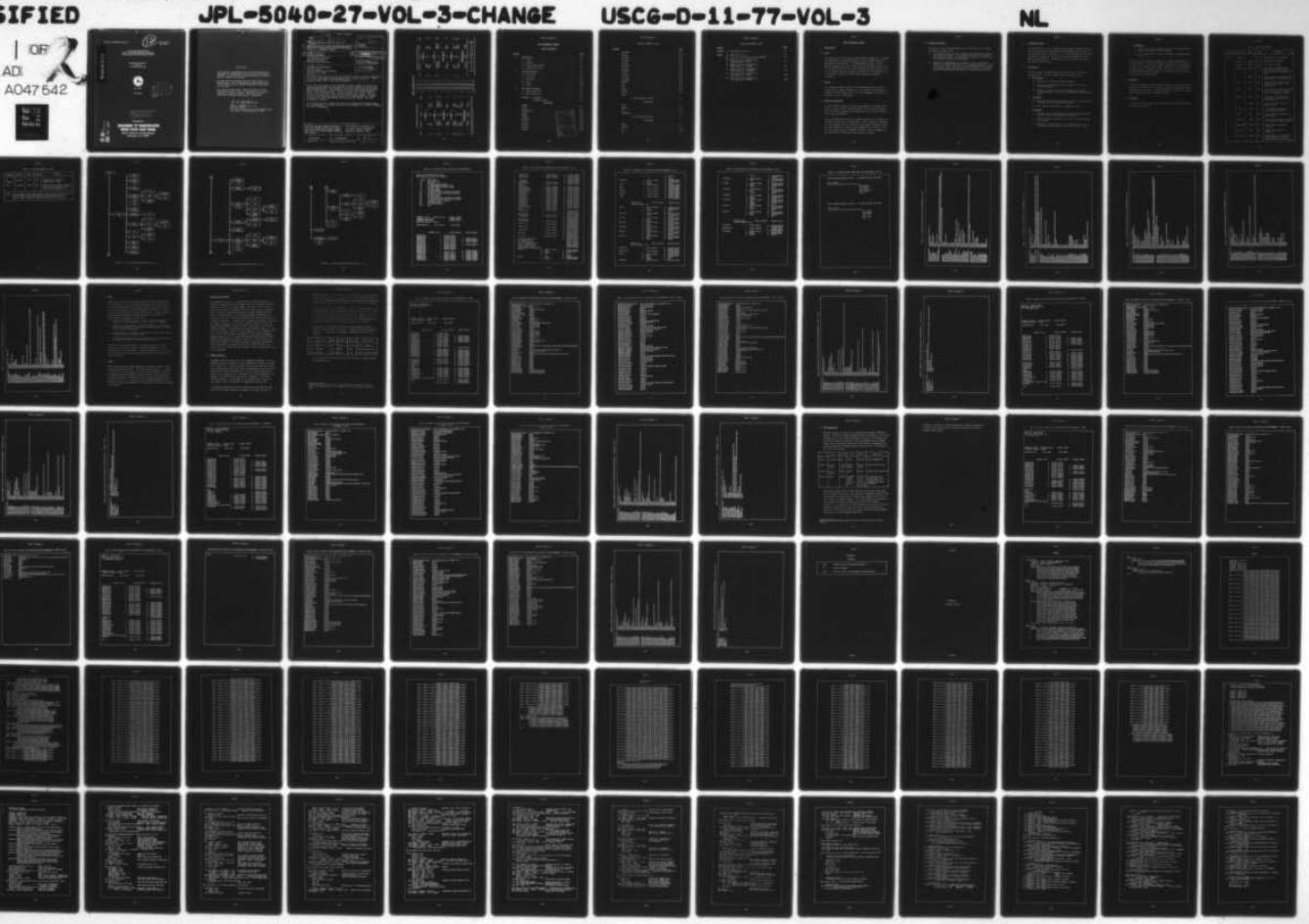
AD-A047 542 JET PROPULSION LAB PASADENA CALIF F/G 9/2
COMPUTER PROGRAM FOR DESIGN AND PERFORMANCE ANALYSIS OF NAVIGAT--ETC(U)
JUL 77 G GOLTZ, H WEINER
UNCLASSIFIED JPL-5040-27-VOL-3-CHANGE USCG-D-11-77-VOL-3 NL



Report No. 5040-27 (Change 1)

CG-D-11-77
July 1977

AD A047542

COMPUTER PROGRAM FOR DESIGN AND PERFORMANCE ANALYSIS OF NAVIGATION-AID POWER SYSTEMS

Program Documentation
Volume III
Programmer's Manual

July 1977

Final Report

DDC
RECEIVED
DEC 13 1977
F

DISTRIBUTION STATEMENT A
Approved for public release;
Distribution Unlimited

AD No.
DDC FILE COPY

Prepared for

DEPARTMENT OF TRANSPORTATION
UNITED STATES COAST GUARD
Office of Research and Development
Washington, D.C. 20590

NOTICE

This document is disseminated under the sponsorship of the Department of Transportation in the interest of information exchange. The United States Government assumes no liability for its contents or use thereof.

The United States Government does not endorse products or manufacturers. Trade or manufacturers' names appear herein solely because they are considered essential to the object of this report.

The contents of this report reflect the views of the Jet Propulsion Laboratory, which is responsible for the facts and accuracy of data presented. This report does not constitute a standard, specification or regulation.

D. L. Birkimer

DONALD L. BIRKIMER, Ph.D., P.E.
Technical Director
U.S. Coast Guard Research and Development Center
Avery Point, Groton, Connecticut 06340

(18) USCG, CGR/DC

5040-27 (Change 1)

Technical Report Documentation Page

1. Report No. (19) CG-D-11-77-VOL-3, 18/76-VOL-3	2. Government Accession No.	3. Recipient's Catalog No. (14) (12) 166p.
4. Title and Subtitle (6) COMPUTER PROGRAM FOR DESIGN AND PERFORMANCE ANALYSIS OF NAVIGATION-AID POWER SYSTEMS PROGRAM DOCUMENTATION, Volume III - Programmer's Manual		5. Report Date July 1977
		6. Performing Organization Code
7. Author(s) (10) G. Goltz H. Weiner (14)		8. Performing Organization Report No. JPL [redacted]-5040-27-VOL-3
9. Performing Organization Name and Address Jet Propulsion Laboratory 4800 Oak Grove Drive Pasadena, California 91103		10. Work Unit No. (TRAIS) Change 1
		11. Contract or Grant No. MIPR No. Z-70099-5-50352
12. Sponsoring Agency Name and Address Department of Transportation U.S. Coast Guard Office of Research and Development Washington, D.C. 20590		13. Type of Report and Period Covered (9) Final Report,
		14. Sponsoring Agency Code

15. Supplementary Notes

The contract under which this report was prepared was under the technical supervision of the Coast Guard Research and Development Center, Groton, Connecticut 06340. R&D Center report number 18/76 has been assigned.

16. Abstract

2 - AD49 356

The Jet Propulsion Laboratory has developed a computer program for designing and analyzing the performance of solar array/battery power systems for the U.S. Coast Guard Navigational Aids. This program is called the Design Synthesis/Performance Analysis (DSPA) Computer Program. The basic function of the Design Synthesis portion of the DSPA program is to evaluate functional and economic criteria to provide specifications for viable solar array/battery power systems. The basic function of the Performance Analysis portion of the DSPA program is to simulate the operation of solar array/battery power systems under specific loads and environmental conditions.

This document provides a detailed description of the DSPA Computer Program system and its subprograms. This manual will assist the programmer in revising or updating the several subprograms.

17. Key Words	18. Distribution Statement		
Batteries, Computer Programs, Navigation Aids, Power Systems, Solar Arrays, Solar Insolation, Statistical Analysis, Temperature, Terrestrial Power Systems, Weather, Wind Velocity	This document is available to the U.S. public through the National Technical Information Service, Sprinfield, Virginia 22161		
19. Security Classif. (of this report)	20. Security Classif. (of this page)	21. No. of Pages	22. Price
Unclassified	Unclassified	184	

Form DOT F 1700.7 (8-72) Reproduction of completed page authorized

191 150

METRIC CONVERSION FACTORS

Approximate Conversions to Metric Measures

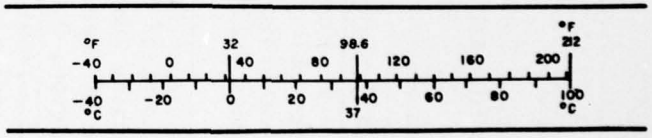
Symbol	When You Know	Multiply by	To Find	Symbol
		LENGTH		
in	inches	*2.5	centimeters	cm
ft	feet	30	centimeters	cm
yd	yards	0.9	meters	m
mi	miles	1.6	kilometers	km
		AREA		
in^2	square inches	6.5	square centimeters	cm^2
ft^2	square feet	0.09	square meters	m^2
yd^2	square yards	0.8	square meters	m^2
mi^2	square miles	2.6	square kilometers	km^2
	acres	*0.4*	*hectares*	ha
		MASS (weight)		
oz	ounces	28	grams	g
lb	pounds	0.45	kilograms	kg
	short tons (2000 lb)	0.9	tonnes	t
		VOLUME		
tsp	teaspoons	5	milliliters	ml
Tbsp	tablespoons	15	milliliters	ml
fl oz	fluid ounces	30	milliliters	ml
c	cups	0.24	liters	l
pt	pints	0.47	liters	l
qt	quarts	0.95	liters	l
gal	gallons	3.8	liters	l
ft^3	cubic feet	0.03	cubic meters	m^3
yd^3	cubic yards	0.76	cubic meters	m^3
		TEMPERATURE (exact)		
°F	Fahrenheit temperature	5/9 (after subtracting 32)	Celsius temperature	°C

*1 in = 2.54 (exactly). For other exact conversions and more detailed tables, see NBS Misc. Publ. 286, Units of Weights and Measures, Price $2.25, SD Catalog No. C13.10:286.



Approximate Conversions from Metric Measures

Symbol	When You Know	Multiply by	To Find	Symbol
		LENGTH		
mm	millimeters	0.04	inches	in
cm	centimeters	0.4	inches	in
m	meters	3.3	feet	ft
m	meters	1.1	yards	yd
km	kilometers	0.6	miles	mi
		AREA		
cm^2	square centimeters	0.16	square inches	in^2
m^2	square meters	1.2	square yards	yd^2
km^2	square kilometers	0.4	square miles	mi^2
ha	hectares (10,000 m^2)	2.5	acres	
		MASS (weight)		
g	grams	0.035	ounces	oz
kg	kilograms	2.2	pounds	lb
t	tonnes (1000 kg)	1.1	short tons	
		VOLUME		
ml	milliliters	0.03	fluid ounces	fl oz
l	liters	2.1	pints	pt
l	liters	1.06	quarts	qt
l	liters	0.26	gallons	gal
m^3	cubic meters	35	cubic feet	ft^3
m^3	cubic meters	1.3	cubic yards	yd^3
		TEMPERATURE (exact)		
°C	Celsius temperature	9/5 (then add 32)	Fahrenheit temperature	°F



DSPA PROGRAMMER'S MANUAL

TABLE OF CONTENTS

SECTION		PAGE
1.	INTROUDCTION	1-1
	1.1 Scope	1-1
	1.2 Purpose	1-1
	1.3 Computer Requirements	1-1
	1.4 Program Flow Charts	1-2
2.	SYSTEM DESCRIPTION	2-1
	2.1 Subroutines	2-1
	2.2 Structure	2-2
	2.3 Collection	2-2
	2.4 Input	2-20
	2.5 Output	2-20
3.	DSPA SUPPORT PROGRAMS	3-1
	3.1 MERGE Program Set	3-1
	3.2 STAT Program Set	3-21
	APPENDIX: A GLOSSARY	A-1
	B PROGRAM LISTING	B-1
	SUBPROGRAMS	
	ACOMONS	B-3
	BLKDTA	B-5
	BLKDTA/NI-CD	B-13
	CENTER	B-19
	DSDRVR	B-21
	DS-BATEFC	B-35
	DS-CDSI	B-36
	DS-ESGIV	B-37
	DS-PCDGC	B-38
	DS-SAGC	B-40



TABLE OF CONTENTS (contd)

SECTION		PAGE
	DS-TERMC	B-42
	PADRVR	B-43
	PA-BATEFC	B-47
	PA-CRVPLT	B-48
	PA-ESGC	B-50
	PA-INTER	B-52
	PA-PCDGC	B-54
	PA-PRTPLT	B-56
	PA-PSGC	B-57
	PA-SLIVC	B-62
	PA-SUMARY	B-64
	READTAPE	B-69
	SLUP	B-73
	SORT	B-75
	TB2GET	B-76
	TBLCOEF	B-78
	TBSR	
C	MERGE PROGRAM LISTINGS	C-1
	SUBPROGRAMS	
	TDF14	C-3
	DECK280	C-8
	IOW	C-12
	LISTMERGE	C-14
D	STAT PROGRAMS LISTINGS	D-1
	SUBPROGRAMS	
	STATS	D-3
	PROFILE	D-7
	IOW	D-12
	SLUP	D-14

TABLE OF CONTENTS (contd)

SECTION			PAGE
FIGURES:	2-1	DSPA Program Structure	2-6
TABLES:	2-1	DSPA Subroutines	2-3
	2-2	Collection and Subroutine Cross-Reference	2-9
	3A	MERGE Collection and Subroutine Cross-Reference - TDF14	3-3
	3B	MERGE Collection and Subroutine Cross-Reference - DECK280	3-9
	3C	MERGE Collection and Subroutine Cross-Reference - LISTMERGE	3-15
	4A	STAT Collection and Subroutine Cross-Reference - STATS	3-23
	4B	STAT Collection and Subroutine Cross-Reference - PROFILE	3-28

DSPA PROGRAMMER'S MANUAL

1. INTRODUCTION

1.1 Scope

The Design Synthesis/Performance Analysis (DSPA) program set is a package of subprograms for computing Navigation Aid Power System design and performance characteristics. The DSPA programs are used to obtain ambient temperature and solar radiation data from weather tapes, to provide specifications for viable solar array/battery power systems for use in the flashing lamp buoys employed as Maritime Aids to Navigation, and to simulate the operation of these power systems under specific conditions.

1.2 Purpose

The Programmer's Manual provides a detailed description of the DSPA system and the programs comprising it. The purpose of this manual is to document the DSPA programs and to assist the programmer in revising or updating the several subprograms.

1.3 Computer Requirements

The DSPA computer program is currently programmed for the UNIVAC 1108 computer using the EXEC 8 Operating System. Since the DSPA program is coded in FORTRAN V language, the program may be run on other computers with minimal modification.

Use of the DSPA program requires a computer having at least 30,000 core locations available for the program and at least 2 tape units available for mount/dismount service in addition to the normal input/output units. The core requirement is based on using segmentation or overlay. Software requirements include a FORTRAN V compiler, standard mathematical and input/output routines and CALCOMP plotting routines.

1.4 Program Flow Charts

Flow charts of the DSPA subprograms were not furnished in the Program Documentation volumes since:

- Most computer facilities have programs which automatically produce subroutine flow charts. If such charts are desired, the program user can easily select the subroutine of interest and obtain a copy of the latest version of the subroutine.

- Preparation, reproduction, and inclusion of all of the present versions of the DSPA subroutines in the Program Documentation would be more costly than if the flow charts were prepared by the program user automatically. Additionally, these flow charts would become obsolete as modifications were made to the DSPA computer program.

2. SYSTEM DESCRIPTION

The DSPA system is a set of related, digital computer programs which are used for designing and analyzing solar array/battery power systems for use as Maritime Aids to Navigation. All of the DSPA programs are written in FORTRAN V language for use on the UNIVAC 1108 computer under the EXEC-8 operating system. Operation is in batch mode via card input as described in Section 2.4.

2.1 Subroutines

The DSPA Program is segmented along functional lines to reduce core occupancy costs. The four segments and their functions are:

a. MAIN Segment

(1) Contains main DSPA driver program (CENTER)

(2) Contains all subprograms which are common to two or more auxiliary segments.

(3) Contains all data areas which are common to two or more auxiliary segments or which must be maintained between segment loads.

b. DS Segment

(1) Evaluates functional and economical criteria to provide design specifications for a viable power system.

(2) Writes design characteristic information tables.

c. PA Segment

(1) Simulates solar array/battery power system performance under specified loads and environmental conditions.

(2) Writes performance information to a file for later printing and plotting.

(3) Generates (on request) current vs. voltage plots for all of the power system subelments at a given "instant" in time.

d. SUM Segment

(1) Produces (on request) summary plots from the performance data generated and stored by the PA segment.

(2) Writes summary tables from the PA segment output performance data.

Loading of the three auxiliary segments is controlled automatically by the EXEC-8 system. An auxiliary segment is loaded into core only as it is required by the main driver program (CENTER) and is released when a different segment is requested. Table 2-1 gives a list of the DSPA program elements grouped by segment. The table also lists the element entry points, type, and function.

2.2 Structure

A schematic diagram of the DSPA program structure is given in Figure 2-1. Each box is identified by its Entry Point name with the Element name in parentheses if different. Internal subroutines are indicated in dotted boxes attached to the main subroutine element.

2.3 Collection

A collection and subroutine cross reference table for the DSPA program is provided in Table 2-2.

TABLE 2-1.-DSPA SUBROUTINES

SEGMENT	ELEMENT	ENTRY	TYPE	FUNCTION
MAIN	ACOMONS	-	PDP	Common data for MAIN segment
	BLKDTA	BLKDTA	FOR	Block Data for ACOMONS
	CENTER	(MAIN)	FOR	Main driver program
	READTAPE	RDTAPE	FOR	Read ambient temperature and solar insolation for a given day and hour from a weather data tape
		TMNMX	FOR	Find minimum and maximum ambient temperature over one-year period
	SLUP	SLUP	FOR	Single-variable look-up using Legrangian interpolation and extrapolation
	SORT	SORT	FOR	Sort an array in ascending order, eliminate any duplicate values, zero-fill remainder of array
	TB2GET	TB2GET	FOR	Two-variable look-up using Legrangian interpolation and extrapolation
		TB2SET	FOR	Build two-variable table for TB2GET search
	TBLCOEF	LCOEF	FOR	Find Legrange interpolation coefficients for TB2GET
	TBSR	TBSR	FOR	Perform TB2GET interpolation or extrapolation
DS	DSDRVR	DSDRVR	FOR	Perform Design Synthesis calculations and write output tables
	DS-BATEFC	DSBEFC	FOR	Compute battery operating efficiency
	DS-CDSI	CDSI	FOR	Compute clear-day solar insolation
	DS-ESGIV	DSESGC	FOR	Compute battery I-V characteristic arrays for the given temperature and state-of-charge

TABLE 2-1.-DSPA SUBROUTINES (contd)

SEGMENT	ELEMENT	ENTRY	TYPE	FUNCTION
DS (contd)	DS-PCDGC	DSPCDG	FOR	Initialize Power Conditioning and Distribution Group parameters
		DSPCDC	FOR	Compute Power Conditioning and Distribution Group I-V characteristic arrays
	DS-SAEC	SAEC	FOR	Compute Solar Array I-V characteristic arrays
	DS-TERMC	TERMC	FOR	Compute terminator characteristics (sunrise and sunset times)
PA	PADRVR	PADRVR	FOR	Execute Performance Analysis calculations and write output file for later printing
	PA-BATEFC	BATEFC	FOR	Compute battery operating efficiency
	PA-DRVPLT	CRVPLT	FOR	Generate "instantaneous" equipment I-V characteristic curve plots
	PA-ESGC	ESGC	FOR	Compute Energy Storage Unit and Group I-V characteristic arrays
	PA-INTER	INTER	FOR	Locate stable intersection for Difference and Energy Storage Group characteristic I-V curves
	PA-PCDGC	PCDG	FOR	Initialize Power Conditioning and Distribution Group parameters
		PCDGC	FOR	Compute Power Conditioning and Distribution Group I-V characteristic arrays
	PA-PRTPLT	PRTPLT	FOR	Write Performance Analysis summary data to a file for later printout
	PA-PSGC	PSGC	FOR	Compute Solar Array and Power Source Group I-V characteristic arrays

TABLE 2-1.-DSPA SUBROUTINES (contd)

SEGMENT	ELEMENT	ENTRY	TYPE	FUNCTION
PA (contd)	PA-SLIVC	SLIVC	FOR	Compute Shunt Limiter I-V characteristic arrays
SUM	PA-SUMARY	SUMARY	FOR	Read Stored Performance Analysis output data and product tabular printout and summary plots

NOTE: The MAIN Segment also contains four system library routines which were modified to facilitate simpler interfacing with the JPL coding. These routines are AXIS, PSCALE, QLINE, and QSCALE.

