RA
RK

RECEIVED
LAWRENCE
RADIATION LABORATORY

UCRL-20659 c.2

DOCUMENTS SECTION

SCEPTRE USERS MANUAL

Bradford B. McMillan
(M.S. Thesis)

March 1971

AEC Contract No. W-7405-eng-48

TWO-WEEK LOAN COPY

This is a Library Circulating Copy which may be borrowed for two weeks. For a personal retention copy, call Tech. Info. Division, Ext. 5545

LAWRENCE RADIATION LABORATORY
UNIVERSITY of CALIFORNIA BERKELEY

UCRL-20659 c.2

DISCLAIMER

This document was prepared as an account of work sponsored by the United States Government. While this document is believed to contain correct information, neither the United States Government nor any agency thereof, nor the Regents of the University of California, nor any of their employees, makes any warranty, express or implied, or assumes any legal responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by its trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof, or the Regents of the University of California. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof or the Regents of the University of California.

SCEPTRE USERS MANUAL

Bradford B. McMillan

Lawrence Radiation Laboratory
University of California
Berkeley, California 94720

FORWARD

The version of SCEPTRE available at LRL was originally received as SCEPTRE/6600 version 10-69-A in December of 1969. Version 10-69-A is an adaptation of the original SCEPTRE written for the IBM 7094. The only user's manual available at that time was the one that had been written for the original SCEPTRE. Because of the many changes that the version 10-69-A had been through, the user's manual was in many ways not relevant to this newer SCEPTRE. There were features that the original user's manual spoke of that just simply did not work on our version; and, in going through the coding, features were found that were not mentioned in the manual.

During the course of the project, it was determined which of the omitted features had been purposely deleted from our version and which had been aborted by programming errors, machine differences, etc. Some of these determinations were made by studying the coding involved. Others were done by direct conversations with Captain John Anderson and Lieutenant R. B. McBride of AFWL Albuquerque, New Mexico, and an author of the original SCEPTRE, Stephen R. Sedore of IBM Electronics Systems Center, Owego, New York. Those features purposely deleted, the coding was changed to conform with its intended purpose.

D. Defined Parameters:

A special section has been created to enable the user to define quantities that may be output other than sources or passive currents and voltages. The user may enter systems of first-order differential equations that may or may not have anything to do with a particular electrical network.

E. Output:

In addition to the conventional output format, which allows all sources and passive currents and voltages at each solution increment, the user may request as output any defined parameter from item D. He may also select any element value, step size, and pass count. Time is not the only independent variable for these outputs, the user may select others from a fairly large list.

F. Subprogram Capability:

The user who is familiar with computer programming may write FORTRAN subroutines and insert them in otherwise conventional SCEPTRE runs. This option permits handling special situations--even though these should be rare.

G. Automatic Termination:

Runs may be automatically terminated contingent on the behavior of specified network quantities.

H. Flexibility:

Nonconventional source dependencies and network topologies can be accommodated.

To acquaint the reader with the input format, a simple inverter circuit (see Fig. 1) and the input cards that would describe it appear here.

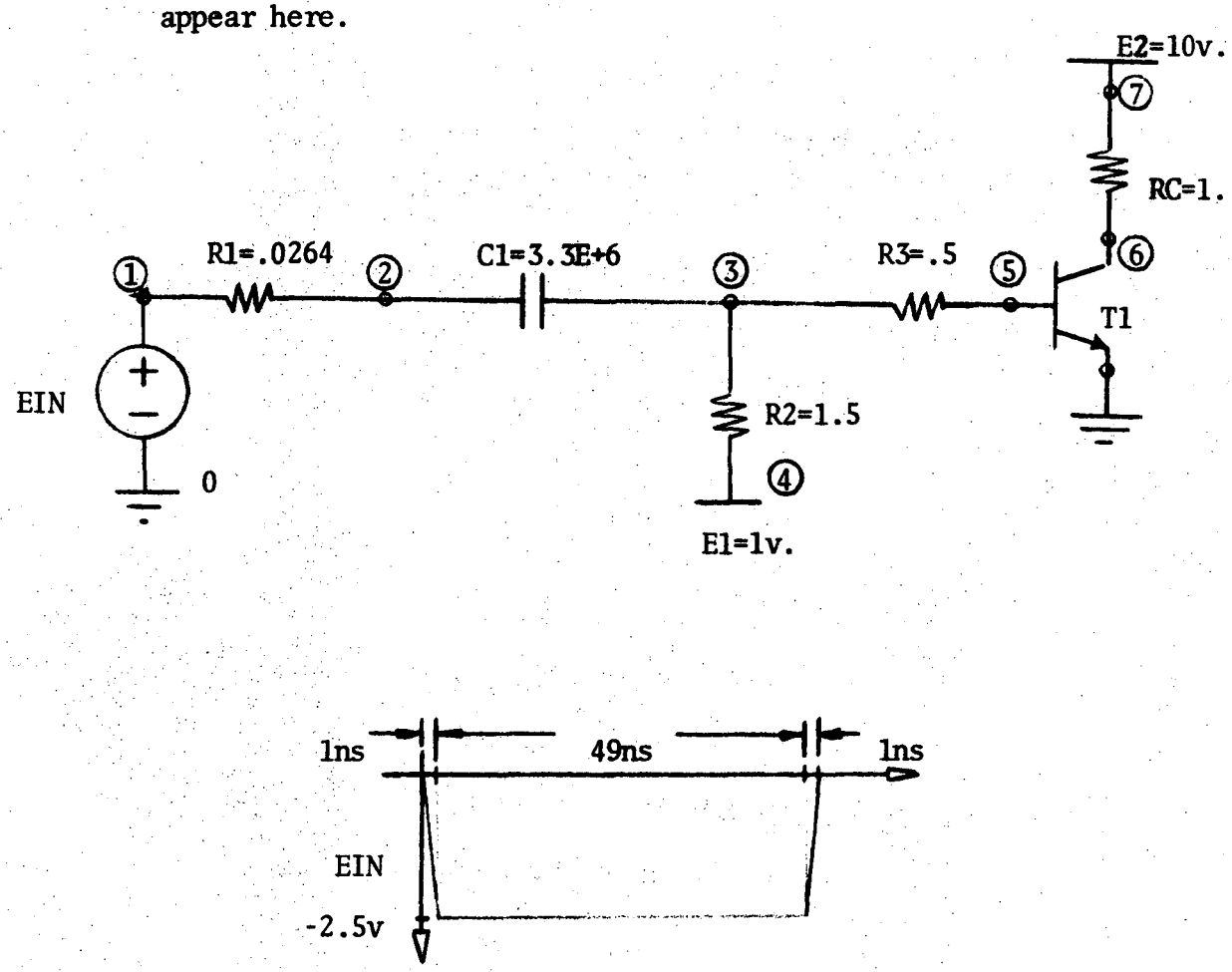


FIG. 1

SIMPLE INVERTER CIRCUIT

See the transistor models subsection, Chapter 3. for a detailed description of the transistor model.

DATA CARDS

```
MODEL DESCRIPTION (INITIAL,PRINT)
MODEL 2N2369 (PERM) (B-E-C)
ELEMENTS
CE,1-E=EQUATION 1(VCE,JE,5.,.9,.30,1.8E-11,37.2,.25)
CC,1-2=EQUATION 1(VCC,JC,3.7,.9,.22,5.75E-11,38.5,6.2)
JE,1-E=DIODE EQUATION(1.8E-11,37.2)
JC,1-2=DIODE EQUATION(5.75E-11,38.5)
JN,2-1=EQUATION 2(TABLE BTA(JE))*JE
JI,E-1=.5*JC
RB,B-1=.1
RC,C-2=.05
R1,E-1=1.E+6
R2,1-2=1.E+6
FUNCTIONS
EQUATION 1(C,G,A,B,D,H,E,F)=(A/ABS(B-C)**D+(E*F)*(G+H))
EQUATION 2(A)=(A/(A+1.0))
TABLE BTA
.1,50., 1.,100., 5.,75.
CIRCUIT DESCRIPTION
   ---   SCEPTRE EXAMPLE CIRCUIT   ---
   ---   ONE TRANSISTOR INVERTER   ---
ELEMENTS
EIN,0-1=TABLE 1(TIME)
E1,0-4=1
E2,0-7=10
R1,1-2=.0264
R2,3-4=1.5
R3,3-5=.5
RC,7-6=1
C1,2-3=3.3E6
T1,5-0-6=MODEL 2N2369 (PERM)
DEFINED PARAMETERS
PVOUT=(10.0-VRC)
PWRIN=(EIN*IEIN)
INITIAL CONDITIONS
VJET1=.5
OUTPUTS
PVOUT(VOUT),PWRIN,VCET1,PLOT
PVOUT(VOUT),PLOT(EIN(VIN))
FUNCTIONS
TABLE 1
0.,0., 1.,-2.5, 50.,-2.5, 51.,0., 52.,0.
RUN CONTROLS
STOP TIME=100
RUN INITIAL CONDITIONS
END
```

Each element is described by a card with the element name, the nodes to which it is connected, and its value. The element type is usually implicit in the name. The name of a resistor starts with an R, the name of a capacitor starts with a C, and so on. The nodes are entered in a from-to direction of assumed positive current flow. For voltage sources the assumed direction of positive current flow is from the negative node to the positive node. Note the voltage source specifications for EIN, E1, and E2 (node 0 is assumed to be ground).

All values for this example are in units of kΩ, pFs, and nsec. It is recommended that all circuits input to SCEPTRE be normalized in this manner.

The value for the capacitor C1, 3.3E6, implies a value of 3300000. This is the standard E format where the integer following the letter E is a power of ten.

This example will be used extensively here to demonstrate the various features of SCEPTRE.

CHAPTER 2. INPUT DECK

The input deck consists of control cards and data input. The control cards are usually the same for each run and will change only when certain special features of SCEPTRE are exercised. The data cards describe the circuit to be analyzed and they therefore change with each circuit. These cards are described below.

A. CONTROL CARDS

At LRL Berkeley, the control cards that are to be used with each run appear below. These cards should be followed by a 7-8-9 card (a card with a 7, 8, and 9 punched in the first column) and then by the data cards. The last card in the input deck should in all cases be the white LRL end-of-file card.

```
JOB CARD.    (NAME,PRIORITY,TIME,126000.JOB NUMBER,USER NAME)
LIBCOPY(EEBINARY,SCEPTRE,SCEPTRE)
LIBCOPY(EEBINARY,TAPE60,SCEPTR2)
SCEPTRE.
RUNF(S,,,TAPE8,NULL)
LOAD(I=LGO,L=TAPE60,O=NOMAP)
NGO.
SCEPTR2.
```

a) Permanent Library:

If the user wishes to keep a permanent library of stored models, he must request that a magnetic tape be mounted to store them. To have this magnetic tape mounted, the card,

```
REQUEST LIBTAPE,nnnn,O.
```

must appear directly after the job card. The nnnn is an LRL library tape number. One permanent library already exists at LRL with an extensive list of possible transistor and diode models from which to choose. The library number for this tape is 12042. The transistors and diodes that are on this tape appear in Appendix C.

b) TV Plots/Cal-Comp Plots:

Any time the user requests printed plots as outputs, he has the additional options of getting either 35 mm photographic film or Cal-Comp outputs as well. To request film outputs the cards

```
LIBCOPY(EEBINARY,PLOTTV,PLOTTV)
PLOTTV.
```

should appear at the end of the other control cards. The film can be used to make 8 x 11 copies of the desired plots. To request Cal-Comp plots, the cards

```
LIBCOPY(EEBINARY,PLOTCC,PLOTCC)
PLOTCC.
```

should appear.

