Appendix C

Computer programs

Many versatile computer programs are now available to the circuit designer; one such program, SPICE (Simulation Program for Integrated Circuit Electronics), written at the University of California, is freely available on the computers of most universities and polytechnics, and at the later stages of a degree course students, particularly those intending to specialise in electrical subjects, will benefit from an acquaintance with such a program.

In the introductory stages of an electrical circuits course, however, assimilation of principles is of paramount importance; the programs described and listed in this appendix have been written with this in mind – they are not intended as circuit design tools but rather to assist the student in working through the examples and problems given in the text. In particular, the program for solving simultaneous equations and the circuit calculator program will remove much of the drudgery associated with the manipulation of complex quantities in a.c. circuit problems.

The programs will run on any personal computer supporting GW BASIC or similar versions of the BASIC language allowing the use of defined functions.*

C1 Simultaneous equations

Program SIMUL solves the set of equations:

$$(a_{11}+jb_{11})x_1+(a_{12}+jb_{12})x_2+\ldots+(a_{1n}+jb_{1n})x_n=c_1\underline{/\theta_1}$$
$$(a_{21}+jb_{21})x_1+(a_{22}+jb_{22})x_2+\ldots+(a_{2n}+jb_{2n})x_n=c_2\underline{/\theta_2}$$
$$\ldots \qquad \ldots \qquad \ldots \qquad \ldots$$
$$(a_{n1}+jb_{n1})x_1+(a_{n2}+jb_{n2})x_2+\ldots+(a_{nn}+jb_{nn})x_n=c_n\underline{/\theta_n}$$

* Minor variations of the BASIC language have, as far as possible, been taken into account in writing these programs. Users of BBC BASIC should, however, note that the pseudo variable INKEY$ takes an argument, viz: INKEY$(10).

https://doi.org/10.1017/CBO9781139170093.013 Published online by Cambridge University Press

This set corresponds to the general mesh equations (B.1) given in Appendix B, or to the nodal equations of similar form. The independent variables in the right-hand column represent, correspondingly, the magnitudes and phases of the net mesh voltages in the case of the mesh equations, or the net injected nodal currents in the case of the nodal equations. The coefficients $(a_{ik}+jb_{ik})$ correspond to impedances or admittances as appropriate.

The program uses the method of Gaussian elimination and back substitution, with partial pivoting.* Real and imaginary parts of coefficients are stored in arrays AA(R,S), BB(R,S), R and S running from 1 to N where N is the number of equations. The independent variables are also stored in these arrays as AA(R,N+1), BB(R,N+1); for brevity in the program instructions these variables are also referred to as coefficients. To avoid corruption of the original data, arrays AA and BB are copied to working arrays A and B.

The dependent variables are stored in the arrays XA(R), XB(R). The program proceeds by eliminating in turn each of the variables $X(1)\ldots X(N-1)$ to produce a solution for $X(N)$. This is back-substituted into the preceding pivotal equation to yield $X(N-1)$, the procedure being repeated for all pivotal equations.

When using the program to solve the standard mesh or nodal equations, it must be remembered that mutual terms are negative (see section 2.5); this can lead to changes of sign which may cause confusion when entering data, particularly if the circuit described by the equations contains both inductive and capacitive elements. It is recommended, therefore, that the equations are written out with coefficients enclosed within brackets, the real and imaginary parts being given the appropriate sign. This is illustrated in the example given below.

An option in the program allows correction of data if a mistake is made during entry. This option also enables changes to be made to one or more circuit parameters and for the program to be rerun so that the effect of such changes may be ascertained.

Example. Use program SIMUL to solve the following set of equations (see section 5.3.5 and Fig. 5.9).

$$(24+j18)I_1+(-2-j4)I_2+(-20-j10)I_3=415/\underline{30}$$
$$(-2-j4)I_1+(24+j18)I_2+(-20-j10)I_3=415/\underline{150}$$
$$(-20-j10)I_1+(-20-j10)I_2+(60+j30)I_3=0$$

* A discussion of the underlying principles of the method can be found in: *Basic Numerical Methods* by R.E. Scraton (Edward Arnold, 1984).

Program printout:
Entry complete; check coefficients

Row 1	24	18
	−2	−4
	−20	−10
	415	30
Row 2	−2	−4
	24	18
	−20	−10
	415	150
Row 3	−20	−10
	−20	−10
	60	30
	0	0

Are coefficients correct (Y/N)? Y
Solutions are:

	MAGNITUDE	PHASE (DEG)
X(1)	21.1047266	19.7636417
X(2)	21.1047266	79.7636416
X(3)	12.1848196	49.7636416

Do you wish to change parameters (Y/N)? N

C2 Circuit calculator

The circuit calculator (program CALC) contains two separate but linked programs: program RLC, which computes the impedance of a series or parallel branch, and program CMPLX, which enables one to perform chained complex arithmetic. On entry to CMPLX, results from RLC are automatically transferred. Program RCL extends from line 10 and calls a subroutine at line 2000; program COMPLX extends from line 1010 and calls subroutines at lines 1500, 2000, 3000 and 4000. Linking between RLC and CMPLX is effected by lines 3–8. Error trapping is provided at line 5. The two programs are self-contained and may be entered in the computer and run as separate programs, in which case it will be necessary to include the dimension statement at line 4 early on in CMPLX. It is also advisable to incorporate error trapping in CMPLX to avoid loss of data on error. Other minor modifications required will be obvious to the programmer. (Note: to escape from CALC it is necessary to type ESC followed by Q.)