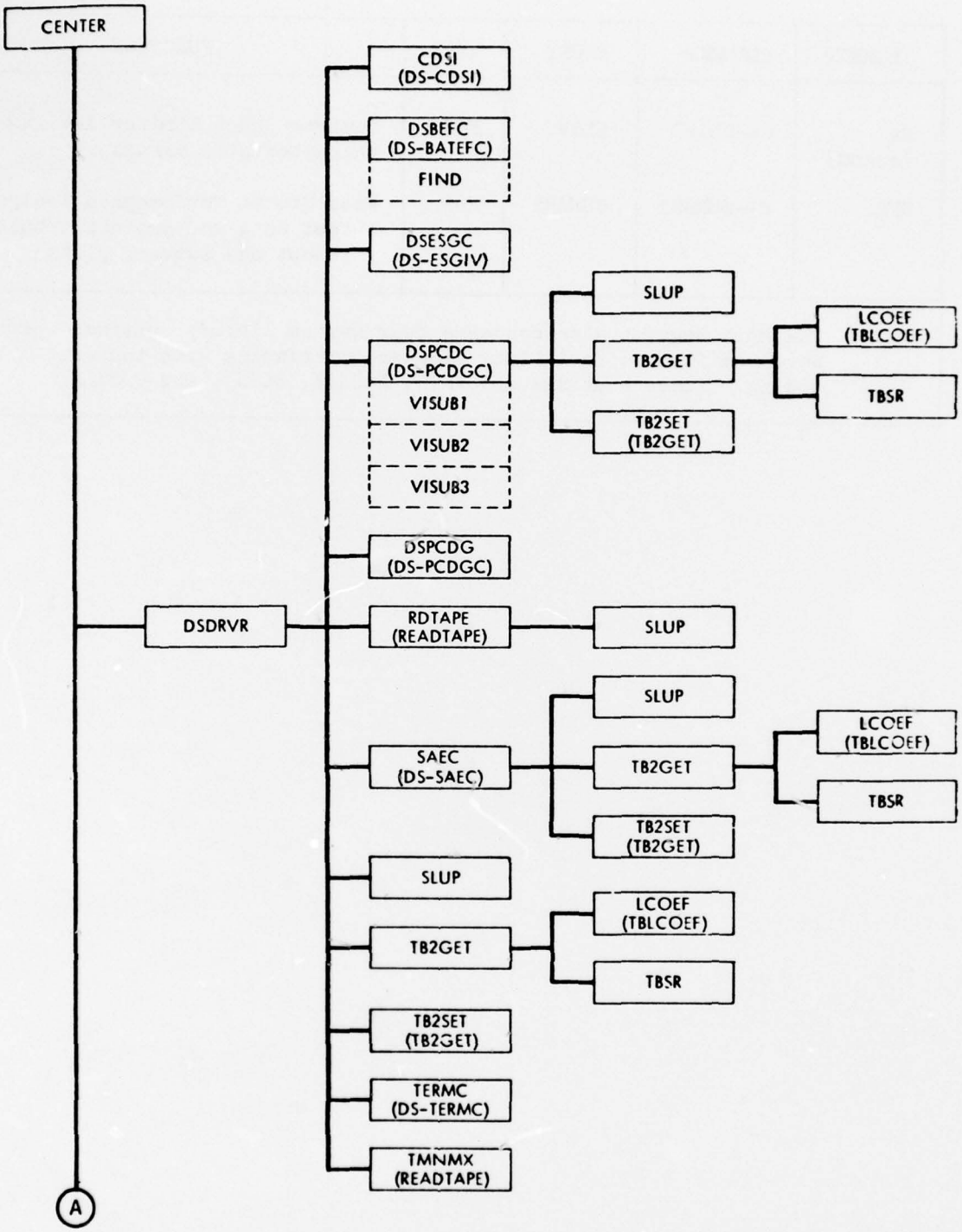


FIGURE 2-1. DSPA PROGRAM STRUCTURE (Sheet 1 of 3)

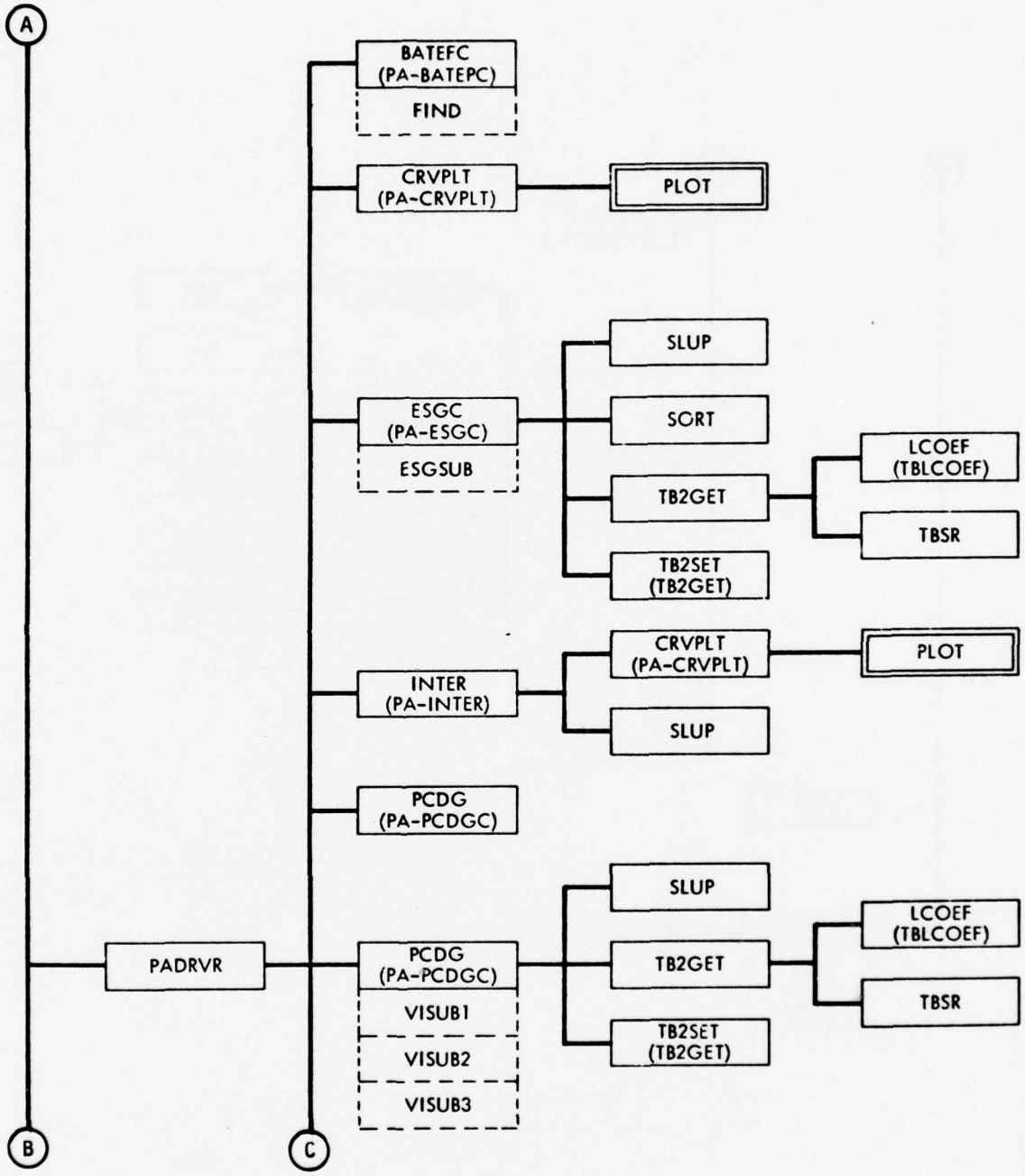


FIGURE 2-1. DSPA PROGRAM STRUCTURE (Sheet 2 of 3)

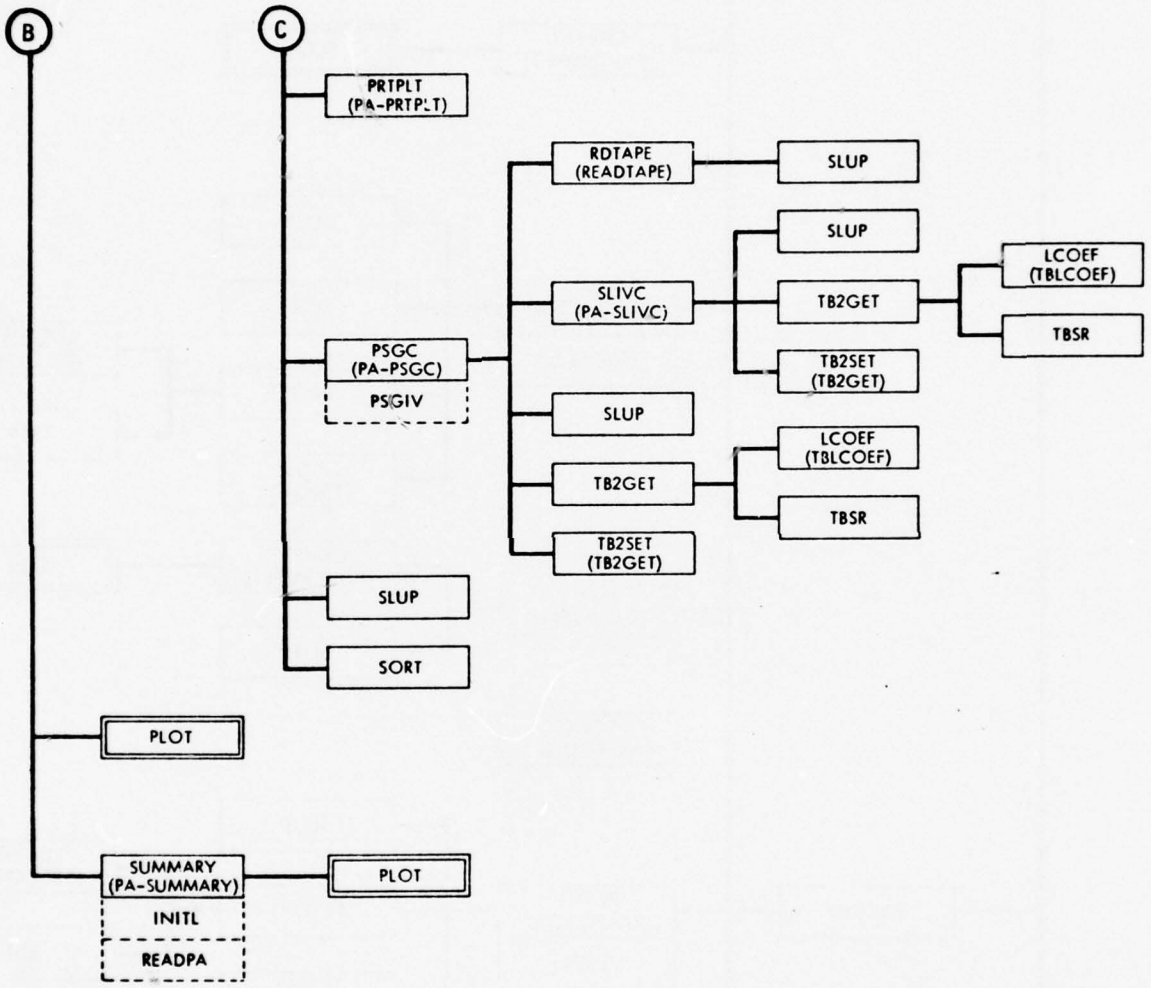


FIGURE 2-1. DSPA PROGRAM STRUCTURE (Sheet 3 of 3)

TABLE 2-2.-COLLECTION AND SUBROUTINE CROSS-REFERENCE

```
@MAP,XS DSPA.MAPDECK,DSPA.DSPA
MAP 23-Q 02/03-11:28 (0,) MAPDECK

     1.NEW   LOGIC FULL
     2.      LIB LIB*PLOTS
     3.      SEG MAIN
     4.      IN DSPA.CENTER,.BLKDTA
     5.      IN DSPA.READTAPE,.SLUP,.SORT
     6.      IN DSPA.TB2GET,.TBLCOEF,.TBSR
     7.      SEG DS*,(MAIN)
     8.      IN DSPA.DSDRVR
     9.      IN DSPA.DS-BATEFC,.DS-CDSI,.DS-ESGIV
    10.      IN DSPA.DS-PCDGC,.DS-SAEC,.DS-TERMC
    11.      SEG PA*,(MAIN)
    12.      IN DSPA.PADRVR
    13.      IN DSPA.PA-BATEFC,.PA-CRVPLT,.PA-ESGC
    14.      IN DSPA.PA-INTER,.PA-PCDGC,.PA-PRTPLT
    15.      IN DSPA.PA-PSGC,.PA-SLIVC
    16.      SEG SUM*,(MAIN)
    17.      IN DSPA.PA-SUMARY
    18.      END



ADDRESS LIMITS     001000 036105      040000 074105
SEGMENT LOAD TABLE                    040000 040017
INDIRECT LOAD TABLE                   040020 040127
STARTING ADDRESS  022135

WORDS DECIMAL       14918 IBANK        14406 DBANK


            SEGMENT MAIN              001000 025472      040130 066002

NSWTCS/FOR                  1   001000 001021
NRBLKS/FOR                  1   001022 001044
NRWNDS/FOR                  1   001045 001126        2   040130 040141
NWEFS/FOR                   1   001127 001313        2   040142 040161
NFTCHS/FOR                  1   001314 001627        2   040162 040207
NFTVS/FOR                   1   001630 001652
NCLOSS/FOR                  1   001653 002054        2   040210 040241
NWBLKS/FOR                  1   002055 002166
NBSBLS/FOR                  1   002167 002227
NUPDAS/FOR                  1   002230 002263
NBFOOS/FOR                                           2   040242 042443
NBDCVS/FOR                  1   002264 002411        2   042444 042506
NOTINS/FOR                  1   002412 002714        2   042507 042520
NOUTS/FOR                   1   002715 004041        2   042521 042555
NCNVTS/FOR                  1   004042 004263        2   042556 042652
NIOERS/FOR                  1   004264 004500        2   042653 043023
NININS/FOR                  1   004501 004671        2   043024 043027
```

TABLE 2-2.-COLLECTION AND SUBROUTINE CROSS-REFERENCE (contd)

```
NINPT$/FOR                  1   004672 005556      2   043030 043053
NFMT$/FOR                   1   005557 006442      2   043054 043131
NFCHK$/FOR                  1   006443 007433      2   043132 043305
                                                   4   043306 043357
NTAB$/FOR                                          2   043360 043534
NEXP6$/FOR                  1   007434 007630      2   043535 043606
NEXP5$/FOR                  1   007631 007716      2   043607 043616
ERU$
EXP$/FOR                    1   007717 010006      2   043617 043637
ASINCOS$/FOR                1   010007 010223      0   043640 043665
TANCOTAN$/FOR               1   010224 010421      2   043666 043706
SQRT$/FOR-JPL               1   010422 010460      2   043707 043713
SINCOS$/FOR                 1   010461 010613      2   043714 043735
ALOG$/FOR                   1   010614 010733      2   043736 043776
NERR$/FOR                   1   010734 011363      2   043777 044206
NLOUT$/FOR                  1   011364 012437      2   044207 044244
NSTOP$/FOR                  1   012440 012475      2   044245 044261
NOBUF$/FOR                  1   012476 012537      2   044262 044262
NIINP$/FOR                  1   012540 014430      2   044263 044467
NIER$/FOR                   1   014431 014602      2   044470 044607
NIBUF$/FOR                  1   014603 014642      2   044610 044610
NINTR$/FOR-JPL              1   014643 015067      2   044611 044674
NTRAN$/FOR-JPL              1   015070 016741      2   044675 045353
IDL$                        1   016742 017010
SETMSG/CALC                 1   017011 017101      0   045354 045400
                                                   2   BLANK$COMMON
SND777/CALC                                        0   045401 045432
PCTAB1/CALC                 1   017102 017133      0   045433 045475
JPLOGO/CALC                 1   017134 017526      0   045476 045705
                                                   2   BLANK$COMMON
PCTABL/CALC                                        0   045706 045745
PHEADER/CALC                1   017527 017761      0   045746 046062
                                                   2   BLANK$COMMON
PLHOR/CALC                  1   017762 020054      0   046063 046127
                                                   2   BLANK$COMMON
NUMBER/CALC                 1   020055 020367      0   046130 046175
                                                   2   BLANK$COMMON
SYMBOL/CALC                 1   020370 020576      0   046176 046205
                                                   2   046206 046521
AXIS/CALC                   1   020577 021266      0   046522 046617
                                                   2   BLANK$COMMON
SPLT/CALC                   1   021267 022134      0   046620 046634
                                                   2   046635 046643
CMNSUM (COMMON BLOCK)                                  046644 046710
PA (COMMON BLOCK)                                      046711 047150
DS (COMMON BLOCK)                                      047151 050634
INPUT2 (COMMON BLOCK)                                  050635 053765
INPUT1 (COMMON BLOCK)                                  053766 056552
DSPA (COMMON BLOCK)                                    056553 063425
BLANK$COMMON (COMMON BLOCK)
CENTER                      1   022135 022360      0   063426 065024
                            3   DSPA               2   BLANK$COMMON
                            5   INPUT2             4   INPUT1
                            7   PA                 6   DS
                                                   8   CMNSUM
BLKDTA                      3   DSPA               0   065025 065027
                            5   INPUT2             2   BLANK$COMMON
```