When requesting TV plots or Cal-Comp plots the user should also submit a COS card to the I/O desk specifying the number of plots requested.

B. DATA CARDS

The data input deck consists of heading and subheading cards and, under these, statements to describe the circuit to be analyzed and to control the run. The headings, and the subheadings allowed under each, appear below.

i) MODEL DESCRIPTION(INITIAL,PRINT)

MODEL NAME (PERM, TEMP OR DELETE)(NODE-NODE-..-NODE)
(COMMENT OR MESSAGE CARDS, IF ANY, UP TO 11 ALLOWED)
1. ELEMENTS
2. DEFINED PARAMETERS
3. OUTPUTS
4. FUNCTIONS

XSTER NAME (PERM, TEMP, OR DELETE)
USER SPECIFIED PARAMETER VALUES
1. FUNCTIONS

The MODEL DESCRIPTION heading is used when it is desired to store one or more models. See the stored model feature section of this report for a more detailed description of this option.

ii) CIRCUIT DESCRIPTION

(COMMENT OR MESSAGE CARDS, IF ANY, UP TO 11 ALLOWED)
1. ELEMENTS
2. DEFINED PARAMETERS
3. OUTPUTS
4. INITIAL CONDITIONS
5. FUNCTIONS
6. RUN CONTROLS

The CIRCUIT DESCRIPTION heading is always used when any network is presented for analysis. Any or all of the six subheadings listed under the heading may be used. The comment cards can include up to 72 columns of alphanumeric information.

iii) RERUN DESCRIPTION(N)

(COMMENT OR MESSAGE CARDS, IF ANY, UP TO 11 ALLOWED)
1. ELEMENTS
2. DEFINED PARAMETERS
3. INITIAL CONDITIONS
4. FUNCTIONS
5. RUN CONTROLS

The RERUN DESCRIPTION Heading is used whenever the rerun feature is exercised. All changes to the master network must appear under this heading.

iv) END

This card must always appear.

The heading and subheading cards represent the general sequence of information input. The formats and types of information to be input under each subheading are described below.

a) ELEMENTS:

All elements introduced in MODEL DESCRIPTION, CIRCUIT DESCRIPTION, or RERUN DESCRIPTION must appear under this subheading. The general form for cards under elements is

ELEMENT NAME,NODE-NODE=VALUE

where ELEMENT NAME is a maximum of five alphanumeric characters. The connection nodes are specified in a from-to order corresponding to the assumed direction of current flow. NODE is a maximum of six alphanumeric characters. VALUE may be a constant, a table name (with an independent variable), a defined parameter, an equation name (with an argument list), or a mathematical expression. Possible entries under the ELEMENTS section are summarized on the following page (Table 1.).

TABLE 1.

ENTRIES UNDER ELEMENTS

ELEMENT NAME STARTS WITH		NODES		VALUE SPECIFICATION
R (RESISTOR) C (CAPACITOR) L (INDUCTOR) E (VOLTAGE SOURCE) J (CURRENT SOURCE)	, , , , ,	NODE-NODE	=	CONSTANT, TABLE NAME (INDEPENDENT VARIABLE), DEFINED PARAMETER, OR EQUATION NAME (ARGUMENT LIST)
M (MUTUAL INDUCTANCE)	,	LNAME-LNAME		
LINEARLY DEPENDENT SOURCES				
E J	, ,	NODE-NODE NODE-NODE	= =	CONSTANT*VR CONSTANT*IR
PRIMARY DEPENDENT CURRENT SOURCES				
J	,	NODE-NODE	=	DIODE TABLE NAME, OR DIODE EQUATION (X1,X2)
SECONDARY DEPENDENT CURRENT SOURCES				
J	,	NODE-NODE	=	CONSTANT*JNAME (A PRIMARY DEP CURRENT SRC)
VOLTAGE AND CURRENT SOURCE DERIVATIVES				
DE DJ			= =	CONSTANT CONSTANT
MODEL CALLS				
ANY NAME	,	NODE-NODE-...-NODE	=	MODEL NAME (PERM OR TEMP)

The special values included for various sources are the only forms of mathematical expressions that should be used under the value specification. Although SCEPTRE will accomodate other expressions under this specification, the format for these expressions is more restricted than it is under the FUNCTIONS subheading. It is therefore recommended that these expressions be entered under FUNCTIONS.

The value specifications under primary dependent current sources, DIODE TABLE NAME and DIODE EQUATION(X1,X2), are included to accomodate those current sources that represent diode or transistor junctions. DIODE EQUATION(X1,X2) should be used when the closed form representation J=X1[EXP(X2*VJ)-1] is desired. Note that this is the conventional form for the diode equation where X1 is the saturation current, IS, and X2 is q/kT. The program will automatically use the voltage across the current source as the independent variable.

The first character M implies that a mutual inductance between two inductors is being specified. The usual mutual inductance polarities apply as sign conventions. Note that in this case the node specifications are replaced by the names of the inductors involved. For example, if coupling exists between inductors L1 and L2, the appropriate entry might be

MX,L1-L2=32.4

in addition, the user must ensure that MX is less than the $\sqrt{L1*L2}$. This ensures that the coefficient of coupling is less than unity.

If a variable voltage source is connected in a loop containing only capacitors and other voltage sources, or a variable current source is connected in a cut set containing only inductors and other current sources, the time derivatives of these sources must be supplied. The derivatives are supplied under the format indicated. Note that in this case no node specification is required.

Some cards from our inverter circuit example of Chapter 1 are repeated here as possible entries under this subheading.

```
CE,1-E=Q1(VCE,JE,5.,.9,.30,1.8E-11,37.2,.25)
JC,1-2=DIODEQ(5.75E-11,38.5)
EIN,0-1=T1(TIME)
C1,2-3=3.3E6
T1,5-0-6=MODEL 2N2369 (PERM)
```

For the input data, the word EQUATION and the letter Q are equivalent as are the word TABLE and the letter T. For example, note that the specification TABLE 1(TIME) has been replaced here by T1(TIME).

The tabular entry for TABLE 1 and the analytical expression for EQUATION 1 must be entered under the FUNCTIONS subheading. A good general rule to follow throughout the program is that all constants in parenthesis must include decimal points.

b) DEFINED PARAMETERS:

This subheading will permit the user considerable flexibility in the use of SCEPTRE. Any variable that can be described in terms of any network variable and/or any number may be defined and this quantity may be used as an element value, an argument in an equation or table, or an output at each time step of the problem, in the same manner as any conventional output. The general form for entries under DEFINED PARAMETERS is

```
PNAME=VALUE      ,or
DPNAME=VALUE
```

where the D indicates the derivative of the defined parameter PNAME. Under this format it is possible to have the program simulate any system of non-linear first-order differential equations that may or may not have anything to do with any particular circuit.[3] For example, if we wished to simulate the differential equation, $\frac{dN}{dt} = -50 \times N$, $N(0)=70$. We could input

```
CIRCUIT DESCRIPTION
DEFINED PARAMETERS
DPN=(-50.0*PN)
PN=70.
RUN CONTROLS
```

Note that the value of N (PN) at time zero is also entered under the defined parameters section. This manner of inputting can extend to any number of simultaneous first-order differential equations. Since the value of the variable can be in any range,

the user should make sure that the error criteria used is proper for his particular problem (see the discussion on integration variables under the run controls section of this report).

c) OUTPUTS:

Any output must consist of some dependent variable which is a function of some independent variable. SCEPTRE outputs consist of printed tabular listings of requested dependent variables as functions of time, and/or plots of the dependent variables as functions of time or some other independent variable. SCEPTRE will not give a tabular listing of a dependent variable as a function of any independent variable other than time. The following quantities may serve as either dependent or independent variables.

- The voltage or current associated with any element (for example, VR1 implies that the desired output is the voltage across resistor R1).
- Any element value.
- Any defined parameter.

The general form for entries under OUTPUTS is

VARIABLE1,VARIABLE2,VARIABLE3 or if a plotted output is desired,

VARIABLE1,VARIABLE2,VARIABLE3,PLOT

If an independent variable other than time is desired for the plotted output, the proper entry is

VARIABLE1,VARIABLE2,VARIABLE3,PLOT(INDVAR)

where INDVAR is the desired independent variable. For example, the entry

PVOUT,PLOT(EIN)

will give a plot of PVOUT as a function of EIN. To rename any output variable, the user need only supply the new name for it in parenthesis following its entry in the output list. For example, the entry

PVOUT(VOUT),PLOT(EIN(VIN))

in our inverter circuit will give the same plotted output as the above entry, but here the variables will be named VOUT and VIN respectively.

d) INITIAL CONDITIONS:

Under this subheading, the user may specify the initial value of any circuit variable (a voltage or a current). The general format for entries under INITIAL CONDITIONS is

VNAME=CONSTANT or,
INAME=CONSTANT

multiple entries may appear on one card if they are separated by commas.

Any entry under this subheading specifies the value of the variable at the beginning of the analysis phase. This is true irregardless or whether the program is to do an initial condition analysis or begin the transient analysis right away. Entries included under this subheading, then, serve one of two possible uses.

First, if an initial condition analysis is being made, it assists convergence to a solution if an initial guess of some of the circuit variables is made (a more complete discussion of this is included under the run controls section of this report). This is especially true if a diode junction is expected to be forward biassed in the initial condition solution. It is always a good idea, for example, to give an initial guess of .5 volts across the internal base-emitter junction of any silicon transistors that are expected to be initially forward biassed in a circuit for which an initial condition solution is requested. Note that this has been done in the inverter circuit example.

Second, if a transient only run is being made, the set of all capacitor voltages and inductor currents should be supplied for a complete solution. These variables are the only meaningful entries in the transient mode of operation. Those not specified are assumed to be zero.

e) FUNCTIONS:

In this data group, each of the tables and equations referred to under ELEMENTS and/or DEFINED PARAMETERS must be defined in detail.

- For tabular entries, the general form for entries under FUNCTIONS is

```
TABLE NAME OR DIODE TABLE NAME    [or TNAME]
NUMBER,NUMBER,NUMBER,NUMBER,ETC.
```

where each pair of numbers represents a unique point. And for each point pair, the independent variable appears first. Note that this form is used to specify TABLE 1 in our inverter circuit example of Chapter 1. An alternative format for specifying this table would be

```
T1 = 0.,0., 1.,-2.5, 50.,-2.5, 51.,0., 52.,0.
```

The table values are updated at each time step by linear interpolation and extrapolation. The reader should note that by giving the point (52.,0.) it is assured that extrapolation will give a value of zero for all values of time greater than fifty-one units of time.

- For equation definition entries, the general form for entries under FUNCTIONS is

```
EQUATION NAME (DUMMY VARIABLE LIST) = (MATHEMATICAL EXPRESSION)
[OR  QNAME(DUMMY VARIABLE LIST) = (MATH EXP)]
```

The dummy variable list must contain the same number of entries as does the argument list in the original equation reference. Each dummy variable may contain up to six alphanumeric characters, the first of which must not be a number or any of the letters I thru N inclusive. All mathematical expressions that appear in SCEPTRE should be enclosed in parenthesis.

