CCICAP USER's GUIDE

Bob Smither, Ph.D.

November 6, 2008

November 6, 2008

CCICAP Version 6

Contents

Disclaimer of Warranty

This software and user's guide are provided "as is" without warranty of any kind, either expressed or implied, including but not limited to the implied warranties of merchantability and fitness for a particular purpose.

CCICAP is developed and maintained by:

Bob Smither, Ph.D.

Circuit Concepts, Inc.

2600 Ware Dairy Road

Friendswood, Texas 77546

Office : 218-331-2744

Internet : smither@C-C-I.Com
WWW.C-C-I.Com

1 INTRODUCTION

CCICAP is based on a modified nodal formulation as described by J. Vlach and K. Singhal [1].

CCICAP is a powerful, easy to use, linear circuit analysis program. The program can be used to study a wide range of linear electronic circuit behavior. CCICAP includes passive circuit elements such as resistors, capacitors, and inductors as well as controlled elements such as voltage to current converters and ideal operational amplifiers. There are built-in linear models for bipolar junction transistors, field effect transistors, and operational amplifiers.

Four digital elements are included to allow the analysis of digital filters and other digital systems in the frequency domain.

The user of CCICAP can specify voltmeters and ammeters at any point in the circuit so that circuit operation can be easily monitored. In addition, user specified calculated relationships among inputs, outputs, and element values can be obtained. Output is produced to a printable ASCII file specified by the user, and optionally to binary data files and plot files.

CCICAP provides both frequency and time domain analysis capabilities.

In the frequency analysis mode, CCICAP provides circuit responses at specified frequencies.

In addition, CCICAP can provide frequency dependent response sensitivities to specified network elements. Multi-parameter, worst case, and RMS sensitivities to a user specified component set are available.

Circuit noise from resistors and active elements may also be obtained. The noise analysis can provide the total circuit noise at a requested node or the noise resulting from a selected set of circuit elements at a requested node. Integrated noise over a user specified band width is provided as part of the noise analysis. A Noise Table can be generated which provides a detailed, sorted list of the elements and their noise contributions at requested frequencies.

A circuit can be optimized in the frequency domain, wherein circuit element values will be searched for that result in a user specified response.

As part of a frequency analysis the user may request that selected circuit element values be varied and the corresponding circuit responses be calculated.

In the time domain analysis mode, CCICAP provides either impulse or step responses.

CCICAP uses an easy to understand and use input syntax. The circuit nodes are assigned unique names by the user. For example, an output node could be named 'Eout'. This capability results in easy to read circuit descriptions. The only pre-assigned node name is that for the ground or reference node. Ground may be called '0', 'GND', 'Gnd', or 'gnd'. The circuit elements are also assigned unique names by the user, adding to the self-documenting nature of the input circuit description. Comments can be added on the element and control records and as separate comment records.

Several convenience features are provided by CCICAP. The units used for the AC analysis results can be set as real and imaginary, dB and phase, or magnitude and phase. The frequency and phase units for an AC analysis can be specified as either Hertz and degrees or radians per second and radians. Standard passive element decades can be specified and the specified element values will be adjusted to the closest standard decade values before analysis. Frequency and impedance scaling is easily done. Parameters can be specified as constants or calculated based on other parameters and circuit element values. Calculated response relationships can be specified using defined inputs, defined outputs, circuit element values, parameters, and other calculated responses as variables. Labeled, automatically scaled, line printer and high resolution graphics response plots can be obtained. Page length can be set by the user. Information about the system matrices can be obtained.

CCICAP can be run from an input file in batch mode or interactively from the keyboard.

2 STARTING CCICAP

CCICAP is invoked by the following command line:

$ccicap inputfile outputfile

where 'inputfile' is the name of an existing circuit description file. 'inputfile' may include path information if the file does not exist in the same directory from which CCICAP is initiated. 'outputfile' is the name given by CCICAP to the ASCII output (results) file. 'outputfile' may include path information. If no path is given, 'outputfile' is created in the directory from which CCICAP is initiated. 'inputfile' and 'outputfile' must not be the same file name if they are to exist in the same directory.

Examples:

$ ccicap my_ckt.inp my_ckt.out

$ ccicap /home/smither/ccicap/my_ckt.inp /home/smither/ccicap/results/my_ckt.out

3 PROGRAM REQUIREMENTS, LIMITS

Hardware Requirements: IBM 386 or higher based PC

Operating System : Linux-ELF.

Co-Processor support : Provided by Linux

Program Item	Limit
Circuit Equations*	≤ 100
Circuit Elements	≤ 300
Models **	≤ 10
Parameters / model	≤ 15
Requested outputs	≤ 30
Requested CALCulations / run	≤ 10
Requested VARY elements / run	≤ 5
Requested PLOTs / run	≤ 7
Variable elements in SYMBOL analysis	≤ 9
Points / output / PLOT	≤ 600
Input sources	≤ 10
Requested SENSitivities	≤ 60
Specified initial conditions	≤ 30
Specified noise source elements	≤ 60
Specified parameters / run ***	≤ 20
Real values / record	≤ 60
Integer values / record	≤ 60
Text values / record	≤ 60
Values in resistor standard decade	≤ 192
Values in capacitor standard decade	≤ 24
Values in inductor standard decade	≤ 24
Characters in 'filename' on .INC records	≤ 60

If the number of nodes, elements, models, calculated outputs, outputs, or input sources is exceeded, an error warning is issued to the ASCII output file and processing continues with the NEXT circuit description (if

present - see description of the .NEXT record). If the number of plots, VARY elements, sensitivities, initial conditions, or noise source elements is exceeded, the excess requests are ignored, a warning is issued to the ASCII output file, and processing of the current circuit continues.

* A circuit equation is generated for each node defined in the circuit description (excluding the ground node). Additional equations are generated for certain element types as follows:

Element Type	Additional Equations
CCC	1
VVC	1
V	1
IOA	1
L	1
CVC	2
AM	1
BJT	1
FET	1
OA	2
K	2

** An individual model may be used multiple times within a circuit description. The limit is on the number of distinct models which may be defined within one analysis.

*** 2 parameters are predefined, 18 are available for definition by the user.

4 INPUT FILE DESCRIPTION

CCICAP is controlled from an input file. This file is a standard ASCII formatted file which contains the circuit description, analysis requests, and model definitions. The file can be generated using an editor which produces ASCII files. Most word processors can be used in a straight ASCII mode. The following definitions apply to the input file records:

Any record which starts with a '*' or a single apostrophe (') in column one is ignored by CCICAP. Any information on a record beyond a '*' or a single apostrophe (') is ignored by CCICAP. This is useful for adding comments to the input circuit description file.

Definitions:

```
Integer :  number, up to four digits, no embedded ' ', ',', or '.'  allowed in the field.

Text :  up to eight characters, no embedded ' ', ',', or '.'  allowed in the field.

Real :  number, must have a '.'  in the field.  No embedded ' ' or ',' in the field.  Real
numbers can also be defined as the result of a calculation between '{' and '}'.

I1, I2, ...  :  integer parameters on input records.

T1, T2, ...  :  text parameters on input records.

F1, F2, ...  :  real parameters on input records.
```

4.1 Postfix operators for reals

Real parameters may optionally be modified with a postfix operator. Postfix operators are used to specify powers of ten.

The following postfix operators are supported:

Operator	Scale Factor
f, F	$10.^{-15}$
p, P	$10.^{-12}$
n, N	$10.^{-9}$
u, U	$10.^{-6}$
l, L, m	$10.^{-3}$
k, K	$10.^{+3}$
M	$10.^{+6}$
g, G	$10.^{+9}$
t, T	$10.^{+12}$

The following entries in an F parameter field all represent the same real number:

`1234000000000000.0f, 1234000000000000.0F, 1234000000000.0p,`

`1234000000000.0P, 1234000000.0n, 1234000000.0N, 1234000.0u,`

`1234000.0U, 1234.0l, 1234.0L, 1234.0m, 1.234, 0.001234k, 0.001234K,`

`0.000001234M, 0.000000001234g, 0.000000001234G, .000000000001234t,`

`.000000000001234T`

Any text following the postfix operator, up to the next delimiter, is ignored. In addition, the percent sign may be used as a postfix operator in some types of records to indicate a percentage deviation from a nominal value. See for example the description of the VARY record.

Input records can be up to 78 characters in length.

The circuit analysis is controlled by several run control records. The available run control records and their usage are documented on the following pages. Run control records start with a period ('.') in column one.

CCICAP is sensitive to alphabetic case. The element names and node names may be in mixed case. Note that the following two element records describe two distinct resistors between two distinct node pairs:

```
R, r1, es, eo, 1.e3 * Resistor r1 between nodes es and eo.
R, R1, eS, Eo, 10.e4 * Resistor R1 between nodes eS and Eo.
```

See element entry descriptions under CKT record below.

4.2 .TITLE Record

The title is printed at the top of each page of the output file and on each plot.

Example:

```
.TITLE Analysis of FDNR Based Active 8'th Order Filter
```

4.3 .INC Record

A CCICAP input file can refer to other files by use of the INClude record. The structure of the INClude record is:

```
.INC filename
```

where 'filename' is the name of an existing file which is to be included in the input file. The filename is not quoted. The filename may include path information. The records of the named file are logically added in place of the INC record. This feature is useful for adding often used models, control records, standard decades, circuit alteration records, etc. The items may be described in files by themselves and referred to by any number of different input files.

INCluded files may contain .INC records referring to additional INClude files as long as the resulting input file is correct. After an included file is processed control returns to the calling file at the record following the .INC record. The depth of nesting of INClude files is limited by the number of files which may be open at any one time.

Examples:

```
* include contents of file '/home/models/op21.mdl' here.

.INC /home/models/op21.mdl

.INClude lpf.ckt * include contents of file 'lpf.ckt'.

* include contents of file op270.mdl located in the parent directory.

.INC ../op270.mdl
```

4.4 .PAGE Record

The PAGE record enables automatic pagination at a specified number of lines for the ASCII output file. If the PAGE record is not included, form feeds will be generated at the beginning of each section in the ASCII output file but will not be generated within a section. For example, the AC analysis results will start at the top of a page but the results may extend beyond the end of this page with no form feeds.

If the PAGE record is included, form feeds will be generated at the beginning of each section in the ASCII output file and will additionally be generated to prevent more than a specified number of lines to be output to any one page.

Examples:

```
.PAGE * lines / page is unlimited (same as not

* including the .PAGE record)

.PAGE 60 * limits lines / page to ≤60.
```

4.5 .KB Record

The KB (keyboard) record redirects input from the input file to the keyboard. After the KB (or KB ON) record, all input to be processed comes from the keyboard until another KB (or KB OFF) record is processed.

Example:

```
...

.CKT * beginning of circuit description

...
```

```
.END * end of circuit description.
.GO * specified analysis is done here.
.KB * switch input to keyboard.  New plots, title, and / or
* element alter records can be input and executed.
* After a .KB record is input from the keyboard,
* the next line of the input file is processed:
.NEXT
...
```

4.6 .ECHO Record

The ECHO record provides operator feedback. When processed, the text on the ECHO record is displayed to the CRT.

Example:

```
...
.ALTEL r1 1.3k
* report change to operator
.ECHO Value for R1 changed to 1.3K.
.GO
...
```

4.7 .OPTions Record

The .OPTion record is used to set certain operational aspects of CCICAP (see below) and to request information about the equation formulation used by CCICAP. The .OPTion record is parsed from left to right so that in the event of conflicting requests, the last request will dominate. Options may be placed in any order on the .OPTions record.