TABLE 2-2.-COLLECTION AND SUBROUTINE CROSS-REFERENCE (contd)

	7	PA		4	INPUT1
				6	DS
				8	CMNSUM
READTAPE	1	022361	023603	0	065030 065506
	3	DSPA		2	BLANKSCOMMON
	5	INPUT2		4	INPUT1
SLUP	1	023604	024120	0	065507 065546
				2	BLANKSCOMMON
SORT	1	024121	024304	0	065547 065575
				2	BLANKSCOMMON
TB2GET	1	024305	024741	0	065576 065676
				2	BLANKSCOMMON
TBLCOFF	1	024742	025076	0	065677 065746
				2	BLANKSCOMMON
TBSR	1	025077	025472	0	065747 066002
				2	BLANKSCOMMON
SEGMENT DS.			025473 035267		066003 074105
FOLLOWS SEGMENT MAIN					
DSDRVR	1	025473	032355	0	066003 072753
	3	DSPA		2	BLANKSCOMMON
	5	INPUT2		4	INPUT1
				6	DS
DS-BATEFC	1	032356	032555	0	072754 073073
	3	INPUT1		2	BLANKSCOMMON
				4	INPUT2
DS-CDSI	1	032556	033103	0	073074 073232
	3	DSPA		2	BLANKSCOMMON
	5	INPUT2		4	INPUT1
DS-ESGIV	1	033104	033300	0	073233 073274
	3	DSPA		2	BLANKSCOMMON
	5	INPUT2		4	INPUT1
DS-PCDGC	1	033301	034177	0	073275 073467
	3	DSPA		2	BLANKSCOMMON
	5	INPUT2		4	INPUT1
DS-SAEC	1	034200	035051	0	073470 073770
	3	DSPA		2	BLANKSCOMMON
	5	INPUT2		4	INPUT1
DS-TERMC	1	035052	035267	0	073771 074105
	3	DSPA		2	BLANKSCOMMON
	5	INPUT2		4	INPUT1
SEGMENT PA.			025473 036105		066003 070712
FOLLOWS SEGMENT MAIN					
QLINE/CALC	1	025473	026017	0	066003 066064
				2	BLANKSCOMMON
QSCALE/CALC	1	026020	026303	0	066065 066133
				2	BLANKSCOMMON
PADRVR	1	026304	030006	0	066134 067030
	3	DSPA		2	BLANKSCOMMON
	5	INPUT2		4	INPUT1
	7	CMNSUM		6	PA
PA-BATEFC	1	030007	030174	0	067031 067125

TABLE 2-2.-COLLECTION AND SUBROUTINE CROSS-REFERENCE (contd)

	3	INPUT1		2	BLANK$COMMON
				4	INPUT2
PA-CRVPLT	1	030175 030714		6	067126 067240
	3	DSPA		2	BLANK$COMMON
				4	PA
PA-ESGC	1	030715 031542		0	067241 067357
	3	DSPA		2	BLANK$COMMON
	5	INPUT2		4	INPUT1
				6	PA
PA-INTER	1	031543 032143		0	067360 067551
	3	DSPA		2	BLANK$COMMON
	5	INPUT2		4	INPUT1
				6	PA
PA-PCDGC	1	032144 033030		0	067552 067652
	3	DSPA		2	BLANK$COMMON
	5	INPUT2		4	INPUT1
PA-PRTPLT	1	033031 033243		0	067653 067725
	3	DSPA		2	BLANK$COMMON
	5	INPUT2		4	INPUT1
	7	CMNSUM		6	PA
PA-PSGC	1	033244 035333		0	067726 070531
	3	DSPA		2	BLANK$COMMON
	5	INPUT2		4	INPUT1
				6	PA
PA-SLIVC	1	035334 036105		0	070532 070712
	3	DSPA		2	BLANK$COMMON
	5	INPUT2		4	INPUT1
				6	PA
SEGMENT SUM•			025473 030216		066003 067173
FOLLOWS SEGMENT MAIN					
LINE/CALC	1	025473 026031		0	066003 066066
				2	BLANK$COMMON
PSCALE/CALC	1	026032 026353		0	066067 066127
				2	BLANK$COMMON
PA-SUMARY	1	026354 030216		0	066130 067173
	3	INPUT1		2	BLANK$COMMON
	5	CMNSUM		4	INPUT2

TABLE 2-2.-COLLECTION AND SUBROUTINE CROSS-REFERENCE (contd)

```
IBANK SEGMENTS DRAWN TO SCALE:    300 WORDS DECIMAL PER DASH


MAIN (10555)
11-1-11-1-11-1-1-11-1-11-1-1-11-1-1
                                          DS* (3965)
                                          1-1-11-1-11-1
                                          PA* (4363)
                                          1-1-11-1-11-1-1
                                          SUM* (1364)
                                          1-1-1


DBANK SEGMENTS DRAWN TO SCALE:    300 WORDS DECIMAL PER DASH


MAIN (11179)
-1-11-1-11-1-1-11-1-11-1-1-11-1-11-1-1
                                                DS* (3139)
                                                -1-1-11-1-
                                                PA* (1480)
                                                -1-1
                                                SUM* (633)
                                                -1
```

TABLE 2-2.-COLLECTION AND SUBROUTINE CROSS-REFERENCE (contd)

```
ENTRY/BLOCK(ELEMENT)  **** REFERENCED BY ELEMENT ****
ACOS(ASINCOSS)        DS-TERMC,PA-PSGC
ALOG(ALOGS)           DSDRVR
ALOG10(ALOGS)         NUMBER,PSCALE,QSCALE
ASIN(ASINCOSS)        DS-CDSI,PA-PSGC
ASTOFF(NIERS)         NOUTS
AWAITS(ERUS)          NTRANS
AXIS(AXIS)            PA-CRVPLT,PA-SUMARY
BATEFC(PA-BATEFC)     PADRVR
BLANKSCOMMON(COMMON BLOCK)   AXIS,BLKDTA,CENTER,DSDRVR,DS-BATEFC,DS-CDSI,DS-ESGIV,DS-PCDGC,DS-SAEC,DS-TERMC,JPLOGO,LINE,NUMBER
                      PADRVR,PA-BATEFC,PA-CRVPLT,PA-ESGC,PA-INTER,PA-PCDGC,PA-PRTPLT,PA-PSGC,PA-SLIVC,PA-SUMARY,PHEADER,PLMDR,PSCALE
                      QLINE,QSCALE,READTAPE,SETMSG,SLUP,SORT,TBLCOEF,TBSR,TB26ET
BLS(NBFOOS)           NFCHKS
BRDS(ERUS)            NTRANS
BRKPT(SPLT)           PHEADER
BSIBLS(NBSBLS)        NFCHKS,NOTINS,NWEFS
B1LS(NBFOOS)          NFCHKS
B1OS(NBFOOS)          NFCHKS
B2LS(NOT-DEFINED)     NFCHKS
B2OS(NOT-DEFINED)     NFCHKS
CDSI(DS-CDSI)         DSDRVR
CENDS(ERUS)           NINTRS
CFE(NFCHKS)           NOTINS
CLOSES(NCLOSS)        NFCHKS,NWEFS
CMNSUM(COMMON BLOCK)   BLKDTA,CENTER,PADRVR,PA-PRTPLT,PA-SUMARY
COMS(ERUS)            NSTOPS
COSISINCOSS)          AXIS,DS-CDSI,DS-TERMC,PA-PSGC,SYMBOL
CRVPLT(PA-CRVPLT)     PADRVR,PA-INTER
CSFS(ERUS)            NCLOSS,NFCHKS,NTRANS,SPLT
DATES(ERUS)           PCTABI
DRAINS(NWBLKS)        NFCHKS,NOTINS,NRWNDS,NWEFS
DS(COMMON BLOCK)      BLKDTA,CENTER,DSDRVR
DSBEFC(DS-BATEFC)     DSDRVR
DSDRVR(DSDRVR)        CENTER
DSESGC(DS-ESGIV)      DSDRVR
DSPA(COMMON BLOCK)    BLKDTA,CENTER,DSDRVR,DS-CDSI,DS-ESGIV,DS-PCDGC,DS-SAEC,DS-TERMC,PADRVR,PA-CRVPLT,PA-ESGC,PA-INTER,PA-PCDGC
                      PA-PRTPLT,PA-PSGC,PA-SLIVC,READTAPE
DSPCDC(DS-PCDGC)      DSDRVR
DSPCDG(DS-PCDGC)      DSDRVR
ERRS(ERUS)            NIOERS,NSTOPS,NTRANS,SPLT
ESGC(PA-ESGC)         PADRVR
EXIT(NSTOPS)          NFCHKS,NINTRS
EXITS(ERUS)           NSTOPS,NTRANS
EXP(EXPS)             DS-CDSI,PA-PSGC
EXTPLS(NSTOPS)        NINTRS
FACILS(ERUS)          NTRANS
FHSIS(NINPTS)         NIBUFS
FHSIOS(NOUTS)         NOBUFS
FHS2S(NINPTS)         NIBUFS
FHS2OS(NOUTS)         NOBUFS
FIELDS(NERRS)         NINTRS
FITEMS(ERUS)          NCLOSS,NFCHKS
```

TABLE 2-2.-COLLECTION AND SUBROUTINE CROSS-REFERENCE (contd)

```
ENTRY/BLOCK(ELEMENT)   **** REFERENCED BY ELEMENT ****
FMTOP(NFMTS)           NOUTS
FNCTBS(NFCHKS)         NFTCHS,NOTINS,NSWTCS
FORKS(ERUS)            NTRANS
IALLS(ERUS)            NINTRS,NSTOPS
INPUT1(COMMON BLOCK)   BLKDTA,CENTER,DSDRVR,DS-BATEFC,DS-CDSI,DS-ESGIV,DS-PCDGC,DS-SAEC,DS-TERMC,PADRVR,PA-BATEPC,PA-ESGC,PA-INTER
                       PA-PCDGC,PA-PRTPLT,PA-PSGC,PA-SLIVC,PA-SUMARY,READTAPE
INPUT2(COMMON BLOCK)   BLKDTA,CENTER,DSDRVR,DS-BATEFC,DS-CDSI,DS-ESGIV,DS-PCDGC,DS-SAEC,DS-TERMC,PADRVR,PA-BATEPC,PA-ESGC,PA-INTER
                       PA-PCDGC,PA-PRTPLT,PA-PSGC,PA-SLIVC,PA-SUMARY,READTAPE
INTER(PA-INTER)        PADRVR
IOCODS(NIERS)          NFCHKS,NFMTS,NFTCHS,NIBUFS,NLINPS,NLOUTS,NOBUFS,NRWNDS,NWEFS
IOERRS(NERRS)          NFCHKS
IOIS(ERUS)             NTRANS
IOWIS(ERUS)            NTRANS
IOWS(ERUS)             NBSBLS,NCLOSS,NFCHKS,NFTCHS,NIOERS,NTRANS
IOXIS(ERUS)            NTRANS
IOS(ERUS)              NCLOSS,NRBLKS,NWBLKS
LCOEF(TBLCOEF)         TB2GET
LINE(LINE)             PA-SUMARY
LOADS(ERUS)            IDLS
LOGNOS(NOBUFS)         NLOUTS
LOGO(JPLOGO)           PLHDR
MBS(ERUS)              NBSBLS,NCLOSS,NFCHKS,NFTCHS,NOBUFS,NRWNDS,NTRANS,NUPDAS
MFS(ERUS)              NFTCHS,NTRANS
MSGCHK(SND777)         PHEADER
NABOS(NFTVS)           NFMTS
NAB1S(NFTVS)           NFMTS
NAB2S(NFTVS)           NFMTS
NAB3S(NFTVS)           NFMTS
NAB4S(NFTVS)           NFMTS
NAB5S(NFTVS)           NFMTS
NAB6S(NFTVS)           NFMTS
NAB7S(NFTVS)           NFMTS
NAVCS(NFTVS)           NFMTS
NBCWS(NIOERS)          NININS
NBFGTS(NFCHKS)         NFTCHS,NOTINS,NWEFS
NBFMGS(NFCHKS)         NFTCHS,NOTINS,NWEFS
NBFRLS(NFCHKS)         NFTCHS,NOTINS,NWEFS
NBFRSS(NFCHKS)         NFTCHS,NOTINS
NBIPAS(NINPTS)         NININS
NBIS(NBDCVS)           NLOUTS,NOUTS
NBLNKS(NOUTS)          NOBUFS
NBTODS(NERRS)          NFCHKS,NTRANS
NCAS(NIERS)            NFMTS
NCCCS(NOUTS)           NOTINS
NCDOFS(NINTRS)         NCNVTS
NCEFS(NSTOPS)          NCLOSS
NCHARS(NIERS)          NFMTS
NCJNIOS(NIERS)         NIOERS,NLOUTS,NOTINS
NCNVOS(NCNVTS)         NINPTS,NLINPS
NCONES(NFMTS)          NCNVTS,NINPTS
NCSPS(NIERS)           NININS,NINPTS,NLINPS,NLOUTS,NOTINS,NOUTS
```

TABLE 2-2.-COLLECTION AND SUBROUTINE CROSS-REFERENCE (contd)

```
ENTRY/BLOCK(ELEMENT)   **** REFERENCED BY ELEMENT ****
NC1ULO(NIERS)          NBDCVS,NFCHKS,NLINPS,NLOUTS,NOTINS
NC1ULI(NIERS)          NBDCVS,NFCHKS,NLINPS
NDBCVS(NCNVTS)         NFMTS,NINPTS,NLINPS
NDBFIS(NCNVTS)         NINPTS,NLINPS
NDBINS(NCNVTS)         NFMTS,NINPTS,NLINPS
NDBIS(NCNVTS)          NFMTS,NLINPS
NDBLTS(NFMTS)          NINPTS
NDIGS(NBDCVS)          NLOUTS,NOUTS
NDOUTS(NBDCVS)         NLOUTS,NOUTS
NEES(NSTOPS)           NERRS,NINTRS
NEFCLS(NIOERS)         NFTCHS,NININS,NINPTS,NOTINS
NERCRS(NIOERS)         NCNVTS,NFMTS,NINPTS,NLINPS
NERCTS(NIOERS)         NCNVTS,NFTCHS,NLINPS
NERRAS(NERRS)          ALOGS,ASINCOSS,EXPS,NEXP5S,NEXP6S,SINCOSS,SQRTS,TANCOTANS
NERRBS(NERRS)          ASINCOSS,EXPS,NEXP5S,NEXP6S,SINCOSS,TANCOTANS
NERRCS(NERRS)          NEXP5S,NEXP6S,TANCOTANS
NERRS(NERRS)           SQRTS
NERR2S(NERRS)          PA-SLIVC,TBSR,TB2GET
NERR3S(NERRS)          AXIS,DSDRVR,DS-BATEFC,DS-CDSI,DS-ESGIV,DS-PCDGC,DS-SAEC,DS-TERMC,JPLOGO,LINE,NUMBER,PADRVR,PA-BATEFC,PA-CRVPLT
                       PA-ESGC,PA-INTER,PA-PCDGC,PA-PRTPLT,PA-PSGC,PA-SLIVC,PA-SUMARY,PHEADER,PLHDR,PSCALE,QLINE,QSCALE,READTAPE
                       SETMSG,SLUP,SORT,TBLCOEF,TBSR,TB2GET
NERR4S(NERRS)          DS-PCDGC,PADRVR,PA-PCDGC,PA-PSGC,PA-SUMARY,READTAPE
NERUS(NIOERS)          NFCHKS,NININS,NLOUTS,NOBUFS,NOTINS
NETFS(NOUTS)           NFMTS
NEXITS(NOUTS)          NOTINS
NEXP5S(NEXP5S)         NUMBER,PSCALE,QSCALE
NEXP6S(NEXP6S)         AXIS,PSCALE
NFARS(NFMTS)           NINPTS,NOUTS
NFBYIS(NIOERS)         NFTCHS
NFCAS(NCNVTS)          NFMTS
NFCHKS(NFCHKS)         NIBUFS,NLINPS,NLOUTS,NOBUFS,NRWNDS,NWEFS
NFCIS(NCNVTS)          NINPTS,NLINPS
NFCMS(NCNVTS)          NINPTS,NLINPS
NFCSS(NCNVTS)          NFMTS
NFDBS(NCNVTS)          NINPTS,NLINPS
NFDPS(NCNVTS)          NBDCVS
NFGCS(NFMTS)           NINPTS,NOUTS
NFGTS(NFMTS)           NINPTS
NFHNGS(NFMTS)          NOUTS
NFMTRS(NCNVTS)         NFMTS
NFMTS(NFMTS)           NIBUFS,NOBUFS,NOUTS
NFM96S(NFMTS)          NINPTS,NOUTS
NFNIO1DS(NIERS)        NFMTS
NFNIO1S(NIERS)         NFMTS
NFNS1S(NBDCVS)         NOUTS
NFNS2S(NBDCVS)         NLOUTS,NOUTS
NFNS3S(NBDCVS)         NLOUTS,NOUTS
NFPCS(NIERS)           NLOUTS,NOTINS,NOUTS
NFPKTS(NFCHKS)         NCLOSS
NFRAS(NFMTS)           NIBUFS,NININS,NLINPS,NOTINS,NOUTS
NFRCS(NFMTS)           NINPTS
```

TABLE 2-2.-COLLECTION AND SUBROUTINE CROSS-REFERENCE (contd)