In addition to the usual mathematical operations of multiplication, division, addition and subtraction, any algebraic operational functions available in FORTRAN are also available in SCEPTRE. These include SQRT(X), SIN(X), COS(X), EXP(X), etc. As an example, if the value of a variable capacitance, say C1, is given by

$C1=10[e^{38*V_{C1}}]$ the cards

```
C1=EQUATION 1 (10,38,VC1)              under ELEMENTS, and
EQUATION 1 (A,B,C) = (A*(EXP(B*C)))    under FUNCTIONS
```

would produce the desired result.

f) RUN CONTROLS:

This is the subheading under which all auxiliary information needed to control the run is entered. First, the user may specify which of the three possible modes of operation he desires. An initial condition only run, a transient only run, or both an initial condition and transient run. If an initial condition run is being made, he may control the criteria used for convergence

of the Newton-Raphson scheme that SCEPTRE employs.[2] He can control the start time and stop time for a transient run and can choose the type and accuracy of the numerical integration routine used. Additional control, including automatic termination, of the run is available as outlined below.

Under normal conditions, a transient only analysis will be made. This mode of operation can be changed by either of the following entries under RUN CONTROLS

RUN I. C. ONLY	--	Only an initial conditions run will be made in this case. There will be no transient calculations done.
RUN INITIAL CONDITIONS	--	This will cause both an initial condition and transient run to be made.

Depending on the mode of operation, various control quantities can be supplied. Most of these quantities have automatically preset values that hold unless specific entries are supplied. The general form for these entries is

QUANTITY = VALUE SPECIFICATION

possible entries under the RUN CONTROLS subheading that are of this general type, along with their preset information, are given in Table 2. on the following page.

TABLE 2.

QUANTITY	PRESET TO	VALUE SPECIFICATION
INITIAL CONDITION VARIABLES		
NEWTON-RAPHSON PASS LIMIT	100	Number [see the following discussion
RELATIVE CONVERGENCE	.001	Number on initial condition control
ABSOLUTE CONVERGENCE	.0001	Number variables]
TRANSIENT RUN VARIABLES		
START TIME	0	Number [A transient run may be started at $t \neq 0$]
STOP TIME	NONE	Number [This must be specified in all transient runs]
MAXIMUM PRINT POINTS	1000	Number [Max outputed irregardless of number calculated]
INTEGRATION VARIABLES		
INTEGRATION ROUTINE	XPO	TRAP or RUK [see the following discussion
MAXIMUM INTEGRATION PASSES	20,000	Number on integration variables]
MINIMUM STEP SIZE	E-5(STOP TIME)	Number
MAXIMUM STEP SIZE	E-2(STOP TIME)	Number
STARTING STEP SIZE	E-3(STOP TIME)	Number
MAXIMUM ABSOLUTE ERROR	.0075	Number
MAXIMUM RELATIVE ERROR	0	Number
MINIMUM ABSOLUTE ERROR	.0002	Number
MINIMUM RELATIVE ERROR	0	Number

RUN CONTROL VARIABLES

- Initial Condition Control Variables -

For the initial condition solution, SCEPTRE employs a modified Newton-Raphson iteration scheme[2] which linearizes all non-linear variables about their previous operating point for any given iteration. This procedure continues until each variable is within a specified tolerance of its previous value. More specifically, assuming preset tolerances, when the value of each and every variable has not changed by more than the quantity (.001+.0001x(variable value)) in two successive iterations, convergence is assumed to have occured. The user may change the values .001 (ABSOLUTE CONVERGENCE) and .0001 (RELATIVE CONVERGENCE) to attain any tolerance he wishes for convergence.

If, after 100 iterations, there are variables that are still changing between successive iterations by more than their allowed tolerance, a diagnostic message is outputted and the program aborts. The upper limit of 100 iterations (THE NEWTON-RAPHSON PASS LIMIT) can also be changed by the user. Experience to date has shown, however, that most circuits that will not converge in 100 iterations will not converge at all, and those circuits that do converge do so in less than 30 iterations.

Convergence problems in initial condition solutions are the direct result of the gross non-linearities in the circuit being solved. The exponential relationship between the current and the voltage of a diode or transistor junction is one such non-linearity, and the way in which this relationship is handled has become a classic problem in programs of this sort.[5] In general, any non-linear set of equations will converge to a

solution if the iteration process is begun sufficiently close to it. SCEPTRE begins iterating assuming that all circuit variables (voltages and currents) are zero except those representing independent sources. It then converges to a solution from this state. Therefore, if convergence problems arise, the user should begin the iteration process closer to the actual solution. To do this, he can take advantage of the fact that the value of any variable at the beginning of any run (initial condition or transient) can be specified under the INITIAL CONDITIONS subheading. That is, he can specify initial guesses to an initial condition solution under the INITIAL CONDITIONS card to assist convergence.

As an example, note that in our inverter circuit example, the voltage source E1 assures that the transistor will be initially forward biassed. For circuits in which there are forward biassed silicon transistors, the base-emitter voltages for each transistor will be in the range of .5 to .8 volts. since this variable represents the dominant current through the transistor, and represents a gross non-linearity, it would help ensure convergence if the user specified an initial guess of .5 volts. Here, this non-linearity is represented by the current source JE. Therefore, the card VJET1=.5 serves as an initial guess to the value of this voltage and the program will converge easier.

- Integration Variables -

Three integration routines are available for use with SCEPTRE. These routines are called XPO, TRAP, and RUK (exponential, trapezoidal, and Runge-Kutta integration respectively). Studies to date indicate that the XPO integration scheme is usually, though not always, faster than the other methods. For this reason, this routine is used unless one of the others is specifically requested by an INTEGRATION ROUTINE card.

In general, integration routines use the value of each variable at a given time and an estimate of the amount that it changes to calculate what its value is at a later time. The error acquired at each time step depends on the accuracy of the estimate of the change over the length of the time step. It can be shown that this error is proportional to a power of the step size.[4] In the XPO routine, for example, it is proportional to the cube of the step size. This error is not directly available to the integration scheme (otherwise, it could use them for a more accurate result in the first place), but its upper limit is available, and this is used to control the size of the time step taken.

SCEPTRE begins the transient solution by arbitrarily taking a step size one one-thousandth of the specified stop time. At each step (including the first one) it calculates the values of all the circuit variables at the new time and estimates the size of the error acquired for each variable. If at any time this error is greater than the quantity (MAXIMUM ABSOLUTE ERROR + MAXIMUM RELATIVE ERROR x (variable value)) for any variable, the step size is halved. This process is repeated

until the estimated error sizes for all variables are less than their allowed tolerances. If the error estimates for all variables are less than (MINIMUM ABSOLUTE ERROR + MINIMUM RELATIVE ERROR x (variable value)) for seven consecutive steps, the step size is doubled. If at any time the required step size becomes less than the minimum step size, the run is stopped and it is indicated in the output that a smaller minimum step is needed to keep the errors within their specified tolerances. At no time will the step size be allowed to become larger than the maximum step size.

The reader can see from the list of integration variables he has control over, that he may specify any degree of accuracy he desires. Since increased accuracy implies smaller step sizes, which in turn, implies more computer time, we see that this increased accuracy can get very expensive. Experience has shown that the values to which the integration variables are preset give an acceptable accuracy-execution time trade off for most circuits.

For some circuits the required integration accuracy will force the step size to become prohibitively small. This situation occurs most often when a circuit has time constants that are small compared to the time interval over which the user requests a solution. Except in borderline cases, where a slight adjustment will produce an acceptable result, this represents a real limitation to the usefulness of programs such as SCEPTRE. If this happens, the user has the option of either accepting relatively large errors or using alternate methods of analysis.

In a transient run, the user may wish to monitor certain voltages or currents in a network to determine their relation to some predetermined quantity. If the relation is satisfied, there may be no further interest in continuing the run. The run can be terminated at that point by the entry

TERMINATE IF (xxx..NN.yyy)

[where .NN. can be .LT. (less than),
.LE. (less than or equal to),
.GT. (greater than),
.GE. (greater than or equal to),
.EQ. (equal to),
.NE. (not equal to)]

As an example of this last entry under RUN CONTROLS, if the card TERMINATE IF (VCC.GE.0) appears, the run will terminate if the voltage across capacitor CC becomes positive.

CHAPTER 3. STORED MODEL FEATURE

Most users of circuit analysis programs find that repeated use is made of special combinations of elements such as filter sections, biasing networks, or in modeling transistors, etc. A convenient approach in handling this situation is to describe the network or model once and store it for future use. In SCEPTRE, a model may be stored temporarily or permanently. If it is stored temporarily, it is available for the run in which it is stored only. If it is stored permanently, it is available for any number of runs until it is deleted from permanent storage. To select whether a particular model is to be stored temporarily, permanently, or deleted from a permanent model library, one of the key words TEMP, PERM, or DELETE must appear on the corresponding MODEL NAME card. Whenever the permanent stored model feature is used, a model library magnetic tape must be mounted for the run. A discussion on the procedure for doing this is included under the control cards section of this report.

All stored models are transferred from storage to the main circuit, where they are used by reference to their external nodes. Corresponding external nodes in both the main circuit and the stored model must match in sequence. As an example of the use of this feature, the following sequence of cards will store, as a temporary model, a series R-L-C circuit, and will call this model into use under the circuit designation T1 and T2.

```
MODEL DESCRIPTION
MODEL SERIES RLC CIRCUIT (TEMP) (EXT01-EXT02)
ELEMENTS
R,EXT01-1=1E3
L,1-2=1E-6
C,2-EXT02=1E-6
OUTPUTS
VC,IL,PLOT
CIRCUIT DESCRIPTION
COMMENT CARD ... RLC MODEL EXAMPLE ... COMMENT CARD
ELEMENTS
      .
      .
T1,5-6=MODEL SERIES RLC CIRCUIT (TEMP)
      .
T2,11-4=MODEL SERIES RLC CIRCUIT (TEMP)
      .
      .
END
```

-- The name of this model is SERIES RLC CIRCUIT. Its external nodes are designated EXT01 and EXT02.

-- The capacitor voltage and inductor current will be output in both tabular and plotted form (as a function of time) for each place this model is used.

-- Note that under the circuit designation T1, the node EXT01 of the model becomes node 5 and the node EXT02 becomes node 6 in the main circuit. Internal nodes are discussed below.

The name of any internal node, element, table, or equation of a stored model becomes a combination of its name under the MODEL DESCRIPTION and the circuit designation of that model in the main circuit. In the above example, the inductor L becomes LT1 when referred to as a part of T1, and becomes LT2 when referred to as a part of T2. Because of this, and because of the restrictions on the length of names for nodes (6 characters), elements (5 characters), tables (5 characters) and equations (5 characters), the length of these names when used in a stored model and the length of circuit designations of models should be kept small.

On both the model name card and the circuit designation cards, if neither TEMP or PERM (or DELETE) appear for the specified model, temporary storage is assumed.

The model name (in this case, SERIES RLC CIRCUIT) must be kept to a maximum of 18 characters. As specified earlier, the subheadings elements, defined parameters, outputs, and functions are valid subheadings under the MODEL NAME card.

If a permanent library is being used, the words INITIAL or PRINT or both (separated by a comma) may appear in parenthesis on the MODEL DESCRIPTION card. INITIAL should appear for the special case of the first model permanently stored on any individual tape. PRINT should appear whenever a printed listing of all models on that particular tape is desired. These words are never used on the MODEL DESCRIPTION card for temporary storage.

A word of warning is in order here. If the word INITIAL appears and the model tape that was mounted for the run already has models on it, they will be erased. Also, the permanent library on tape number 12042 has over two hundred models on it. If PRINT appears while using this library tape, over seventy pages of output will result.