(a) Program RLC

This program computes the complex impedance $(R_S + jX)$ and complex admittance $(G + jB)$ of any combination of the elements R,L,C connected either all in series or all in parallel. Additionally, if both L and C are present, the frequency at which the circuit resonates is computed together with the Q-factor and half-power bandwidth. The definition of resonant frequency is that given by (3.90). When entering data, a zero (0) is used to indicate that a particular element is absent; e.g., if only L and C are present in a parallel circuit their numerical values are entered but a zero is placed against R (the program then takes into account the fact that the circuit is lossless and R infinite). Values for R,L,C and f are entered in units of ohm, henry, farad and hertz respectively. Note that for a series circuit containing resistance, the value of the series resistance R_S printed out by the program will be identical to the input value of R. However, for a parallel circuit R_S will be a function of the values of all elements present (see section 3.8).

(b) Program CMPLX

In this program complex numbers are held in seven registers designated Z1–Z7. Registers Z1 and Z2 are used for manual entry of data in either Cartesian or polar form. When transferring from program RLC, results are automatically placed in Z1. These registers may also be used for temporary storage of numbers during calculations. Registers Z3–Z5 are used for permanent storage, while the results of arithmetical operations are displayed in registers Z6 or Z7, the default being Z6. Operations may be carried out using any two registers as operands, e.g., the instruction 6*7 multiplies the number in Z6 by that in Z7 placing the result in Z6; the instruction 6*77 places the result in Z7. Data is, of course, overwritten in Z6 or Z7 so that these registers should not be used as operands if their contents are required for further calculation. A number in any register may be placed in any other register using the equality operator, e.g., the instruction 3 = 1 sets the number in Z3 equal to the number in Z1. Reciprocals are obtained by means of the operator R: instruction 4R47, for example, takes the inverse of the number in Z4 and places it in Z7.

Example. The printouts given below illustrate the use of CALC for solving the a.c. circuit problem given in section 3.7. The three impedances in the circuit (designated Z_1,Z_2,Z_3 in section 3.7) have been calculated using RLC, and these are displayed in registers Z3, Z4 and Z5. The impedance of the two branches connected in parallel $(Z_1//Z_2)$ has been computed (using the 'product-over-sum rule') and added to the series impedance Z_3 to give

the total impedance of the circuit; this is displayed in register Z7. The applied circuit voltage has been entered manually via register Z1 and, finally, the operation 1/7 has produced in Z6 the value of the main circuit current.

Program printout:

PROGRAM RLC
Enter R,L,C,F (if element is ABSENT enter a ZERO)

R(ohm) = ?2
L(H) = ?15.9E—3
C(F) = ?0
F(Hz) = ?50

Select circuit: series(S) or parallel(P) S

Series circuit parameters are:

Rs(ohm) = 2
X(ohm) = 4.99513232
Z(ohm) = 5.38064558
AZ(deg) = 68.17934
G(S) = 6.90814147E − 2
B(S) = −0.172535404
Y(S) = 0.185851304
AY(deg) = −68.17934

Product LC = 0: fo,Q,Bw undefined

Select prog. RLC(R) or prog. CMPLX(X) X

PROGRAM CMPLX
Result of Z(1) − (/) − Z(7) − > Z(6) is:

	REAL	IMAGINARY	MAGNITUDE	ANGLE(DEG)
ENTER & STORE				
Z(1)	240	0	240	0
Z(2)	0	0	0	0
STORE				
Z(3)	6	20.9858389	21.8267138	74.0443918
Z(4)	30	−7.99773583	31.0477661	−14.9273799
Z(5)	2	4.99513232	5.38064558	68.17934

https://doi.org/10.1017/CBO9781139170093.013 Published online by Cambridge University Press

RESULT

Z6	7.4013709	−7.63636498	10.6345833	−45.8952859
Z7	15.7066099	16.205296	22.5678801	45.8952859

Select: Operation(O);Entry(E);Program RLC(R)

C3 Tee-Pi transformation

Given the impedances ZA,ZB,ZC connected in a T (or star) configuration, or the impedances Z1,Z2,Z3 connected in a π (or delta) configuration, program TEE-PI computes the transformation given by the relations (8.25). These relations are printed out on the VDU together with schematics of the circuit configurations. The operation of this program is self-explanatory.

C4 Roots of polynomials

Program 'ROOTS' finds the roots of a polynomial of the form

$$A(m+1)x^m + A(m)x^{m-1} + \ldots + A(2)x + A(1) = 0$$

where m is the degree of the polynomial and the coefficients A may be complex.* The real and imaginary parts of the coefficients are retained in arrays A(K) and B(K) respectively (K running from 1 to M+1). Values held in these arrays are transferred to working arrays AR(K) and AI(K) as necessary. Roots are found by Laguerre's method which operates iteratively. For a trial value of X, a correction term DX is computed which is a function of the value of the polynomial and its first and second derivative; $X-DX$ then becomes the next trial value, the procedure being repeated until DX is sufficiently small. After each root is found, forward deflation is used to reduce the degree of the polynomial by one, the next root is then found for this new polynomial. Finally, when all roots have been found, their values are refined (polished), again using Laguerre's method, by insertion, in turn, into the unmodified polynomial.

The coefficients of polynomials arising in electrical circuit theory are always real, consequently, the imaginary part of each coefficient must be set to zero. Alternatively, for the purposes of the examples in this book, reference to array B(K) in line 80 of the program may be omitted.

* This program is a modified version of a program, written in FORTRAN, given in *Numerical Recipes* by W.H. Press, B.P. Flannery, S.A. Teukolsky and W.T. Vetterling, (Cambridge University Press, 1986).