ENTRY/BLOCK(ELEMENT)	•••• REFERENCED BY ELEMENT ••••
NFRGS(NIERS)	NFMTS,NINPTS
NFRHS(NIERS)	NIBUFS,NININS,NINPTS,NLINPS
NFRJS(NIERS)	NLOUTS,NOTINS,NOUTS
NFRONFS(NIOERS)	NFTCHS
NFRZS(NIERS)	NFMTS,NINPTS,NOUTS
NFRZSS(NIERS)	NFMTS
NFSGS(NCNVTS)	NINPTS,NLINPS
NFTCBS(NFTCHS)	NININS
NFTCHS(NFTCHS)	NININS
NFTGLS(NIERS)	NCNVTS,NFMTS,NINPTS
NGC9S(NFMTS)	NINPTS,NOUTS
NHPFAS(NIERS)	NIBUFS,NLINPS,NLOUTS,NOBUFS,NRWNDS,NWEFS
NHVCS(NFTVS)	NFMTS
NIICS(NINPTS)	NININS
NINDS(NBDCVS)	NLOUTS,NOUTS
NINIIS(NININS)	NIBUFS,NLINPS
NINTRS(NINTRS)	CENTER
NIOERS(NIOERS)	NBSRLS,NCLOSS,NFCHKS,NFTCHS,NOTINS,NRBLKS,NRWNDS,NWBLKS,NWEFS
NIOERSA(NIOERS)	NOBUFS
NIOIVS(NIERS)	NFMTS
NIOIS(NIERS)	DSDRVR,PA-SUMARY
NIO2VS(NIERS)	NIBUFS,NOBUFS
NIO2S(NIERS)	CENTER,DSDRVR,DS-PCDGC,DS-SAEC,PADRVR,PA-ESGC,PA-INTER,PA-PCDGC,PA-PRTPLT,PA-PSGC,PA-SLIVC,PA-SUMARY,READTAPE SETMSG
NIO3VAS(NIERS)	NFMTS
NIO3VS(NIERS)	NFMTS
NIO2S(NIERS)	NFMTS,NIBUFS,NOTINS
NKLNS(NIERS)	NIBUFS,NININS,NLINPS,NOTINS
NKL2S(NIERS)	NININS,NLINPS,NOTINS
NLIOS(NIERS)	NLINPS,NLOUTS
NLLCS(NIERS)	NINPTS,NLINPS
NLLNS(NIERS)	NININS,NLINPS
NLRTS(NIERS)	NIOERS,NLINPS,NLOUTS
NLTBS(NIERS)	NIOERS,NLINPS,NLOUTS
NNG9OS(NIERS)	NIBUFS,NINPTS,NLINPS
NNRSXS(NIERS)	NLINPS,NLOUTS
NOLCS(NIERS)	NLOUTS
NOLNS(NIERS)	NOTINS
NOTIIS(NOTINS)	NLOUTS,NOBUFS
NPCTS(NFMTS)	NINPTS,NOUTS
NPRS(NOUTS)	NOTINS
NPUS(NOUTS)	NOTINS
NPW2S(NFMTS)	NINPTS
NP9IS(NFMTS)	NINPTS,NOUTS
NRBFAS(NFCHKS)	NWEFS
NRBFS(NFCHKS)	NCLOSS,NRWNDS
NRDUS(NIBUFS)	CENTER,DSDRVR,PADRVR
NRDS(NINPTS)	NININS
NRECS(NIOERS)	NININS,NLINPS,NOTINS
NRETOS(NERRS)	NINTRS
NREWS(NRWNDS)	NCLOSS

TABLE 2-2.-COLLECTION AND SUBROUTINE CROSS-REFERENCE (contd)

ENTRY/BLOCK(ELEMENT)	**** REFERENCED BY ELEMENT ****
NRM92S(NIERS)	NLOUTS,NOUTS
NRNLS(NLINPS)	CENTER
NRSFS(NERRS)	NIOERS,NSTOPS,NTRANS
NRSXS(NIERS)	NIBUFS,NLINPS,NLOUTS,NOBUFS
NRTRS(NIERS)	NFMTS,NIBUFS,NININS,NINPTS,NLINPS
NR91S(NFMTS)	NIBUFS,NOBUFS,NOUTS
NR92S(NFMTS)	NINPTS,NOUTS
NR93S(NFMTS)	NIERS,NINPTS,NOUTS
NSAOS(NERRS)	NIOERS
NSFS(NCNVTS)	NINPTS
NSLS(NBDCVS)	NLOUTS,NOUTS
NSTATS(NIERS)	NCNVTS,NFCHKS,NFMTS,NFTCHS,NINPTS,NIOERS,NLINPS
NSTOPS(NSTOPS)	CENTER,DSDRVR,DS-PCDGC,DS-SAEC,PADRVR,PA-ESGC,PA-INTER,PA-PCDGC,PA-PRTPLT,PA-PSGC,PA-SLIVC,PA-SUMARY,READTAPE SETMSG,T82GET
NSTSVS(NIERS)	NCNVTS,NFMTS,NIBUFS,NINPTS,NLINPS,NOTINS
NSWTCS(NSWTCS)	NWEFS
NSIIS(NSTOPS)	NCLOSS,NERRS,NFCHKS,NIERS,NIOERS,NRWNDS,NTRANS,NWEFS
NTABS(NTABS)	NBSBLS,NCLOSS,NFCHKS,NFMTS,NFTCHS,NIBUFS,NININS,NIOERS,NLINPS,NLOUTS,NOBUFS,NOTINS,NRBLKS,NRWNDS,NSWTCS,NTRANS NUPDAS,NWBLKS,NWEFS
NTBSZS(NTABS)	NCLOSS,NFCHKS,NTRANS
NTENDS(NFMTS)	NIOERS,NLOUTS,NOTINS
NTRAN(NTRANS)	CENTER,PA-PRTPLT,PA-SUMARY,READTAPE
NTSTOS(NFMTS)	NOBUFS
NTIOS(NFMTS)	NINPTS,NOUTS
NUMBER(NUMBER)	AXIS,PA-CRVPLT
NVECS(NFTVS)	NFMTS,NINPTS,NOUTS
NWALKS(NERRS)	NCLOSS,NFCHKS,NIOERS,NOTINS,NRWNDS,NTRANS,[illegible]FS
NWDUS(NOBUFS)	CENTER,DSDRVR,DS-PCDGC,DS-SAEC,PADRVR,PA-E[illegible],PA-INTER,PA-PCDGC,PA-PRTPLT,PA-PSGC,PA-SLIVC,PA-SUMARY,READTAPE SETMSG
NWEFS(NWEFS)	NCLOSS
NWNLS(NLOUTS)	CENTER,DSDRVR,DS-BATEFC,DS-CDSI,DS-ESGIV,DS-PCDGC,DS-SAEC,DS-TERMC,PADRVR,PA-INTER,PA-PSGC
NXVCS(NFTV*)	NFMTS
OPTS(ERUS)	NSTOPS
PA(COMMON BLOCK)	BLKDTA,CENTER,PADRVR,PA-CRVPLT,PA-ESGC,PA-INTER,PA-PRTPLT,PA-PSGC,PA-SLIVC
PACKTS(NIERS)	NFCHKS,NININS,NIOERS,NOBUFS,NOTINS,NWEFS
PADRVR(PADRVR)	CENTER
PCDG(PA-PCDGC)	PADRVR
PCDGC(PA-PCDGC)	PADRVR
PCTABL(PCTABL)	PLHDR
PCTABI(PCTABI)	PHEADER
PCTS(ERUS)	NTRANS,PCTABL,PCTABI
PLHDR(PLHDR)	SPLT
PLHDRI(PHEADER)	SPLT
PLOT(SPLT)	AXIS,CENTER,JPLOGO,LINE,PA-CRVPLT,PA-SUMARY,PHEADER,PLHDR,QLINE,SETMSG,SYMBOL
PLOTS(SPLT)	CENTER
PLOTI(SPLT)	PHEADER
PLS(NBFOOS)	NFCHKS
PNCHAS(ERUS)	NOTINS,NWEFS
PPPS(NOUTS)	NOTINS
PRINTS(ERUS)	NCLOSS,NERRS,NFCHKS,NFMTS,NINTRS,NIOERS,NOTINS,NOUTS,NRWNDS,NSTOPS,NTRANS,NWEFS,SPLT
PRNTAS(ERUS)	NOTINS,NOUTS

TABLE 2-2.-COLLECTION AND SUBROUTINE CROSS-REFERENCE (contd)

ENTRY/BLOCK(ELEMENT)	**** REFERENCED BY ELEMENT ****
PRTPLT(PA-PRTPLT)	PADRVR
PSCALE(PSCALE)	PA-SUMARY
PSGC(PA-PSGC)	PADRVR
PUNCHS(ERUS)	NOUTS,NWEFS,SND777,SPLT
QLINE(QLINE)	PA-CRVPLT
QSCALE(QSCALE)	PA-CRVPLT
RDBLKS(NRBLKS)	NFTCHS,NWEFS
RDTAPE(READTAPE)	DSDRVR,PA-PSGC
READAS(ERUS)	NININS
READS(ERUS)	NINPTS
RESTS(NERRS)	NSTOPS
REWIS(ERUS)	NTRANS
REWS(ERUS)	NTRANS
RS(ERUS)	NFTCHS,NRBLKS,NTRANS
SAEC(DS-SAEC)	DSDRVR
SDS(ERUS)	NTRANS
SETMSG(SETMSG)	PHEADER
SIN(SINCOSS)	AXIS,DS-CDSI,DS-TERMC,PA-PSGC,SYMBOL
SLIVC(PA-SLIVC)	PA-PSGC
SLTS(NO-ELEMENT)	IDLS
SLUP(SLUP)	DSDRVR,DS-PCDGC,DS-SAEC,PADRVR,PA-CRVPLT,PA-ESGC,PA-INTER,PA-PCDGC,PA-PSGC,PA-SLIVC,READTAPE
SNS(ERUS)	NTRANS
SNAPS(ERUS)	NSTOPS
SND777(SND777)	SETMSG
SORT(SORT)	PADRVR
SQRT(SQRTS)	DS-CDSI,PA-PSGC
SRDS(ERUS)	NTRANS
STREGS(NERRS)	NCLOSS,NCNVTS,NFCHKS,NFMTS,NFTCHS,NINPTS,NINTRS,NIOERS,NLINPS,NRWNDS,NTRANS,NWEFS
SUMARY(PA-SUMARY)	CENTER
SYMBOL(SYMBOL)	AXIS,LINE,NUMBER,PA-CRVPLT,PA-SUMARY,PHEADER,PLHDR,QLINE,SETMSG
TAN(TANCOTANS)	DS-CDSI,DS-TERMC,PA-PSGC
TBSR(TBSR)	TB2GET
TB2GET(TB2GET)	DSDRVR,DS-BATEFC,DS-ESGIV,DS-PCDGC,DS-SAEC,PA-BATEFC,PA-ESGC,PA-PCDGC,PA-PSGC,PA-SLIVC
TB2SET(TB2GET)	DSDRVR,DS-BATEFC,DS-ESGIV,DS-PCDGC,DS-SAEC,PA-BATEFC,PA-ESGC,PA-PCDGC,PA-PSGC,PA-SLIVC
TEMPS(NIERS)	NFCHKS,NININS,NOTINS
TERMC(DS-TERMC)	DSDRVR
TINTLS(ERUS)	NRWNDS,NTRANS
TNNMX(READTAPE)	DSDRVR
TPCKS(NTABS)	NOUTS
TSAS(ERUS)	NTRANS
TSWAPS(ERUS)	NIOERS,NTRANS
TWAITS(ERUS)	NTRANS
UNITS(NIERS)	NCLOSS,NFCHKS,NFTCHS,NININS,NIOERS,NLINPS,NOTINS,NRBLKS,NWBLKS,NWEFS
UPDDAS(NUPDAS)	NBSBLS,NFCHKS,NFTCHS,NOTINS,NRBLKS,NWBLKS,NWEFS
WAITS(ERUS)	NBSBLS,NCLOSS,NFCHKS,NFTCHS,NOBUFS,NOTINS,NRBLKS,NRWNDS,NUPDAS,NWBLKS,NWEFS
WEFS(ERUS)	NTRANS
WHERE(SPLT)	LINE,PLHDR,QLINE
WRBLKS(NWBLKS)	NFCHKS,NSWTCS
WS(ERUS)	NCLOSS,NFCHKS,NTRANS,NWBLKS
XFORS(NFMTS)	NOUTS
ZONER(PA-SLIVC)	PADRVR

2.4 Input

There are three forms of inputs required for use of the DSPA program: (1) control cards to direct the program execution, (2) namelist data, and (3) time-variant, free-format data. Each of these forms depends on the portion of portions of the DSPA program to be utilized for a particular run as described in the DSPA User's Manual. Certain conventions have been established and used wherever possible to provide some simplification with regards to input data:

a. Free-format input data consists of a list of values separated by commas; if an item is omitted, the value of zero is assumed.

b. Namelist input data consists of variable names with their values; if an item or portions of an array are omitted, zeroes are assumed for those items not specified.

c. Automatic typing of variables is used except for MAXI, MAXV, MARSA, and MSAPWR which are defined as real.

d. Multidimensional arrays follow the convention that the first index will vary most rapidly, then the second, and so on.

There are two different procedures for executing the DSPA program, depending on the manner in which the ambient temperature and solar insolation data is to be obtained. Detailed descriptions of the DSPA input data and usage, along with sample runstreams, are provided in the DSPA User's Manual.

2.5 Output

There are two kinds of outputs generated by the DSPA programs: tabular printout and graphical output. The tabular output consists of a number of 132-character lines arranged in various tables depending on which of the Design Synthesis/Performance Analysis routines are being executed. The graphical output is produced only for Performance Analysis runs and consists of current vs. voltage plots and/or summary performance plots. Detailed descriptions and samples of DSPA output are provided in the DSPA User's Manual.

3. DSPA SUPPORT PROGRAMS

As indicated in Section 2.1 of the DS/PA User's Manual, there are, in addition to manual entry, two alternate forms of environmental data input available for use with the DSPA program. The first of these alternatives obtains input from a MERGE file which contains up to 12 sequential years of hourly environmental information directly extracted from NOAA TDF-14 and DECK-280 tapes. The second method uses a STAT file which consists of one year of hourly environmental data formed by profiling the n years of MERGE data. To produce these alternate environmental data files, two support program sets were written: MERGE (consisting of the TDF14, DECK280, and LISTMERGE programs) and STAT (consisting of the STATS and PROFILE programs). These programs are described further below. Note that both sets of support programs (with the exception of the LISTMERGE routine which, like the DSPA program, uses NTRAN) utilize a special assembly-language subroutine, IOW, to facilitate rapid, efficient, and economical random access reading and writing of data files. This subroutine also permits the reading of the NOAA environmental tapes which, because they utilize fractional word length records (82-1/3 words for TDF-14 and 133-2/3 words for DECK-280 records), cannot be read by conventional Univac 1108 I/O packages.

3.1 MERGE Program Set

The MERGE program set consists of three stand-alone programs: TDF14 and DECK280 for producing MERGE files, and LISTMERGE for viewing the contents of the MERGE files. The TDF14 program reads one day's data from a NOAA TDF-14 weather tape, strips off the date, temperature, and wind velocity information and writes a MERGE record. Since the TDF-14 tape data is sorted, this initial writing of the MERGE file is done sequentially from January 1 of a user-specified start year to December 31 of a user-specified end year. Any missing days are automatically filled in by the program. Bad or missing data is flagged by the value -1000.

The DECK280 program reads ten hours of data from a NOAA DECK-280 tape, strips off the solar radiation (in Langley's), converts the data to

watts/square meter, and inserts the insolation values into the appropriate MERGE file date and hour. It should be noted that the DECK-280 tape data is unsorted and unedited, thereby preventing the program from verifying the integrity of an entire day's information until completion of the program.

To facilitate checking of the MERGE file contents, a reader program, LIST-MERGE, was written. This program will display any number of days of MERGE data beginning at any desired date within the file. Viewing is completely random, and the user may skip around in the file as much as he desires. Also, either the skeletal MERGE file produced by the TDF14 program or the completed file generated by the DECK280 program may be reviewed.

Descriptions for the usage of these three MERGE programs are given in Section 4.1 of the DS/PA User's Manual. Program listings are provided in Appendix C of this manual. The format of the MERGE file records is as follows:

	Word 1	Word 2	Word 3	Word 4	Words 5-28
Sector 1*	Station No.	Date	Sector No.	EOF Date	Hourly Temperature
Sector 2	Blank	Blank	Blank	Blank	Hourly Wind Velocity
Sector 3	Blank	Blank	Blank	Blank	Hourly Solar Radiation

A collection and subroutine cross reference table for the MERGE programs are provided below.

*A Sector is defined as 28 words. Both NTRAN and IOW use sector count to locate read/write positions. Each MERGE record is 3 sectors, or 84 words, long.

TABLE 3A.-MERGE COLLECTION AND SUBROUTINE CROSS-REFERENCE - TDF14

```
@MAP,XS  ,USER.TDF14
MAP 23-R 05/21-07:18 (,0)
IN  USER.TDF14,.IOW




ADDRESS LIMITS     001000 013172      040000 045370
STARTING ADDRESS   011270

WORDS DECIMAL         5243 IBANK          2809 DBANK



            SEGMENT MAIN              001000 013172      040000 045370

NSWTC$/FOR                  1    001000 001021
NRBLK$/FOR                  1    001022 001044
NRWND$/FOR                  1    001045 001126        2    040000 040011
NWEF$/FOR                   1    001127 001313        2    040012 040031
NBDCV$/FOR                  1    001314 001441        2    040032 040074
NFTCH$/FOR                  1    001442 001755        2    040075 040122
NCLOS$/FOR                  1    001756 002157        2    040123 040154
NWBLK$/FOR                  1    002160 002271
NBSBL$/FOR                  1    002272 002332
NUPDA$/FOR                  1    002333 002366
NBFOO$/FOR                                           2    040155 042356
NFTV$/FOR                   1    002367 002411
NCNVT$/FOR                  1    002412 002633        2    042357 042453
NOTIN$/FOR                  1    002634 003136        2    042454 042465
NOUT$/FOR                   1    003137 004263        2    042466 042522
NININ$/FOR                  1    004264 004454        2    042523 042526
NFCHK$/FOR                  1    004455 005445        2    042527 042702
                                                     4    042703 042754
NOSYM$/FOR                  1    005446 005707        2    042755 042761
NIOER$/FOR                  1    005710 006124        2    042762 043132
NINPT$/FOR                  1    006125 007011        2    043133 043156
NFMT$/FOR                   1    007012 007675        2    043157 043234
NTAB$/FOR                                            2    043235 043411
ERUS
NERR$/FOR                   1    007676 010325        2    043412 043621
NSTOP$/FOR                  1    010326 010363        2    043622 043636
NOBUF$/FOR                  1    010364 010425        2    043637 043637
NIBUF$/FOR                  1    010426 010465        2    043640 043640
NINTR$/FOR-JPL              1    010466 010712        2    043641 043724
NIER$/FOR                   1    010713 011064        2    043725 044044
NISYM$/FOR                  1    011065 011267        2    044045 044050
BLANK$COMMON (COMMON BLOCK)
TDF14                       1    011270 013137        0    044051 045357
                                                     2    BLANK$COMMON
IOW                         1    013140 013172        2    045360 045370
```