A. CHANGES IN STORED MODELS

The user who frequently makes use of the stored model feature of SCEPTRE will often encounter the situation in which the topology of his stored model is satisfactory but the size of some of the model elements must be changed. Changes can be effected easily for any individual run, but no permanent changes to the stored model are possible. (The user has the option of storing a second model which contains a different version of the original.) All changes must be made on the circuit designation card that appears under the ELEMENTS subsection of the circuit description.

a) Changes in elements or defined parameters:

The method by which element values or defined parameters are changed in a stored model is best illustrated by an example. Suppose that the user wished to make the value of the capacitor in the above RLC model example a tabular function of the voltage across it in the model designated T1. The following would make the necessary change

```
T1,5-6=MODEL SERIES RLC CIRCUIT (CHANGE C=TABLE7(VCT1))
```

of course, the tabular function (TABLE7) would have to appear under the FUNCTIONS subheading of the main circuit. If, at the same time, he wished to change the resistor to 500 Ω, he could insert the following entry

T1,5-6=MODEL SERIES RLC CIRCUIT (CHANGE C=TABLE7(VCT1),R=5E2)

which would produce the desired result.

b) Changes in outputs:

The RLC model example above calls for the outputs VC and IL. If other quantities from the model are desired as outputs, they may be requested under the OUTPUTS subheading of the main circuit. To request the current in the resistor R in T2, for example, the specification IRT2 would appear under outputs.

Outputs requested in a stored model may also be suppressed in any given circuit designation. To inhibit the outputs for T2 either

T2,11-4=MODEL SERIES RLC CIRCUIT (PERM,SUPPRESS VC,IL) , or

T2,11-4=MODEL SERIES RLC CIRCUIT (PERM, SUPPRESS ALL)

would appear. Under the first format any number of the previously requested outputs can be suppressed.

The normal output routine will produce a listing of the entire circuit without a detailed printout of any stored models. To get such a printout the circuit designation card should be changed to read

T2,11-4=MODEL SERIES RLC CIRCUIT (PERM,PRINT)

c) Changes in functions:

A table or equation that appears under the FUNCTIONS subheading of the stored model can be changed as illustrated in the following example

Q4,7-4-6=MODEL2N44D (PERM, CHANGE TABLE1=TABLE7) ,or

Q17,3-18-2=MODEL 2N44D (PERM, CHANGE Q1=Q2)

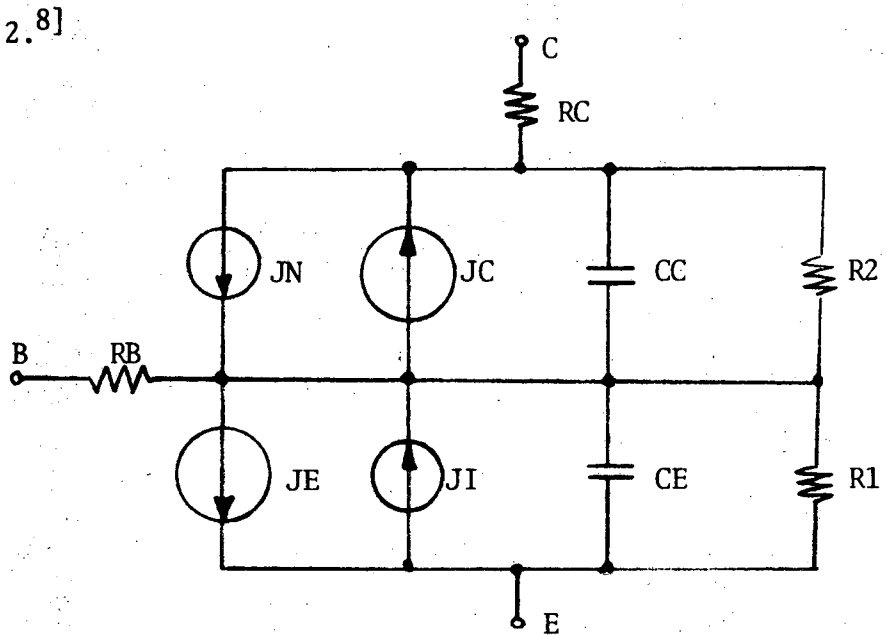
For the table change, no correlation is required between the number of point pairs in the original and new tables. For the equation change, the independent variables that held for the original equation must also apply for the new equation. A table cannot be changed to an equation, and an equation cannot be changed to a table, but this type of change can easily be effected by making the appropriate change in the values of the affected element or defined parameter.

d) Multiple changes:

Any combination of the above changes to stored models can be accomodated by including them all, separated by commas, within one set of parentheses.

B. TRANSISTOR MODELS

Because of the expected high use of the Ebers-Moll transistor model,[7] a version for which the user need only supply the parameter values is available. The model, and the equations that describe it are shown in Fig. 2.[8]



where

$$JE = I_{ES}\left(e^{\theta_E \times V_{JE}} - 1\right)$$

$$JC = I_{CS}\left(e^{\theta_C \times V_{JC}} - 1\right)$$

$$JN = \alpha_N \times JE$$

$$JI = \alpha_I \times JC$$

$$CE = \frac{a_E}{\left|\phi_E - V_{CE}\right|} n_E + \theta_E \times \tau_E \, (JE \times I_{ES})$$

$$CC = \frac{a_E}{\left|\phi_C - V_{CC}\right|} n_C + \theta_C \times \tau_C \, (JC \times I_{CS})$$

FIG. 2. SCEPTRE TRANSISTOR MODEL

To use this model, use the XSTER NAME card and specify the parameter values on the following cards. Each of the parameters has a preset value and the user need specify only those he wishes to change. The parameters, their SCEPTRE name, and their preset values are listed in Table 3. below.

PARAMETER	SCEPTRE NAME	PRESET VALUE
a_E	AE	1.
ϕ_E	PHIE	.8
n_E	NE	.5
I_{ES}	IES	1.E-11
θ_E	THETAE	38.5
τ_E	TAUE	.25
α_I	ALPHAI	.5
R_B	RB	.1
R_1	R1	1.E+6
a_C	AC	1.
ϕ_C	PHIC	.8
n_C	NC	.5
I_{CS}	ICS	2.E-11
θ_C	THETAC	38.5
τ_C	TAUC	25.
α_C	ALPHAN	.99
R_C	RC	.05
R_2	R2	1.E+6

TABLE 3.

The designation PNP (or NPN) can appear anywhere among the parameters to declare that a PNP (or NPN) transistor is desired. If neither designation appears, an NPN transistor is assumed.

As an example of the format for using this feature, the following is a listing for the original version of the transistor model used in the inverter circuit example of Chapter 1, and the format for storing the same model using the abbreviated (XSTER) version.

ORIGINAL VERSION

```
MODEL 2N2369 (PERM) (B-E-C)
ELEMENTS
CE,1-E=EQUATION 1(VCE,JE,5.,.9,.30,1.8E-11,37.2,.25)
CC,1-2=EQUATION 1(VCC,JC,3.7,.9,.22,5.75E-11,38.5,6.2)
JE,1-E=DIODE EQUATION91.8E-11,37.2)
JC,1-2=DIODE EQUATION95.75E-11,38.5)
JN,2-1=EQUATION 2(TABLE BTA(JE))*JE
JI,E-1=.5*JC
RB,B-1=.1
RC,C-2=.05
R1,E-1=1.E+6
R2,1-2=1.E+6
FUNCTIONS
EQUATION 1(C,G,A,B,D,H,E,F)=(A/ABS(B-C)**D+(E*F)*(G+H))
EQUATION 2(A)=(A/(A+1.0))
TABLE BTA
.1,50., 1.,100., 5.,75.
```

ABBREVIATED VERSION

```
XSTER 2N2369 (PERM) (B-E-C)
AE=5.,PHIE=.9,NE=.30,IES=1.8E-11,THETAE=37.2
AC=3.7,PHIC=.9,NC=.22,ICS=5.75E-11,TAUC=6.2
ALPHAN=Q2(TBTA(JE)),NPN
FUNCTIONS
Q2(A)=(A/(A+1.0))
TBTA=.1,50., 1.,100., 5.,75.
```

The transistor model stored will always have the element names shown in the original version. Note that in the abbreviated version we were able to specify the forward beta as a function of JE, the emitter current, even though JE did not explicitly appear as an element.

Since the name EQUATION1 is used for the equation that describes the junction capacitances CE and CC, the user should not use this name to define any other equations for the transistor model. The nodal sequence (B-E-C) on the XSTER NAME card will mean that the three nodes specified on the circuit designation card will be base, emitter, and collector respectively. If no nodal sequence is included, the sequence (B-E-C) is assumed.

Note that in the abbreviated version, only those parameters that differ from the corresponding preset value are specified.

The FUNCTION subheading is the only one allowed under the XSTER NAME card.

CHAPTER 4. RERUN FEATURE

The rerun feature of SCEPTRE allows the user to run multiple versions of a circuit. Numerical values may change for each version, but the topology and element form must remain the same.

The RERUN DESCRIPTION (N) heading can appear as many times as desired after the CIRCUIT DESCRIPTION. (N) specifies the number of reruns desired under that heading. If (N) does not appear, only one rerun will be made. Each individual rerun will utilize all information from the master run except that which is specifically modified for that particular rerun. Any intermediate reruns will have no effect on a subsequent rerun.

For each desired rerun, a complete transient only run will be made. The initial conditions used will be those calculated, or specified, for the master run. If the nature of the change is such that new initial conditions should be calculated, a RUN INITIAL CONDITIONS card should appear under the RUN CONTROLS section.

Any element, defined parameter, initial condition, or run control variable that had a numerical value for the master run may be changed under rerun. The format under these subheadings is best illustrated by an example. Suppose the user wished to change the value of the capacitor C1, in the inverter circuit example of Chapter 1 to 3.7E6, then to 4.1E6, and then to 4.5E6 in three consecutive reruns. Suppose also that he

wished to change the problem duration to 110, then to 120, and then to 125 for the three values of C1. The cards

```
        .
        .
RERUN DESCRIPTION (3)
ELEMENTS
C1 = 3.7E6, 4.1E6, 4.5E6
RUN CONTROLS
STOP TIME=110,120,125
END
```

would accomplish the desired result. The user may not describe the values of more than one variable on the same card under this heading.

Under the functions subheading, only tables can be changed. Equations or equation arguments cannot be directly changed under rerun. It should be noted in passing though, that any argument that has a numerical value in the argument list of an equation specified under elements can be changed to a defined parameter. The numerical value of this defined parameter can, in turn, be changed in any subsequent reruns. For example, note the forth argument, ϕ_E, of CE, the nonlinear capacitor, in the transistor model of the inverter circuit example of Chapter 1. This argument has a value of .9 in the master run. Suppose now that the user wished to change its value to .85 and then to .8 in two reruns. If the following changes were made to the transistor model cards

```
        .
ELEMENTS
CE,1-E=EQUATION 1(VCE,JE,5.,PRM,.30,1.8E-11,37.2,.25)
        .
        .
DEFINED PARAMETERS
PRM=.9
FUNCTIONS
        .
```

we see that the value of ϕ_E would be .9 in the master run. Its value could then be changed to .85 and .8 in subsequent reruns by placing the cards

```
        .
        .
RERUN DESCRIPTION (2)
PRMT1=.85,.8
END
```

at the end of the CIRCUIT DESCRIPTION cards.

When specifying a variable under rerun, the user must use its name under the main circuit. The user cannot change the value of a variable in a model and have that change apply to all places where that model is used in the main circuit. Note that in the example above, the defined parameter PRM of the transistor model was referred to as PRMT1 under rerun.