The following options and their actions are supported:

- `TAB`: Causes CCICAP to print out the variable headings for .AC, .DC, .VARY, and .TRAN analyses along with the generated tabular data to the ASCII output file. This is the default mode.
- `NTAB`: Causes CCICAP to print out the variable headings for .AC, .DC, .VARY, and .TRANsient analyses but to not print out the tabular data in the ASCII output file. This option might be used to suppress tabular data and reduce the size of the output file when high resolution plots are going to be produced and used as the primary output. It can also be used when the output is directed to the screen (using 'con' as the output file) and the user doesn't want to see the tabular data scroll by.
- `HUSH`: Suppresses informative messages to the console during the run.
- `NHUSH`: Enables informative messages to the console during the run. This is the default mode.
- `FILE`: Activates automatic creation of binary output data files. A `.PLOT` request will also activate the creation of binary data files. The data files will be created in the default directory and will have names given by

  - DATAmmnn.AC and/or DATAmmnn.TR and/or VARYmmnn.AC
  - where mm refers to the current circuit description and nn refers to the alteration number for the current circuit. The first analysis of a run generates files named DATA0000 and/or VARY0000. The first alteration of this circuit (see ALTEL and ALTGO records below) generates files named DATA0001 and/or VARY0001. The second circuit, defined after a .NEXT record, generates files named DATA01nn and/or VARY01nn. The data calculated during the AC analysis will be placed in DATAmmnn.AC and/or VARYmmnn.AC and the data produced during the TRANsient analysis will be placed in DATAmmnn.TR.
  - If a .FILE option or a .PLOT record is included in an input file, the title is saved to a file named as
  - TITLmmnn.TXT
  - where the mm and nn fields are incremented as described for the DATAmmnn files.
  - If no FILE option or .PLOT record is included in a circuit description, the binary data files are not produced. Data files from the individual analyses within a multiple analysis run can be controlled by the use of the FILE option. Binary data files are always produced if one or more .PLOT requests are made, as they are used after the analysis to produce the plots. The binary output data files can be used by post processors.

- NFILE: turns off the FILE option described above.
- DB: CCICAP will produce output from a requested AC analysis in dB and phase format. This is the default format.
- MAG: CCICAP will produce output from a requested AC analysis in magnitude and phase format.
- R&I: CCICAP will produce output from a requested AC analysis in real and imaginary format.
- HZ: CCICAP will use Hertz and degrees as the units of frequency and phase for AC analyses. These are the default units.
- RPS: CCICAP will use radians per second and radians as the units of frequency and phase for AC analyses.
- STEP: CCICAP will generate the step response for any requested TRansient analyses. This is the default.
- IMP: CCICAP will generate the impulse response for any requested TRansient analyses.
- SIZE: CCICAP will print out the following information about the circuit and the circuit description matrix:
  - #outs: number of requested outputs.
  - #ins: number of specified inputs.
  - #elem: number of elements in the circuit description.
  - nodes: number of user specified nodes in the circuit (does not include the ground node).
  - szmat: size of the circuit description matrix (number of equations).
  - order: order of the circuit.
- NSIZE: turns off the SIZE option described above.
- MAT: CCICAP will print out the circuit description matrices and the source vector. This option can produce copious amounts of output for large circuits and should be used accordingly.
- NMAT: turns off the MAT option described above.

- `TXT`: CCICAP will generate an ASCII data file for any requested analyses. The data files will be created in the default directory and will have names given by:
  - `TEXTmmnn.AC` and/or `TEXTmmnn.TR` and / or `TEXTmmnn.NT`
  - Where the `mm` and `nn` fields are as described under the `FILE` option. The file format is as follows:
  - `Independent variable  1 [SP] data1 [SP] data2 [SP] ...[SP] dataN2.`
  - `Independent variable  2 [SP] data1 [SP] data2 [SP] ...[SP] dataN2.`
  - `...`
  - `Independent variable N1 [SP] data1 [SP] data2 [SP] ...[SP] dataN2.`
  - where Independent variable i is the i'th value of the independent variable (frequency or time), data1 through dataN2 are the corresponding values of the dependent variables, and [SP] is the user specified separation character (see description of OPTions entry SPnn below).
  - File `TEXTmmnn.AC` will contain the results of the AC analysis (if requested), file `TEXTmmnn.TR` will contain the results of the TRansient analysis (if requested), and file `TEXTmmnn.NT` will contain the unsorted results of the Noise Table analysis (if requested).
- `NTXT`: turns off the `TXT` option described above.
- `SPnn`: used to establish a separator character for use in the raw ASCII text files described in the TXT option above. The default separator character is a space. The field `nn` represents the hexadecimal value of an ASCII character. Recommended separators include:

`nn`	Separator
`09`	Tab
`20`	Space ' '
`2C`	Comma ','
`3A`	Colon ':'
`3B`	Semi-colon ';'

- `STDIO`: Causes CCICAP to report all numerical results to standard output. Normal informational messages to standard output are suppressed.
- `NSTDIO`: turns off the `STDIO` mode described above.

Options Record Examples:

```
.OPT SIZE * request information about formulation.
.OPT SIZE MAT * request above and the matrices.
.OPT FILE * create binary data files.
.OPT NTAB * suppress all tabular data printout.
.OPT SIZE MAT NTAB PLTP * all of the above.
.OPT MAG RPS * use magnitude and phase, radians per second and radians,
* for AC analysis units.
.OPT IMP MAG * impulse for TRansient response, magnitude for AC response.
.OPT TXT SP3A * produce raw ascii data files, use ';' as the field separator.
```

4.8 Circuit Optimization Control Records

CCICAP can perform circuit optimization in the frequency domain. A circuit optimization attempts to reduce the error between a *desired* circuit response and the *actual* circuit response by adjusting specified circuit element values.

The optimization used in CCICAP is based on a genetic algorithm whereby a 'generation' of sets of circuit element values (each member of the set is called an 'individual' and represents a circuit with specific element values) is derived from a previous 'generation' of individual sets of circuit element values. The 'genetic makeup' of each individual is simply the set of component values corresponding to that individual. Associated with each individual is an error defined as the difference between that individual's circuit response and the desired circuit response.

As each new generation is created, the individuals in the current generation are 'bred' in order to produce new individuals. In CCICAP, a user specified subset of the current generation consisting of the individuals with the lowest associated errors are used to create the next full generation. The individual with the lowest error is always carried unchanged into the next generation. New members of the next generation are obtained by randomly mixing the 'genetic makeup' (circuit element values) from randomly selected individuals in the current generation. The 'breeders' are selected so that individuals with lower associated errors are selected more often than individuals with higher errors.

Each generation consists of a user specified population size. As each generation is created, there is a user specified probability of mutations. A mutation is a random change in the value of a randomly selected circuit element in a randomly selected individual.

4.8.1 .OPP Record

The OPP record is one of several records that are used to request and control a circuit optimization. CCICAP uses a genetic optimization algorithm. The OPP record contains 6 parameters that control the routine:

```
.OPP F1 F2 F3 F4 I1 I2 I3 I4
```

where

`F1` : sets the maximum allowed normalized component change per generation. Restricted to 0.0 (no change allowed, not very interesting) to 1.0 (allowing up to a 100% change in a component value per generation). Typical : 0.1 - 0.3. This parameter is used to generate the first, randomly seeded, generation and is used to set limits on the changes allowed by mutation during the creation of subsequent generations.

`F2` : stop condition for the optimization. When the calculated error becomes equal to or less than this number the optimization will stop. The units are those specified for the current analysis.

The error measure is the weighted (see the `.OPW` record) RMS average of the errors at each specified frequency.

`F3` : sets the probability of a crossover as individual circuits are combined. Restricted to 0.0 to 1.0. Typical : 0.7 - 1.0.

`F4` : sets the mutation rate of the population. Restricted to 0.0 to 1.0. Typical : 0.01 - 0.3. The mutation rate is applied to each new generated circuit element value. For example if each individual has 10 associated element values and the mutation rate is 0.1 then on average there would be one mutated value for each individual in the next generation.

`I1` : limits the number of generations attempted. After `I1` generations the optimization will end regardless of the error.

`I2` : population size. At each generation, `I2` - 1 new individuals are created using `I3` breeders. Note that the best individual from each generation is carried unchanged into the next generation.

`I3` : number of breeders. As each generation is created the `I2` - 1 new individuals are obtained from the `I3` best individuals from the current generation. Note that `I3` must be less than or equal to `I2`.

I4 : random number generator seed. Set to any integer. If not set, 0, which is as good as any other value, is used as the seed. Note that the same sequence of random uniform deviates will be generated for any specific seed. To obtain a different sequence, the seed should be changed.

Example

```
; allow 20% max changes, stop when average error is .01,
; apply crossover to 70% of new generation,
; allow 5% mutation rate, run for 500 generations max,
; each generation has 30 individuals derived from the best
; 10 individuals from the previous generation.
.OPP 0.2 0.01 0.7 0.05 500 30 10
; if response is in dB, then this stops if average error
; is reduced to 0.01 dB or 500 generations are tried.
```

4.8.2 .OPM Record

The .OPM record sets the optimization mode. The mode is specified as:

```
.OPM T1 T2
```

where T1 is one of:

T1 = AMPlitude - optimize the amplitude response; interpret the numbers on the .OPR record as amplitude values.

T1 = PHAse - optimize the phase response; interpret the numbers on the .OPR record as phase values.

T1 = DELAY - optimize the delay response; interpret the numbers on the .OPR record as delay values.

If T2 is not specified, the delay values are assumed to be relative delays.

If T2 is specified, it is used in conjunction with the DELAY mode and can be one of:

T2 = ABSolute - interpret the numbers on the .OPR record as absolute delay values.

T2 = RELative - interpret the numbers on the .OPR record as relative delay values. When T2 is specified as RELative, the values on the .opr record(s) are interpreted as relative to a weighted average of the delay. For example, to obtain a flat delay value, use the relative delay mode and specify the desired responses as all zeros.

T2 is ignored for the AMP and PHA modes.

The optimization mode defaults to AMP.

Examples:

```
; optimize the phase response to values specified on .opr records
.opm phase
; optimize the delay response to a flat delay
.opr eout 1.0 0.0 0.0 0.0 0.0
.opm delay relative
```

4.8.3 .OPF Record

This record is used with the .OPR record below. The .OPF record specifies the frequencies at which the responses specified in the .OPR record are to occur. The error is evaluated at the frequencies on the .OPF record. The frequency unit is that specified for the current analysis.

4.8.4 .OPR Record

This record is used with the .OPF and .OPM records above. The .OPR record specifies the circuit output to be optimized, the weight given to errors at the specified output, and the desired circuit responses at the specified output at the frequencies specified in the .OPF record. The response unit is that specified for the current analysis.

There can be multiple .OPR records specifying multiple outputs to be optimized.

Example:

```
; establish desired responses at four frequencies,
; 1.0, 10.0, 100.0, and 1000.0 ...
.OPF 1.0 10.0 100.0 1000.0
; the four desired responses at eout
; (which must be a specified output), with a relative
; weight of 2.5, are 0.0, 3.16, ...
.OPR eout 2.5 0.0 3.16 10.0 31.6
; and the four desired responses at i33
; (which must be a specified output),
; with a relative weight of 0.5, are 10.0 3.16, ...
.OPR i33 0.5 10.  3.16 1.0 1.0
```

4.8.5 .OPW Record

This record establishes weights for the responses set by the .OPF and .OPR records. The weights default to 1.0 for each response. If the response at certain frequencies is of more or less importance an appropriate set of weights can be specified.

Example

```
; the third specified response is the most important ...
.OPW 1.0 1.0 10.0 1.0
```

4.8.6 .OPE Record

This record specifies the circuit elements that are to be allowed to change during the optimization. The entries on this record are the text names of circuit elements specified in the .CKT - .END records:

```
.OPE T1 T2 ...
```

Example

```
; let r1, r20, and c2 be optimized:
.OPE r1 c2 r20
```

4.8.7 .OPLIM Record

This record is used to set low and high limits for the values of the optimization circuit elements specified on the `.OPE` record. If a restricted range of allowed values is desired for a particular element it is entered on this record.

```
.OPLIM T1 F1 F2 T2 F3 F4 ...
```

where

`T1` = circuit element name whose value is to be limited

`F1` = lowest allowed value for element `T1`

`F2` = highest allowed value for element `T1`

`T2` = next element to be limited ...

Each limit set consists of the element name along with the low and high limit values.

After each new generation is created in the optimization algorithm, and before the responses of the new individuals are evaluated, the limits are imposed.

Example

```
; limit the range of c2 and r1 as shown:

.OPLIM c2 .001u 1.0u r1 1.k 100.k
```

The circuit elements named on the OPLIM Record must also be named on the OPE Record.

4.8.8 .OPC Record

When there are relationships (constraints) that must be maintained between pairs of element values they can be specified with the `.OPC` Record. The constaints are specified as:

```
.OPC T1 T2 T3 F1, T4 T5 T6 F2 ...
```

where

`T1` = reference circuit element name

`T2` = constraint operation

`T3` = constrained circuit element name

`F1` = contraint value

`T4` = next reference circuit element name ...