TABLE 3A.-MERGE COLLECTION AND SUBROUTINE CROSS-REFERENCE - TDF14 (contd)

```
ENTRY/BLOCK(ELEMENT)    **** REFERENCED BY ELEMENT ****
ASTOFF(NIERS)           NOUTS
BLANKSCOMMON(COMMON BLOCK)   TDF14
BLS(NBFOOS)             NFCHKS
BSIBLS(NBSBLS)          NFCHKS,NOTINS,NWEFS
BILS(NBFOOS)            NFCHKS
BIOS(NBFOOS)            NFCHKS
B2LS(NOT-DEFINED)       NFCHKS
B2OS(NOT-DEFINED)       NFCHKS
CENDS(ERUS)             NINTRS
CFE(NFCHKS)             NOTINS
CLOSES(NCLOSS)          NFCHKS,NWEFS
COMS(ERUS)              NSTOPS
CSFS(ERUS)              NCLOSS,NFCHKS
DRAINS(NWBLKS)          NFCHKS,NOTINS,NRWNDS,NWEFS
ENDECS(NIERS)           NISYMS,NOSYMS
ERRS(ERUS)              NIOERS,NSTOPS
EXIT(NSTOPS)            NFCHKS,NINTRS
EXITS(ERUS)             NSTOPS
EXTPLS(NSTOPS)          NINTRS
FHSIS(NINPTS)           NIBUFS,NISYMS
FHSIOS(NOUTS)           NOBUFS,NOSYMS
FHS2S(NINPTS)           NIBUFS,NISYMS
FHS2OS(NOUTS)           NOBUFS,NOSYMS
FIELDS(NERRS)           NINTRS
FITEMS(ERUS)            NCLOSS,NFCHKS
FMTOP(NFMTS)            NOUTS
FNCTBS(NFCHKS)          NFTCHS,NOTINS,NSATCS
IALLS(ERUS)             NINTRS,NSTOPS
INSTAT(NIERS)           TDF14
INS(NOSYMS)             NISYMS
IOCODS(NIERS)           NFCHKS,NFMTS,NFTCHS,NIBUFS,NISYMS,NOBUFS,NOSYMS,NRWNDS,NWEFS
IOERRS(NERRS)           NFCHKS
IOW(IOW)                TDF14
IOWS(ERUS)              IOW,NBSBLS,NCLOSS,NFCHKS,NFTCHS,NIOERS
IOS(ERUS)               NCLOSS,NRBLKS,NWBLKS
LOGNOS(NOBUFS)          NOSYMS
MBS(ERUS)               NBSBLS,NCLOSS,NFCHKS,NFTCHS,NOBUFS,NRWNDS,NUPDAS
MFS(ERUS)               NFTCHS
NABOS(NFTVS)            NFMTS
NAB1S(NFTVS)            NFMTS
NAB2S(NFTVS)            NFMTS
NAB3S(NFTVS)            NFMTS
NAB4S(NFTVS)            NFMTS
NAB5S(NFTVS)            NFMTS
NAB6S(NFTVS)            NFMTS
NAB7S(NFTVS)            NFMTS
NAVCS(NFTVS)            NFMTS
NBCWS(NIOERS)           NININS
NBFGTS(NFCHKS)          NFTCHS,NOTINS,NWEFS
NBFMGS(NFCHKS)          NFTCHS,NOTINS,NWEFS
NBFRLS(NFCHKS)          NFTCHS,NOTINS,NWEFS
```

TABLE 3A.-MERGE COLLECTION AND SUBROUTINE CROSS-REFERENCE - TDF14 (contd)

ENTRY/BLOCK(ELEMENT)	**** REFERENCED BY ELEMENT ****
NBFRS$(NFCHK$)	NFTCH$,NOTIN$
NBIPA$(NINPT$)	NININ$,NISYM$
NBI$(NBDCV$)	NOUT$
NBLNK$(NOUT$)	NOBUF$,NOSYM$
NBTOD$(NERR$)	NFCHK$
NCA$(NIER$)	NFMT$,NISYM$,NOSYM$
NCCC$(NOUT$)	NOTIN$
NCDOF$(NINTR$)	NCNVT$
NCEF$(NSTOP$)	NCLOS$
NCHAR$(NIER$)	NFMT$,NISYM$,NOSYM$
NCJNI02$(NIER$)	NIOER$,NOTIN$
NCNV9$(NCNVT$)	NINPT$
NCOM3$(NFMT$)	NCNVT$,NINPT$
NCSP$(NIER$)	NININ$,NINPT$,NOTIN$,NOUT$
NC1ULO(NIER$)	NBDCV$,NFCHK$,NOTIN$
NC1UL1(NIER$)	NBDCV$,NFCHK$
NDBCV$(NCNVT$)	NFMT$,NINPT$
NDBFI$(NCNVT$)	NINPT$
NDBIN$(NCNVT$)	NFMT$,NINPT$
NDBI$(NCNVT$)	NFMT$
NDBLT$(NFMT$)	NINPT$
NDCOD$(NISYM$)	TDF14
NDIG$(NBDCV$)	NOUT$
NDOUT$(NBDCV$)	NOUT$
NEE$(NSTOP$)	NERR$,NINTR$
NEFCL$(NIOER$)	NFTCH$,NININ$,NINPT$,NOTIN$
NERCR$(NIOER$)	NCNVT$,NFMT$,NINPT$
NERCT$(NIOER$)	NCNVT$,NFTCH$
NERR3$(NERR$)	TDF14
NERR6$(NERR$)	TDF14
NERU$(NIOER$)	NFCHK$,NININ$,NOBUF$,NOSYM$,NOTIN$
NETF$(NOUT$)	NFMT$,NOSYM$
NEXIT$(NOUT$)	NOSYM$,NOTIN$
NFAR$(NFMT$)	NINPT$,NOUT$
NFBY1$(NIOER$)	NFTCH$
NFCA$(NCNVT$)	NFMT$
NFCHK$(NFCHK$)	NIBUF$,NOBUF$,NRWND$,NWEF$
NFCI$(NCNVT$)	NINPT$
NFCM$(NCNVT$)	NINPT$
NFCS$(NCNVT$)	NFMT$
NFDB$(NCNVT$)	NINPT$
NFDP$(NCNVT$)	NBDCV$
NFGC$(NFMT$)	NINPT$,NOUT$
NFGT$(NFMT$)	NINPT$
NFMNG$(NFMT$)	NOUT$
NFMTR$(NCNVT$)	NFMT$
NFMT$(NFMT$)	NIBUF$,NISYM$,NOBUF$,NOSYM$,NOUT$
NFM96$(NFMT$)	NINPT$,NOUT$
NFNIO1D$(NIER$)	NFMT$
NFNIO1$(NIER$)	NFMT$
NFNS1$(NBDCV$)	NOUT$

TABLE 3A.-MERGE COLLECTION AND SUBROUTINE CROSS-REFERENCE - TDF14 (contd)

ENTRY/BLOCK(ELEMENT)	**** REFERENCED BY ELEMENT ****
NFNS2$(NBDCV$)	NOUT$
NFNS3$(NBDCV$)	NOUT$
NFPC$(NIER$)	NOSYM$,NOTIN$,NOUT$
NFPKT$(NFCHK$)	NCLOS$
NFRA$(NFMT$)	NIBUF$,NININ$,NISYM$,NOSYM$,NOTIN$,NOUT$
NFRC$(NFMT$)	NINPT$
NFRG$(NIER$)	NFMT$,NINPT$
NFRH$(NIER$)	NIBUF$,NININ$,NINPT$,NISYM$
NFRJ$(NIER$)	NOSYM$,NOTIN$,NOUT$
NFRONF$(NIOER$)	NFTCH$
NFRZ$(NIER$)	NFMT$,NINPT$,NOUT$
NFRZ$S(NIER$)	NFMT$
NFSG$(NCNVT$)	NINPT$
NFTCB$(NFTCH$)	NININ$
NFTCH$(NFTCH$)	NININ$
NFTGL$(NIER$)	NCNVT$,NFMT$,NINPT$
NGC9$(NFMT$)	NINPT$,NOUT$
NHPFA$(NIER$)	NIBUF$,NISYM$,NOBUF$,NOSYM$,NRWND$,NWEF$
NHPFB$(NIER$)	NISYM$,NOSYM$
NHVC$(NFTV$)	NFMT$
NIIC$(NINPT$)	NININ$,NISYM$
NIND$(NBDCV$)	NOUT$
NINI1$(NININ$)	NIBUF$
NINTR$(NINTR$)	TDF14
NIOER$(NIOER$)	NBSBL$,NCLOS$,NFCHK$,NFTCH$,NOTIN$,NRBLK$,NRWND$,NWBLK$,NWEF$
NIOERSA(NIOER$)	NOBUF$
NIO1V$(NIER$)	NFMT$
NIO1$(NIER$)	NOSYM$,TDF14
NIO2V$(NIER$)	NIBUF$,NISYM$,NOBUF$,NOSYM$
NIO2$(NIER$)	TDF14
NIO3VA$(NIER$)	NFMT$
NIO3V$(NIER$)	NFMT$
NIO2$(NIER$)	NFMT$,NIBUF$,NISYM$,NOTIN$
NKLN$(NIER$)	NIBUF$,NININ$,NISYM$,NOSYM$,NOTIN$
NKL2$(NIER$)	NININ$,NOTIN$
NLLC$(NIER$)	NINPT$
NLLM$(NIER$)	NININ$
NLRT$(NIER$)	NIOER$
NLTB$(NIER$)	NIOER$
NNG90$(NIER$)	NIBUF$,NINPT$,NISYM$
NOLM$(NIER$)	NOTIN$
NOTI1$(NOTIN$)	NOBUF$
NPCT$(NFMT$)	NINPT$,NOUT$
NPR$(NOUT$)	NOSYM$,NOTIN$
NPU$(NOUT$)	NOSYM$,NOTIN$
NPW2$(NFMT$)	NINPT$
NP91$(NFMT$)	NINPT$,NOUT$
NRBFA$(NFCHK$)	NWEF$
NRBF$(NFCHK$)	NCLOS$,NRWND$
NRDU$(NIBUF$)	TDF14
NRD$(NINPT$)	NININ$,NISYM$

TABLE 3A.-MERGE COLLECTION AND SUBROUTINE CROSS-REFERENCE - TDF14 (contd)

ENTRY/BLOCK(ELEMENT)	**** REFERENCED BY ELEMENT ****
NRECS(NIOERS)	NININS,NISYMS,NOSYMS,NOTINS
NRETOS(NERRS)	NINTRS
NREWS(NRWNDS)	NCLOSS
NRM92S(NIERS)	NOUTS
NRSFS(NERRS)	NIOERS,NSTOPS
NRSXS(NIERS)	NIBUFS,NISYMS,NORUFS,NOSYMS
NRTRS(NIERS)	NFMTS,NIBUFS,NININS,NINPTS,NISYMS
NR91S(NFMTS)	NIBUFS,NISYMS,NORUFS,NOSYMS,NOUTS
NR92S(NFMTS)	NINPTS,NOUTS
NR93S(NFMTS)	NIERS,NINPTS,NOUTS
NSAOS(NERRS)	NIOERS
NSFS(NCNVTS)	NINPTS
NSLS(NBDCVS)	NOUTS
NSTATS(NIERS)	NCNVTS,NFCHKS,NFMTS,NFTCHS,NINPTS,NIOERS
NSTOPS(NSTOPS)	TDF14
NSTSVS(NIERS)	NCNVTS,NFMTS,NIBUFS,NINPTS,NOSYMS,NOTINS
NSWTCS(NSWTCS)	NWEFS
NSIIS(NSTOPS)	NCLOSS,NERRS,NFCHKS,NIERS,NIOERS,NRWNDS,NWEFS
NTABS(NTABS)	NBSBLS,NCLOSS,NFCHKS,NFMTS,NFTCHS,NIBUFS,NININS,NIOERS,NISYMS,NOBUFS,NOSYMS,NOTINS,NRBLKS,NRWNDS,NSWTCS,NUPDAS NWBLKS,NWEFS
NTBSZS(NTABS)	NCLOSS,NFCHKS
NTENDS(NFMTS)	NIOERS,NOTINS
NTSTOS(NFMTS)	NOBUFS,NOSYMS
NTIOS(NFMTS)	NINPTS,NOUTS
NVECS(NFTVS)	NFMTS,NINPTS,NOUTS
NWALKS(NERRS)	NCLOSS,NFCHKS,NIOERS,NOTINS,NRWNDS,NWEFS
NWDUS(NOBUFS)	TDF14
NWEFS(NWEFS)	NCLOSS
NXVCS(NFTVS)	NFMTS
OPTS(ERUS)	NSTOPS
OUTS(NOSYMS)	NISYMS
PACKTS(NIERS)	NFCHKS,NININS,NIOERS,NISYMS,NORUFS,NOSYMS,NOTINS,NWEFS
PLS(NBFOOS)	NFCHKS
PNCHAS(ERUS)	NOTINS,NWEFS
PPPS(NOUTS)	NOTINS
PRINTS(ERUS)	NCLOSS,NERRS,NFCHKS,NFMTS,NINTRS,NIOERS,NOTINS,NOUTS,NRWNDS,NSTOPS,NWEFS
PRNTAS(ERUS)	NOTINS,NOUTS
PUNCHS(ERUS)	NOUTS,NWEFS
RDBLKS(NRBLKS)	NFTCHS,NWEFS
READAS(ERUS)	NININS
READS(ERUS)	NINPTS
RESTS(NERRS)	NSTOPS
RS(ERUS)	NFTCHS,NRBLKS
SNAPS(FRUS)	NSTOPS
STREGS(NERRS)	NCLOSS,NCNVTS,NFCHKS,NFMTS,NFTCHS,NINPTS,NINTRS,NIOERS,NRWNDS,NWEFS
TEMPS(NIERS)	NFCHKS,NININS,NOTINS
TINTLS(ERUS)	NRWNDS
TPCKS(NTABS)	NOUTS
TSWAPS(ERUS)	NIOERS
UNITS(NIERS)	NCLOSS,NFCHKS,NFTCHS,NININS,NIOERS,NOSYMS,NOTINS,NRBLKS,NWBLKS,NWEFS
UPDDAS(NUPDAS)	NBSBLS,NFCHKS,NFTCHS,NOTINS,NRBLKS,NWBLKS,NWEFS

TABLE 3A.-MERGE COLLECTION AND SUBROUTINE CROSS-REFERENCE - TDF14 (contd)

ENTRY/BLOCK(ELEMENT)	**** REFERENCED BY ELEMENT ****
WAITS(ERUS)	NBSBLS, NCLOSS, NFCHKS, NFTCHS, NOBUFS, NOTINS, NRBLKS, NRWNDS, NUPDAS, NWBLKS, NWEFS
WRBLKS(NWBLKS)	NFCHKS, NSWTCS
WS(ERUS)	NCLOSS, NFCHKS, NWALKS
XFORS(NFMTS)	NOUTS

TABLE 3B.-MERGE COLLECTION AND SUBROUTINE CROSS-REFERENCE - DECK280

```
@MAP,XS  ,USER,DECK280
MAP 23-R 05/21-07:19 (,0)
IN  USER.DECK280,.10W




ADDRESS LIMITS     001000 012510      040000 045165
STARTING ADDRESS   011270

WORDS DECIMAL         4937 IBANK          2678 DBANK



            SEGMENT MAIN              001000 012510      040000 045165

NSWTCS/FOR                  1    001000 001021
NRBLKS/FOR                  1    001022 001044
NRWNDS/FOR                  1    001045 001126        2    040000 040011
NWEFS/FOR                   1    001127 001313        2    040012 040031
NBDCVS/FOR                  1    001314 001441        2    040032 040074
NFTCHS/FOR                  1    001442 001755        2    040075 040122
NCLOSS/FOR                  1    001756 002157        2    040123 040154
NWBLKS/FOR                  1    002160 002271
NBSBLS/FOR                  1    002272 002332
NUPDAS/FOR                  1    002333 002366
NBFOOS/FOR                                           2    040155 042356
NFTVS/FOR                   1    002367 002411
NCNVTS/FOR                  1    002412 002633        2    042357 042453
NOTINS/FOR                  1    002634 003136        2    042454 042465
NOUTS/FOR                   1    003137 004263        2    042466 042522
NININS/FOR                  1    004264 004454        2    042523 042526
NFCHKS/FOR                  1    004455 005445        2    042527 042702
                                                     4    042703 042754
NOSYMS/FOR                  1    005446 005707        2    042755 042761
NIOERS/FOR                  1    005710 006124        2    042762 043132
NINPTS/FOR                  1    006125 007011        2    043133 043156
NFMTS/FOR                   1    007012 007675        2    043157 043234
NTABS/FOR                                            2    043235 043411
ERUS
NERRS/FOR                   1    007676 010325        2    043412 043621
NSTOPS/FOR                  1    010326 010363        2    043622 043636
NOBUFS/FOR                  1    010364 010425        2    043637 043637
NIBUFS/FOR                  1    010426 010465        2    043640 043640
NINTRS/FOR-JPL              1    010466 010712        2    043641 043724
NIERS/FOR                   1    010713 011064        2    043725 044044
NISYMS/FOR                  1    011065 011267        2    044045 044050
BLANK$COMMON (COMMON BLOCK)
DECK280                     1    011270 012455        0    044051 045154
                                                     2    BLANK$COMMON
IOW                         1    012456 012510        2    045155 045165
```

TABLE 3B.-MERGE COLLECTION AND SUBROUTINE CROSS-REFERENCE - DECK280 (contd)

```
ENTRY/BLOCK(ELEMENT)  **** REFERENCED BY ELEMENT ****
ASTOFF(NIERS)         NOUTS
BLANKSCOMMON(COMMON BLOCK)   DECK280
BLS(NBFOOS)           NFCHKS
BSIBLS(NBSBLS)        NFCHKS,NOTINS,NWEFS
BILS(NBFOOS)          NFCHKS
BIOS(NBFOOS)          NFCHKS
B2LS(NOT-DEFINED)     NFCHKS
B2OS(NOT-DEFINED)     NFCHKS
CENDS(ERUS)           NINTRS
CFE(NFCHKS)           NOTINS
CLOSES(NCLOSS)        NFCHKS,NWEFS
COMS(ERUS)            NSTOPS
CSFS(ERUS)            NCLOSS,NFCHKS
DRAINS(NWBLKS)        NFCHKS,NOTINS,NRWNDS,NWEFS
ENDECS(NIERS)         NISYMS,NOSYMS
ERRS(ERUS)            NIOERS,NSTOPS
EXIT(NSTOPS)          NFCHKS,NINTRS
EXITS(ERUS)           NSTOPS
EXTPLS(NSTOPS)        NINTRS
FMS1S(NINPTS)         NIBUFS,NISYMS
FMS1OS(NOUTS)         NOBUFS,NOSYMS
FMS2S(NINPTS)         NIBUFS,NISYMS
FMS2OS(NOUTS)         NOBUFS,NOSYMS
FIELDS(NERRS)         NINTRS
FITEMS(ERUS)          NCLOSS,NFCHKS
FMTOP(NFMTS)          NOUTS
FNCTBS(NFCHKS)        NFTCHS,NOTINS,NSWTCS
IALLS(ERUS)           NINTRS,NSTOPS
INSTAT(NIERS)         DECK280
INS(NOSYMS)           NISYMS
IOCODS(NIERS)         NFCHKS,NFMTS,NFTCHS,NIBUFS,NISYMS,NOBUFS,NOSYMS,NRWNDS,NWEFS
IOERRS(NERRS)         NFCHKS
IOW(IOW)              DECK280
IOWS(ERUS)            IOW,NBSBLS,NCLOSS,NFCHKS,NFTCHS,NIOERS
IOS(ERUS)             NCLOSS,NRBLKS,NWBLKS
LOGNOS(NOBUFS)        NOSYMS
MBS(ERUS)             NBSBLS,NCLOSS,NFCHKS,NFTCHS,NOBUFS,NRWNDS,NUPDAS
MFS(ERUS)             NFTCHS
NAB0S(NFTVS)          NFMTS
NAB1S(NFTVS)          NFMTS
NAB2S(NFTVS)          NFMTS
NAB3S(NFTVS)          NFMTS
NAB4S(NFTVS)          NFMTS
NAB5S(NFTVS)          NFMTS
NAB6S(NFTVS)          NFMTS
NAB7S(NFTVS)          NFMTS
NAVCS(NFTVS)          NFMTS
NBCWS(NIOERS)         NININS
NBFGTS(NFCHKS)        NFTCHS,NOTINS,NWEFS
NBFMGS(NFCHKS)        NFTCHS,NOTINS,NWEFS
NBFRLS(NFCHKS)        NFTCHS,NOTINS,NWEFS
```