When changing a table, the independent variable values can be changed only once for each RERUN DESCRIPTION heading and the dependent variable values can be changed for each individual rerun. The format for changing a table is also best illustrated by an example. Suppose that for a first rerun we wished the Table T1 (in the inverter circuit example of Chapter 1.) to become 0.,0., 5.,-3., 25.,-3., 30.,0., 31.,0. and for the second rerun we wished it to become 0.,0., 5.,-5., 25.,-5., 30.,0., 31.,0. (Note that the independent variable values 0., 5., 25., 30., 31. are the same in both cases and we can therefore include both changes under the same RERUN DESCRIPTION heading.) The entry

```
        .
        .
RERUN DESCRIPTION (2)
FUNCTIONS
T1 = 0.,0.,0.
     5.,-3.,-5.
     25.,-3.,-5.
     30.,0.,0.
     31.,0.,0.
END
```

would produce the desired result. The reader can see from the format why only one set of independent variable values can be specified under a single RERUN DESCRIPTION heading. A change in the independent variable values of a table between reruns will be the only case in which the RERUN DESCRIPTION heading will have to be used more than once.

All outputs that applied for the master run will apply to all subsequent runs. Termination conditions also apply to all reruns.

CHAPTER 5. SUBROUTINE CAPABILITY

This feature is intended for the user with some experience in FORTRAN programming and with an occasional need for special computation that is not directly provided by SCEPTRE. Any subroutine may be written according to the rules of FORTRANIV function subroutines and input as a part of the input deck. The subroutine name must be unique and begin with any letter except I, J, K, L, M, or N. It must not exceed six alphanumeric characters. Uniqueness would be insured if the letter F were used as the first character of any subroutine created by the user.

To illustrate a possible use for this feature, and the method for using it, suppose we wanted to describe a voltage source, EIN, whose value was a nonconventional function of time. An appropriate procedure would include under ELEMENTS

```
EIN,0-1=Q7(TIME)
```

and under FUNCTIONS

```
Q7(A)=(FGEN(A)).
```

The subroutine itself should be included before the regular data cards and after a card with the word SUBPROGRAM punched in the first ten columns. A possible entry for the example above could be

```
(7-8-9) CARD
SUBPROGRAM
      FUNCTION FGEN (Z)
      IF (Z.LE.10.) GO TO 1
      FGEN=5.
      RETURN
    1 FGEN=-.05*(Z**2)+Z
      RETURN
      END
      .
      .
      .
```

CHAPTER 6. PROGRAM LIMITATIONS AND RESTRICTIONS

The storage of data is restricted only by the amount of memory allocated in FORTRAN DIMENSION statements. The data limits shown in Table 4., therefore, reflect only the current sizes of storage blocks and they can be increased by an experienced programmer.

TABLE 4.

DESCRIPTION OF DATA	MAX
Heading Cards	11
Nodes	301
Defined Parameters	100
Mutual Industances	50
Model Table Changes	15
Model Output Suppressions	10
Supplied Initial Conditions	100
Cards Per Equation Function	20
Optional Termination Conditions	10
Model Terminals (External Nodes)	25
Elements	300
Source Derivatives	50
Defined Parameter Differential Equations	100
Arguments in Equation Value Specification	50
Model Equation Changes	15
Output Requests	100
Equation Functions (1 Equation Per Card)	80
Table Functions	80
Models on Library Tape (Combined)	250
Model Internal Nodes	301

Additional restrictions to the use of SCEPTRE include topological restrictions on the circuit introduced. The user must recognize the following restrictions

- No run may contain a loop composed exclusively of voltage sources or a cut set composed exclusively of current sources.
- No initial condition run may contain a loop composed exclusively of voltage sources and inductors or a cut set composed exclusively of current sources and capacitors.
- No resistor current can be used as an independent variable in a table or equation used in an initial conditions run.

The user may note that a wide variety of network quantities are allowed as arguments in equation and table construction. It is sometimes true that the use of certain quantities can cause a computational delay in transient runs. That is, computation at the nth time step will begin with independent variable values that are valid at the $(n-1)^{th}$ time step. The amount of error introduced by these delays is difficult to assess. As a rule, the error will be proportional to the degree of nonlinearity exhibited by the equation or table. Computational delays will not occur if the independent variables are time, capacitor voltages, or inductor currents. The validity of the use of other independent variables cannot be unambiguously stated. The status is topology dependent. The program will always print out a warning message if a computational delay occurs, provided that the proper equation or table formats are used.

During simulation, the message THE NEWTON-RAPHSON PASS LIMIT HAS BEEN EXCEEDED WITHOUT ATTAINING CONVERGENCE may appear. The reader is referred to the discussion on convergence in the run controls section of this report for a discussion of this error. Another possible message during simulation is MATRIX ****** IS SINGULAR. If this message is encountered, it is suggested that the user examine the parameter values of his circuit very closely for any possible errors. This message is often the result of errors of this type.

CHAPTER 7. ERROR DIAGNOSTICS

The free form input data format of SCEPTRE is intended to minimize formatting errors. Other types of errors such as those of omission, ambiguity, inconsistency, and violation (of syntax, program limits, etc.) must be detected and diagnosed and the user alerted. For this purpose, the program possesses a comprehensive input data diagnostic capability. As the input data cards are read in by the input processor, each card is printed out and then scanned for errors. If an error is found, an error message stating the trouble is printed immediately following the card in error. The severity of errors detected in this manner is also indicated. There are three levels of severity

- LEVEL 1 -- WARNING ONLY - execution continues
- LEVEL 2 -- SIMULATION DELETED - error scan continues, but analysis phase is aborted
- LEVEL 3 -- EXECUTED TERMINATED - processing is immediately stopped

The program also contains a diagnostic capability to inform the user if any errors are detected while the network topology is being analyzed. During this phase of execution, the program will detect any violations of the topological restrictions discussed in the previous section. This phase of execution will also inform the user of any possible computational delays (also discussed in the previous section) that will occur during simulation.

FOOTNOTE AND REFERENCES

This work was carried out as a part of the research program of the Physics Instrumentation Division of the Lawrence Radiation Laboratory, University of California which is supported by the U. S. Atomic Energy Commission Contract W-7405-eng-48.

1. H. W. Mathers, S. R. Sedore, J. R. Sents, "SCEPTRE Support", Technical Report No. AFWL-TR-67-124 Vol. 1-2, April 1968.

2. H. W. Mathers, S. R. Sedore, J. R. Sents, "Automated Digital Computer Program for determining Responses of Electronic Circuits to Transient Nuclear Radiation (SCEPTRE)", Technical Report No. AFWL-TR-66-126 Vol. 2, February 1967.

3. J. B. Portasik, C. M. Glass, M. L. Goode, "Systems Analysis Using the SCEPTRE Computer Program", Technical Report No. AFWL-TR-68-98, September 1968.

4. A. Ralston, "A First Course on Numerical Analysis", McGraw-Hill, Inc. pp. 159-228, 1965.

5. W. J. McCalla, D. O. Pederson, "The Elements of Computer-Aided Circuit Analysis", IEEE Trans. on Circuit Theory, Vol. CT-18 No. 1, January 1971.

6. W. J. McCalla, W. G. Howard, "Bias-3, A Program for the Non-linear DC Analysis of Bipolar Transistor Circuits", IEEE Journal of Solid-State Circuits, Vol. SC-5, No. 6, December 1970.

7. D. O. Pederson, "Electronic Circuits", McGraw-Hill, Inc., pp. 34-59, 1966.

8. C. M. Glass, G. K. Pritchard, "Model Conversion for CIRCUS, TRAC, SCEPTRE, and NET-1", Technical Report No. AFWL-TR-69-37, July 1969.

9. E. K. Chen, "A Comprehensive Study of CIRCUS", University of California M. S. Thesis, June 1969.

ACKNOWLEDGEMENTS

The author wishes to express his sincere gratitude to H. G. Jackson for the guidance and encouragement received throughout the entirety of this project and to D. O. Pederson for his inspirational teaching. The author also wishes to acknowledge the many helpful discussions and assistances given by J. Katz, T. Shimizu, R. Wilson and W. McCalla, and I. Getreu. The SCEPTRE program used in Berkeley is a modification of the version graciously supplied by Captain J. Anderson of AFWL Albuquerque, New Mexico.

The author further wishes to express his deepest gratitude to Vicky D. Donelson for her assistance in helping prepare this report.

APPENDICES

APPENDIX A. SCEPTRE OPERATION AND PROGRAMMING NOTES

The version of SCEPTRE currently availiable at LRL was originally SCEPTRE/6600 VERSION 10-69-A. It has since been adapted for use on the BKY (LRL) SCOPE system. It takes a state-space approach to circuit analysis. This approach allows the user to input a wide variety of real and theoretical circuit types. This versatility comes at no small cost. The coding and logic sequence is quite cumbersome at some points. And, especially for larger circuits, the amount of computer time necessary for a full run can become prohibative. The user who is familiar with the operation of SCEPTRE, however, can in many cases take full advantage of the versatility and minimize unnecessary computations. The following outline of the control cards and their purpose will familiarize the reader with SCEPTRE's operation.

1. LIBCOPY (EEBINARY,SCEPTRE/BR,SCEPTRE)

   This card loads an unrelocatable (cross-referenced) object deck onto a disk file called SCEPTRE.

2. LIBCOPY (EEBINARY, TAPE60/BR,SCEPTRE2)

   This card loads two files onto a disk file called TAPE60. The first file is a set of card images utilized by SCEPTRE. The second file is an object deck of relocatable subroutines that will be used in the second phase.

3. SCEPTRE.

This card loads the object deck, SCEPTRE, into the computer and begins execution. During execution, a FORTRAN program, SCEPTRE2, is written onto a disk file called TAPE8. The first file of TAPE60 is used here as data. In addition, element values, table values, control variables, etc. are written on TAPE8 as data to be read by SCEPTR2. An additional record of data is written for each rerun requested. To get a printed listing of the SCEPTR2 program and the data associated with it, the user need only put a card with the words WRITE SIMUL8 DATA in the RUN CONTROLS subsection of the main circuit.

4. RUNF(S,,,TAPE8,NULL)

This card compiles the program written on TAPE8. The word NULL is there to suppress any output listing from this compilation.

5. LODE(I=LGO,L=TAPE60,0=NOMAP)

This card loads the object deck compiled above, and the compiled subroutines that are on the second file of TAPE60. The 0=NOMAP suppresses the load-map output.

6. NGO.

This card does the cross-referencing for the header program SCEPTR2, and the subroutines on TAPE60. The unrelocatable object deck is then written onto a disk file called SCEPTR2, and

7. SCEPTR2.

Loads and executes this program.

We can see from the above outline that one possible time savings would result if we were running the same circuit many times changing only element values between runs. In this case, the user could store the compiled SCEPTR2 object deck and use it each time for execution instead of recompiling for each run. The only information needed from Step 3. would be the data.

For additional information on SCEPTRE operation as well as information on the mathematical formulations used, the reader is referred to technical report Nos. AFWL-TR-66-126 and AFWL-TR-67-124.

APPENDIX B. FEATURES UNIQUE AT LRL BERKELEY

This section is intended for the user who is familiar with other versions of SCEPTRE. The following is a list of the major differences between the version of SCEPTRE available at LRL and many versions found elsewhere.

The LRL version does not have:

1. A RE-OUTPUT option.
2. A CONTINUE option.

The LRL version does have:

1. A TV plot capability.
2. A Cal-Comp plot capability.
3. A built-in transistor model.

In addition, the LRL version of SCEPTRE has been modified to handle the characteristic represented by the value specification DIODE EQUATION (X1,X2) differently.[6] This modification was made to improve the convergence to an initial condition solution. Therefore, many circuits that will not converge on other versions of SCEPTRE will converge on the version available here.