Example. Denominator polynomial for a fourth-order Butterworth filter (see section 6.11).

$$s^4 + 2.613s^3 + 3.414s^2 + 2.613s + 1 = 0$$

Program printout:

PROGRAM ROOTS

Degree of polynomial: M? 4

Enter coefficients starting with A(M + 1)

A(5)(Real,Imag) = ?1,0
A(4)(Real,Imag) = ?2.613,0
A(3)(Real, Imag) = ?3.414,0
A(2)(Real,Imag) = ?2.613,0
A(1)(Real,Imag) = ?1,0

Entry complete; coefficients OK(Y/N)? Y

Roots of polynomial are:

REAL	IMAGINARY	MODULUS	ARGUMENT(DEG)
−0.382629302	0.923901953	1	112.496643
−0.382629302	−0.923901952	1	−112.496643
−0.923870697	−0.382704758	0.999999998	−157.498677
−0.9238707	0.382704758	1	157.498677

Program SIMUL

```
10 CLS:PRINT "PROGRAM SIMUL":PRINT
20 REM SOLUTION OF SIMULTANEOUS EQUATIONS
30 REM BY GAUSSIAN ELIMINATION
40 REM*DATA INPUT*
50 INPUT "Number of equations: N ";N: PRINT
60 DIM AA(N,N+1),BB(N,N+1),A(N,N+1),B(N,N+1),XA(N),XB(N):PRINT
70 PRINT "Enter coefficients":PRINT
80 FOR R=1 TO N
90 FOR S=1 TO N
100 PRINT"A(";R;",";S;") = ";:INPUT AA(R,S)
110 PRINT"B(";R;",";S;") = ";:INPUT BB(R,S)
120 NEXT S
130 PRINT"C(";R;") = ";:INPUT AA(R,N+1)
140 PRINT"THETA(";R;") = ";:INPUT BB(R,N+1): PRINT
150 NEXT R
160 CLS:PRINT:PRINT"Entry complete; check coefficients":PRINT
170 FOR R=1 TO N
180 PRINT"Row";R;
190 FOR S=1 TO N+1
200 PRINT TAB(6); AA(R,S) TAB(20); BB(R,S)
210 NEXT S: PRINT
220 NEXT R
230 PRINT:PRINT"Are coefficients correct (Y/N)? ";:GOSUB 1500
240 IF AN<>1 AND AN<>2 GOTO 230
250 IF AN=1 GOTO 300
```

```
260 PRINT:INPUT"Enter Row,Column,Values(A,B or C,THETA)";R,S,VA,VB
270 IF R<=N AND R>=1 AND S<=N+1 AND S>=1 GOTO 290
280 PRINT:PRINT"Incorrect row and/or column numbers":GOTO 260
290 AA(R,S)=VA:BB(R,S)=VB:GOTO 170
300 REM*FUNCTIONS FOR COMPLEX ARITHMETIC*
310 DEF FNCABS(D,E)=SQR(D*D+E*E)
320 DEF FNPA(D,E,F,G)=D*F-E*G
330 DEF FNPB(D,E,F,G)=D*G+E*F
340 DEF FNQA(D,E,F,G)=(D*F+E*G)/(F*F+G*G)
350 DEF FNQB(D,E,F,G)=(E*F-D*G)/(F*F+G*G)
360 REM*COPY COEFFS TO WORKING ARRAYS*
370 FOR R=1 TO N
380 FOR S=1 TO N+1
390 A(R,S)=AA(R,S):B(R,S)=BB(R,S)
400 NEXT S
410 NEXT R
420 REM*CONVERT (C,THETA) VALUES TO CARTESIAN
430 PIE=3.14159265#
440 FOR R=1 TO N
450 AC=(B(R,N+1))*PIE/180
460 A=A(R,N+1)*COS(AC)
470 B(R,N+1)=A(R,N+1)*SIN(AC)
480 A(R,N+1)=A
490 NEXT R
500 REM*ELIMINATION WITH PARTIAL PIVOTING*
510 FOR Z=1 TO N-1
520 W=0
530 FOR R=Z TO N
540 MA=FNCABS(A(R,Z),B(R,Z))
550 IF MA>W THEN U=R:W=MA
560 NEXT R
570 FOR S=Z TO N+1
580 P=A(U,S):A(U,S)=A(Z,S):A(Z,S)=P
590 P=B(U,S):B(U,S)=B(Z,S):B(Z,S)=P
600 NEXT S
610 FOR R=Z+1 TO N
620 PA=FNQA(A(R,Z),B(R,Z),A(Z,Z),B(Z,Z))
630 PB=FNQB(A(R,Z),B(R,Z),A(Z,Z),B(Z,Z))
640 FOR S=Z+1 TO N+1
650 A(R,S)=A(R,S)-FNPA(PA,PB,A(Z,S),B(Z,S))
660 B(R,S)=B(R,S)-FNPB(PA,PB,A(Z,S),B(Z,S))
670 NEXT S
680 NEXT R
690 NEXT Z
700 REM*BACK SUBSTITUTION*
710 FOR R=N TO 1 STEP-1
720 PA=A(R,N+1)
730 PB=B(R,N+1)
740 IF R=N GOTO 790
750 FOR S=R+1 TO N
760 PA=PA-FNPA(A(R,S),B(R,S),XA(S),XB(S))
770 PB=PB-FNPB(A(R,S),B(R,S),XA(S),XB(S))
780 NEXT S
790 XA(R)=FNQA(PA,PB,A(R,R),B(R,R))
800 XB(R)=FNQB(PA,PB,A(R,R),B(R,R))
810 NEXT R
820 REM* CONVERT RESULTS TO POLAR FORM AND PRINT*
830 PRINT:PRINT"Solutions are:":PRINT
840 PRINT TAB(8)"MAGNITUDE" TAB(23)"PHASE(DEG)":PRINT
850 FOR R=1 TO N
860 U=XA(R):V=XB(R):GOSUB 2000
870 PRINT"X(";R;")"TAB(8);MZ TAB(23);AZ
880 NEXT R
890 PRINT:PRINT"Do you wish to change parameters (Y/N)? ";:GOSUB 1500
900 IF AN<>1 AND AN<>2 GOTO 890
910 IF AN=1 GOTO 170
920 END
1500 REM SUBROUTINE: OPTIONS
1510 Q$=INKEY$:IF Q$<>"" GOTO 1510
1520 Q$=INKEY$:IF Q$="" GOTO 1520
1530 IF Q$>"a" THEN Q$=CHR$(ASC(Q$)-32)
1540 PRINT Q$:AN=0
1550 IF Q$="Y" THEN AN=1
1560 IF Q$="N" THEN AN=2
1570 RETURN
2000 REM SUBROUTINE: POLAR CONVERSION
2010 MZ=FNCABS(U,V)
2020 IF ABS(U)<1E-36 GOTO 2080
```

```
2030 AZ=(ATN(V/U))*180/PIE
2040 IF AZ=0 AND U<0 THEN AZ=180
2050 IF U<0 AND V>0 THEN AZ=180+AZ
2060 IF U<0 AND V<0 THEN AZ=AZ-180
2070 RETURN
2080 IF V>0 THEN AZ=90 ELSE AZ=-90
2090 IF U=0 AND V=0 THEN AZ=0
2100 RETURN
```