The reference element (`T1`, `T4`, ...) must be a circuit element that is specified on the `.OPE` record and whose value is allowed to change during the optimization. The constraint will not change the value of the reference element. The constrained element (`T3`, `T6`, ...) can be any circuit element and is the element whose value is changed by the constraint. Note that it is computationally inefficient to specify the constrained circuit element on the `.OPE` record since it's value will be determined by the constraint and any optimized value will be discarded.

For the first constraint set `T1` and `T3` are the names of the two circuit elements involved. `T2` is an operation involving the two circuit element values. `F1` is the value of the constraint operation. `T2` can be any one of the following operations:

`+` addition

- subtraction

X multiplication

/ division.

Each constraint set consists of the two element names along with the constraint operation and the constraint value. The corresponding constraint equations are

```
T1 T2 T3 = F1
```

```
T4 T5 T6 = F2
```

...

After each new generation is created in the optimization algorithm, and before the responses of the new individuals are evaluated, the contraints are imposed.

Example

```
' constrain r2 to be equal in value to r1
```

```
.OPC r1 - r2 0.0
```

The corresponding constraint equation is

```
r1 - r2 = 0.0
```

or

```
r2 = r1.
```

For this example `r1` must be named on the `.OPE` Record. The *constraint* does not alter the value of `r1`.

Example

```
' constrain c1 to be the reciprocal of c3
```

```
' and constrain r5 to be twice the value of r66
```

```
.OPC c3 X c1 1.0 r66 / r5 0.5
```

The corresponding constraint equations are

```
c3 * c1 = 1.0
```

or

```
c1 = 1.0 / c3
```

and

```
r66 / r5 = 0.5
```

or

```
r5 = 2.0 * r66
```

For this example `c3` and `r66` must be named on the `.OPE` record. The *constraints* do not alter the values of `c3` or `r66`.

4.9 .AC Record

Requests a small signal AC analysis for the circuit. The '.' before AC must be in the first column of the record. The `.AC` record contains four parameters which control the AC analysis:

```
.AC I1, I2, F1, F2
```

where

`I1`	Number of frequency intervals to be calculated. If frequency spacing is not linear (see I2) then I1 is the number of intervals per frequency factor. Defaults to zero intervals (one value of frequency).
`I2`	Frequency factor. I2 = 0 or 1 results in linear spacing of the calculated frequency intervals between F1 and F2. I2 = 2 results in I1 intervals per octave. I2 = 10 results in I1 intervals per decade. I2 $\geq$ 0. Defaults to linear spacing.
`F1`	Starting frequency ($\geq$0).
`F2`	Ending frequency ($>$ F1).

The frequency and phase units default to Hertz and degrees. This selection can be changed to radians per second and radians on the `.OPT` record.

The results format of an AC analysis defaults to dB and phase. This format can be changed to either real and imaginary or magnitude and phase on the `.OPT` record.

Examples:

```
.AC 100 10.e3 20.e3 * 100 intervals from 10KHz to 20KHZ (or 10Krps to 20Krps)

.AC 50 10 1.0 100.0 * 50 pts/decade from 1.  to * 100.Hz or rps.

.AC 5, 2 10.0, 40.0 * 5 pts/octave from 10 to 40 Hz or rps.

.AC 25 10 6.  6.k * 25 pts/decade from 6 to 6K Hz or rps.

; use of parameters

; sweep from fmid / 3 to fmid * 3

.par fmid 100.k

.ac 100 10 { fmid 3.  / } { fmid 3.  X }

;

.AC 100.  * zero intervals (one point) at 100.  Hz or rps.

.AC CLR * clears existing AC record.  Use with * .ALTEL/.ALTGO control.
```

4.10 .DC Record

Requests a DC analysis of the network. The DC analysis is actually an AC analysis with the analysis frequency set to zero. Note that the analysis will be invalid if any element has infinite conductance at DC (such as an inductor). Note also that a NOISE request will be invalid if any circuit element is specified as having 1/f noise. There are no parameters on the DC record.

Examples:

```
.DC
```

requests a DC analysis of the current circuit. A DC analysis request is equivalent to the following:

```
.AC * zero intervals, zero frequency.
```

4.11 .VARY Record

The VARY record is used to specify a circuit element whose value is to be varied over a specified range of values. The format of the VARY record is as follows:

```
.VARY, T1, T2, I1, F1, F2, F3
```

where

`T1`	'AC' to indicate a variation during the AC analysis.
`T2`	the name of the circuit element whose value is to be varied.
`I1`	the number of intervals in the variation.
`F1`	the starting value of the variation.
`F2`	the ending value of the variation.
`F3`	the frequency at which to perform the variation.

During an AC analysis, the circuit will first be analyzed with the nominal element values specified in the `.CKT` to `.END` section. Following the nominal values analysis, the circuit will be analyzed at the specified frequency `F3` for each of the `I1` + 1 values specified on the `.VARY` record. The nominal value of the circuit element named by `T2` is restored at the end of each element variation.

The entire analysis is repeated for each value specified on the `.VARY` record. This includes any requested responses, sensitivities, noises, and calculations. Integrated noise is not calculated during a variation analysis. The value of integrated noise is set to zero in any variation results.

`T1` = 'CLR' will clear any existing `.VARY` requests. This can be useful in combination with the `.ALTGO` record so that revised `.VARY` requests can be implemented.

Examples:

```
.VARY AC r34 10 .0001 100.
```

causes the value of `r34` to be varied between .0001 and 100. ohms over 10 intervals and for the analysis frequency to be set to 0.0 Hz. Note that specifying a zero value for a resistor will cause an error. The value of `r34` is restored to whatever value is specified in the `.CKT` to `.END` section after the variation analysis is complete.

```
.VARY AC Vin 100 -10.% +10.% 10.K
```

causes the value of circuit element `Vin` to be varied from 10% below its nominal value to 10% above its nominal value in 100 intervals. The frequency is set to 10.0e3. The nominal value of `Vin` is the value specified in the `.CKT` to `.END` section.

```
.VARY AC c44 9.44ufd 9.44ufd 10.0KHz
```

causes the circuit to be analyzed with the value of `c44` set to 9.44 ufd (ie, zero intervals is the default) at a frequency of 10.0 KHz. This may be easier in some situations than using the `.ALTEL` / `.ALTGO` syntax. The value of `c44` is restored to whatever value is specified in the `.CKT` to `.END` section after the variation analysis is complete.

```
.VARY CLR
```

Clears all previous VARY requests and allows new requests to be stored. This can be used with a `.ALTGO` record as follows :

...

```
.END * end of CKT section.
.go * do requested analyses, including variations
.VARY CLR
.VARY ...  (set new requests)
...
.ALTGO * starts new analysis ...
```

4.12 .TR Record

This record requests a time domain or TRansient analysis of the circuit. The '.' before TR must be in the first column of the record. The `.TR` record contains three parameters which control the analysis:

```
.TR I1, F1, F2
```

where

`I1`	integer number of time steps to be calculated.
`F1`	real, the analysis starting time.
`F2`	real, the analysis end time.

The transient analysis always starts at time = 0.0 with the first response calculated for time = (`F2` - `F1`) / `I1`. Only the data generated after time $\geq$`F1` is written to output file(s).

The number of time intervals refers to the number of intervals calculated between the start and end times. The calculation time step will be (`F2` - `F1`) / `I1`. This time step will be used before `F1` if `F1` $>$ 0.0.

Examples:

```
.TRAN 100 0.0 1.0 * 100 steps from 0 to 1 second.
.TRAN 500 5.e-9 20.nsec * 500 intervals from 5 to 20 nsec.
```

The TRansient analysis will produce the step response for the specified circuit unless an impulse response is requested on the `.OPT` record.

4.13 .IC Record

This record is used to establish initial conditions for the transient analysis. The '.' before IC must be in the first column of the record. The IC record has the following structure:

```
.IC name1 ic1 name2 ic2 ...
```

where namei is the name of the element whose initial condition follows and ici is the value of the initial condition for the element named by namei. Only C and L elements may have initial conditions. If the named element is a capacitor, the ic is a voltage. If the named element is an inductor, the ic is a current. The ic values must be real numbers with an embedded '.'.

4.14 .SYMBOL Record

The SYMBOL record requests a symbolic analysis of the current circuit. The SYMBOL record can be used in one of two forms:

```
.SYMBOL
```

requests a symbolic analysis based on the values of the circuit elements. The resulting transfer function is in terms of powers of the Laplace variable S and constant coefficients.

```
.SYMBOL T1 T2 ...
```

requests a symbolic analysis wherein those elements named on the SYMBOL record (T1, T2, ...) are referred to by name. All other circuit elements are included by using their values. The resulting transfer function is in terms of powers of the Laplace variable S and coefficients that are functions of the named variables (T1, T2, ...).

The output of the symbolic analysis is the S domain transfer function between the first specified input source and the first specified output.

Note that symbolic analysis operates best on normalized networks. Any coefficient found whose magnitude is less than 1.e-4 is discarded. Consider the following example:

```
...

.symbol

.ckt v ein 1 0 1.0

r r1 1 2 1.k

c c1 2 gnd 1.u

vm eout 2 gnd

.end

.go

...
```

The transfer function for this RC low pass filter is:

$$T(S) = \frac{1/r1}{[S*c1+(1/r1)]},$$

which, if no scaling is applied, becomes:

$$T(S) = \frac{1.e{-}3}{[S*1.e{-}6+1.e{-}3]}.$$

Since the magnitude of the coefficient of S in the denominator is less than 1.e-4, it will be discarded and the reported (incorrect) transfer function will be:

$$T(S) = \frac{1.e{-}3}{1.e{-}3}.$$

If the example above is frequency scaled by 1.e-3, using an FSCALE record, the capacitor becomes equal to 1.e-3 and the numerical transfer function is reported (correctly) as:

$$T(S) = \frac{1.e{-}3}{[S*1.e{-}3+1.e{-}3]}$$

with a note that a frequency scaling of 1.e-3 had been applied.

For best results when using symbolic analysis, use a combination of frequency and impedance scaling to scale the network to component values near unity. The output file will document the scale factors used to facilitate scaling of the resulting transfer function.

When the second form of the SYMBOL analysis is used the values of the selected circuit elements are not used and are irrelevant.

4.15 .DELAY Record

The DELAY record requests the calculation of group delay during an AC or VARY analysis. There are no parameters for this record. An AC or VARY analysis must be requested for this record to have an effect. If present, the DELAY record causes the group delay, in seconds, of the first selected output with respect to the first defined input to be recorded to both the ASCII output file and to the binary output file or files, if any.

4.16 .MAKECKT Record

The MAKECKT record is used to make a new circuit description file from the currently defined circuit. The new filename is specified on the MAKECKT record as follows:

```
.MAKECKT filename
```

The new circuit description file begins with a .CKT record and ends with an .END record. The circuit element values in the new description reflect any .STDR, .STDC, .STDL, .FSCALE, .ZSCALE, and .INCluded circuits in the currently defined circuit.

One use for this feature is to easily scale filter circuits. A normalized design can be stored and then impedance and or frequency scaled with the resulting scaled circuit written to a new file. The new file can then be used in a subsequent analysis.

Example:

```
.TITLE Run to scale filter stored as 'normfil.ckt'

.MAKECKT scaled.ckt

' include standard 1% resistor decade values

.INC stdr.1

' include standard 10% capacitor decade values

.INC stdc.10

.ZSCALE 10.k

.FSCALE { PI 2.  X 2.5k X }

.INC normfil.ckt

.go

.STOP
```

Assuming that 'normfil.ckt' contains the description of a filter normalized to a cutoff frequency of 1 radian per second and an impedance level of 1 ohm, the above run would produce a file named 'scaled.ckt' containing the component values for a filter with a cutoff frequency of 2.5 KHz and an impedance level of 10 Kohms. In the above example, the new circuit would use standard 1% resistors (see description of STDR record) and 10% capacitors (see description of STDC record). The file 'scaled.ckt' could be INCluded as the circuit description portion of a subsequent CCICAP analysis.

4.17 .TEMP Record

The TEMP record is used to change the assumed temperature in the circuit. If a TEMP record is not included, all calculations assume a temperature of 298.16 degrees K (25 degrees C). The TEMP record contains one parameter:

```
.temp F1
```

where F1 is the circuit temperature in degrees C.

The temperature is used in calculating the noise from resistors according to the Johnson noise formula:

$En = \sqrt{4.0 * k * T * B * R}$

where k is Boltzman's constant (1.38e-23), T is the resistor temperature in degrees K, B is the noise bandwidth in Hertz, and R is the resistance in ohms.