APPENDIX C.

PERMANENT LIBRARY MODELS CURRENTLY AVAILABLE AT LRL

The following is a list of the diodes and transistors for which a model exists on tape number 12042, the LRL permanent SCEPTRE library tape. For instructions on the use of this tape, see the control card section of this write-up.

DIODES

1N63	1N93	1N100	1N140
1N191	FD200	UT262	1N270
1N279	FD300	UT484	1N486A
SD500	FD600	FD624	FDA630
1N645	1N646	1N647	1N648
1N649	1N658	1N659	1N661
1N695	FD700	1N746A	1N747A
1N748A	1N749A	1N750A	1N751A
1N752A	1N753A	1N754A	1N755A
1N756A	1N757A	1N758A	1N759A
1N827A	1N903	1N914	1N914B
1N961	1N962B	1N963B	1N964B
1N965B	1N966B	1N967B	1N968B
1N969B	1N970B	1N971B	1N972B
1N973B	1N995	FDM1000	1N2199
H2969	1N2997	1N3016B	1N3017B
1N3018B	1N3019B	1N3020B	1N3021B
1N3022B	1N3023B	1N3024B	1N3025B
1N3026B	1N3027B	1N3028B	ID3050T
1N3070	1N3071	1N3600	1N3605
1N3669	1N4001	1N4003	UT4410
1N4572	1N4610	FD6666	

TRANSISTORS

3TX002	2N174	A210	TC229	2N315	2N329A	2N335
2N336	2N343	2N356	2N375	2N384	2N385	2N393
2N398	2N404	MPS404	MPS404A	2N414	2N457	NS480
GE489A	2N491B	GE493	GE494A	MJE521	2N585	2N597
2N598	2N657	2N697	2N705	2N706	2N706A	2N711A
2N718	2N718A	2N720A	2N722	2N743	2N797	2N834
2N835	2N910	2N914	2N915	2N916	2N918	2N955A
2N964	2N976	2N995	2N1016B	2N1016E	2N1037	2N1039
2N1099	2N1131	2N1132	2N1184	2N1225	2N1228	2N1289
2N1301	2N1304	2N1306	2N1307	2N1308	2N1342	2N1483
2N1490	2N1499A	2N1506A1	2N1613	2N1711	2N1724	MM1737
PT1867	2N1893	2N1900	2N2048	2N2060	2N2087	2N2102
2N2126	2N2187	2N2188	2N2192	2N2222	2N2223	2N2243A
2N2258	MM2258	2N2369	2N2411	2N2453	2N2481	2N2484
2N2538	GE2646	MO2646	MO2647	GE2647	2N2656	2N2695
2N2708	2N2784	2N2808	2N2845	2N2887	2N2894	2N2905
2N3017	2N3019	2N3021	2N3026	2N3055	2N3108	2N3117
2N3119	2N3227	2N3244	2N3251	2N3252	2N3287	2N3309
2N3468	2N3498	2N3499	2N3501	2N3502	2N3503	2N3507
2N3600	2N3635	2N3737	2N3738	2N3766	2N3792	2N3828
2N3866	2N3904	2N3906	2N3913	2N3914	2N3915	2N3959
2N3960	2N4125					

APPENDIX D. EXAMPLE CIRCUITS AND COMPUTER TIMING NOTES

In this appendix, we will try to give the reader some idea of the amount of computer time necessary to solve various types of circuits. Three examples; a simple inverter, a one stage amplifier, and a monostable multivibrator were run on both SCEPTRE and the CIRCUS program, another transient analysis program available at LRL.[9] The SCEPTRE input data and the respective output waveforms for these examples are shown on the next few pages. A summary of the computer (cp) time necessary to solve these circuits on each program is given in Table D1. following the examples.

EXAMPLE D1. INPUT DATA

SIMPLE INVERTER CIRCUIT

```
MODEL DESCRIPTION
XSTER 2N2369 (TEMP)
RB=.025,RC=.005
AE=5.,PHIE=.9,NE=.30,IES=1.8E-11,THETAE=37.2
AC=3.7,PHIC=.9,NC=.22,ICS=5.75E-11,THETAC=34.5
ALPHAN=Q2(TBN(JE)),ALPHAI=Q2(TBI(JC))
TAUE=TTN(JE),TAUC=TTI(JC)
FUNCTIONS
Q2(A)=(A/(A+1.))
TBN = 1.,65.5, 2.,68., 5.,69.4, 10.,65.5, 20.,27.4, 50.,27.8
TBI = .75,.15, 1.7,.17, 2.5,.18
TTN = 1.,.36, 2.,.39, 5.,.39, 10.,.4
TTI = 1.,64., 2.,48.5, 5.,62.
CIRCUIT DESCRIPTION
   ---     SIMPLE INVERTER CIRCUIT    ---
ELEMENTS
EIN,0-1=T1(TIME)
E1,0-4=-1
E2,0-7=10
RS,1-2=.0264
R1,3-4=.5
R2,3-5=.5
RC,7-6=1
C1,2-3=3.3E6
T1,5-0-6=MODEL 2N2369
DEFINED PARAMETERS
PVOUT=(10.0-VRC)
INITIAL CONDITIONS
VCET1=-1., VCCT1=-11., VC1=1.
OUTPUTS
PVOUT(VOUT),PLOT
FUNCTIONS
T1 = 0.,0., 1.,2.9, 110.,2.9, 111.,0., 113.,0.
RUN CONTROLS
STOP TIME = 200.
END
```

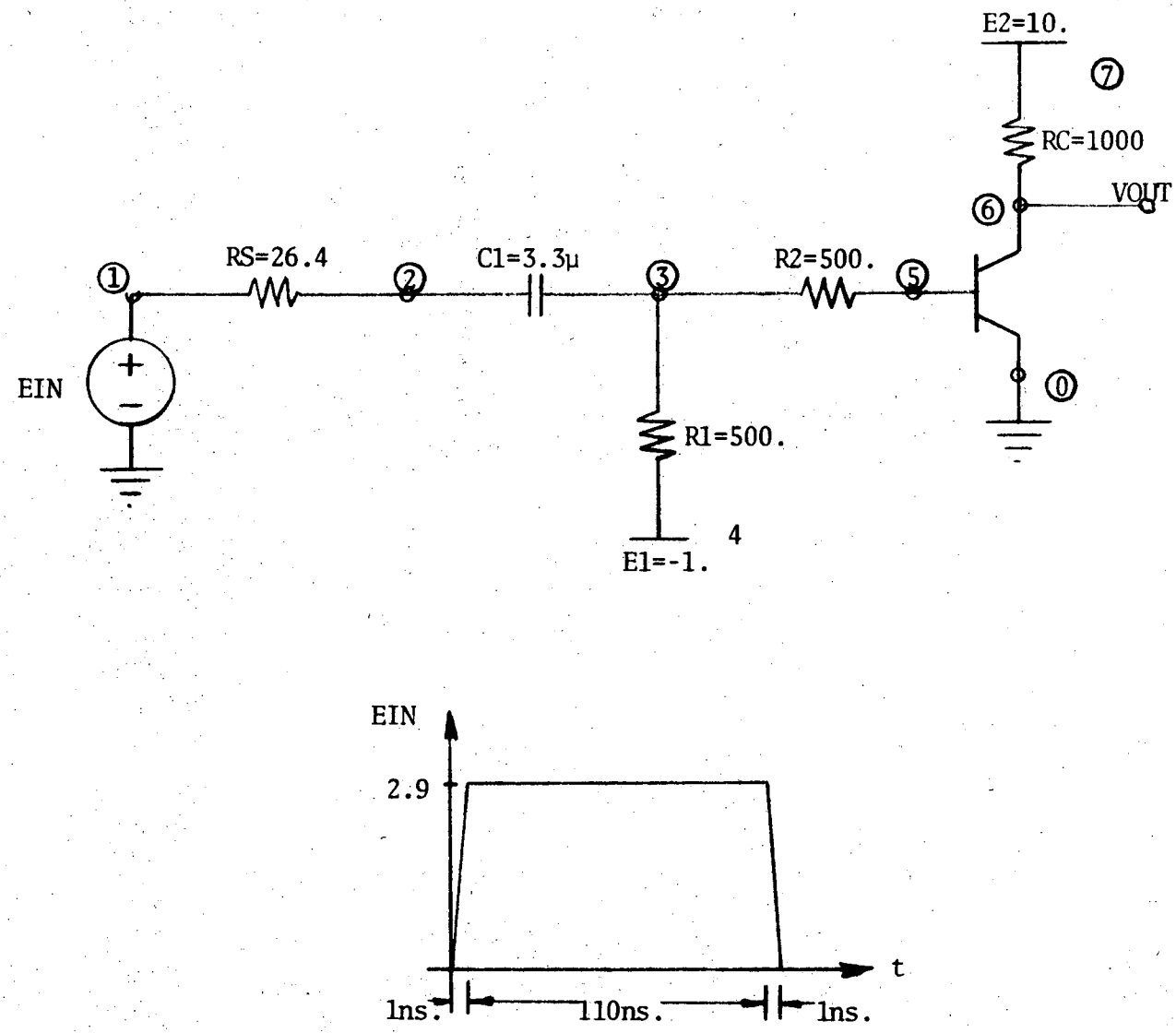
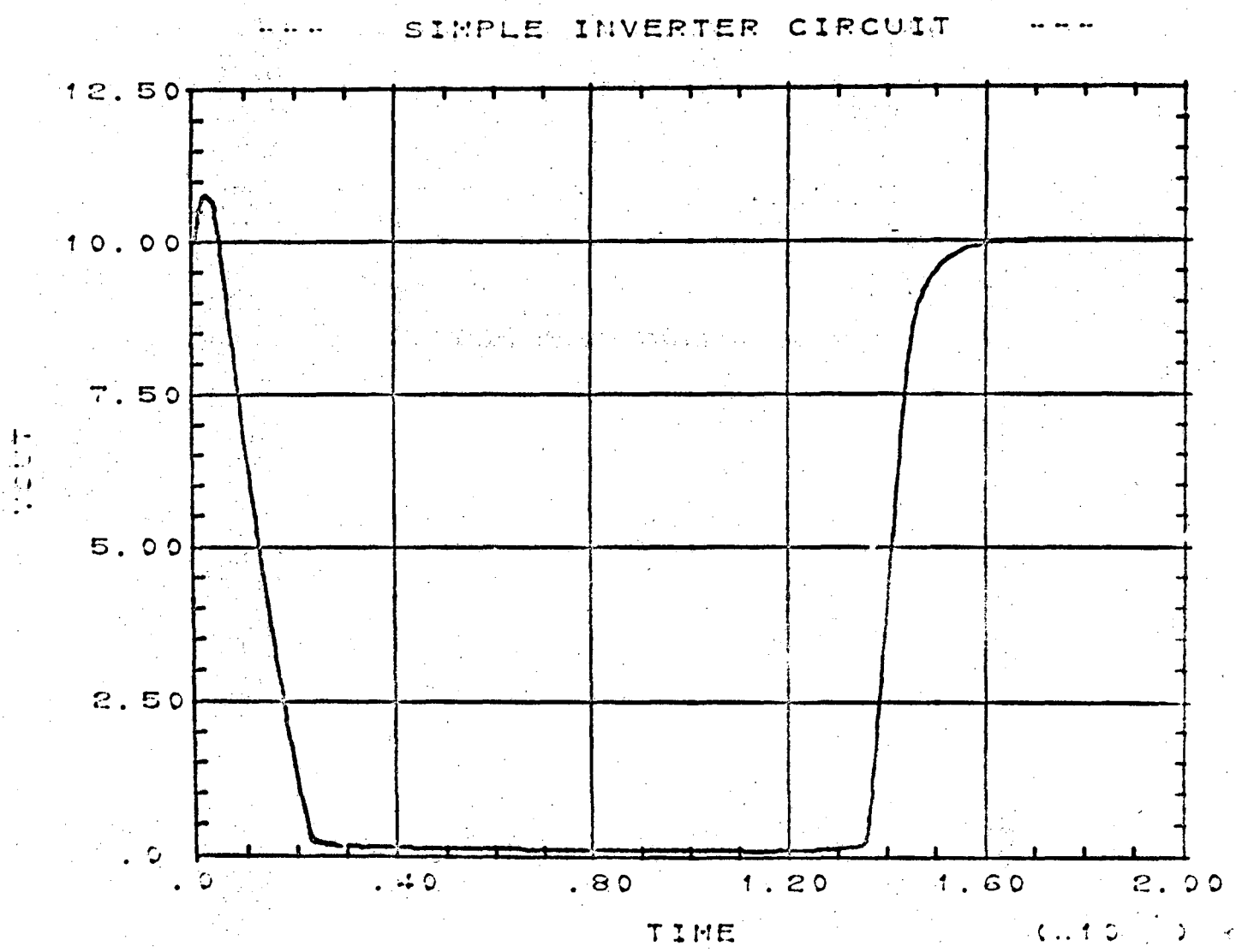