Program CALC

```
3 CLS:PRINT"PROGRAM CALC"
4 DIM R(7),X(7),M(7),A(7)
5 ON ERROR GOTO 999
6 PRINT:PRINT"Select: prog.RLC(R); prog.CMPLX(X) or quit(Q) ";:GOSUB 1500
7 IF AN=1 GOTO 1000
8 IF AN<>2 GOTO 6
10 CLS:PRINT"PROGRAM RLC":PRINT
20 REM*PARAMETERS OF RLC SERIES/PARALLEL CIRCUITS*
30 PRINT"Enter R,L,C,F (if element is ABSENT enter a ZERO)":PRINT
40 INPUT"R(ohm)= ";R
50 INPUT"L(H)= "; L
60 INPUT"C(F)= ";C
70 INPUT"F(Hz)= ";F
80 PIE=3.14159265#
90 W=2*PIE*F
100 F0=0
110 IF L>0 AND C>0 THEN F0=1/(2*PIE*SQR(L*C))
120 W0=2*PIE*F0
130 REM*TEST FOR RESONANCE*
140 RE=0
150 IF F0>0 GOTO 160 ELSE GOTO 170
160 IF ABS((F-F0)/F0)<.000001 THEN RE=1
170 DEF FNR(U,V)=U/(U*U+V*V)
180 DEF FNI(U,V)=-V/(U*U+V*V)
190 PRINT:PRINT"Select circuit: series(S) or parallel(P) ";
200 QU$=INKEY$:IF QU$<>"" GOTO 200
210 QU$=INKEY$:IF QU$="" GOTO 210
220 PRINT QU$:IF QU$>="a"THEN QU$=CHR$(ASC(QU$)-32)
230 IF QU$<>"S" AND QU$<>"P" GOTO 190
240 IF QU$="P" GOTO 450
250 REM*COMPUTE PARAMETERS FOR SERIES CIRCUIT*
260 IF C=0 GOTO 380
270 IF RE=1 AND R=0 GOTO 280 ELSE GOTO 290
280 PRINT"Resonance(f=fo):Q infinite,Z zero for Rs=0":END
290 X=W*L-1/(W*C)
300 U=R:V=X:GOSUB 2000
310 Z=M:AZ=A
320 G=FNR(R,X):B=FNI(R,X)
330 U=G:V=B:GOSUB 2000
340 Y=M:AY=A
350 IF R=0 OR F0=0 GOTO 670
360 Q=W0*L/R:BW=F0/Q
370 GOTO 670
380 X=W*L
390 U=R:V=X:GOSUB 2000
400 Z=M:AZ=A
410 G=FNR(R,X):B=FNI(R,X)
420 U=G:V=B:GOSUB 2000
430 Y=M:AY=A
440 GOTO 670
450 REM*COMPUTE PARAMETERS FOR PARALLEL CIRCUIT*
460 IF RE=1 AND R=0 GOTO 470 ELSE GOTO 490
470 PRINT:PRINT"Resonance(f=fo):Q and Z infinite for G=1/R=0"
480 END
490 IF R=0 THEN G=0 ELSE G=1/R
500 IF L=0 GOTO 600
510 B=W*C-1/(W*L)
520 U=G:V=B:GOSUB 2000
530 Y=M:AY=A
540 RS=FNR(G,B):X=FNI(G,B)
550 U=RS:V=X:GOSUB 2000
560 Z=M:AZ=A
570 IF R=0 OR F0=0 GOTO 670
580 Q=R/(W0*L):BW=F0/Q
590 GOTO 670
600 B=W*C
```