TABLE 3B.-MERGE COLLECTION AND SUBROUTINE CROSS-REFERENCE - DECK280 (contd)

ENTRY/BLOCK(ELEMENT)	**** REFERENCED BY ELEMENT ****
NBFRS$(NFCHKS)	NFTCHS,NOTINS
NBIPA$(NINPTS)	NININS,NISYMS
NBIS(NBDCVS)	NOUTS
NBLNK$(NOUTS)	NOBUFS,NOSYMS
NBTOD$(NERRS)	NFCHKS
NCA$(NIERS)	NFMTS,NISYMS,NOSYMS
NCCC$(NOUTS)	NOTINS
NCDOF$(NINTRS)	NCNVTS
NCEF$(NSTOPS)	NCLOSS
NCHAR$(NIERS)	NFMTS,NISYMS,NOSYMS
NCJN102$(NIERS)	NIOERS,NOTINS
NCNV9$(NCNVTS)	NINPTS
NCOM3$(NFMTS)	NCNVTS,NINPTS
NCSP$(NIERS)	NININS,NINPTS,NOTINS,NOUTS
NC1ULO(NIERS)	NBDCVS,NFCHKS,NOTINS
NC1UL1(NIERS)	NBDCVS,NFCHKS
NDBCVS(NCNVTS)	NFMTS,NINPTS
NDBFI$(NCNVTS)	NINPTS
NDBIN$(NCNVTS)	NFMTS,NINPTS
NDBIS(NCNVTS)	NFMTS
NDBLT$(NFMTS)	NINPTS
NDCOD$(NISYMS)	DECK280
NDIGS(NBDCVS)	NOUTS
NDOUTS(NBDCVS)	NOUTS
NEE$(NSTOPS)	NERRS,NINTRS
NEFCL$(NIOERS)	NFTCHS,NININS,NINPTS,NOTINS
NERCR$(NIOERS)	NCNVTS,NFMTS,NINPTS
NERCT$(NIOERS)	NCNVTS,NFTCHS
NERR3$(NERRS)	DECK280
NERR4$(NERRS)	DECK280
NERR6$(NERRS)	DECK280
NERU$(NIOERS)	NFCHKS,NININS,NOBUFS,NOSYMS,NOTINS
NETF$(NOUTS)	NFMTS,NOSYMS
NEXIT$(NOUTS)	NOSYMS,NOTINS
NFAR$(NFMTS)	NINPTS,NOUTS
NFBY1$(NIOERS)	NFTCHS
NFCA$(NCNVTS)	NFMTS
NFCHK$(NFCHKS)	NIBUFS,NOBUFS,NRWNDS,NWEFS
NFCI$(NCNVTS)	NINPTS
NFCM$(NCNVTS)	NINPTS
NFCS$(NCNVTS)	NFMTS
NFDB$(NCNVTS)	NINPTS
NFDP$(NCNVTS)	NBDCVS
NFGC$(NFMTS)	NINPTS,NOUTS
NFGT$(NFMTS)	NINPTS
NFMNG$(NFMTS)	NOUTS
NFMTR$(NCNVTS)	NFMTS
NFMT$(NFMTS)	NIBUFS,NISYMS,NOBUFS,NOSYMS,NOUTS
NFM96$(NFMTS)	NINPTS,NOUTS
NFNIO1D$(NIERS)	NFMTS
NFNIO1$(NIERS)	NFMTS

TABLE 3B.-MERGE COLLECTION AND SUBROUTINE CROSS-REFERENCE - DECK280 (contd)

ENTRY/BLOCK(ELEMENT)	**** REFERENCED BY ELEMENT ****
NFNS1$(NBDCV$)	NOUT$
NFNS2$(NBDCV$)	NOUT$
NFNS3$(NBDCV$)	NOUT$
NFPC$(NIER$)	NOSYM$,NOTIN$,NOUT$
NFPKT$(NFCHK$)	NCLOS$
NFRA$(NFMT$)	NIBUF$,NININ$,NISYM$,NOSYM$,NOTIN$,NOUT$
NFRC$(NFMT$)	NINPT$
NFRG$(NIER$)	NFMT$,NINPT$
NFRH$(NIER$)	NIBUF$,NININ$,NINPT$,NISYM$
NFRJ$(NIER$)	NOSYM$,NOTIN$,NOUT$
NFRONF$(NIOER$)	NFTCH$
NFRZ$(NIER$)	NFMT$,NINPT$,NOUT$
NFRZS$(NIER$)	NFMT$
NFSG$(NCNVT$)	NINPT$
NFTCB$(NFTCH$)	NININ$
NFTCH$(NFTCH$)	NININ$
NFTGL$(NIER$)	NCNVT$,NFMT$,NINPT$
NGC9$(NFMT$)	NINPT$,NOUT$
NHPFA$(NIER$)	NIBUF$,NISYM$,NOBUF$,NOSYM$,NRWND$,NWEF$
NHPFB$(NIER$)	NISYM$,NOSYM$
NHVC$(NFTV$)	NFMT$
NIIC$(NINPT$)	NININ$,NISYM$
NIND$(NBDCV$)	NOUT$
NINI1$(NININ$)	NIBUF$
NINTR$(NINTR$)	DECK280
NIOER$(NIOER$)	NBSBL$,NCLOS$,NFCHK$,NFTCH$,NOTIN$,NRBLK$,NRWND$,NWBLK$,NWEF$
NIOERSA(NIOER$)	NOBUF$
NIO1V$(NIER$)	NFMT$
NIO1$(NIER$)	DECK280,NOSYM$
NIO2V$(NIER$)	NIBUF$,NISYM$,NOBUF$,NOSYM$
NIO2$(NIER$)	DECK280
NIO3VA$(NIER$)	NFMT$
NIO3V$(NIER$)	NFMT$
NIO2$(NIER$)	NFMT$,NIBUF$,NISYM$,NOTIN$
NKLN$(NIER$)	NIBUF$,NININ$,NISYM$,NOSYM$,NOTIN$
NKL2$(NIER$)	NININ$,NOTIN$
NLLC$(NIER$)	NINPT$
NLLM$(NIER$)	NININ$
NLRT$(NIER$)	NIOER$
NLTB$(NIER$)	NIOER$
NNG90$(NIER$)	NIBUF$,NINPT$,NISYM$
NOLM$(NIER$)	NOTIN$
NOTI1$(NOTIN$)	NOBUF$
NPCT$(NFMT$)	NINPT$,NOUT$
NPR$(NOUT$)	NOSYM$,NOTIN$
NPU$(NOUT$)	NOSYM$,NOTIN$
NPW2$(NFMT$)	NINPT$
NP91$(NFMT$)	NINPT$,NOUT$
NRBFA$(NFCHK$)	NWEF$
NRBF$(NFCHK$)	NCLOS$,NRWND$
NRDU$(NIBUF$)	DECK280

TABLE 3B.-MERGE COLLECTION AND SUBROUTINE CROSS-REFERENCE - DECK280 (contd)

ENTRY/BLOCK(ELEMENT)	**** REFERENCED BY ELEMENT ****
NADS(NINPTS)	NININS,NISYMS
NAECS(NIOERS)	NININS,NISYMS,NOSYMS,NOTINS
NAETOS(NERRS)	NINTRS
NAEWS(NRWNDS)	NCLOSS
NAM92S(NIERS)	NOUTS
NASFS(NERRS)	NIOERS,NSTOPS
NASXS(NIERS)	NIBUFS,NISYMS,NOBUFS,NOSYMS
NATRS(NIERS)	NFMTS,NIBUFS,NININS,NINPTS,NISYMS
NAT1S(NFMTS)	NIBUFS,NISYMS,NOBUFS,NOSYMS,NOUTS
NAT2S(NFMTS)	NINPTS,NOUTS
NAT3S(NFMTS)	NIERS,NINPTS,NOUTS
NBAOS(NERRS)	NIOERS
NBFS(NCNVTS)	NINPTS
NBLS(NBDCVS)	NOUTS
NBTATS(NIERS)	NCNVTS,NFCHKS,NFMTS,NFTCHS,NINPTS,NIOERS
NBTOPS(NSTOPS)	DECK280
NBTSVS(NIERS)	NCNVTS,NFMTS,NIBUFS,NINPTS,NOSYMS,NOTINS
NBWTCS(NSWTCS)	NWEFS
NBIIS(NSTOPS)	NCLOSS,NERRS,NFCHKS,NIERS,NIOERS,NRWNDS,NWEFS
NTABS(NTABS)	NBSBLS,NCLOSS,NFCHKS,NFMTS,NFTCHS,NIBUFS,NININS,NIOERS,NISYMS,NOBUFS,NOSYMS,NOTINS,NRBLKS,NRWNDS,NSWTCS,NUPDAS,NWBLKS,NWEFS
NTBSZS(NTABS)	NCLOSS,NFCHKS
NTENDS(NFMTS)	NIOERS,NOTINS
NTSTOS(NFMTS)	NOBUFS,NOSYMS
NTIOS(NFMTS)	NINPTS,NOUTS
NVECS(NFTVS)	NFMTS,NINPTS,NOUTS
NWALKS(NERRS)	NCLOSS,NFCHKS,NIOERS,NOTINS,NRWNDS,NWEFS
NWDUS(NOBUFS)	DECK280
NWEFS(NWEFS)	NCLOSS
NXVCS(NFTVS)	NFMTS
OPTS(ERUS)	NSTOPS
OUTS(NOSYMS)	NISYMS
PACKTS(NIERS)	NFCHKS,NININS,NIOERS,NISYMS,NOBUFS,NOSYMS,NOTINS,NWEFS
PLS(NBFOOS)	NFCHKS
PNCHAS(ERUS)	NOTINS,NWEFS
PPPS(NOUTS)	NOTINS
PRINTS(ERUS)	NCLOSS,NERRS,NFCHKS,NFMTS,NINTRS,NIOERS,NOTINS,NOUTS,NRWNDS,NSTOPS,NWEFS
PRNTAS(ERUS)	NOTINS,NOUTS
PUNCHS(ERUS)	NOUTS,NWEFS
RDBLKS(NRBLKS)	NFTCHS,NWEFS
READAS(ERUS)	NININS
READS(ERUS)	NINPTS
RESTS(NERRS)	NSTOPS
RS(ERUS)	NFTCHS,NRBLKS
SNAPS(ERUS)	NSTOPS
STREGS(NERRS)	NCLOSS,NCNVTS,NFCHKS,NFMTS,NFTCHS,NINPTS,NINTRS,NIOERS,NRWNDS,NWEFS
TEMPS(NIERS)	NFCHKS,NININS,NOTINS
TINTLS(ERUS)	NRWNDS
TPCKS(NTABS)	NOUTS
TSWAPS(ERUS)	NIOERS
UNITS(NIERS)	NCLOSS,NFCHKS,NFTCHS,NININS,NIOERS,NOSYMS,NOTINS,NRBLKS,NWBLKS,NWEFS

TABLE 3B.-MERGE COLLECTION AND SUBROUTINE CROSS-REFERENCE - DECK280 (contd)

```
ENTRY/BLOCK(ELEMENT)   **** REFERENCED BY ELEMENT ****
UPDDAS(NUPDAS)         NBSBLS,NFCHKS,NFTCHS,NOTINS,NRBLKS,NWBLKS,NWEFS
WAITS(ERUS)            NBSBLS,NCLOSS,NFCHKS,NFTCHS,NOBUFS,NOTINS,NRBLKS,NRWNDS,NUPDAS,NWBLKS,NWEFS
WRBLKS(NWBLKS)         NFCHKS,NSWTCS
WS(ERUS)               NCLOSS,NFCHKS,NWBLKS
XFORS(NFMTS)           NOUTS
```

TABLE 3C.-MERGE COLLECTION AND SUBROUTINE CROSS-REFERENCE - LISTMERGE

```
@MAP,XS  ,USER,LISTMERGE
MAP 23-R 05/21-07:19 (,0)
IN  USER.LISTMERGE



ADDRESS LIMITS     001000 013207      040000 045033
STARTING ADDRESS   012475

WORDS DECIMAL        5256 IBANK          2588 DBANK



            SEGMENT MAIN              001000 013207      040000 045033

NSWTC$/FOR                 1    001000 001021
NRBLK$/FOR                 1    001022 001044
NRWND$/FOR                 1    001045 001126        2    040000 040011
NWEF$/FOR                  1    001127 001313        2    040012 040031
NFTCH$/FOR                 1    001314 001627        2    040032 040057
NBDCV$/FOR                 1    001630 001755        2    040060 040122
NFTV$/FOR                  1    001756 002000
NCNVT$/FOR                 1    002001 002222        2    040123 040217
NCLOS$/FOR                 1    002223 002424        2    040220 040251
NWBLK$/FOR                 1    002425 002536
NBSBL$/FOR                 1    002537 002577
NUPDA$/FOR                 1    002600 002633
NBF00$/FOR                                           2    040252 042453
NININ$/FOR                 1    002634 003024        2    042454 042457
NINPT$/FOR                 1    003025 003711        2    042460 042503
NOTIN$/FOR                 1    003712 004214        2    042504 042515
NOUT$/FOR                  1    004215 005341        2    042516 042552
NFMT$/FOR                  1    005342 006225        2    042553 042630
NIOER$/FOR                 1    006226 006442        2    042631 043001
NFCHK$/FOR                 1    006443 007433        2    043002 043155
                                                     4    043156 043227
ERUS
NTAB$/FOR                                            2    043230 043404
ATH$$                                                0    043405 043405
NERR$/FOR                  1    007434 010063        2    043406 043615
NIBUF$/FOR                 1    010064 010123        2    043616 043616
NIER$/FOR                  1    010124 010275        2    043617 043736
NOBUF$/FOR                 1    010276 010337        2    043737 043737
NSTOP$/FOR                 1    010340 010375        2    043740 043754
NINTR$/FOR-JPL             1    010376 010622        2    043755 044040
NTRAN$/FOR-JPL             1    010623 012474        2    044041 044517
BLANKSCOMMON (COMMON BLOCK)
LISTMERGE                  1    012475 013207        0    044520 044664
                           5    BLANKSCOMMON         2    044665 044707
                                                     4    044710 045033
```

TABLE 3C.-MERGE COLLECTION AND SUBROUTINE CROSS-REFERENCE - LISTMERGE (contd)

```
ENTRY/BLOCK(ELEMENT)   **** REFERENCED BY ELEMENT ****
ASTOFF(NIERS)          NOUTS
AWAITS(ERUS)           NTRANS
BLANKSCOMMON(COMMON BLOCK)   LISTMERGE
BLS(NBFOOS)            NFCHKS
BRDS(ERUS)             NTRANS
BSIBLS(NBSBLS)         NFCHKS,NOTINS,NWEFS
B1LS(NBFOOS)           NFCHKS
B1OS(NBFOOS)           NFCHKS
B2LS(NOT-DEFINED)      NFCHKS
B2OS(NOT-DEFINED)      NFCHKS
CENDS(ERUS)            NINTRS
CFE(NFCHKS)            NOTINS
CLOSES(NCLOSS)         NFCHKS,NWEFS
COMS(ERUS)             NSTOPS
CSFS(ERUS)             NCLOSS,NFCHKS,NTRANS
DRAINS(NWBLKS)         NFCHKS,NOTINS,NRWNDS,NWEFS
ERRS(ERUS)             NIOERS,NSTOPS,NTRANS
EXIT(NSTOPS)           NFCHKS,NINTRS
EXITS(ERUS)            NSTOPS,NTRANS
EXTPLS(NSTOPS)         NINTRS
FACILS(ERUS)           NTRANS
FHS1S(NINPTS)          NIBUFS
FHS1OS(NOUTS)          NOBUFS
FHS2S(NINPTS)          NIBUFS
FHS2OS(NOUTS)          NOBUFS
FIELDS(NERRS)          NINTRS
FITEMS(ERUS)           NCLOSS,NFCHKS
FMTOP(NFMTS)           NOUTS
FNCTBS(NFCHKS)         NFTCHS,NOTINS,NSWTCS
FORKS(ERUS)            NTRANS
IALLS(ERUS)            NINTRS,NSTOPS
IOCODS(NIERS)          NFCHKS,NFMTS,NFTCHS,NIBUFS,NOBUFS,NRWNDS,NWEFS
IOERRS(NERRS)          NFCHKS
IOIS(ERUS)             NTRANS
IOWIS(ERUS)            NTRANS
IOWS(ERUS)             NBSBLS,NCLOSS,NFCHKS,NFTCHS,NIOERS,NTRANS
IOXIS(ERUS)            NTRANS
IOS(ERUS)              NCLOSS,NRBLKS,NWBLKS
MBS(ERUS)              NBSBLS,NCLOSS,NFCHKS,NFTCHS,NOBUFS,NRWNDS,NTRANS,NUPDAS
MFS(ERUS)              NFTCHS,NTRANS
NAB0S(NFTVS)           NFMTS
NAB1S(NFTVS)           NFMTS
NAB2S(NFTVS)           NFMTS
NAB3S(NFTVS)           NFMTS
NAB4S(NFTVS)           NFMTS
NAB5S(NFTVS)           NFMTS
NAB6S(NFTVS)           NFMTS
NAB7S(NFTVS)           NFMTS
NAVCS(NFTVS)           NFMTS
NBCWS(NIOERS)          NININS
NBFGTS(NFCHKS)         NFTCHS,NOTINS,NWEFS
```

TABLE 3C.-MERGE COLLECTION AND SUBROUTINE CROSS-REFERENCE - LISTMERGE (contd)