The temperature is not used in the calculation of noise from operational amplifiers or transistors. The noise spectral densities set in the model descriptions are used at all temperatures.

Examples:

```
.TEMP * sets temperature to 0.0 C.

.TEMP -55.  * sets temperature to -55.0 C.
```

4.18 .MODEL Record

If models are used, the model name and parameters are given on a .MODEL record. Each model to be used in the circuit description must have an associated .MODEL record and must be named and defined before the .CKT section of the input file. The structure of the .MODEL record is :

```
.MODEL name F1, F2, ...
```

where 'name' is the text name given to the model and is referred to from the circuit description and F1, F2, ... are the parameters for the model.

Models are available for op-amps, field effect transistors, and bipolar junction transistors.

.MODEL records may span more than one line by using a multiple line record. Multiple line records are indicated by ending each line with the "\" character. Comments and .INClude records are *not* allowed within a multiple line record. See the examples below.

4.18.1 Op-Amp Parameters:

Parameter	Definition
F1	Differential input resistance $R_{in}(\Omega)$
F2	Common mode input resistance $R_{cm}(\Omega)$
F3	Gain bandwidth product $GBWP(Hz)$
F4	Open loop gain $A_o(V/V)$
F5	Output resistance $R_o(\Omega)$
F6	Second pole frequency $(Hertz)$
F7	Input noise voltage spectral density $e_n(V/\sqrt{Hz})$
F8	input noise current spectral density $i_n(A/\sqrt{Hz})$
F9	Input noise voltage 1/f corner $(Hertz)$
F10	Input noise current 1/f corner $(Hertz)$

The second pole frequency defaults to infinity. The noise sources default to zero amplitude. The 1/f corner frequencies default to zero Hertz. See Appendix B for more information on the op-amp noise sources.

Examples:

```
* lm741 model, Rin = 2M, Rcm = 5G, GBWP = 1MHz,
* Ao = 200k, Ro = 75, no second pole, no noise.
.MODEL 741 2.e6 5.e9 1.e6 2.e5 75.
;
* op21 model, Rin = 10M, Rcm = 10G, GBWP = 600KHz, Ao = 2M
* Ro = 1k, second pole at 500KHz
* en = 20 nV/SQRT(Hz), in = .3 pA/SQRT(Hz)
* 1/f(en) = 8 Hz, 1/f(in) = 30 Hz.
.MODEL op21 10.e6 10.e9 6.e5 2.e6 \
     1.e3 5.e5 \
     2.e-8 .3e-12 \
     8.  30.
```

The comment (*) records are ignored by CCICAP.

4.18.2 FET parameters:

Parameter	Definition
F1	Drain to source resistance $R_{ds}(\Omega)$
F2	Transconductance $G_m(1/\Omega)$
F3	Gate to source capacitance $C_{gs}(Farads)$
F4	Gate to drain capacitance $C_{gd}(Farads)$
F5	Drain to source capacitance $C_{ds}(Farads)$
F6	Drain bias current $I_d(Amps)$
F7	DC gate current $I_g(Amps)$
F8	Drain shot noise 1/f corner $(Hertz)$
F9	Gate shot noise 1/f corner $(Hertz)$

The noise sources are determined from the Gm, Id, and Ig parameters. See Appendix B for more information on FET noise sources.

Example:

```
* FET with Rds = 10K, Gm = .005, Cgs = 6pfd, Cgd = 2pfd,
* Cds = 1.5pfd, Id = 200uA, Ig = 50 pA,
* 1/f(drain) = 100Hz, 1/f(gate) = 1KHz.
.MODEL FET1 1.e4, .005, 6.e-12, 2.e-12, 1.5e-12, 2.e-4 \
     5.e-11, 100., 1.e3
```

4.18.3 BJT parameters:

Parameter	Definition
F1	Base spreading resistance $R_{bb'}(\Omega)$
F2	Active base to emitter resistance $R_{b'e}(\Omega)$
F3	Collector to active base resistance $R_{cb'}(\Omega)$
F4	Collector to emitter resistance $R_{ce}(\Omega)$
F5	Transconductance $G_m(1/\Omega)$
F6	Active base to emitter capacitance $C_{b'e}(Farads)$
F7	Collector to active base capacitance $C_{cb'}(Farads)$
F8	Collector to emitter capacitance $C_{ce}(Farads)$
F9	Collector bias current $I_c(Amps)$
F10	Base bias current $I_b(Amps)$
F11	Collector shot noise 1/f corner $(Hertz)$
F12	Base shot noise 1/f corner $(Hertz)$

The noise sources are determined from the Rb'e, Ic, and Ib parameters. See Appendix B for more information on BJT noise sources.

Example:

```
*2N2222 with Rbb' = 60, Rb'e = 2.58K, Rcb' = 5.16M
* Rce = 333K, Gm = .0388, Cb'e = 30pfd
* Ccb' = 2.5pfd, Cce = 8pfd, Ic, Ib defaulted
* 1/f(collector) = 200Hz, 1/f(base) = 100Hz.
.MODEL 2N2222 60., 2.58e3, 5.16e6 \
     3.33e5, .0388, 30.e-12 \
     2.5e-12, 8.e-12, 0., 0.  \
     200., 100.
```

The parameters Gm, Rb'e, Ic and Ib are used as follows:

If Gm and Rb'e are specified and Ic and Ib are set to 0.0 then Ic and Ib are estimated by CCICAP as:

`Ic = Vt * Gm Ib = Vt / Rb'e`

where `Vt = kT/q`. The estimated values are used in any subsequent noise calculations.

If Ic and Ib are specified as positive and Gm and Rb'e are set to 0.0 then Gm and Rb'e are estimated by CCICAP as:

`Gm = Ic / Vt Rb'e = Ic / (Ib * Gm).`

If Ic and Ib are specified as positive and Gm and Rb'e are specified then Gm and Rb'e are not changed and the specified Ic and Ib are used in the noise calculation for the device.

If Ic and Ib are specified as negative, their values are set to zero and there will be no noise contribution from them. Gm and Rb'e are not changed and must be specified.

4.19 .PAR Record

The .PARameter record is used to define constants for use in subsequent .PARameter, .CALCulation, or circuit element value records. The .PARameter record can take one of two forms:

```
.PAR name F1
```

or

```
.PAR name { expression }
```

where 'name' is the name given to the specified parameter, F1 is a real number, and 'expression' is a calculated expression in Reverse Polish Notation (RPN). A .PAR calculation is specified using Reverse Polish Notation (RPN) operators and a stack data structure. See the section on RPN for details of this syntax and of the CCICAP implementation.

The parameter records within a run are processed as the input file is read in. The expressions used on parameter records can reference *previously* defined parameters.

There are 2 named parameter constants:

Parameter Name	Value
E	2.718281828
PI	3.141592654

Examples:

```
* first form of .par record:

.PAR fo 16.384e6 * set fo in Hertz.

* using built in parameters '2' and 'PI':

.PAR 2pi { PI 2.  X } * defines the value of 2 * pi.

* using predefined parameters 'fo' and '2pi':

.PAR wo { fo 2pi X } * set wo in rps.
```

4.20 .CALC Record

Each .CALC record defines a calculated response. The format for the .CALC record is as follows:

```
.CALC name { calculation }
```

where 'name' is the name given by the user to the calculated response variable. The calculation is specified between a '{' and a '}'. A .CALC calculation is specified using Reverse Polish Notation (RPN) operators and a stack data structure. See the section on RPN for details of this syntax and of the CCICAP implementation.

.CALC calculations are performed during the circuit analysis and are processed in the order that they appear in the input file. .CALC calculations can refer to parameters (see the .PAR Record description); inputs (V or I elements); outputs (VM or AM elements); element values that are defined in the .CKT to .END section; and *previously* defined .CALC variables.

Calculations are performed in the natural units (volts, amps, etc.) and then converted to any requested output format such as dB or magnitude.

If an attempted calculation would result in an illegal operation (such as divide by 0.0) an error message is added to the ASCII output file and the calculated response is set to 0.0.

Examples:

```
.CALC Gm { Eout Iin / }
```

calculates the ratio of the variable 'Eout' to the variable 'Iin' and names the ratio 'Gm'. 'Eout' and 'Iin' must be requested inputs or outputs for the current analysis.

```
.CALC Watt { e45 i53 X }
```

calculates the product of the variables 'e45' and 'i53' and names the result 'Watt'. 'e45' and 'i53' must be requested inputs or outputs for the current analysis.

```
.CALC Wsqr { Watt Watt X }
```

calculates the square of the previously calculated Watt as Wsqr. Watt must be a unique variable name and must be defined as a .CALC variable *before* Wsqr is defined.

```
.CALC Gm { Icq Vt / }
```

calculates Gm from the variables Icq and Vt. If Vt represents the thermal voltage (k*T/q) it can be defined in a PAR record as follows:

```
.PAR Vt .0258
```

The value of Vt must be defined before it is used in the Gm calculation.

```
.CALC pwr1 { vr1 vr1 X r1 / }
```

calculates the power in element `r1` as `vr1 * vr1 / r1`. The requested VM output element named `vr1` would define the voltage across the specified resistor `r1`.

4.21 .STDR Record

This record is used to define an optional standard resistor decade. If present before the .CKT record (see below), the .STDR record will force the use of resistor values taken from a standard decade. Standard decades are defined by resistor manufacturers and indicate what values are available for a particular resistor series. For example, the one percent standard resistor decade has 96 values between 1.0 and 10.0. The one percent standard decade could be defined to CCICAP with the following multiple line record:

```
.STDR 1.00 1.02 1.05 1.07 1.10 1.13 1.15 1.18 1.21 1.24 1.27 1.30 \
1.33 1.37 1.40 1.43 1.47 1.50 1.54 1.58 1.62 1.65 1.69 1.74 \
1.78 1.82 1.87 1.91 1.96 2.00 2.05 2.10 2.15 2.21 2.26 2.32 \
2.37 2.43 2.49 2.55 2.61 2.67 2.74 2.80 2.87 2.94 3.01 3.09 \
3.16 3.24 3.32 3.40 3.48 3.57 3.65 3.74 3.83 3.92 4.02 4.12 \
4.22 4.32 4.42 4.53 4.64 4.75 4.87 4.99 5.11 5.23 5.36 5.49 \
5.62 5.76 5.90 6.04 6.19 6.34 6.49 6.65 6.81 6.98 7.15 7.32 \
7.50 7.68 7.87 8.06 8.25 8.45 8.66 8.87 9.09 9.31 9.53 9.76
```

The values must be $\geq$1.0 and $<$ 10.0 and must be in increasing order as in the example above.

If a .STDR record is processed before the .CKT record, any resistor values specified in the circuit will be changed to the closest standard decade value. For example, a resistor specified as:

```
R r23 nd1 nd2 3.50843e3
```

will be changed to a resistor with value `3.48e3` ohms. The new value, selected from the standard decade, will be shown on the circuit listing in the ASCII output file.

Standard resistor decades are easily added and removed from an analysis by the use of the .INC feature. For example, the following record would add a standard decade to an analysis:

```
.INC stdr.1
```

assuming that the file stdr.1 exists and consists of the .STDR record shown above. To remove the standard decade, the .INC record could be commented out:

```
*.INC stdr.1
```

To remove the effect of the STDR record for selected components, see the description of the .STD record.

There are two ways to specify resistor values that are not modified by a specified standard decade. A resistor can be specified as an equivalent conductance using a G type element (see .CKT section). The values of G elements are not changed by the presence of a standard decade. Additionally, elements whose values are changed via an .ALTEL record are not forced to be from a specified standard decade.

4.22 .STDC Record

This record is used to define an optional standard capacitor decade. If present before the .CKT record (see below), the .STDC record will force the use of capacitor values taken from a standard decade. Standard decades are defined by capacitor manufacturers and indicate what values are available for a particular capacitor series. For example, the ten percent standard capacitor decade has 12 values between 1.0 and 10.0. The ten percent standard capacitor decade could be defined to CCICAP with the following record:

```
.STDC 1.0 1.2 1.5 1.8 2.2 2.7 3.3 3.9 4.7 5.6 6.8 8.2
```

The values must be $\geq$1.0 and $<$ 10.0 and must be in increasing order as in the example above. If needed, multiple line records (see the example .STDR record above) can be used.