https://doi.org/10.1017/CBO9781139170093.013 Published online by Cambridge University Press

```
610 U=G:V=B:GOSUB 2000
620 Y=M:AY=A
630 RS=FNR(G,B):X=FNI(G,B)
640 U=RS:V=X:GOSUB 2000
650 Z=M:AZ=A
660 REM*PRINT RESULTS*
670 PRINT:IF QU$="S" THEN PRINT"Series circuit parameters are:":PRINT
680 IF QU$="P" THEN PRINT"Parallel circuit parameters are:":PRINT
690 IF QU$="S" THEN RS=R
700 PRINT"Rs(ohm)= ";RS
710 PRINT"X(ohm)= ";X
720 PRINT"Z(ohm)= ";Z
730 PRINT"AZ(deg)= ";AZ
740 PRINT"G(S)= ";G
750 PRINT"B(S)= ";B
760 PRINT"Y(S)= ";Y
770 PRINT"AY(deg)= ";AY
780 IF F0>0 AND R>0 GOTO 790 ELSE GOTO 820
790 PRINT"fo(Hz)= ";F0
800 PRINT "Q= ";Q
810 PRINT "Bw= ";BW:GOTO 860
820 IF F0>0 AND R=0 GOTO 830 ELSE GOTO 850
830 PRINT"fo(Hz)= ";F0
840 PRINT:PRINT"Lossless  circuit:Q infinite,Bw zero":GOTO 860
850 PRINT:PRINT"Product LC=0:fo,Q,Bw undefined"
860 R(1)=RS:X(1)=X:M(1)=Z:A(1)=AZ
870 GOTO 6
880 END
999 CLS:PRINT:PRINT"ERROR NUMBER ";ERR:RESUME 6
1000 REM*COMPLEX ARITHMETIC*
1010 CLS:PRINT"PROGRAM CMPLX"
1020 PIE=3.14159265#
1030 GOSUB 4000
1040 PRINT:PRINT"Select:Operation(O);Entry(E);Prog.RLC(R);Quit(Q)";:GOSUB 1500
1050 IF AN<2 OR AN>4 GOTO 1040
1060 ON AN-1 GOTO 10,1120,1070
1070 CLS:GOSUB 4000:GOSUB 3000
1080 IF I<>7 THEN I=6
1090 IF O$="=" THEN I=G
1100 CLS:PRINT"Result of Z(";G;")-(";O$;")-Z(";H;")->Z(";I;") is:":PRINT
1110 GOSUB 4000:GOTO 1040
1120 CLS:GOSUB 4000
1130 PRINT:PRINT"Entry in Cartesian(C) or Polar(P)?";:GOSUB 1500
1140 IF AN<>5 AND AN<>6 THEN GOTO 1130
1150 IF AN=6 GOTO 1200
1160 PRINT:FOR J=1 TO 2
1170 PRINT"Enter Z";J;"(Real,Imag)";:INPUT R(J),X(J)
1180 U=R(J):V=X(J):GOSUB 2000:M(J)=M:A(J)=A
1190 NEXT J:GOTO 1250
1200 PRINT:FOR J=1 TO 2
1210 PRINT"Enter Z";J;"(Mag,Angle(deg))";:INPUT M(J),A(J)
1220 AA=A(J)*PIE/180
1230 R(J)=M(J)*COS(AA):X(J)=M(J)*SIN(AA)
1240 NEXT J
1250 CLS:GOSUB 4000
1260 GOTO 1040
1500 REM SUBROUTINE: OPTIONS
1510 Q$=INKEY$:IF Q$<>"" GOTO 1510
1520 Q$=INKEY$:IF Q$="" GOTO 1520
1530 IF Q$>="a" THEN Q$=CHR$(ASC(Q$)-32)
1540 PRINT Q$:AN=0
1550 IF Q$="X" THEN AN=1
1560 IF Q$="R" THEN AN=2
1570 IF Q$="E" THEN AN=3
1580 IF Q$="O" THEN AN=4
1590 IF Q$="C" THEN AN=5
1600 IF Q$="P" THEN AN=6
1610 IF Q$="Q" GOTO 9999
1620 RETURN
2000 REM SUBROUTINE: POLAR CONVERSION
2010 M=SQR(U*U+V*V)
2020 IF ABS(U)<1E-36 GOTO 2080
2030 A=(ATN(V/U))*180/PIE
2040 IF A=0 AND U<0 THEN A=180
2050 IF U<0 AND V>0 THEN A=180+A
2060 IF U<0 AND V<0 THEN A=A-180
2070 RETURN
```

```
2080 IF V>0 THEN A=90 ELSE A=-90
2090 IF U=0 AND V=0 THEN A=0
2100 RETURN
3000 REM SUBROUTINE: OPERATIONS
3010 PRINT:PRINT"Operations:";
3020 PRINT"+-*/= and R"
3030 I=0:PRINT
3040 INPUT"Z(?)-(OP?)-Z(?)->Z(6-7)";X$
3050 IF LEN(X$)<3 OR LEN(X$)>4 GOTO 3040
3060 IF LEN(X$)=3 GOTO 3090
3070 I=VAL(RIGHT$(X$,1)):IF I<>7 GOTO 3040
3080 X$=LEFT$(X$,3)
3090 G=VAL(LEFT$(X$,1)):H=VAL(RIGHT$(X$,1))
3100 O$=MID$(X$,2,1)
3110 IF G<1 OR G>7 GOTO 3040
3120 IF H<1 OR H>7 GOTO 3040
3130 IF O$<>"+" AND O$<>"-" GOTO 3140 ELSE GOTO 3160
3140 IF O$<>"*" AND O$<>"/" GOTO 3150 ELSE GOTO 3160
3150 IF O$<>"R" AND O$<>"=" GOTO 3040 ELSE GOTO 3160
3160 REM*SELECT OPERANDS*
3170 FOR J=1 TO 7
3180 IF G=J THEN P=R(J)
3190 IF G=J THEN Q=X(J)
3200 IF H=J THEN S=R(J)
3210 IF H=J THEN T=X(J)
3220 NEXT J
3230 REM*EXECUTE OPERATOR*
3240 IF O$="+" GOTO 3250 ELSE GOTO 3260
3250 U=P+S:V=Q+T:GOSUB 2000:GOTO 3430
3260 IF O$="-" GOTO 3270 ELSE GOTO 3280
3270 U=P-S:V=Q-T: GOSUB 2000:GOTO 3430
3280 IF O$="*" GOTO 3290 ELSE GOTO 3300
3290 U=P*S-Q*T:V=P*T+Q*S: GOSUB 2000:GOTO 3430
3300 IF O$="/" GOTO 3310 ELSE GOTO 3330
3310 U=(P*S+Q*T)/(S*S+T*T)
3320 V=(Q*S-P*T)/(S*S+T*T):GOSUB 2000:GOTO 3430
3330 IF O$="R" GOTO 3340 ELSE GOTO 3360
3340 U=P/(P*P+Q*Q)
3350 V=-Q/(P*P+Q*Q):GOSUB 2000:GOTO 3430
3360 IF O$<>"=" GOTO 3430
3370 FOR J=1 TO 7
3380 IF G<>J THEN GOTO 3420
3390 R(J)=S:X(J)=T
3400 U=S:V=T:GOSUB 2000
3410 M(J)=M:A(J)=A
3420 NEXT J:RETURN
3430 IF I=7 GOTO 3450
3440 R(6)=U:X(6)=V:M(6)=M:A(6)=A:RETURN
3450 R(7)=U:X(7)=V:M(7)=M:A(7)=A
3460 RETURN
4000 REM SUBROUTINE:DISPLAY
4010 PRINT TAB(5)"REAL" TAB(18)"IMAGINARY";
4020 PRINT TAB(31)"MAGNITUDE" TAB(44)"ANGLE(DEG)"
4030 FOR J%=1 TO 7
4040 IF J%=1 THEN PRINT"ENTER & STORE"
4050 IF J%=3 THEN PRINT"STORE"
4060 IF J%=6 THEN PRINT"RESULT"
4070 PRINT"Z";J% TAB(5);R(J%) TAB(18);X(J%) TAB(31);M(J%) TAB(44);A(J%)
4080 NEXT
4090 RETURN
9999 PRINT"Quitting prog.CALC"
```