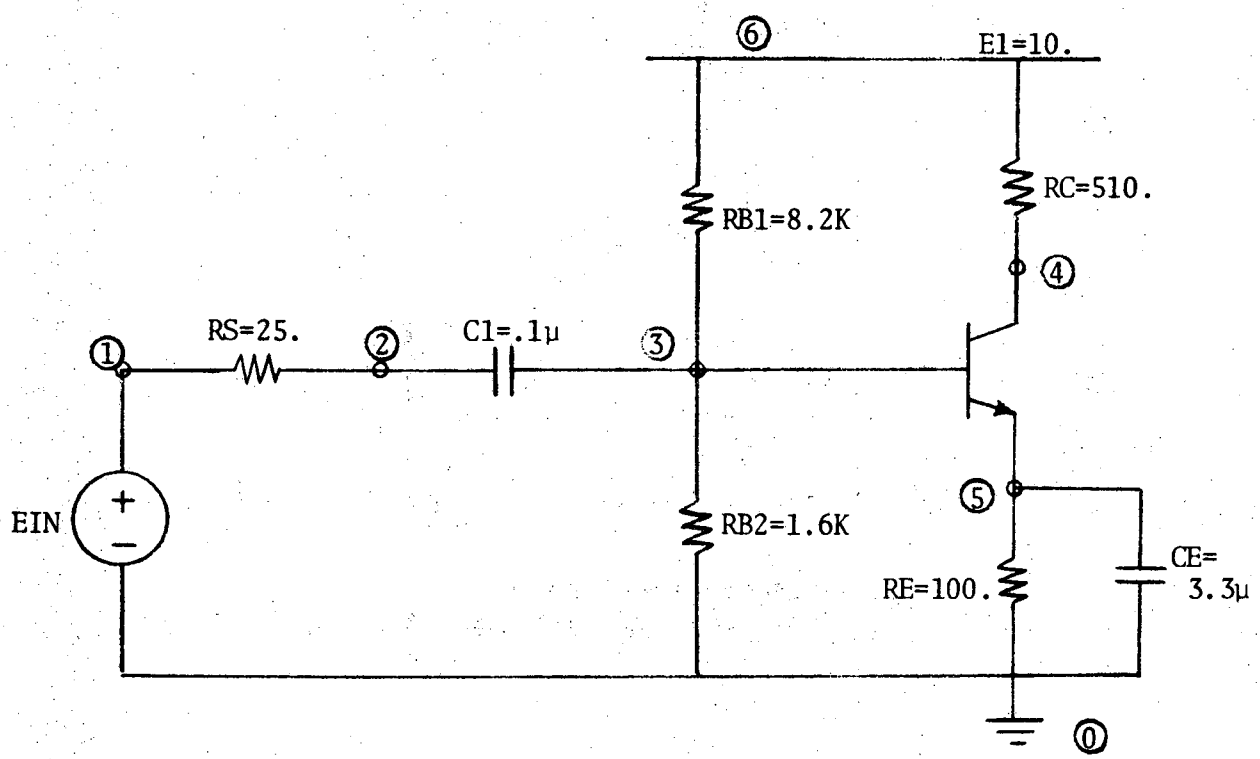
FIG. 3. SIMPLE INVERTER CIRCUIT


--- SIMPLE INVERTER CIRCUIT ---
12.50
10.00
7.50
5.00
2.50
.0
.0
.40
.80
1.20
1.60
2.00
TIME

EXAMPLE D2. INPUT DATA

AMPLIFIER CIRCUIT

```
MODEL DESCRIPTION
XSTER 2N2369 (TEMP)
RB=.025,RC=.005
AE=5.,PHIE=.9,NE=.30,IES=1.8E-11,THETAE=37.2
AC=3.7,PHIC=.9,NC=.22,ICS=5.75E-11,THETAC=34.5
ALPHAN=Q2(TBN(JE)),ALPHAI=Q2(TBI(JC))
TAUE=TTN(JE),TAUC=TTI(JC)
FUNCTIONS
Q2(A)=(A/(A+1.))
TBN =  1.,65.5,  2.,68.,  5.,69.4,  10.,65.5,  20.,27.4,  50.,27.8
TBI =  .75,.15,  1.7,.17,  2.5,.18
TTN = 1.,.36,  2.,.39,  5.,.39,  10.,.4
TTI = 1.,64.,  2.,48.5,  5.,62.
CIRCUIT DESCRIPTION
   ---   AMPLIFIER CIRCUIT   ---
ELEMENTS
EIN,0-1=T1(TIME)
E1,0-6=10
RS,1-2=.025
C1,2-3=1.E5
RB1,3-6=8.2
RB2,3-0=1.6
RC,6-4=.51
RE,5-0=.1
CE,5-0=3.3E6
T1,3-5-4=MODEL 2N2369
DEFINED PARAMETERS
PVOUT=(10.-VRC)
INITIAL CONDITIONS
VCET1=.7196, VCCT1=-4.656, VC1=-1.484, VCE=.7612
FUNCTIONS
T1=0.,0.,  10.,.014,  110.,.014,  124.,0.,  125.,0.
OUTPUTS
PVOUT(VOUT),PLOT
RUN CONTROLS
STOP TIME=200.
END
```



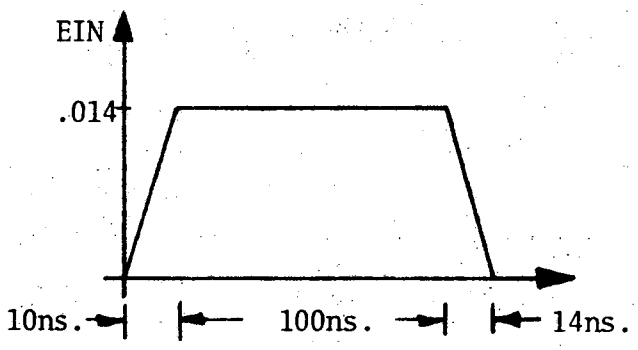
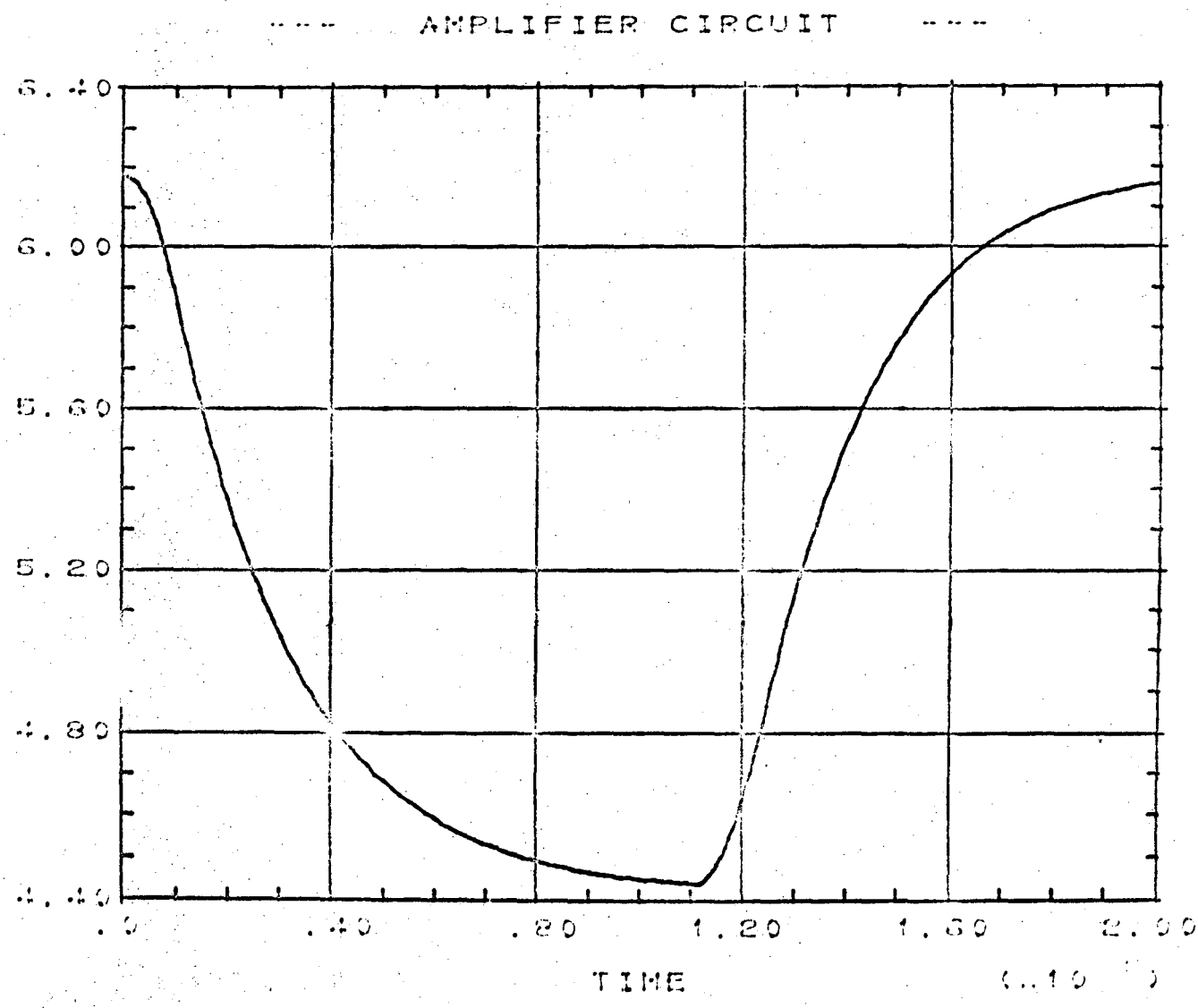