If a .STDC record is processed before the .CKT record, any capacitor values specified in the circuit will be changed to the closest standard decade value. For example, a capacitor specified as:

```
C c56 nd3 nd4 1.6582e-6
```

will be changed to a capacitor with value `1.8e-6` farads. The new value, selected from the standard decade, will be shown on the circuit listing in the ASCII output file.

Standard capacitor decades are easily added and removed from an analysis by the use of the .INC feature. For example, the following record would add a standard decade to an analysis:

```
.INC stdc.10
```

assuming that the file stdc.10 existed and consisted of the .STDC record shown above. To remove the standard decade, the .INC record could be commented out:

```
*.INC stdc.10
```

Elements whose values are changed via an .ALTEL record are not forced to be from a specified standard decade. This allows for nonstandard capacitor values for selected capacitors during a subsequent analysis of the circuit.

4.23 .STDL Record

This record is used to define an optional standard inductor decade. If present before the .CKT record (see below), the .STDL record will force the use of inductor values taken from a standard decade. Standard decades are defined by inductor manufacturers and indicate what values are available for a particular inductor series. For example, the ten percent standard inductor decade has 12 values between 1.0 and 10.0. The ten percent standard inductor decade could be defined to CCICAP with the following record:

```
.STDL 1.0 1.2 1.5 1.8 2.2 2.7 3.3 3.9 4.7 5.6 6.8 8.2
```

The values must be $\geq$1.0 and $<$ 10.0 and must be in increasing order as in the example above.

If a .STDL record is processed before the .CKT record, any inductor values specified in the circuit will be changed to the closest standard decade value. For example, an inductor specified as:

```
L 123 nd1 nd2 8.3759e-3
```

will be changed to an inductor with a value of `8.2e-3` Henries. The new value, selected from the standard decade, will be shown on the circuit listing in the ASCII output file.

Standard inductor decades are easily added and removed from an analysis by the use of the .INC feature. For example, the following record would add a standard decade to an analysis:

```
.INC stdl.10
```

assuming that the file stdl.10 exists and consists of the .STDL record shown above. To remove the standard decade, the .INC record could be commented out:

```
*.INC stdl.10
```

Elements whose values are changed via an .ALTEL record are not forced to be from a specified standard decade. This allows for nonstandard inductor values for selected inductors during a subsequent analysis of the circuit.

4.24 .SENSA Record

Requests that absolute sensitivities be calculated for specified circuit elements with respect to the first requested output variable. The '.' before SENSA must be in the first column of the record. The elements for which the absolute sensitivity will be calculated are listed on the .SENSA record. Absolute sensitivity can be calculated for the following element types: G, R, L, C, IOA, VCC, CCC, VVC, CVC, and MUL. The sensitivity reported for R elements will be that for the corresponding G element. The sensitivity to the R value is obtained by multiplying the reported sensitivity value by -1.0.

Sensitivities are calculated as part of an .AC or .VARY analysis. An .AC or .VARY analysis (see .AC ad .VARY record descriptions) must be requested for a .SENSA analysis to be performed. Sensitivities are calculated at the frequencies requested in the .AC or .VARY analysis request.

Absolute sensitivities are defined as:

```
dO(jw)/dh
```

where `O(jw)` is the first requested output variable for the analysis and h is the specified network element.

All sensitivities calculated are presented in real and imaginary form. The sensitivity format is independent of the format requested for the .AC analysis.

Note that only a single .SENSA or a single .SENSR request can be made for any one analysis.

Example:

```
.SENSA c1, r23, g4, c2, rx
```

4.25 .SENSR Record

Requests that relative sensitivities be calculated for specified circuit elements with respect to the first requested output variable. The '.' before SENSR must be in the first column of the record. The elements for which the relative sensitivity will be calculated are listed on the SENSR record. Relative sensitivity can be calculated for any conductor (G type element), resistor (R type element), capacitor (C type element),

inductor (L type element), or multiplier (MUL type element) in the circuit. The sensitivity reported for R elements will be that for the corresponding G element. The sensitivity to the R value is obtained by multiplying the reported sensitivity value by -1.0.

If sensitivities to the other elements listed under the SENSA record description are requested as relative sensitivities, absolute sensitivities will be calculated and reported.

Sensitivities are calculated as part of an AC or VARY analysis. An AC or VARY analysis (see AC ad VARY record descriptions) must be requested for a SENSR analysis to be performed. Sensitivities are calculated at the frequencies requested in the AC or VARY analysis request.

Relative sensitivities are defined as:

```
dO(jw)/dh * (h/O(jw))
```

where O(jw) is the first requested output variable for the analysis and h is the specified network element.

All sensitivities calculated are presented in real and imaginary form. The sensitivity format is independent of the format requested for the AC analysis.

Note that only a single .SENSA or a single .SENSR request can be made for any one analysis.

Example:

```
.SENSR c3, r32, g8, cl, rb
```

4.26 .MSENS Record

Requests a multiparameter sensitivity [4, p. 204] calculation for the circuit elements named on the .SENSA or .SENSR record. The multiparameter sensitivity is defined as the complex sum of the individual complex element sensitivities. A .SENSA or a .SENSR record must also be included in the analysis. If a .SENSA record is included, the multiparameter sensitivity is the sum of the appropriate absolute element sensitivities. If a .SENSR record is included, the multiparameter sensitivity is the sum of the appropriate relative element sensitivities.

4.27 .WSENS Record

Requests a worst case multiparameter sensitivity [4, p. 207] calculation for the circuit elements named on the .SENSA or .SENSR record. The worst case multiparameter sensitivity is defined as the sum of the magnitudes of the individual complex element sensitivities. A .SENSA or a .SENSR record must also be included in the analysis. If a .SENSA record is included, the multiparameter sensitivity is the worst case sum of the appropriate absolute element sensitivities. If a .SENSR record is included, the multiparameter sensitivity is the worst case sum of the appropriate relative element sensitivities.

4.28 .SSENS Record

Requests an RMS (square root of the sum of the squares) multipa rameter sensitivity [4, p. 204] calculation for the circuit elements named on the .SENSA or .SENSR record. The RMS multiparameter sensitivity is defined as the square root of the sum of the squared magnitudes of the individual complex element sensitivities. A .SENSA or a .SENSR record must also be included in the analysis. If a .SENSA record is included, the multiparameter sensitivity is the RMS value of the appropriate absolute element sensitivities. If a .SENSR record is included, the multiparameter sensitivity is the RMS value of the appropriate relative element sensitivities.

4.29 .NOISE Record

The NOISE record requests a frequency dependent noise analysis of the circuit. The noise analysis is performed along with an .AC or .VARY analysis. The noise analysis uses the noise parameters from the OA, BJT, and FET models and generates noise sources for each real resistor and conductance in the circuit. The magnitude of the noise spectral density calculated at the first requested output is provided as the first part of the noise result. The units (either dB[magnitude / $\sqrt{Freq}$] or magnitude / $\sqrt{Freq}$) are specified by the units selection parameters on the .OPT record. The second part of the noise result is the integrated noise as described below. The units for integrated noise are either Vrms or Arms. Note that a zero noise result can only occur as a result of an error. For example, if the noise from an op-amp is requested but the op-amp model does not have specified noise parameters, a zero noise results.

The '.' before NOISE must be in the first column of the record.

4.29.1 NOISE record options

The NOISE record may optionally contain text parameters:

`T1, T2, T3, ...`

where `T1, T2, T3, ...` are element names whose noises are to be included in the calculated output noise. If element names are not specified on the .NOISE record, then all circuit noise sources are included in the calculated output noise. If element names are included on the .NOISE record, then only the noise sources from the named elements are included in the noise calculation.

The .NOISE record may optionally contain two real number parameters:

`F1, F2`

where F1 and F2 define a noise bandwidth. If F1 is less than the starting frequency specified on the .AC record, F1 is reset to the starting frequency. If F2 is greater than the final frequency specified on the AC record, F2 is reset to the final frequency. If F1 and F2 are not specified on the .NOISE record, F1 is set to the starting frequency and F2 is set to the final frequency specified on the .AC record. The total integrated noise at the first requested output that results from the specified circuit elements is calculated and displayed as the second part of the noise result. At each frequency point in an .AC analysis, the integrated noise represents the total noise in a noise bandwidth between F1 and the current frequency point. The final result is duplicated at the termination of the AC analysis. The integrated noise calculation is done using the Trapezoidal rule [3] for numerical integration. It is up to the user to assure that the frequency steps taken are sufficiently small to assure accuracy.

Examples:

```
.NOISE * ask for total circuit noise density at first
* requested output.
* Also provides total noise from all circuit
* elements between F1 and F2 specified on .AC record.
.NOISE r3, amp1 * noise from circuit elements r3 and amp1 only.
.NOISE r44 120.  3000.  * integrated noise between 120.  and
* 3000.  Hz (or rps) from r44.
```

4.30 .N_TABLE Record

The .N_TABLE record is used in conjunction with the .NOISE record. A noise table is useful in identifying which circuit elements are limiting the overall circuit noise performance. If the .N_TABLE record is included, a new text file is created named `NTBLmmnn.AC` that contains a noise table. The values of `'mmnn'` are described under the .OPTion FILE. The noise table lists, at each frequency requested on the .AC record, the name of each circuit element specified on the .NOISE record along with the contribution to the total noise from each requested element. The elements listed are determined by the .NOISE record. The listing is sorted at each frequency so that the elements that contribute most to the noise at that frequency are listed first.

An additional parameter on the .N_TABLE record is used to specify whether SPOT or total (INTEgrated) noise is to be used in the listing and sort.

A third parameter on the .N_TABLE record may be used to limit the size of the noise table file. If no third parameter is specified, or if '1' is entered as the third parameter, noise table data is generated at every frequency specified on the .AC record. An entry of '2' as the third parameter would cause noise table data to be generated at every second frequency, etc.

Examples:

```
.NOISE * noise from all elements.
.N_TABLE SPOT * noise table with an entry from each circuit
* element at each frequency.  The contributions
* to spot noise at each frequency will
* be listed and used in the sort.

.NOISE r1, r2, r3 * only consider r1, r2, and r3 for noise
* calculations.
.N_TABLE INTE * noise table with an entry for only r1,
* r2, and r3.  The total (integrated)
* noise will be listed and used in the sort.

.AC 100 10 1.  10.k * 100 points / decade.
.NOISE
.N_TABLE SPOT 100 * noise table data generated at each
* decade between 1.0 and 10.KHz.
* (5 frequencies)
```

4.31 .FOFFSET Record

The FOFFSET record can be used to offset the frequency that CCICAP reports in its tabular and plotted outputs. The frequency variable used in internal calculations is not affected by the .FOFFSET record. The .FOFFSET record can be used to increase the resolution of printed or plotted results. This feature is useful in studying narrow band circuits such as band pass filters and crystal oscillators where the analysis frequency is over a very narrow range and the variation of interest may not be sufficiently resolved in the standard output format. The .FOFFSET record has a single real parameter:

```
.FOFFSET F1
```

where F1 is a real number. F1 is subtracted from the internal frequency variable and the resulting offset frequency is displayed in tabular and plotted outputs.

4.32 .FSCALE Record

The .FSCALE record can be used to frequency scale an entire circuit. The .FSCALE record has a single parameter, the frequency scale factor:

```
.FSCALE F1
```

where F1 must be a real number. If the .FSCALE record is encountered prior to the .CKT record, the capacitors and inductors in the circuit description will be divided by F1. The frequency scaling is applied before any adjustment to a standard decade is performed (see .STDR and .STDC record descriptions).

Frequency scaling is applied only to elements specified via C and L element records. The scaling is not applied to elements within models.

The analysis frequencies specified on the .AC record are multiplied by the frequency scale factor.

The analysis times specified on the .TRAN record are divided by the frequency scale factor.

4.33 .ZSCALE Record

The .ZSCALE record can be used to impedance scale an entire circuit. The .ZSCALE record has a single real parameter, the impedance scale factor:

```
.ZSCALE F1
```

where F1 must be a real number. If the .ZSCALE record is encountered prior to the .CKT record, the inductor and resistor values in the circuit description will be multiplied by F1 and the capacitor values will be divided by F1. The impedance scaling is applied before any adjustment to a standard decade is performed (see .STDR and .STDC record descriptions). Elements within models are not scaled by F1.