Program TEE-PI

```
10 CLS:PRINT"PROGRAM TEE-PI":PRINT
20 DIM Z$(6),R(6),X(6),M(6),A(6)
30 Z$(1)="Z1":Z$(2)="Z2":Z$(3)="Z3"
40 Z$(4)="ZA":Z$(5)="ZB":Z$(6)="ZC"
50 DEF FNR(P,Q,R,S,U,V)=(U*P*R-U*Q*S+V*P*S+V*Q*R)/(U*U+V*V)
60 DEF FNI(P,Q,R,S,U,V)=(U*P*S+U*Q*R-V*P*R+V*Q*S)/(U*U+V*V)
70 GOSUB 1000
80 PRINT "ZA=Z1*Z2/(Z1+Z2+Z3)"TAB(21)"Z1=ZA+ZC+ZA*ZC/ZB"
90 PRINT "ZB=Z2*Z3/(Z1+Z2+Z3)"TAB(21)"Z2=ZB+ZA+ZB*ZA/ZC"
100 PRINT "ZC=Z3*Z1/(Z1+Z2+Z3)"TAB(21)"Z3=ZC+ZB+ZC*ZB/ZA":PRINT
110 PRINT "Select:TEE-to-PI(T) or PI-to-TEE(P) ";
120 Q$=INKEY$:IF Q$<>"" GOTO 120
130 Q$=INKEY$:IF Q$="" GOTO 130
```