FIG 4. AMPLIFIER CIRCUIT


--- AMPLIFIER CIRCUIT ---
6.40
6.00
5.60
5.20
4.80
4.40
.0
.40
.80
1.20
1.60
2.00
TIME

EXAMPLE D3. INPUT DATA

MONOSTABLE MULTIVIBRATOR CIRCUIT

```
MODEL DESCRIPTION
XSTER 2N709 (TEMP)
RB=.048,RC=.007
AE=.96,PHIE=.9,NE=.38,IES=3.55E-11,THETAE=33.5
AC=1.9,PHIC=.9,NC=.18,ICS=8.7E-11,THETAC=30.8
ALPHAN=Q2(TBN(JE)),ALPHAI=Q2(TBI(JC))
TAUE=TTN(JE),TAUC=TTI(JC)
FUNCTIONS
Q2(A)=(A/(A+1.))
TBN =  1.,74.1,  2.,72.,  5.,60.3,  10.,44.3,  20.,21.8
TBI =  5.7,.18,  9.7,.21,  20.,.24
TTN = 1.,.105,  2.,.0988,  5.,.127,  10.,.142
TTI = 1.,31.4,  2.,30.4,  5.,31.4,  10.,28.5
CIRCUIT DESCRIPTION
   ---   MONOSTABLE MULTIVIBRATOR CIRCUIT   ---
ELEMENTS
EIN,0-6=T1(TIME)
E1,0-5=10
RF,1-4=9.1
RC1,2-5=1
RB2,3-5=10
RB,1-6=10
RC2,5-4=1
C1,2-3=33
T1,1-0-2=MODEL 2N709
T2,3-0-4=MODEL 2N709
DEFINED PARAMETERS
PVOUT=(10.-VRC2)
INITIAL CONDITIONS
VCCT1=-10.,VC1=9.3,VCET2=.7
OUTPUTS
PVOUT(VOUT),PLOT
FUNCTIONS
T1=0.,0.,  1.,3.,  101.,3.,  102.,0.,  106.,0.
RUN CONTROLS
STOP TIME=300., MAXIMUM PRINT POINTS=250.
END
```

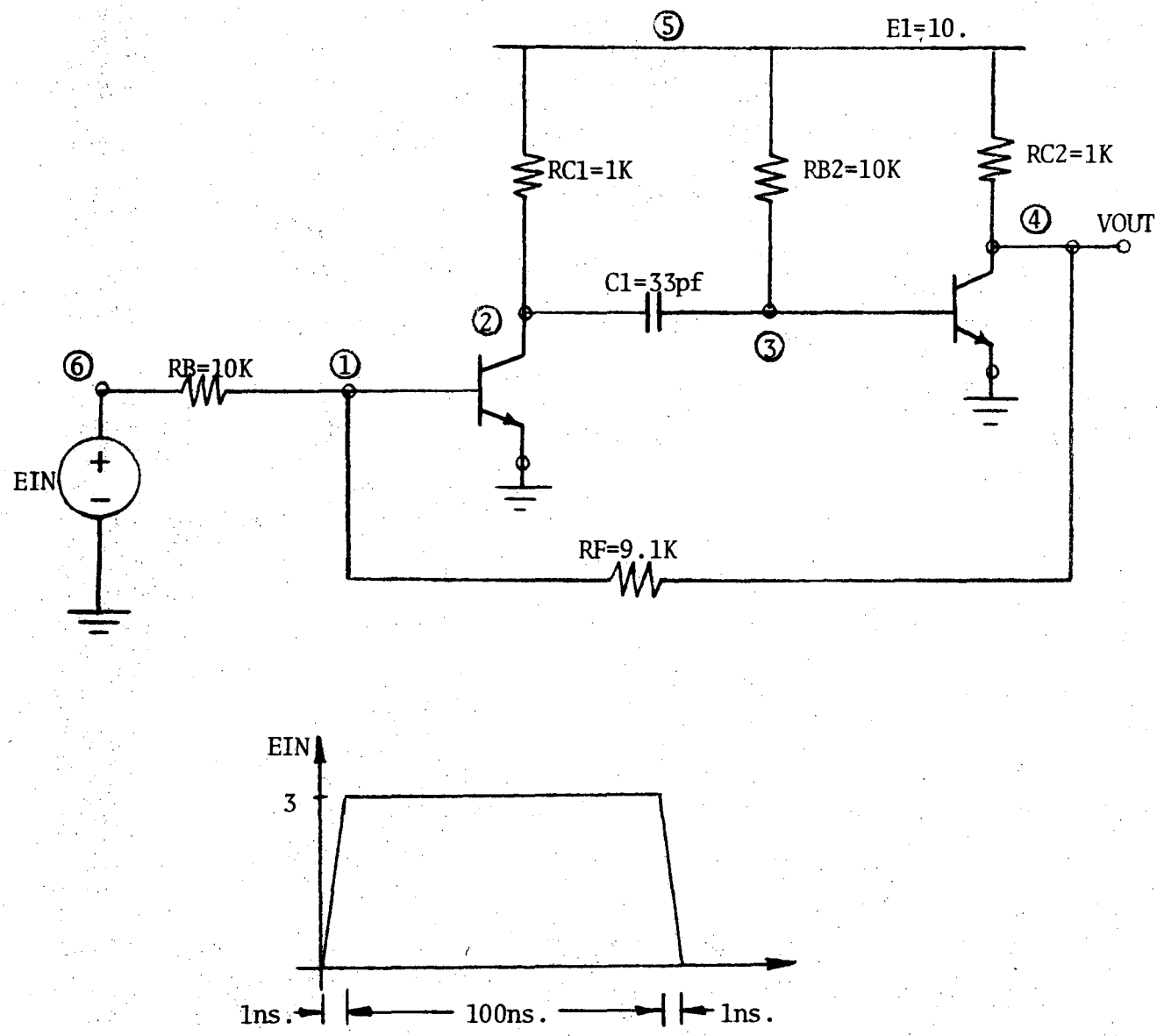
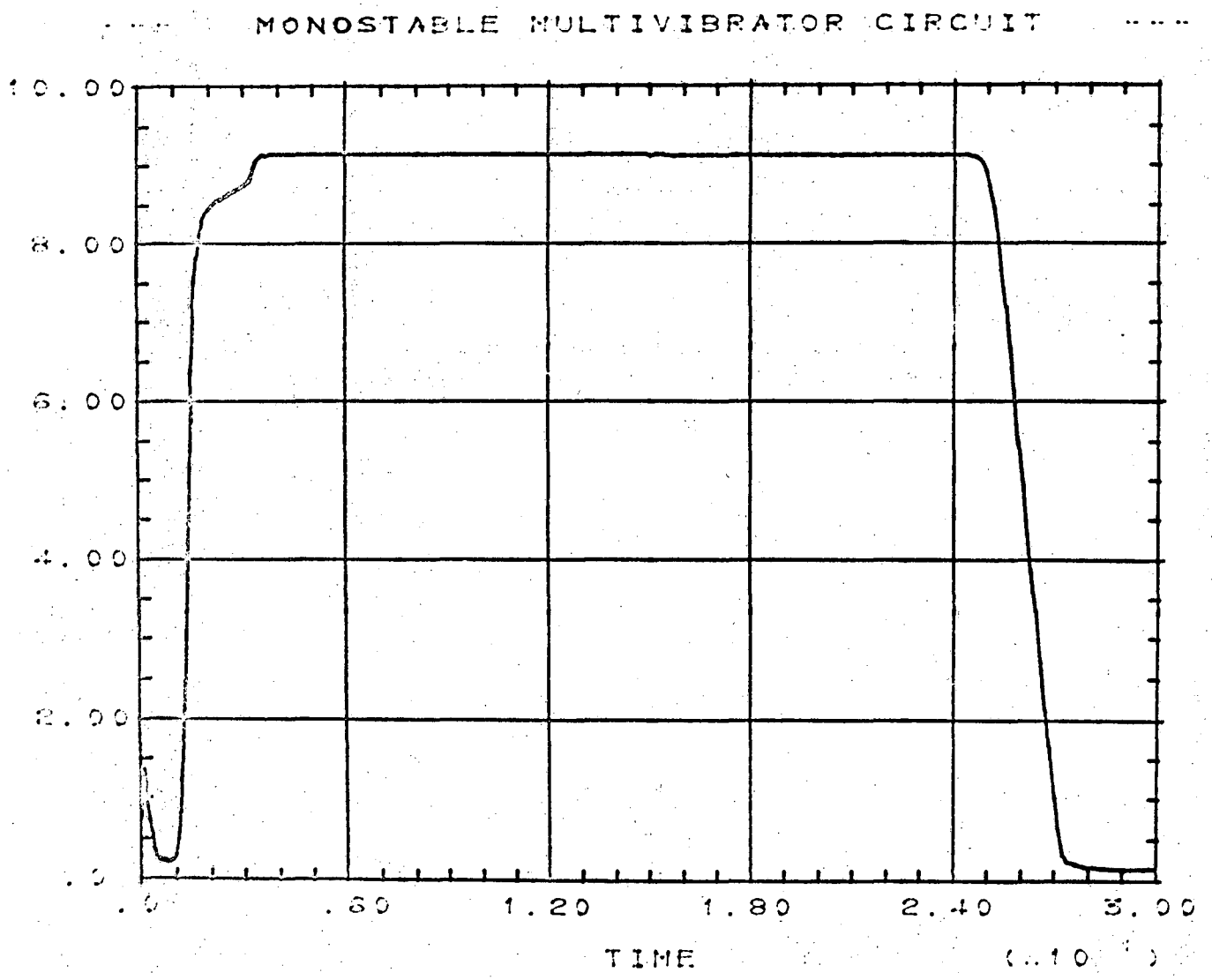


FIG. 5. MONOSTABLE MULTIVIBRATOR CIRCUIT


MONOSTABLE MULTIVIBRATOR CIRCUIT
10.00
8.00
6.00
4.00
2.00
.0
.0
.60
1.20
1.80
2.40
3.00
TIME

The running times of each circuit for both SCEPTRE and CIRCUS are shown in Table D1. below.

TABLE D1.

RUNNING TIMES FOR (in secs.)	SIMPLE INVERTER	AMPLIFIER	MONOSTABLE MULTIVIBRATOR
SCEPTRE	11.5	11.1	28.0
CIRCUS	2	2	15

These running times resulted from inputting exactly the data shown on the preceeding pages. The reader may remember the discussion on integration variables in the run controls section of Chapter 2. In it, we discussed the accuracy--computer time trade-off that was available to the user by proper adjustment of the error criteria for the numerical integration. We will try to give some perspective here as to what this trade off entails.

The monostable multivibrator circuit was run a total of seven times with larger errors allowed in the integration routine for each successive run. The computer time necessary for each run is tabulated on the following in Table D2. as a function of the MAXIMUM ABSOLUTE ERROR.

TABLE D2.

MAXIMUM ABSOLUTE ERROR	COMPUTER (cp) TIME (sec)
.005	29.5
.0075 (Preset Value)	28.0
.01	28.0
.05	23.5
.1	22.2
.5	21.4
1.	21.3

Since the MAXIMUM RELATIVE ERROR is preset to zero, the MAXIMUM ABSOLUTE ERROR represents the largest value the estimated error can go to without halving the time step size. If the error is estimated to be greater than this number, the time step is halved and more computer time is necessary for solution. Note the decrease in computer time as the allowed error increases.

LEGAL NOTICE

This report was prepared as an account of work sponsored by the United States Government. Neither the United States nor the United States Atomic Energy Commission, nor any of their employees, nor any of their contractors, subcontractors, or their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness or usefulness of any information, apparatus, product or process disclosed, or represents that its use would not infringe privately owned rights.

TECHNICAL INFORMATION DIVISION
LAWRENCE RADIATION LABORATORY
UNIVERSITY OF CALIFORNIA
BERKELEY, CALIFORNIA 94720