4.34 .PLOT Record

Each .PLOT record produces a plot file of requested outputs using frequency, time, or values of a .VARY element as the independent variable. The format for PLOT records is as follows:

```
.PLOT T1 T2 T3 T4 ...  F1 F2 ...
```

where

T1 : Analysis type code. Plots of the .AC analysis results are requested with T1 = 'AC'. Plots of the TRANsient analysis results are requested with T1 = 'TR'.

T1 = 'CLR' will clear any previous plot requests. This can be useful in combination with the ALTGO feature so that new or revised plots can be requested. If T1 = 'CLR' the remaining fields on the .PLOT record, if any, are ignored.

T2 : Plot format code. Currently supported forms are:

'80' - 80 column ASCII printer plot.

'132' - 132 column ASCII printer plot.

'PNG' - Generates a Portible Network Graphics image of the requested plot. The image file will have the extension '.png'.

'X' - Display the plot in a new X window. An X server must be running to support this option.

'PS' - Create a PostScript file of the plot. The postscript file will have the extension '.ps'.

T3 : This field is used for two different purposes. If the plot is to have either frequency or time as the independent variable then T3 identifies the first dependent variable (see below). If the plot is to use the

values of a variable element (specified on a .VARY record) as the independent variable, then T3 is used to identify the variable element by name and the variable name will be used to label the independent axis of the plot. If T3 is the name of a .VARY element then the values of the .VARY element will be used as the values of the independent variable.

(T3), T4, T5, ... : If T3 names a .VARY element, the plot will use the values of the named element as the independent variable and T4, T5, ... are used to name the requested dependent (output) variables. If T3 does not name a .VARY element, then T3, T4, T5, ... are used to name the requested dependent (output) variables. Up to 5 dependent variables may be requested on a single .PLOT record. Variable names must exist in the current circuit description and are limited to the names of VM and AM elements in the circuit description, the names defined on .CALC records (see .CALC record description), or the pre-defined names listed below.

Each output variable name may optionally be FOLLOWED by a '<' or a '>' to indicate the first or second part of a two part output variable respectively. This convention allows plotting, for example, the magnitude (first part, '<') or phase (second part, '>') from an output request in the .AC analysis mode. Note that the options '<' and '>' only make sense for 'AC' plot requests where the variables have two parts and are ignored if encountered on 'TR' plot requests. If a '<' or '>' is not included, the first part of the requested output variable is assumed.

4.34.1 Predefined PLOT names

Several pre-defined names are available. These names provide access to noise, delay, and multi-parameter sensitivity measures. The pre-defined names are as follows:

Name	Description
`NDM`	integrated noise measure from a `.NOISE` request
`INM`	integrated noise measure from a `.NOISE` request
`MSM`	multiparameter sensitivity measure from a `.MSENS` request
`WSM`	worst case sensitivity measure from a `.WSENS` request
`SSM`	RMS sensitivity measure from a `.SSENS` request
`DLY`	group delay from a `.DELAY` request

4.34.2 PLOT pointer variables

The convention described for T3, T4, T5, ... does not permit requesting individual component sensitivity results. These outputs are not named by CCICAP. To request these variables, or any other variable, the use of 'pointers' is allowed. Plot variables can be requested by their positions in the output data structure. The output data structure is available in the CCICAP ASCII output datafile. At each requested value of the independent variable (frequency, time, or .VARY element value) there will be an array of outputs. The independent variable value is array element 1. For .AC analysis results, element 2 is the first part of the first printed output in the output file, element 3 is the second part of the first printed output in the output file, etc. For .TRANsient analysis results, element 2 is the first printed output in the output file, element 3 is the second printed output in the output file, etc. See the examples below for the way to use position pointers. See Appendix A for a further discussion of the output data structure.

4.34.3 PLOT scaling limits

F1, F2, ... : These fields are optional plot limit sets for the requested variables. The limits allow the user to set the minimum and maximum on-scale values for each variable. Each requested variable requires two

limits. The first limit is the lower plot limit, the second is the upper plot limit. If the limits are not included, or if both limits are specified as 0.0, the plot is automatically scaled so that all data is plotted.

4.34.4 PLOT files

All plot requests normally produce PLOT files in the default disk directory. Plot files are named `PLOTmmnn.AC` or `PLOTmmnn.TR` where the AC extension results from data generated during an .AC analysis and the TR extension results from data generated during a .TRansient analysis. The number `'mm'` refers to the current circuit description and `'nn'` is a plot sequence number within the current circuit analysis and any .ALTERations of the current circuit. The first analysis of a run generates plot files named `PLOT00mm`. The first plot file generated during the analysis is named `PLOT0000`. The second plot file is named `PLOT0001`, etc. The first plot file generated from a second circuit, defined after a .NEXT record, is named `PLOT0100`, etc.

Examples:

```
.PLOT AC 132 eout eout > 0.0 0.0 -180.  180.
```

```
.PLOT AC 132 eout 0.  0.  eout > -180.  180.
```

Produces a 132 column ASCII plot of the AC data for the first part of eout and the second part of eout. The first part of eout will be automatically scaled by CCICAP. The second part of eout will be plotted between the limits of -180. (lower limit) and 180. (upper limit).

```
.PLOT AC PNG eout eout 0.  0.  -.5 .5
```

Produces a PNG graphic plot of the AC data. The variable eout (first part) is plotted with both automatically scaled limits and user specified limits of -.5 to .5.

```
.PLOT AC PNG NDM INM ^19
```

Produces a PNG graphics plot file of the noise density measure, the integrated noise measure, and the 19th value in the output array from the current AC analysis with automatically scaled plot limits. The file will be named `PLOTmmnn.AC.png` where `'mm'` will be the current `.NEXT` number for the run and `'nn'` will be the number of the plot as produced by the current circuit. For example, if the above `.PLOT` request results in the 20th plot produced by the 3rd circuit in a run, the resulting plot file will be named `PLOT0219.AC`.

```
.PLOT TR 132 eout ^7
```

Produces a 132 column ASCII plot of the TRansient data. The variable `eout` and the seventh variable in the output structure are plotted with automatically scaled plot limits.

```
.PLOT AC PNG r45 iout
```

Produces a graphics plot file using the values of `r45` for the independent variable values. The element `r45` must have been specified on a `.VARY .AC` request in the current analysis. The output `iout` is used as the dependent variable and is plotted between automatically scaled limits.

```
.PLOT CLR
```

Clears all previous .PLOT requests and allows new plot requests to be stored. This can be used with an `.ALTGO` record as follows :

```
...
(end of .CKT section)
.END
.GO
.PLOT CLR
.PLOT ...  (new requests)
...
.ALTGO * starts new analysis ...
```

4.35 .ZDLY Record

The ZDLY record is used to change the displayed frequency scale for digital system analysis. Normally each DEL element in a digital system analysis corresponds to a normalized (one second) delay, for which the Nyquist interval is between 0.0 and pi radians per second or 0.0 and 0.5 Hz. The apparent Nyquist interval can be scaled by using the .ZDLY record. The format for the .ZDLY record is as follows:

```
.ZDLY F1
```

where

F1 : implied sampling interval. F1 defaults to 1.0, restricting the useful frequency range on an .AC record to 0.0 to .5 Hz (0.0 to pi rps) for an .AC analysis of a digital system.

Example:

```
.ZDLY 50.usec
```

Sets the apparent sampling rate to 20 KHz and the unit delay to 50 usec. The analysis frequencies on an AC record should be between 0.0 and 10.0 KHz to stay within the Nyquist interval.

Note that the use of the .ZDLY record is strictly for convenience. Network formulation for digital systems are based on a unit delay for each DEL element. The calculations performed by CCICAP use a scaled frequency appropriate for unit delays of one second. Identical results from a digital system analysis will occur by not using the .ZDLY record and restricting the requested analysis frequencies to between 0.0 and 0.5 Hz or 0.0 and pi rps.

4.36 .CKT Record

The circuit description follows the .CKT record. The '.' before CKT must be in the first column of the record. Only comment, network commands, parameter records, and element records can be placed between the .CKT record and the .END record.

See the section on CKT - END Records.

Example:

```
.CKT
```

4.37 .END Record

Marks the end of the circuit description. The '.' before END must be in the first column of the record.

See the section on CKT - END Records.

Example:

```
.END
```

4.38 .ALTEL Record

The ALTEL record provides altered circuit element values for a subsequent analysis. The format for the ALTEL record is:

```
.ALTEL name1 value1 name2 value2 ...
```

where namei is a text element name and must refer to an existing circuit element and valuei is a real value which is to replace the current value for element namei. The values for resistors, capacitors, inductors, VCC

elements, CCC elements, VVC elements, CVC elements, I sources, and V sources can be .ALTered. The phase for I and V sources cannot be .ALTered.

The .ALTEL record must appear after the CKT - END section.

The use of multiple line records is allowed with .ALTEL records. The effect of .ALTEL records is accumulative since the circuit element values are not restored to the original values after an .ALTered circuit is analyzed. Multiple .ALTEL records can be used to alter large numbers of circuit elements.

The changes specified on .ALTEL records are not effective until a subsequent .ALTGO or .GO record is processed.

Example:

```
.ALTEL r45 2.385e3 ein 2.5 \

c3 1.55e-8
```

After the above record the current circuit (defined by a previous .CKT - .END section) will contain modified element values for the elements `r45`, `ein`, and `c3`. The elements `r45`, `c3`, and `ein` must have been defined in the previous .CKT - .END section. The .ALTered circuit will be analyzed when an .ALTGO or .GO record is encountered. The values of any altered elements are scaled according to the current .FSCALE and .ZSCALE parameters, if any.

Only if a legal .ALTEL record is encountered will the alteration number of the run be incremented.

Note that calculated values must be handled carefully. Because of the implementation, the following record will not do what is implied:

```
.ALTEL r3 100.  r4 { r3 199.  X }
```

The value of r3 used to calculate r4 will be the value that existed *before* this `.ALTEL` record. To use the altered value for r3, simply use two `.ALTEL` records:

```
.ALTEL r3 100.

; previous records updates value for r3

; new value of r3 is used in:

.ALTEL r4 { r3 199.  X }
```

4.39 .GO Record

The GO record is used to start an analysis using the currently defined circuit, element values and/or run control records and/or titles.

Note that replacement .NOISE, .TITLE, .SYMBOL, .AC, .TRAN, .IC, .SENSA, and .SENSR records may be inserted between two .GO records. The analyses specified on these replacement records will be initiated by the following .GO record. If no replacement control records are inserted between the two .GO records, the previous run control records are reused.

.PLOT requests will be reused from the previous analysis. If new .PLOT requests are required, the previously defined .PLOT requests can be cleared by the use of the 'CLR' option on a .PLOT request. New .PLOT requests can then be entered for the .GO analysis.

.VARY requests will be reused from the previous analysis. If new .VARY requests are required, the old .VARY requests can be cleared by the use of the 'CLR' option on a .VARY request. New .VARY requests can then be entered for the .GO analysis.

Examples:

...

```
.GO
' now change an element and redo the analysis
.ALTEL r23 3.45k
.GO
...
```

To change a plot request:

```
...
.PLOT CLR * clear old plot requests.
.PLOT AC GP01 eout
* new plot request for this analysis.
.GO * new analysis with revised plot(s).
...
```

To change the .VARY request:

```
...
.VARY CLR
.VARY ...  (new request(s))
.GO
...
```

To generate a new plot and title from existing `DATAmmnn.AC`:

```
...
.GO
' now plot a different variable
...
.AC CLR * get rid of AC request.
.PLOT CLR * clear existing PLOT requests.
.PLOT AC GP02 eout * new PLOT request, part of a plot
* only run since AC request removed.
.TITLE New Title for plot of variable 'eout'
.GO * initiate the run.  ...
```

4.40 .ALTGO Record

Identical to the .GO record.

Example:

```
.ALTGO
```

4.41 .NEXT Record

Sets up for a completely new subsequent circuit definition to follow. The '.' before NEXT must be in the first column of the record.

Example:

```
.NEXT
```

4.42 .STOP Record

Terminates execution of CCICAP. The '.' before STOP must be in the first column of the record.

Example:

```
.STOP
```

5 CKT - END Section Records

5.1 .SCALE OFF

Circuit elements encountered after a .SCALE OFF record and before a subsequent .SCALE ON record (if any) will not have the user specified scaling set by the .FSCALE and .ZSCALE records applied. The values on the following element records are unaltered. The default setting for .SCALE is ON.