https://doi.org/10.1017/CBO9781139170093.013 Published online by Cambridge University Press

```
140 PRINT Q$:IF Q$>="a" THEN Q$=CHR$(ASC(Q$)-32)
150 IF Q$<>"T" AND Q$<>"P" GOTO 110
160 IF Q$="P" GOTO 300
170 PRINT:PRINT"Enter TEE-circuit values":PRINT
180 FOR J=4 TO 6
190 PRINT Z$(J);"(Real,Imag)";:INPUT R(J),X(J)
200 NEXT J
210 REM*COMPUTE PI-IMPEDANCES*
220 R(1)=R(4)+R(6)+FNR(R(4),X(4),R(6),X(6),R(5),X(5))
230 X(1)=X(4)+X(6)+FNI(R(4),X(4),R(6),X(6),R(5),X(5))
240 R(2)=R(5)+R(4)+FNR(R(5),X(5),R(4),X(4),R(6),X(6))
250 X(2)=X(5)+X(4)+FNI(R(5),X(5),R(4),X(4),R(6),X(6))
260 R(3)=R(6)+R(5)+FNR(R(6),X(6),R(5),X(5),R(4),X(4))
270 X(3)=X(6)+X(5)+FNI(R(6),X(6),R(5),X(5),R(4),X(4))
280 GOSUB 2000
290 CLS:GOSUB 1000:GOTO 440
300 PRINT:PRINT"Enter PI-circuit values":PRINT
310 FOR J=1 TO 3
320 PRINT Z$(J);"(Real,Imag)";:INPUT R(J),X(J)
330 NEXT J
340 REM*COMPUTE TEE-IMPEDANCES*
350 D=R(1)+R(2)+R(3):E=X(1)+X(2)+X(3)
360 R(4)=FNR(R(1),X(1),R(2),X(2),D,E)
370 X(4)=FNI(R(1),X(1),R(2),X(2),D,E)
380 R(5)=FNR(R(2),X(2),R(3),X(3),D,E)
390 X(5)=FNI(R(2),X(2),R(3),X(3),D,E)
400 R(6)=FNR(R(3),X(3),R(1),X(1),D,E)
410 X(6)=FNI(R(3),X(3),R(1),X(1),D,E)
420 GOSUB 2000
430 CLS:GOSUB 1000
440 REM*PRINT RESULTS*
450 PRINT TAB(5)"REAL" TAB(18)"IMAGINARY";
460 PRINT TAB(31)"MAGNITUDE" TAB(44)"ANGLE(DEG)"
470 IF Q$="T" GOTO 540
480 PRINT:PRINT"FOR PI-IMPEDANCES:":PRINT
490 FOR J=1 TO 6
500 IF J=4 THEN PRINT:PRINT"TEE-IMPEDANCES ARE:":PRINT
510 GOSUB 900
520 NEXT J
530 END
540 PRINT:PRINT"FOR TEE-IMPEDANCES:":PRINT
550 FOR J=4 TO 6
560 GOSUB 900
570 NEXT J
580 PRINT:PRINT"PI-IMPEDANCES ARE:":PRINT
590 FOR J=1 TO 3
600 GOSUB 900
610 NEXT J
620 END
900 REM SUBROUTINE: PRINT VALUES
910 PRINT Z$(J) TAB(5);R(J) TAB(18);X(J);
920 PRINT TAB(31);M(J) TAB(44);A(J)
930 RETURN
1000 REM SUBROUTINE: DISPLAY CIRCUIT SCHEMATICS
1010 PRINT TAB(5)"TEE" TAB(26)"PI":PRINT
1020 PRINT"**ZA*****ZC**" TAB(21)"*****Z1*****"
1030 PRINT TAB(6)"*"TAB(23)"*"TAB(30)"*"
1040 PRINT TAB(6)"*"TAB(23)"*"TAB(30)"*"
1050 PRINT TAB(6)"ZB"TAB(13)"<------>"TAB(23)"Z2"TAB(30)"Z3"
1060 PRINT TAB(6)"*"TAB(23)"*"TAB(30)"*"
1070 PRINT TAB(6)"*"TAB(23)"*"TAB(30)"*":PRINT
1080 RETURN
2000 REM SUBROUTINE: CARTESIAN/POLAR CONVERSION
2010 PIE=3.14159265#:FOR J=1 TO 6
2020 M(J)=SQR(R(J)*R(J)+X(J)*X(J))
2030 IF ABS(R(J))<1E-36 GOTO 2050
2040 A(J)=(ATN(X(J)/R(J)))*180/PIE:GOTO 2060
2050 IF X(J)>0 THEN A(J)=90 ELSE A(J)=-90
2060 NEXT J
2070 RETURN
```

Program ROOTS

```
10 CLS:PRINT"PROGRAM ROOTS":PRINT
20 REM*EVALUATES ROOTS OF POLYNOMIAL OF DEGREE M*
30 REM*INPUT DATA*
40 INPUT"Degree of polynomial: M";M
50 DIM A(M+1),B(M+1),AR(M+1),AI(M+1),ROOTR(M),ROOTI(M)
60 PRINT:PRINT"Enter coefficients starting with A(M+1)":PRINT
70 FOR K=M+1 TO 1 STEP-1
80 PRINT "A(";K;")(Real,Imag) = ";:INPUT A(K),B(K)
90 NEXT K
100 PRINT:PRINT"Entry complete; coefficients OK(Y/N)? ";
110 QU$=INKEY$:IF QU$<>"" GOTO 110
120 QU$=INKEY$:IF QU$="" GOTO 120
130 PRINT QU$:IF QU$>="a" THEN QU$=CHR$(ASC(QU$)-32)
140 IF QU$<>"Y" AND QU$<>"N" GOTO 100
150 IF QU$="N" GOTO 60
160 REM*COPY COEFFS TO WORKING ARRAYS*
170 FOR K=1 TO M+1
180 AR(K)=A(K): AI(K)=B(K)
190 NEXT K
200 REM*FUNCTIONS FOR COMPLEX ARITHMETIC*
210 DEF FNCABS(P,Q)=SQR(P*P+Q*Q)
220 DEF FNPR(P,Q,R,S)=P*R-Q*S
230 DEF FNPI(P,Q,R,S)=P*S+Q*R
240 DEF FNQR(P,Q,U,V)=(P*U+Q*V)/(U*U+V*V)
250 DEF FNQI(P,Q,U,V)=(Q*U-P*V)/(U*U+V*V)
260 REM*ROOT FINDING DRIVER ROUTINE*
270 EPS=.000001:REM*ACCURACY*
280 FOR J=M TO 1 STEP -1
290 PLSH$="FALSE"
300 XR=0:XI=0:REM*START ROOT FINDING AT ZERO*
310 GOSUB 1000:REM*CALL SUBROUTINE LAGUERRE*
320 IF ABS(XI)<=2*EPS*EPS*ABS(XR) THEN XI=0
330 ROOTR(J)=XR:ROOTI(J)=XI
340 REM*FORWARD DEFLATION*
350 BR=AR(J+1):BI=AI(J+1)
360 FOR K=J TO 1 STEP-1
370 CR=AR(K):CI=AI(K)
380 AR(K)=BR:AI(K)=BI
390 BRR=FNPR(XR,XI,BR,BI)+CR
400 BI=FNPI(XR,XI,BR,BI)+CI:BR=BRR
410 NEXT K
420 NEXT J
430 REM*POLISH ROOTS USING UNDEFLATED COEFFS*
440 FOR K=1 TO M+1
450 AR(K)=A(K):AI(K)=B(K)
460 NEXT K
470 FOR L=1 TO M
480 J=M
490 PLSH$="TRUE"
500 XR=ROOTR(L):XI=ROOTI(L)
510 GOSUB 1000:REM*CALL LAGUERRE*
520 IF ABS(XI)<=.1*EPS*ABS(XR) THEN XI=0
530 ROOTR(L)=XR:ROOTI(L)=XI
540 NEXT L
550 REM*SORT ROOTS BY REAL PARTS*
560 FOR J=2 TO M
570 XR=ROOTR(J):XI=ROOTI(J)
580 FOR K=J-1 TO 1 STEP-1
590 IF ROOTR(K)<=XR GOTO 630
600 ROOTR(K+1)=ROOTR(K):ROOTI(K+1)=ROOTI(K)
610 NEXT K
620 K=0
630 ROOTR(K+1)=XR:ROOTI(K+1)=XI
640 NEXT J
650 REM*OUTPUT RESULTS*
660 PRINT:PRINT"Roots of polynomial are:":PRINT
670 PRINT "REAL" TAB(14)"IMAGINARY";
680 PRINT TAB(28)"MODULUS" TAB(42)"ARGUMENT(DEG)"
690 FOR K=M TO 1 STEP-1
700 ZR=ROOTR(K):ZI=ROOTI(K)
710 GOSUB 2000
720 MO=FNCABS(ZR,ZI)
730 AZ=AZ*180/PIE
740 PRINT ROOTR(K) TAB(14);ROOTI(K) TAB(28);MO TAB(42);AZ
750 NEXT K
760 END
```

