.PARAM` line. The parameters and values are set by the `value` property which is substituted verbatim.

9.8.3.7.5 The Distribution Card

If you include the distribution card in your netlist and you extract a PSPICE netlist, *Sch2spi* will write a `.DISTRIBUTION` line in your netlist. The parameters are supplied by the `value` property which is substituted verbatim.