5.2 .SCALE ON

Circuit elements encountered after a .SCALE ON record and before a subsequent .SCALE OFF record (if any) will have the user specified scaling set by the .FSCALE and .ZSCALE records applied. The default setting for SCALE is ON. This record is normally used after the .SCALE OFF record to re-enable scaling.

5.3 .STD OFF

Circuit elements encountered after a .STD OFF record and before a subsequent .STD ON record (if any) will not have the user specified standard decades set by the .STDC, .STDL, and .STDR records applied. The values on the following element records are unaltered. The default setting for .STD is ON.

5.4 .STD ON

Circuit elements encountered after a .STD ON record and before a subsequent .STD OFF record (if any) will have the user specified scaling set by the .STDC, .STDL, and .STDR records applied. The default setting for .STD is ON. This record is normally used after the .STD OFF record to re-enable standard decades.

5.5 .PARameter Records

The .PARameter record specifies a symbolic constant for later use. Parameters can be used in subsequent parameter calculations involving other parameters or element values. For example, consider specifying a quartz crystal in terms of its center frequency and Q value:

```
.CKT
```

```
...
* specify Q as 75,000
.par q 75.k
* specify crystal center frequency as 10 MHz
.par fo 10.e6
* calculate center frequency in RPS
.par wo { fo 2.  X PI X }
* crystal motional resistance = 25 ohms
r r1 1 2 25.
* calculate value of motional inductance, L1 = R1 * Q / Wo
L L1 2 3 { r1 q X wo / }
* calculate value of motional capacitance, C1 = 1 / L1 / Wo**2
C C1 3 4 { 1.  L1 / wo / wo / }
...
.END
```

In this example, the crystal center frequency or Q is easily changed and CCICAP will calculate the new values for L1 and C1.

Parameter records that involve a calculation define the calculation between '{' and '}'. Parameter calculations within the .CKT - .END section can refer to *previously* defined parameters and the values of *previously* defined circuit elements. The parameter records are parsed as they are read in from the input file.

There are several predefined parameters and allowed operations that are described under the .PAR Record description and the section on Reverse Polish Notation (RPN).

Note that the calculations on parameter records are performed using Reverse Polish Notation (RPN). See the section on RPN for an explanation of its use.

5.6 Circuit Element records

The order of the elements within the circuit description is not important, although it is recommended that source elements be placed at the beginning of a circuit description. The element type code (R, C, IOA, etc.) must be in the first column of an element record.

The first field following the element type code is the user given name for the element being defined. Following the name field are two, three, or four node names followed by a value if required. The name for ground nodes must be either '0', 'GND', 'Gnd', or 'gnd'.

For all circuit elements the element name and node names are text. The value field is a real.

Each element record describes one circuit element.

5.6.1 Input Voltage Source - V

```
V, name, node1, node2, amplitude, phase (degrees)
```

Voltage is Vnode1 - Vnode2, phase is optional, defaults to 0.0.

5.6.2 Input Current Source - I

`I, name, node1, node2, amplitude, phase (degrees)`

Current is from node1 to node2 through the source, phase is optional, defaults to 0.0.

5.6.3 Resistor - R

`R, name, node1, node2, value`

5.6.4 Conductance - G

`G, name, node1, node2, value`

5.6.5 Capacitor - C

`C, name, node1, node2, value`

5.6.6 Inductor - L

`L, name, node1, node2, value`

5.6.7 Voltmeter - VM

`VM, name, node1, node2`

Measured voltage is Vnode1 - Vnode2.

5.6.8 Ammeter - AM

`AM, name, node1, node2`

Measured current is from node1 to node2, nodes 1 and 2 are short circuited.

5.6.9 Ideal Operational Amplifiers - IOA

`IOA, name, node1, node2, node3, node4`

name	circuit element name
node1	non-inverting input
node2	inverting input
node3	output
node4	output reference node

5.6.10 Non-Ideal Operational Amplifiers - OA

`OA, name, node1, node2, node3, model name`

node1 non-inverting input

node2 inverting input

node3 output (wro ground)

model text name of a previously named model

Example:

`OA, amp1, ein+, ein-, eout, 741`

5.6.11 Field Effect Transistor - FET

`FET, name, node1, node2, node3, model name`

node1 gate

node2 source

node3 drain

model text name of a previously named model

Example:

`fet, q43 gate 44 56 2n1234`

5.6.12 Bipolar Junction Transistor - BJT

BJT, name, node1, node2, node3, model name node1 : base. node2 : emitter. node3 : collector. model : refers to previously named model.

5.6.13 CONVERTERS - CCC; CVC; VCC; VVC

Current to Current Converter

`CCC, name, node1, node2, node3, node4, value`

Controlling current is from node1 to node2, output current is from node3 to node4, value is current gain (A/A).

Current to Voltage Converter

`CVC, name, node1, node2, node3, node4, value`

Controlling current is from node1 to node2, output voltage is Vnode3 - Vnode4, value is transresistance (ohms).

Voltage to Current Converter

`VCC, name, node1, node2, node3, node4, value`

Controlling voltage is Vnode1 - Vnode2, output current is from node3 to node4, value is transconductance (mhos).

Voltage to Voltage Converter

`VVC, name, node1, node2, node3, node4, value`

Controlling voltage is Vnode1 - Vnode2, output voltage is Vnode3 - Vnode4, value is voltage gain (V/V).

5.6.14 Transformer - K

`K name N1 N2 N3 N4 Lp Ls K`

Transformer model.

name circuit element name

N1-N2	are the Primary nodes.
N3-N4	are the secondary nodes.
Lp	Primary winding inductance.
Ls	Secondary winding inductance.
K	Winding coupling coefficient. K = 1.0 for an ideal transformer, K < 1.0 for a lossy transformer. The mutual inductance of the transformer is given by:

$$M = K * \sqrt{Lp * Ls}$$

Example:

```
K T3, 6 2 sec+ sec- 1.3e-3 4.5e-6 0.99
```

5.6.15 Input Node (digital system element) - IN

```
IN, name, node1, gnd, amplitude
```

5.6.16 Output Node (digital system element) - OUT

```
OUT, name, node1, gnd
```

5.6.17 Multiplier Branch (digital system element) - MUL

```
MUL, name, node1, node2, value
```

Node1 variable is multiplied by 'value' and added to node2.

5.6.18 Delay Branch (digital system element) - DEL

```
DEL, name, node1, node2
```

A unit delay is inserted between node1 and node2.

6 Suggested Practice

6.1 Order of input records

The input records can be ordered as indicated in the table of contents. There are a few restrictions on the placement of input records.

The .TITLE, .INC, .PAGE, .OPT, .AC, .DC, .VARY, .TR, .IC, .DELAY, .MAKECKT, .TEMP, .MODEL, .CALC, .STDR, .STDC, .STDL, .SENSA, .SENSR, .MSENS, .WSENS, .SSENS, .NOISE, .FSCALE, .ZSCALE, .PLOT, .ZDLY, and the .CKT to .END section must preceed the first .GO record to be effective during the analysis of the circuit.

As previously discussed, there can be only comment, network command, element description, .PAR, and .INC records between the .CKT and .END records and all element description records must be between the .CKT and .END records. The .INC records used in the .CKT - .END section must specify files which contain only comment, network command, .PAR, .INC, and element description records. The .ALTEL, .ALTGO and .GO records should come after the .CKT - .END section and before the .NEXT or .STOP records.

It is suggested that sources (V and I elements) be placed at the beginning of the CKT section.

6.2 File naming convention

The following convention is suggested only, it is not required.

Extension	Contents
.INP	Files which are specified as input files on the command line.
.OUT	Files which are specified as output files on the command line.
.MDL	Model description files to be INCluded from .INP files.
.CTL	Control record files, to be INCluded from .INP files.
.CKT	Circuit description files, starting with .CKT and ending with .END records, to be INCluded from .INP files.
.ALT	Alternate element value files, starting with one or more .ALTEL records and ending with a .ALTGO or .GO record, to be INCluded from .INP files.

Using the above approach, the contents of a .INP file could have this structure:

```
* file circuit.inp - 04/10/97 - mas
.TITLE Test circuit for CCICAP
.INC circuit.ctl * add the run control records.
.INC /ccicap/models/opamp.mdl * add required model
* from models directory.
.INC circuit.ckt * add the circuit description.
' and run ...
.go
.INC circuit.alt * alter the circuit and rerun.
.STOP
```

The above example could be run from the following command line if the output file was to go to the current directory:

$ ccicap circuit.inp circuit.out

6.3 Script file use

A simple script file that will run a circuit prepared as indicated above and plot any resulting plots can be created as follows:

```
#!  /bin/sh

# remove any left over plot files

rm plot*

# run the analysis

ccicap $1.inp $1.out

# print any new plots that resulted

lpr plot*
```

If the script file is named 'caprun', a CCICAP analysis can be done by entering the following command line:

```
$caprun circuit
```

where 'circuit' is the name of an existing CCICAP analysis file with a '.inp' extension. The ASCII output will be placed in a file named 'circuit.out'.

6.4 Excessive output

Runs prepared for GP01 plots (with a large number of calculated intervals) or with several VARY records or ALTER phases can produce copious quantities of output directed to the ASCII output file. To avoid running out of disk space, direct the ASCII output file to the printer (have lots of paper in place!), the console (CON), or the null (NUL) device. The analysis will run fastest with output directed to the null device. The compressed data will be saved to disk if there is a FILE record in the run or if any PLOTs are requested. Plots can then be obtained using plot only runs (see below) to access the compressed data files.

Another method of reducing the size of ASCII output files is to use the NTAB option on the OPTions record. This option will print out the tabular data heading but will suppress the printing of the data itself. The tabular data heading can be used to document what variables are available in the compressed data files.

6.5 Plot only runs

Plot only runs can be used to access preexisting DATA files on a disk. The runs can be used to avoid having to repeat an entire analysis if the desired plot data is stored in the DATA files. To perform a plot only run, simply remove or comment out any AC, DC, or TRAN records and leave the CKT to END section of the run unchanged.

Plot only runs can change the circuit output variables that are plotted only as long as the requested outputs were specified in the original analysis run. The new outputs need not have been plotted in the original run. Plot limits can be altered on the PLOT records of a plot only run. The run title can be changed and the new title will appear on any plots resulting from the plot only run.

The PLOT files will be renumbered according to the request sequence in the plot only run, starting with PLOT0000. Note that this means any pre-existing PLOT files may be overwritten.

7 References

1. Vlach, Jiri and Kishore Singhal, Computer Methods for Circuit Analysis and Design, Van Nostrand Reinhold:New York, c 1983.

2. van der Ziel, A., "Thermal Noise in Field-Effect Transistors," Proceeding of the IRE, Vol. 50, pp1808-1812, 1962.

3. Press, William H., et. al., Numerical Recipes; The Art of Scientific Computing, Cambridge University Press:Cambridge, c 1986.

4. Bruton, Leonard T., RC-Active Circuits; Theory and Design, Englewood Cliffs, New Jersey:Prentice-Hall, Inc., 1980.

8 Appendices

8.1 Appendix A - Binary Data Files Contents and Output Data Structures

CCICAP can produce binary data output files when the FILE record is included and will produce binary data output files when one or more plots are requested. Two types of binary files are produced, one for the normal analysis DATA and one for the data produced by any VARY records. The files are unformatted FORTRAN data files.

The first type of data file, the file which results from an AC or TRANsient request, is named `DATAmmnn.AC` or `DATAmmnn.TR`. The number mm (between 00 and 99) refers to the circuit number of the run. The first circuit described is numbered 00, the circuit described after the first NEXT record is numbered 01, etc. The number nn (between 00 and 99) refers to the circuit alteration number. The original circuit is numbered alteration 00, the circuit resulting at the first ALTGO record is numbered 01, etc.

In the first type of data file the first record consists of three FORTRAN 4 byte integers

`N1, N2, N3`

where `N1` is the number of data sets which are to follow in the file, `N2` is the number of dependent variables in each of the sets, and `N3` is a format descriptor related to the units of the data which follows.

`N3` is defined as follows:

Let bit-n refer to the n'th bit of the 32 bit integer value N3 where bit- is the lsb, etc.

Then

bit-0 = 1 => results are from an AC analysis, for which:

- bit-4 = 0 => Frequency units are rps.
- bit-4 = 1 => Frequency units are Hertz.
- bit-5 = 0 => Response units are real, imaginary.
- bit-6 = 1 => Response units are dB, phase.
- bit-7 = 1 => Response units are magnitude, phase.

bit-0 = 1 => results are from a TRAN analysis, for which:

- bit-4 = 0 => Response is the impulse response.