ENTRY/BLOCK(ELEMENT)	**** REFERENCED BY ELEMENT ****
NBFMG$(NFCHK$)	NFTCH$,NOTIN$,NWEF$
NBFRL$(NFCHK$)	NFTCH$,NOTIN$,NWEF$
NBFRS$(NFCHK$)	NFTCH$,NOTIN$
NBIPA$(NINPT$)	NININ$
NBI$(NBDCV$)	NOUT$
NBLNK$(NOUT$)	NOBUF$
NBTOD$(NERR$)	NFCHK$,NTRAN$
NCA$(NIER$)	NFMT$
NCCC$(NOUT$)	NOTIN$
NCDOF$(NINTR$)	NCNVT$
NCEF$(NSTOP$)	NCLOS$
NCHAR$(NIER$)	NFMT$
NCJNI02$(NIER$)	NIOER$,NOTIN$
NCNV9$(NCNVT$)	NINPT$
NCOM3$(NFMT$)	NCNVT$,NINPT$
NCSP$(NIER$)	NININ$,NINPT$,NOTIN$,NOUT$
NCIULD(NIER$)	NBDCV$,NFCHK$,NOTIN$
NCIULI(NIER$)	NBDCV$,NFCHK$
NDBCV$(NCNVT$)	NFMT$,NINPT$
NDBFI$(NCNVT$)	NINPT$
NDBIN$(NCNVT$)	NFMT$,NINPT$
NDBI$(NCNVT$)	NFMT$
NDBLT$(NFMT$)	NINPT$
NDIG$(NBDCV$)	NOUT$
NDOUT$(NBDCV$)	NOUT$
NEE$(NSTOP$)	NERR$,NINTR$
NEFCL$(NIOER$)	NFTCH$,NININ$,NINPT$,NOTIN$
NERCR$(NIOER$)	NCNVT$,NFMT$,NINPT$
NERCT$(NIOER$)	NCNVT$,NFTCH$
NERR4$(NERR$)	LISTMERGE
NERU$(NIOER$)	NFCHK$,NININ$,NOBUF$,NOTIN$
NERIO$(ATH5$)	LISTMERGE
NETF$(NOUT$)	NFMT$
NEXIT$(NOUT$)	NOTIN$
NFAR$(NFMT$)	NINPT$,NOUT$
NFBYI$(NIOER$)	NFTCH$
NFCA$(NCNVT$)	NFMT$
NFCHK$(NFCHK$)	NIBUF$,NOBUF$,NRWND$,NWEF$
NFCI$(NCNVT$)	NINPT$
NFCM$(NCNVT$)	NINPT$
NFCS$(NCNVT$)	NFMT$
NFDB$(NCNVT$)	NINPT$
NFDP$(NCNVT$)	NBDCV$
NFGC$(NFMT$)	NINPT$,NOUT$
NFGT$(NFMT$)	NINPT$
NFMNG$(NFMT$)	NOUT$
NFMTR$(NCNVT$)	NFMT$
NFMT$(NFMT$)	NIBUF$,NOBUF$,NOUT$
NFM96$(NFMT$)	NINPT$,NOUT$
NFNIOID$(NIER$)	NFMT$
NFNIOI$(NIER$)	NFMT$

TABLE 3C.-MERGE COLLECTION AND SUBROUTINE CROSS-REFERENCE - LISTMERGE (contd)

ENTRY/BLOCK(ELEMENT)	**** REFERENCED BY ELEMENT ****
NFNS1S(NBDCVS)	NOUTS
NFNS2S(NBDCVS)	NOUTS
NFNS3S(NBDCVS)	NOUTS
NFPCS(NIERS)	NOTINS,NOUTS
NFPKTS(NFCHKS)	NCLOSS
NFRAS(NFMTS)	NIBUFS,NININS,NOTINS,NOUTS
NFRCS(NFMTS)	NINPTS
NFRGS(NIERS)	NFMTS,NINPTS
NFRHS(NIERS)	NIBUFS,NININS,NINPTS
NFRJS(NIERS)	NOTINS,NOUTS
NFRONFS(NIOERS)	NFTCHS
NFRZS(NIERS)	NFMTS,NINPTS,NOUTS
NFRZSS(NIERS)	NFMTS
NFSGS(NCNVTS)	NINPTS
NFTCBS(NFTCHS)	NININS
NFTCHS(NFTCHS)	NININS
NFTGLS(NIERS)	NCNVTS,NFMTS,NINPTS
NGC9S(NFMTS)	NINPTS,NOUTS
NHPFAS(NIERS)	NIBUFS,NOBUFS,NRWNDS,NWEFS
NHVCS(NFTVS)	NFMTS
NIICS(NINPTS)	NININS
NINDS(NBDCVS)	NOUTS
NINIIS(NININS)	NIBUFS
NINTRS(NINTRS)	LISTMERGE
NIOERS(NIOERS)	NBSBLS,NCLOSS,NFCHKS,NFTCHS,NOTINS,NRBLKS,NRWNDS,NWBLKS,NWEFS
NIOERSA(NIOERS)	NOBUFS
NIO1VS(NIERS)	NFMTS
NIO1S(NIERS)	LISTMERGE
NIO2VS(NIERS)	NIBUFS,NOBUFS
NIO2S(NIERS)	LISTMERGE
NIO3VAS(NIERS)	NFMTS
NIO3VS(NIERS)	NFMTS
NIO2S(NIERS)	NFMTS,NIBUFS,NOTINS
NKLNS(NIERS)	NIBUFS,NININS,NOTINS
NKL2S(NIERS)	NININS,NOTINS
NLLCS(NIERS)	NINPTS
NLLMS(NIERS)	NININS
NLRTS(NIERS)	NIOERS
NLTBS(NIERS)	NIOERS
NNG90S(NIERS)	NIBUFS,NINPTS
NOLMS(NIERS)	NOTINS
NOTIIS(NOTINS)	NOBUFS
NPCTS(NFMTS)	NINPTS,NOUTS
NPRS(NOUTS)	NOTINS
NPUS(NOUTS)	NOTINS
NPW2S(NFMTS)	NINPTS
NP91S(NFMTS)	NINPTS,NOUTS
NRBFAS(NFCHKS)	NWEFS
NRBFS(NFCHKS)	NCLOSS,NRWNDS
NRDUS(NIBUFS)	LISTMERGE
NRDS(NINPTS)	NININS

TABLE 3C.-MERGE COLLECTION AND SUBROUTINE CROSS-REFERENCE - LISTMERGE (contd)

```
ENTRY/BLOCK(ELEMENT)  **** REFERENCED AT ELEMENT ****
NRECS(NIOERS)         NININS,NOTINS
NRETOS(NERRS)         NINTRS
NREWS(NRWNDS)         NCLOSS
NRM92S(NIERS)         NOUTS
NRSFS(NERRS)          NIOERS,NSTOPS,NTRANS
NRSXS(NIERS)          NIBUFS,NOBUFS
NRTRS(NIERS)          NFMTS,NIBUFS,NININS,NINPTS
NR91S(NFMTS)          NIBUFS,NOBUFS,NOUTS
NR92S(NFMTS)          NINPTS,NOUTS
NR93S(NFMTS)          NIERS,NINPTS,NOUTS
NSAOS(NERRS)          NIOERS
NSFS(NCNVTS)          NINPTS
NSLS(NBDCVS)          NOUTS
NSTATS(NIERS)         NCNVTS,NFCHKS,NFMTS,NFTCHS,NINPTS,NIOERS
NSTOPS(NSTOPS)        LISTMERGE
NSTSVS(NIERS)         NCNVTS,NFMTS,NIBUFS,NINPTS,NOTINS
NSWTCS(NSWTCS)        NWEFS
NS11S(NSTOPS)         NCLOSS,NERRS,NFCHKS,NIERS,NIOERS,NRWNDS,NTRANS,NWEFS
NTABS(NTABS)          NBSBLS,NCLOSS,NFCHKS,NFMTS,NFTCHS,NIBUFS,NININS,NIOERS,NOBUFS,NOTINS,NRBLKS,NRWNDS,NSWTCS,NTRANS,NUPDAS,NWBLKS
                      NWEFS
NTBSZS(NTABS)         NCLOSS,NFCHKS,NTRANS
NTENDS(NFMTS)         NIOERS,NOTINS
NTRAN(NTRANS)         LISTMERGE
NTSTOS(NFMTS)         NOBUFS
NTIOS(NFMTS)          NINPTS,NOUTS
NVECS(NFTVS)          NFMTS,NINPTS,NOUTS
NWALKS(NERRS)         NCLOSS,NFCHKS,NIOERS,NOTINS,NRWNDS,NTRANS,NWEFS
NWBUS(NOBUFS)         LISTMERGE
NWEFS(NWEFS)          NCLOSS
NXVCS(NFTVS)          NFMTS
OPTS(ERUS)            NSTOPS
PACKTS(NIERS)         NFCHKS,NININS,NIOERS,NOBUFS,NOTINS,NWEFS
PCTS(ERUS)            NTRANS
PLS(NBFOOS)           NFCHKS
PNCHAS(ERUS)          NOTINS,NWEFS
PPPS(NOUTS)           NOTINS
PRINTS(ERUS)          NCLOSS,NERRS,NFCHKS,NFMTS,NINTRS,NIOERS,NOTINS,NOUTS,NRWNDS,NSTOPS,NTRANS,NWEFS
PRNTAS(ERUS)          NOTINS,NOUTS
PUNCHS(ERUS)          NOUTS,NWEFS
RDBLKS(NRBLKS)        NFTCHS,NWEFS
READAS(ERUS)          NININS
READS(ERUS)           NINPTS
RESTS(NERRS)          NSTOPS
REWIS(ERUS)           NTRANS
REWS(ERUS)            NTRANS
RS(ERUS)              NFTCHS,NRBLKS,NTRANS
SOS(ERUS)             NTRANS
SNS(ERUS)             NTRANS
SNAPS(ERUS)           NSTOPS
SODS(ERUS)            NTRANS
STRECS(NERRS)         NCLOSS,NCNVTS,NFCHKS,NFMTS,NFTCHS,NINPTS,NINTRS,NIOERS,NRWNDS,NTRANS,NWEFS
```

TABLE 3C.-MERGE COLLECTION AND SUBROUTINE CROSS-REFERENCE - LISTMERGE (contd)

ENTRY/BLOCK(ELEMENT)	•••• REFERENCED BY ELEMENT ••••
TEMPS(NIERS)	NFCHKS,NININS,NOTINS
TINTLS(ERUS)	NRWNDS,NTRANS
TPCKS(NTABS)	NOUTS
TSAS(ERUS)	NTRANS
TSWAPS(ERUS)	NIOERS,NTRANS
TWAITS(ERUS)	NTRANS
UNITS(NIERS)	NCLOSS,NFCHKS,NFTCHS,NININS,NIOERS,NOTINS,NRBLKS,NWBLKS,NWEFS
UPDDAS(NUPDAS)	NBSBLS,NFCHKS,NFTCHS,NOTINS,NRBLKS,NWBLKS,NWEFS
WAITS(ERUS)	NBSBLS,NCLOSS,NFCHKS,NFTCHS,NOBUFS,NOTINS,NRBLKS,NRWNDS,NUPDAS,NWBLKS,NWEFS
WEFS(ERUS)	NTRANS
WRBLKS(NWBLKS)	NFCHKS,NSWTCS
WS(ERUS)	NCLOSS,NFCHKS,NTRANS,NWBLKS
XFORS(NFMTS)	NOUTS

3.2 STAT Program Set

The STAT program set consists of two stand-alone programs: STATS and PROFILE. The STATS program reads MERGE data for the same day of all years contained in the file, computes averages for each hour of the day, computes daily averages for each year, monthly averages, and the average yearly minimum and maximum temperature, and writes an intermediate (not directly readable by the DSPA program) statistical file. The format for this file is as follows:

	Word 1	Word 2	Word 3	Word 4	Words 5-28
Sector 1	Station No.	Day	Sector	EOF Day	Hourly Temperature
Sector 2	Yearly T_{min}*	Yearly σ_{min}*	Yearly T_{max}*	Yearly σ_{max}*	Hourly Wind Velocity
Sector 3	No. of Years*	Blank	Blank	Blank	Hourly Solar Insolation
Sector 4	Blank	Blank	Average daily wind over all years	Average daily solar insolation over all years	12 positions for average daily wind, 12 positions for average daily solar insolation

The PROFILE program reads the STATS file and monthly user-input parameters, computes low or high or mean profiles as requested, computes worst case wind and insolation days if required, and produces a STAT file for use as input to the DSPA program. Input of the monthly user parameters is via namelist format as described in the DS/PA User's Manual, Section 4.2. The format of the PROFILE output is the same as that of the MERGE file except that the first four words of the second and third sectors of the first day contain the yearly minimum and maximum temperature data as in the intermediate STATS file.

*Yearly minimum and maximum temperature data is stored only on the day=1 record.

Listings of the STATS and PROFILE programs are provided in Appendix D. A collection and subroutine cross reference table for the two STAT programs is given below.

TABLE 4A.-STAT COLLECTION AND SUBROUTINE CROSS-REFERENCE - STATS

```
@MAP,XS  ,USER,STATS
MAP 23-R 05/21-07:20 (,0)
IN  USER.STATS,.10W



ADDRESS LIMITS     001000 010607      040000 045307
STARTING ADDRESS   007230

WORDS DECIMAL        3976 IBANK          2760 DBANK



            SEGMENT MAIN              001000 010607      040000 045307

NSWTC$/FOR                  1   001000 001021
NRBLK$/FOR                  1   001022 001044
NRWND$/FOR                  1   001045 001126        2   040000 040011
NWEF$/FOR                   1   001127 001313        2   040012 040031
NBDCV$/FOR                  1   001314 001441        2   040032 040074
NFTV$/FOR                   1   001442 001464
NCNVT$/FOR                  1   001465 001706        2   040075 040171
NCLOS$/FOR                  1   001707 002110        2   040172 040223
NWBLK$/FOR                  1   002111 002222
NBSBL$/FOR                  1   002223 002263
NUPDA$/FOR                  1   002264 002317
NBFOO$/FOR                                           2   040224 042425
NOTIN$/FOR                  1   002320 002622        2   042426 042437
NOUT$/FOR                   1   002623 003747        2   042440 042474
NFMT$/FOR                   1   003750 004633        2   042475 042552
NIOER$/FOR                  1   004634 005050        2   042553 042723
NFCHK$/FOR                  1   005051 006041        2   042724 043077
                                                     4   043100 043151
NTAB$/FOR                                            2   043152 043326
ERU$
NERR$/FOR                   1   006042 006471        2   043327 043536
NSTOP$/FOR                  1   006472 006527        2   043537 043553
SQRT$/FOR-JPL               1   006530 006566        2   043554 043560
NIER$/FOR                   1   006567 006740        2   043561 043700
NOBUF$/FOR                  1   006741 007002        2   043701 043701
NINTR$/FOR-JPL              1   007003 007227        2   043702 043765
BLANK$COMMON (COMMON BLOCK)
STATS                       1   007230 010554        0   043766 045276
                                                     2   BLANK$COMMON
IOW                         1   010555 010607        2   045277 045307
```

TABLE 4A.-STAT COLLECTION AND SUBROUTINE CROSS-REFERENCE - STATS (contd)

ENTRY/BLOCK(ELEMENT)	•••• REFERENCED BY ELEMENT ••••
ASTOFF(NIERS)	NOUTS
BLANKSCOMMON(COMMON BLOCK)	STATS
BLS(NBFOOS)	NFCHKS
BSIBLS(NBSBLS)	NFCHKS,NOTINS,NWEFS
B1LS(NBFOOS)	NFCHKS
B1OS(NBFOOS)	NFCHKS
B2LS(NOT-DEFINED)	NFCHKS
B2OS(NOT-DEFINED)	NFCHKS
CENDS(ERUS)	NINTRS
CFE(NFCHKS)	NOTINS
CLOSES(NCLOSS)	NFCHKS,NWEFS
COMS(ERUS)	NSTOPS
CSFS(ERUS)	NCLOSS,NFCHKS
DRAINS(NWBLKS)	NFCHKS,NOTINS,NRWNDS,NWEFS
ERRS(ERUS)	NIOERS,NSTOPS
EXIT(NSTOPS)	NFCHKS,NINTRS
EXITS(ERUS)	NSTOPS
EXTPLS(NSTOPS)	NINTRS
FHS1OS(NOUTS)	NOBUFS
FHS2OS(NOUTS)	NOBUFS
FIELDS(NERRS)	NINTRS
FITEMS(ERUS)	NCLOSS,NFCHKS
FMTOP(NFMTS)	NOUTS
FNCTBS(NFCHKS)	NOTINS,NSWTCS
IALLS(ERUS)	NINTRS,NSTOPS
IOCODS(NIERS)	NFCHKS,NFMTS,NOBUFS,NRWNDS,NWEFS
IOERRS(NERRS)	NFCHKS
IOW(IOW)	STATS
IOWS(ERUS)	IOW,NBSBLS,NCLOSS,NFCHKS,NIOERS
IOS(ERUS)	NCLOSS,NRBLKS,NWBLKS
MBS(ERUS)	NBSBLS,NCLOSS,NFCHKS,NOBUFS,NRWNDS,NUPDAS
NABOS(NFTVS)	NFMTS
NAB1S(NFTVS)	NFMTS
NAB2S(NFTVS)	NFMTS
NAB3S(NFTVS)	NFMTS
NAB4S(NFTVS)	NFMTS
NAB5S(NFTVS)	NFMTS
NAB6S(NFTVS)	NFMTS
NAB7S(NFTVS)	NFMTS
NAVCS(NFTVS)	NFMTS
NBFGTS(NFCHKS)	NOTINS,NWEFS
NBFMGS(NFCHKS)	NOTINS,NWEFS
NBFRLS(NFCHKS)	NOTINS,NWEFS
NBFRSS(NFCHKS)	NOTINS
NBIS(NBDCVS)	NOUTS
NBLNKS(NOUTS)	NOBUFS
NBTODS(NERRS)	NFCHKS
NCAS(NIERS)	NFMTS
NCCCS(NOUTS)	NOTINS
NCDOFS(NINTRS)	NCNVTS
NCEFS(NSTOPS)	NCLOSS

TABLE 4A.-STAT COLLECTION AND SUBROUTINE CROSS-REFERENCE - STATS (contd)

ENTRY/BLOCK(ELEMENT)	**** REFERENCED BY ELEMENT ****
NCHARS(NIERS)	NFMTS
NCJNIO2S(NIERS)	NIOERS,NOTINS
NCOM3S(NFMTS)	NCNVTS
NCSPS(NIERS)	NOTINS,NOUTS
NC1UL0(NIERS)	NBDCVS,NFCHKS,NOTINS
NC1UL1(NIERS)	NBDCVS,NFCHKS
NDBCVS(NCNVTS)	NFMTS
NDBINS(NCNVTS)	NFMTS
NDBIS(NCNVTS)	NFMTS
NDIGS(NBDCVS)	NOUTS
NDOUTS(NBDCVS)	NOUTS
NEES(NSTOPS)	NERRS,NINTRS
NEFCLS(NIOERS)	NOTINS
NERCRS(NIOERS)	NCNVTS,NFMTS
NERCTS(NIOERS)	NCNVTS
NERRAS(NERRS)	SQRTS
NERRS(NERRS)	SQRTS
NERR3S(NERRS)	STATS
NERR4S(NERRS)	STATS
NERUS(NIOERS)	NFCHKS,NOBUFS,NOTINS
NETFS(NOUTS)	NFMTS
NEXITS(NOUTS)	NOTINS
NFARS(NFMTS)	NOUTS
NFCAS(NCNVTS)	NFMTS
NFCHKS(NFCHKS)	NOBUFS,NRWNDS,NWEFS
NFCSS(NCNVTS)	NFMTS
NFDPS(NCNVTS)	NBDCVS
NFGCS(NFMTS)	NOUTS
NFMNGS(NFMTS)	NOUTS
NFMTRS(NCNVTS)	NFMTS
NFMTS(NFMTS)	NOBUFS,NOUTS
NFM96S(NFMTS)	NOUTS
NFNIO1DS(NIERS)	NFMTS
NFNIO1S(NIERS)	NFMTS
NFNS1S(NBDCVS)	NOUTS
NFNS2S(NBDCVS)	NOUTS
NFNS3S(NBDCVS)	NOUTS
NFPCS(NIERS)	NOTINS,NOUTS
NFPKTS(NFCHKS)	NCLOSS
NFRAS(NFMTS)	NOTINS,NOUTS
NFRGS(NIERS)	NFMTS
NFRJS(NIERS)	NOTINS,NOUTS
NFRZS(NIERS)	NFMTS,NOUTS
NFRZSS(NIERS)	NFMTS
NFTGLS(NIERS)	NCNVTS,NFMTS
NGC9S(NFMTS)	NOUTS
NHPFAS(NIERS)	NOBUFS,NRWNDS,NWEFS
NHVCS(NFTVS)	NFMTS
NINDS(NBDCVS)	NOUTS
NINTRS(NINTRS)	STATS
NIOERS(NIOERS)	NBSBLS,NCLOSS,NFCHKS,NOTINS,NRBLKS,NRWNDS,NWBUKS,NWEFS