https://doi.org/10.1017/CBO9781139170093.013 Published online by Cambridge University Press

```
1000 REM SUBROUTINE: LAGUERRE
1010 EPSS=6E-09:REM*ACCURACY*
1020 MAXIT=100:REM*NUMBER OF ITERATIONS*
1030 DXOLD=FNCABS(XR,XI)
1040 FOR I=1 TO MAXIT
1050 BR=AR(J+1):BI=AI(J+1)
1060 ERO=FNCABS(BR,BI)
1070 DR=0:DI=0
1080 FR=0:FI=0
1090 ABX=FNCABS(XR,XI)
1100 REM*EVALUATION OF POLYNOMIAL AND FIRST TWO DERIVATIVES*
1110 FOR K=J TO 1 STEP-1
1120 FRR=FNPR(XR,XI,FR,FI)+DR
1130 FI=FNPI(XR,XI,FR,FI)+DI:FR=FRR
1140 DRR=FNPR(XR,XI,DR,DI)+BR
1150 DI=FNPI(XR,XI,DR,DI)+BI:DR=DRR
1160 BRR=FNPR(XR,XI,BR,BI)+AR(K)
1170 BI=FNPI(XR,XI,BR,BI)+AI(K):BR=BRR
1180 ERO=FNCABS(BR,BI)+ABX*ERO
1190 NEXT K
1200 ERO=EPSS*ERO
1210 IF FNCABS(BR,BI)<=ERO THEN GOTO 1220 ELSE GOTO 1250
1220 DXR=0:DXI=0
1230 RETURN
1240 REM*COMPUTE CORRECTION TERM*
1250 GR=FNQR(DR,DI,BR,BI)
1260 GI=FNQI(DR,DI,BR,BI)
1270 G2R=FNPR(GR,GI,GR,GI)
1280 G2I=FNPI(GR,GI,GR,GI)
1290 HR=G2R-2*FNQR(FR,FI,BR,BI)
1300 HI=G2I-2*FNQI(FR,FI,BR,BI)
1310 ZR=(J-1)*(J*HR-G2R)
1320 ZI=(J-1)*(J*HI-G2I)
1330 GOSUB 2000:REM*COMPUTES ARG(Z)=AZ*
1340 SRQ=SQR(FNCABS(ZR,ZI))*COS(AZ/2)
1350 SIQ=SQR(FNCABS(ZR,ZI))*SIN(AZ/2)
1360 GPR=GR+SRQ:GPI=GI+SIQ
1370 GMR=GR-SRQ:GMI=GI-SIQ
1380 IF FNCABS(GPR,GPI)<FNCABS(GMR,GMI) GOTO 1390 ELSE GOTO 1400
1390 GPR=GMR:GPI=GMI
1400 DXR=FNQR(J,0,GPR,GPI)
1410 DXI=FNQI(J,0,GPR,GPI)
1420 X1R=XR-DXR:X1I=XI-DXI
1430 IF XR=X1R AND XI=X1I THEN RETURN
1440 XR=X1R:XI=X1I
1450 CDX=FNCABS(DXR,DXI)
1460 IF I>6 AND CDX>=DXOLD THEN RETURN
1470 DXOLD=CDX
1480 IF PLSH$="TRUE" GOTO 1500
1490 IF FNCABS(DXR,DXI)<=EPS*FNCABS(XR,XI) THEN RETURN
1500 NEXT I
1510 PRINT "TOO MANY ITERATIONS"
1520 RETURN
2000 REM SUBRTN 4-QUADRANT ARCTAN ROUTINE
2010 PIE=3.14159265#
2020 IF ABS(ZR)<1E-36 GOTO 2080
2030 AZ=ATN(ZI/ZR)
2040 IF AZ=0 AND ZR<C THEN AZ=PIE
2050 IF ZR<0 AND ZI>0 THEN AZ=PIE+AZ
2060 IF ZR<0 AND ZI<0 THEN AZ=AZ-PIE
2070 RETURN
2080 IF ZI>0 THEN AZ=PIE/2 ELSE AZ=-PIE/2
2090 IF ZI=0 AND ZR=0 THEN AZ=0
2100 RETURN
```