- bit-4 = 1 => Response is the step response.

The subsequent records are the data sets which consist of single precision FORTRAN reals:

```
Independent variable 1, data1, data2, ..., dataN2.

Independent variable 2, data1, data2, ..., dataN2.

...

Independent variableN1, data1, data2, ..., dataN2.
```

The ordering of the variables is the same as that in the ASCII file created during the analysis.

For example, an AC analysis with two requested V or I outputs, a single requested calculated response, two requested element sensitivities, a worst case sensitivity, and a noise output would produce data sets and an ASCII output data structure with the following composition:

Position	Variable
1	Frequency : independent variable
2	First output : real part, magnitude, or dB
3	First output : imaginary part or phase
4	Second output : real part, magnitude, or dB
5	Second output : imaginary part or phase
6	Calculated response: real part, magnitude, or dB
7	Calculated response: imaginary part or phase
8	First sensitivity : real part
9	First sensitivity : imaginary part
10	Second sensitivity : real part
11	Second sensitivity : imaginary part
12	Worst case sens. : value
13	Worst case sens. : no imag. part, set = 0.0
14	Noise output : volts/$\sqrt{Hz}$ or amps/$\sqrt{Hz}$ or dB[volts/$\sqrt{Hz}$] or dB[amps/$\sqrt{Hz}$]
15	Integrated noise : volts or amps

The binary files are created by FORTRAN unformatted writes. The data can be recovered by FORTRAN unformatted reads such as:

```
c open the input file ...
      open(luin, file = infile)
c read in the data structure integers
c n1, n2, n3 are 4 byte integers
      read(luin) n1, n2, n3
c read in the data ...
      do i = 1, n1
c read in one record of data
        do j = 1, n2 + 1
          read(luin) xt(j)
        end do
c extract the independent variable
        x(i) = xt(1)
c extract the dependent variables
        do j = 2, n2 + 1
```

```
      y(i, j - 1) = xt(j)
    end do
  end do
```

where `'infile'` defines the name of the CCICAP generated file from which data is recovered. After the above statements, `x(i)` will contain the i'th value of the independent variable and `y(i, j)` will contain the N2 independent variables associated with `x(i)`.

A slightly different format is used for data resulting from any VARY requests. All VARY data resulting from a given circuit or circuit alteration is stored in a single file named `VARYmmnn.AC` or `VARYmmnn.TR` where mm and nn are as described for the `DATAmmnn` files above. The first record in the file consists of four four-byte FORTRAN integers

`N1, N2, N3, N4`

where `N1` is not currently used (set to one), `N2` is the number of dependent variables in each data set, `N3` is the same as described above for the AC or TRANsient results data files, and `N4` is the number of VARY elements in the run that produced the VARY data file.

Each VARY analysis produces N_VARY data sets with N2 dependent variables. The second record in the file consists of N4 four-byte FORTRAN integers whose value equals the number of intervals requested on the VARY records.

The third and subsequent records consist of N2 + 1 FORTRAN single precision reals. Each record records the value of the independent variable (the VARY element) followed by the values of the N2 dependent variables that result.

8.2 Appendix B - Noise Models

CCICAP models the white noise sources associated with all real resistances and active elements. In addition, a 1/f noise corner can be specified for sources in the active elements.

Resistor noise

Resistor noise is calculated from the thermal noise formula:

$$e_n = \sqrt{4. * k * T * R}\, V_{rms}/\sqrt{Hz}$$

where k is Boltzman's constant (1.38e-23), T is the temperature in degrees Kelvin (set by the TEMP record or defaulted to 298.16K), and R is the resistance in ohms. The calculated noise results are spectral densities with units of volts per root Hertz or amps per root Hertz.

Op-amp noise

Op-amp noise is calculated from the voltage and current spectral densities provided by the user on the MODEL record. These parameters are often available from the manufacturer's data sheets. The corner frequencies for voltage and current 1/f noise can also often be obtained or at least estimated from data sheets. The voltage spectral density source is added at the input to the op-amp, effectively in series with the non-inverting input terminal. The current spectral density source is added at each input terminal to ground.

BJT noise

BJT noise is calculated from three sources. The thermal noise of Rbb' is calculated from the thermal noise equation given above for resistor noise. The shot noise of the collector and base currents is calculated from the shot noise formula:

$$i_n = \sqrt{2. * q * I_{DC}}$$

where q is the electronic charge (1.602e-19 C) and I_{DC} is the DC current in question. The collector and base bias currents can be specified on the MODEL record. If the collector current is not specified it is estimated from the equation for the transconductance parameter:

$$I_C = G_m * k * T/q$$

where Gm is the transconductance parameter provided for the model. Similarly, if the base bias current is not specified on the MODEL record, the base bias current is estimated from the device β which is calculated from the Gm and Rb'e parameters as:

$$\beta = G_m * R_{b'e}$$

so that

$$\text{Ib} = \text{Ic} \ / \ \beta.$$

If the noise contribution from the collector current is not desired, set the collector current equal to a negative value on the MODEL record. If the noise contribution from the base current is not desired, set the base current to a negative value on the MODEL record.

FET noise

FET noise sources include the shot noise and 1/f noise of the drain and gate currents and a thermal noise associated with the resistive channel. The shot noise sources are calculated from the user specified drain and gate bias currents. The thermal source is calculated from an effective channel noise resistance given as [2]:

$$\text{Rn} = .67 \ / \ \text{Gm}$$

where Gm is the user specified device transconductance.

8.3 Appendix C - Reverse Polish Notation (RPN)

Reverse Polish Notation is the notation used on Hewlett Packard and other high end calculators. The notation is elegant, allowing unlimited expression complexity without requiring the use of parentheses.

RPN is based on the use of a stack data structure. In CCICAP, the stack has three storage positions. The three positions can be labeled x, y, and z. Whenever a new value is entered into a calculation, it is placed in position x while the value that was in position x is moved to y, the value that was in position y is moved to z, and the value that was in position z is lost.

Operations involving the values stored in the stack positions are of three types: binary, unary, and stack. The values must be stored in the stack before the operation is executed.

8.3.1 Binary Operations

Binary operations operate on the values stored in positions x and y. The allowed binary operations in CCICAP are:

Operator	**Definition**
+	x = y + x (addition)
-	x = y - x (subtraction)
X	x = y X x (multiplication)
/	x = y / x (division)
^	x = y ^ x (exponentiation)

The original x and y values are used in the calculation and the result of the binary operation is placed in position x. The value that was in position z is copied into position y (and retained in position z).

8.3.2 Unary Operations

The unary operations operate only on the variable x, placing the result of the operation back into position x. The values in positions y and z are not changed. The allowed unary operations in CCICAP are:

Operator	Definition
EXP	x = EXP(x) (exponential)
LN	x = LN(x) (natural log)
SIN	x = SIN(x) (Sin, x in radians)
COS	x = COS(x) (Cos, x in radians)
TAN	x = TAN(x) (Tan, x in radians)
SQRT	x = SQRT(x) (square root)
CHS	x = -x (change sign)
CC	x = CONJG(x)(complex conjugate)
RE	x = REAL(x) (extract real part)
IM	x = IMAG(x) (extract imaginary part)

8.3.3 Stack Operations

Stack operations allow rearranging the values on the stack. The allowed stack operations in CCICAP are:

Operator - Definition

XY - interchange the values in x and y:

Step	x	y	z
-	Xo	Yo	Zo
XY	Yo	Xo	Zo

P - push 'down' the stack:

Step	x	y	z
-	Xo	Yo	Zo
P	Xo	Xo	Yo

R - roll the stack 'down':

Step	x	y	z
-	Xo	Yo	Zo
R	Zo	Xo	Yo

R^ - roll the stack 'up':

Step	x	y	z
-	Xo	Yo	Zo
R^	Yo	Zo	Xo

Examples

```
.PAR x1 { 1.  2.  / }
```

As the expression between '{' and '}' is parsed from left to right, the following values are stored on the stack:

Step	x	y	z
1.	1.0	0.0	0.0
2.	2.0	1.0	0.0
/	0.5	0.0	0.0

The operator '/' takes the then current values of x and y (2.0 and 1.0) and divides them, resulting in the final value for the parameter `x1` of 0.5.

```
.CALC V_dB { eout LN 10.  LN / 10.  X 2.  X }
```

This example expresses the variable 'eout' in dB. The step by step stack values are (assuming that eout has the value 2.0 with no imaginary part):

Step	x	y	z
eout	2.0	0.0	0.0
LN	0.693...	0.0	0.0
10.	10.0	0.693...	0.0
LN	2.302...	0.693...	0.0
/	0.301...	0.0	0.0
10.	10.0	0.301...	0.0
X	3.01...	0.0	0.0
2.	2.0	3.01...	0.0
X	6.02...	0.0	0.0

The equivalent algebraic expression is:

$$V_{dB} = \frac{ln(eout)}{ln(10.0)} * 10.0 * 2.0$$

or

$$V_{dB} = 20.0 * log(eout)$$

where *log* is to the base 10.

```
.CALC pwr { eout CC iout X }
```

This example calulates the complex power associated with the voltage eout and the current iout. The equivalent algebraic expression is:

$$pwr = conj(eout) * iout$$

where $conj(eout)$ denotes the complex conjugate of eout.

```
.CKT
...
* define a 10.00 MHz crystal
* crystal Q:
.PAR q 75.e3
* crystal resonant frequency:
.PAR fo 10.e6
* calculate wo (rps):
.PAR wo { fo 2 X PI X }
* crystal motional resistance:
R R1 1 2 50.
* calculate motional inductance:
L L1 2 3 { R1 q X wo / }
* calculate motional capacitance:
C C1 3 4 { 1 L1 / wo / wo / }
* holder capacitance
C Co 1 4 5.pfd
...
.END
```

This example defines two crystal parameters, the q and resonant frequency fo. The values of wo, the motional inductance and the motional capacitance are then calculated based on the established parameters.

8.4 Appendix D - Internal Data Structures

Key Variables

Variable Name	Description
N	size of system matrices (N x N)
NEL	number of elments
NOUT	number of outputs specified
NIN	number in inputs specified
NODES	number of nodes (does not count ground node)
NORDR	order of the linear system

The internal data structures of CCICAP include the following:

Element Code (j)	Element Name	Number of Nodes	Element Description
1	G	2	Conductance
2	C	2	Capacitance
3	VCC	4	Voltage to Current Converter
4	CCC	4	Current to Current Converter
5	VVC	4	Voltage to Voltage Converter
6	I	2	Current Source
7	V	2	Voltage Source
8	IOA	4	Ideal Op-Amp
9	L	2	Inductance
10	R	2	Resistor
11	K	0	Inductive Coupling
12	CVC	4	Current to Voltage Converter
13	AM	2	Ammeter
14	VM	2	Voltmeter
15	BJT	3	Bipolar Junction Transistor
16	FET	3	Field Effect Transistor
17	OA	3	Operational Amplifier
18	IN	1	Digital Input Node
19	OUT	1	Digital Output Node
20	MUL	2	Digital Multiplier
21	DEL	2	Digital Delay Element

Variable	Value	Description
IM1(j, 5)	0	for G, C, VCC, I, R, VM, FET, K, IN, OUT, MUL, DEL
	1	CCC, VVC, V, IOA, L, AM, BJT
	2	CVC, OA
IM1(j, 2)	1	Valid sensitivity request elements.
IT1(j, 2)	1	analog elements
	0	digital elements
IT1(j, 3)	1	source elements
	0	non-source elements
IT1(j, 4)	1	outputs
	0	non-outputs
IT1(j, 5)	1	C, L, DEL
	2	G, VCC, CCC, VVC, CVC, R, K, MUL
	3	I, V, IN
	4	AM, VM
	5	BJT, FET, OA
	6	IOA
IT1(j, 6)		number of nodes for element type j
IT1G(k, j)		pointer to setup NEMT(6-9, i) for element type j

Variable Name	Description
NEMT(1, i)	Element Code for element i
NEMT(2-5, i)	Node numbers for element i
NEMT(6, i)	model number (temp) for element i
NEMT(6-9, i)	coordinates for entry of value to G or C matrix
EL_NAME(i)	Name of element i
NAMES(.)	Node Names