TABLE 4A.-STAT COLLECTION AND SUBROUTINE CROSS-REFERENCE - STATS (contd)

ENTRY/BLOCK(ELEMENT)	**** REFERENCED BY ELEMENT ****
NIOERSA(NIOERS)	NOBUFS
NIO1VS(NIERS)	NFMTS
NIO13(NIERS)	STATS
NIO2VS(NIERS)	NOBUFS
NIO2S(NIERS)	STATS
NIO3VAS(NIERS)	NFMTS
NIO3VS(NIERS)	NFMTS
NIO2S(NIERS)	NFMTS,NOTINS
NKLNS(NIERS)	NOTINS
NKL2S(NIERS)	NOTINS
NLRTS(NIERS)	NIOERS
NLTBS(NIERS)	NIOERS
NOLMS(NIERS)	NOTINS
NOTI1S(NOTINS)	NOBUFS
NPCTS(NFMTS)	NOUTS
NPRS(NOUTS)	NOTINS
NPUS(NOUTS)	NOTINS
NP91S(NFMTS)	NOUTS
NRBFAS(NFCHKS)	NWEFS
NRBFS(NFCHKS)	NCLOSS,NRWNDS
NRECS(NIOERS)	NOTINS
NRETOS(NERRS)	NINTRS
NREWS(NRWNDS)	NCLOSS
NRM92S(NIERS)	NOUTS
NRSFS(NERRS)	NIOERS,NSTOPS
NRSXS(NIERS)	NOBUFS
NRTRS(NIERS)	NFMTS
NR91S(NFMTS)	NOBUFS,NOUTS
NR92S(NFMTS)	NOUTS
NR93S(NFMTS)	NIERS,NOUTS
NSAOS(NERRS)	NIOERS
NSLS(NBDCVS)	NOUTS
NSTATS(NIERS)	NCNVTS,NFCHKS,NFMTS,NIOERS
NSTOPS(NSTOPS)	STATS
NSTSVS(NIERS)	NCNVTS,NFMTS,NOTINS
NSWTCS(NSWTCS)	NWEFS
NS11S(NSTOPS)	NCLOSS,NERRS,NFCHKS,NIERS,NIOERS,NRWNDS,NWEFS
NTABS(NTABS)	NBSBLS,NCLOSS,NFCHKS,NFMTS,NIOERS,NOBUFS,NOTINS,NRBLKS,NRWNDS,NSWTCS,NUPDAS,NWBLKS,NWEFS
NTBSZS(NTABS)	NCLOSS,NFCHKS
NTENDS(NFMTS)	NIOERS,NOTINS
NTSTOS(NFMTS)	NOBUFS
NT10S(NFMTS)	NOUTS
NVECS(NFTVS)	NFMTS,NOUTS
NWALKS(NERRS)	NCLOSS,NFCHKS,NIOERS,NOTINS,NRWNDS,NWEFS
NWDUS(NOBUFS)	STATS
NWEFS(NWEFS)	NCLOSS
NXVCS(NFTVS)	NFMTS
OPTS(ERUS)	NSTOPS
PACKTS(NIERS)	NFCHKS,NIOERS,NOBUFS,NOTINS,NWEFS
PLS(NBFOOS)	NFCHKS
PNCHAS(ERUS)	NOTINS,NWEFS

TABLE 4A.-STAT COLLECTION AND SUBROUTINE CROSS-REFERENCE - STATS (contd)

ENTRY/BLOCK(ELEMENT)	•••• REFERENCED BY ELEMENT ••••
PPPS(NOUTS)	NOTINS
PRINTS(ERUS)	NCLOSS,NERRS,NFCHKS,NFMTS,NINTRS,NIOERS,NOTINS,NOUTS,NRWNDS,NSTOPS,NWEFS
PRNTAS(ERUS)	NOTINS,NOUTS
PUNCHS(ERUS)	NOUTS,NWEFS
RDBLKS(NRBLKS)	NWEFS
RESTS(NERRS)	NSTOPS
RS(ERUS)	NRBLKS
SNAPS(ERUS)	NSTOPS
SQRT(SQRTS)	STATS
STREGS(NERRS)	NCLOSS,NCNVTS,NFCHKS,NFMTS,NINTRS,NIOERS,NRWNDS,NWEFS
TEMPS(NIERS)	NFCHKS,NOTINS
TINTLS(ERUS)	NRWNDS
TPCKS(NTABS)	NOUTS
TSWAPS(ERUS)	NIOERS
UNITS(NIERS)	NCLOSS,NFCHKS,NIOERS,NOTINS,NRBLKS,NWBLKS,NWEFS
UPDDAS(NUPDAS)	NBSBLS,NFCHKS,NOTINS,NRBLKS,NWBLKS,NWEFS
WAITS(ERUS)	NBSBLS,NCLOSS,NFCHKS,NOBUFS,NOTINS,NRBLKS,NRWNDS,NUPDAS,NWBLKS,NWEFS
WRBLKS(NWBLKS)	NFCHKS,NSWTCS
WS(ERUS)	NCLOSS,NFCHKS,NWBLKS
XFORS(NFMTS)	NOUTS

TABLE 4B.-STAT COLLECTION AND SUBROUTINE CROSS-REFERENCE - PROFILE

```
@MAP,XS  ,USER.PROFILE
MAP 23-R 06/04-09:35 (,0)
 IN  USER.PROFILE,.I@W,.SLUP



ADDRESS LIMITS     001000 016115      040000 050135
STARTING ADDRESS   014063

WORDS DECIMAL        6734 IBANK          4190 DBANK


            SEGMENT MAIN              001000 016115      040000 050135

NSWTCS/FOR                  1   001000 001021
NRBLKS/FOR                  1   001022 001044
NRWNDS/FOR                  1   001045 001126        2   040000 040011
NWEFS/FOR                   1   001127 001313        2   040012 040031
NFTCHS/FOR                  1   001314 001627        2   040032 040057
NINPTS/FOR                  1   001630 002514        2   040060 040103
NFTVS/FOR                   1   002515 002537
NCLOSS/FOR                  1   002540 002741        2   040104 040135
NWBLKS/FOR                  1   002742 003053
NBSBLS/FOR                  1   003054 003114
NUPDAS/FOR                  1   003115 003150
NBFOOS/FOR                                           2   040136 042337
NBDCVS/FOR                  1   003151 003276        2   042340 042402
NCNVTS/FOR                  1   003277 003520        2   042403 042477
NININS/FOR                  1   003521 003711        2   042500 042503
NOTINS/FOR                  1   003712 004214        2   042504 042515
NOUTS/FOR                   1   004215 005341        2   042516 042552
NFMTS/FOR                   1   005342 006225        2   042553 042630
NIOERS/FOR                  1   006226 006442        2   042631 043001
NFCHKS/FOR                  1   006443 007433        2   043002 043155
                                                     4   043156 043227
NTABS/FOR                                            2   043230 043404
ERUS
DEXPS/FOR                   1   007434 007604        2   043405 043440
NEXP9S/FOR                  1   007605 007727        2   043441 043461
NERRS/FOR                   1   007730 010357        2   043462 043671
SQRTS/FOR-JPL               1   010360 010416        2   043672 043676
NSTOPS/FOR                  1   010417 010454        2   043677 043713
NLOUTS/FOR                  1   010455 011530        2   043714 043751
NLINPS/FOR                  1   011531 013421        2   043752 044156
NIERS/FOR                   1   013422 013573        2   044157 044276
NOBUFS/FOR                  1   013574 013635        2   044277 044277
NINTRS/FOR-JPL              1   013636 014062        2   044300 044363
BLANKSCOMMON (COMMON BLOCK)
PROFILE                     1   014063 015545        0   044364 050064
                                                     2   BLANKSCOMMON
I@W                         1   015546 015600        2   050065 050075
```

TABLE 4B.-STAT COLLECTION AND SUBROUTINE CROSS-REFERENCE - PROFILE (contd)

```
SLUP
                    1   015601 016115        0   050076 050135
                                              2   BLANKSCOMMON
```

TABLE 4B.-STAT COLLECTION AND SUBROUTINE CROSS-REFERENCE - PROFILE (contd)

```
ENTRY/BLOCK(ELEMENT)   **** REFERENCED BY ELEMENT ****
ASTOFF(NIER$)          NOUT$
BLANK$COMMON(COMMON BLOCK)   PROFILE,SLUP
BL$(NBF00$)            NFCHK$
BS1BL$(NBSBL$)         NFCHK$,NOTIN$,NWEF$
B1L$(NBF00$)           NFCHK$
B10$(NBF00$)           NFCHK$
B2L$(NOT-DEFINED)      NFCHK$
B20$(NOT-DEFINED)      NFCHK$
CEND$(ERU$)            NINTR$
CFE(NFCHK$)            NOTIN$
CLOSE$(NCLOS$)         NFCHK$,NWEF$
COM$(ERU$)             NSTOP$
CSF$(ERU$)             NCLOS$,NFCHK$
DEXP(DEXP$)            PROFILE
DRAIN$(NWBLK$)         NFCHK$,NOTIN$,NRWND$,NWEF$
ERR$(ERU$)             NIOER$,NSTOP$
EXIT(NSTOP$)           NFCHK$,NINTR$
EXIT$(ERU$)            NSTOP$
EXTPL$(NSTOP$)         NINTR$
FHS10$(NOUT$)          NOBUF$
FHS20$(NOUT$)          NOBUF$
FIELD$(NERR$)          NINTR$
FITEM$(ERU$)           NCLOS$,NFCHK$
FMTOP(NFMT$)           NOUT$
FNCTB$(NFCHK$)         NFTCH$,NOTIN$,NSTC$
IALL$(ERU$)            NINTR$,NSTOP$
IOCOD$(NIER$)          NFCHK$,NFMT$,NFTCH$,NLINP$,NLOUT$,NOBUF$,NRWND$,NWEF$
IOERR$(NERR$)          NFCHK$
IOW(IOW)               PROFILE
IOW$(ERU$)             IOW,NBSBL$,NCLOS$,NFCHK$,NFTCH$,NIOER$
IO$(ERU$)              NCLOS$,NRBLK$,NWBLK$
LOGNO$(NOBUF$)         NLOUT$
MB$(ERU$)              NBSBL$,NCLOS$,NFCHK$,NFTCH$,NOBUF$,NRWND$,NUPDA$
MF$(ERU$)              NFTCH$
NAB0$(NFTV$)           NFMT$
NAB1$(NFTV$)           NFMT$
NAB2$(NFTV$)           NFMT$
NAB3$(NFTV$)           NFMT$
NAB4$(NFTV$)           NFMT$
NAB5$(NFTV$)           NFMT$
NAB6$(NFTV$)           NFMT$
NAB7$(NFTV$)           NFMT$
NAVC$(NFTV$)           NFMT$
NBCW$(NIOER$)          NININ$
NBFGT$(NFCHK$)         NFTCH$,NOTIN$,NWEF$
NBFMG$(NFCHK$)         NFTCH$,NOTIN$,NWEF$
NBFRL$(NFCHK$)         NFTCH$,NOTIN$,NWEF$
NBFRS$(NFCHK$)         NFTCH$,NOTIN$
NBIPA$(NINPT$)         NININ$
NBI$(NBDCV$)           NLOUT$,NOUT$
NBLNK$(NOUT$)          NOBUF$
```

TABLE 4B.-STAT COLLECTION AND SUBROUTINE CROSS-REFERENCE - PROFILE (contd)

ENTRY/BLOCK(ELEMENT)	**** REFERENCED BY ELEMENT ****
NBTOD$(NERR$)	NFCHK$
NCA$(NIER$)	NFMT$
NCCC$(NOUT$)	NOTIN$
NCDOF$(NINTR$)	NCNVT$
NCEF$(NSTOP$)	NCLOS$
NCHAR$(NIER$)	NFMT$
NCJNI02$(NIER$)	NIOER$,NLOUT$,NOTIN$
NCNV9$(NCNVT$)	NINPT$,NLINP$
NCOM3$(NFMT$)	NCNVT$,NINPT$
NCSP$(NIER$)	NININ$,NINPT$,NLINP$,NLOUT$,NOTIN$,NOUT$
NC1ULO(NIER$)	NBDCV$,NFCHK$,NLINP$,NLOUT$,NOTIN$
NC1UL1(NIER$)	NBDCV$,NFCHK$,NLINP$
NDBCV$(NCNVT$)	NFMT$,NINPT$,NLINP$
NDBFI$(NCNVT$)	NINPT$,NLINP$
NDBIN$(NCNVT$)	NFMT$,NINPT$,NLINP$
NDBIS(NCNVT$)	NFMT$,NLINP$
NDBLT$(NFMT$)	NINPT$
NDIG$(NBDCV$)	NLOUT$,NOUT$
NDOUT$(NBDCV$)	NLOUT$,NOUT$
NEE$(NSTOP$)	NERR$,NINTR$
NEFCL$(NIOER$)	NFTCH$,NININ$,NINPT$,NOTIN$
NERCR$(NIOER$)	NCNVT$,NFMT$,NINPT$,NLINP$
NERCT$(NIOER$)	NCNVT$,NFTCH$,NLINP$
NERRA$(NERR$)	DEXP$,NEXP9$,SQRT$
NERRB$(NERR$)	DEXP$,NEXP9$
NERRC$(NERR$)	NEXP9$
NERR$(NERR$)	SQRT$
NERR3$(NERR$)	PROFILE,SLUP
NERU$(NIOER$)	NFCHK$,NININ$,NLOUT$,NOBUF$,NOTIN$
NETF$(NOUT$)	NFMT$
NEXIT$(NOUT$)	NOTIN$
NEXP9$(NEXP9$)	PROFILE
NFAR$(NFMT$)	NINPT$,NOUT$
NFBY1$(NIOER$)	NFTCH$
NFCA$(NCNVT$)	NFMT$
NFCHK$(NFCHK$)	NLINP$,NLOUT$,NOBUF$,NRWND$,NWEF$
NFCI$(NCNVT$)	NINPT$,NLINP$
NFCM$(NCNVT$)	NINPT$,NLINP$
NFCS$(NCNVT$)	NFMT$
NFDB$(NCNVT$)	NINPT$,NLINP$
NFDP$(NCNVT$)	NBDCV$
NFGC$(NFMT$)	NINPT$,NOUT$
NFGT$(NFMT$)	NINPT$
NFMNG$(NFMT$)	NOUT$
NFMTR$(NCNVT$)	NFMT$
NFMT$(NFMT$)	NOBUF$,NOUT$
NFM96$(NFMT$)	NINPT$,NOUT$
NFNI01D$(NIER$)	NFMT$
NFNI01$(NIER$)	NFMT$
NFNS1$(NBDCV$)	NOUT$
NFNS2$(NBDCV$)	NLOUT$,NOUT$

TABLE 4B.-STAT COLLECTION AND SUBROUTINE CROSS-REFERENCE - PROFILE (contd)

ENTRY/BLOCK(ELEMENT)	**** REFERENCED BY ELEMENT ****
NFNS3$(NBDCV$)	NLOUT$,NOUT$
NFPC$(NIER$)	NLOUT$,NOTIN$,NOUT$
NFPKT$(NFCHK$)	NCLOS$
NFRA$(NFMT$)	NININ$,NLINP$,NOTIN$,NOUT$
NFRC$(NFMT$)	NINPT$
NFRG$(NIER$)	NFMT$,NINPT$
NFRH$(NIER$)	NININ$,NINPT$,NLINP$
NFRJ$(NIER$)	NLOUT$,NOTIN$,NOUT$
NFRONF$(NIOER$)	NFTCH$
NFRZ$(NIER$)	NFMT$,NINPT$,NOUT$
NFRZ$$(NIER$)	NFMT$
NFSG$(NCNVT$)	NINPT$,NLINP$
NFTCB$(NFTCH$)	NININ$
NFTCH$(NFTCH$)	NININ$
NFTGL$(NIER$)	NCNVT$,NFMT$,NINPT$
NGC9$(NFMT$)	NINPT$,NOUT$
NHPFA$(NIER$)	NLINP$,NLOUT$,NOBUF$,NRWND$,NWEF$
NHVC$(NFTV$)	NFMT$
NIIC$(NINPT$)	NININ$
NIND$(NBDCV$)	NLOUT$,NOUT$
NINI1$(NININ$)	NLINP$
NINTR$(NINTR$)	PROFILE
NIOER$(NIOER$)	NBSBL$,NCLOS$,NFCHK$,NFTCH$,NOTIN$,NRBLK$,NRWND$,NWBLK$,NWEF$
NIOERSA(NIOER$)	NOBUF$
NIO1V$(NIER$)	NFMT$
NIO1$(NIER$)	PROFILE
NIO2V$(NIER$)	NOBUF$
NIO2$(NIER$)	PROFILE
NIO3VA$(NIER$)	NFMT$
NIO3V$(NIER$)	NFMT$
NIO2$(NIER$)	NFMT$,NOTIN$
NKLN$(NIER$)	NININ$,NLINP$,NOTIN$
NKL2$(NIER$)	NININ$,NLINP$,NOTIN$
NLIO$(NIER$)	NLINP$,NLOUT$
NLLC$(NIER$)	NINPT$,NLINP$
NLLM$(NIER$)	NININ$,NLINP$
NLRT$(NIER$)	NIOER$,NLINP$,NLOUT$
NLTB$(NIER$)	NIOER$,NLINP$,NLOUT$
NNG90$(NIER$)	NINPT$,NLINP$
NNRSX$(NIER$)	NLINP$,NLOUT$
NOLC$(NIER$)	NLOUT$
NOLM$(NIER$)	NOTIN$
NOTI1$(NOTIN$)	NLOUT$,NOBUF$
NPCT$(NFMT$)	NINPT$,NOUT$
NPR$(NOUT$)	NOTIN$
NPU$(NOUT$)	NOTIN$
NPW2$(NFMT$)	NINPT$
NP91$(NFMT$)	NINPT$,NOUT$
NRBFA$(NFCHK$)	NWEF$
NRBF$(NFCHK$)	NCLOS$,NRWND$
NRD$(NINPT$)	NININ$

TABLE 4B.-STAT COLLECTION AND SUBROUTINE CROSS-REFERENCE - PROFILE (contd)

ENTRY/BLOCK(ELEMENT)	**** REFERENCED BY ELEMENT ****
NRECS(NIOERS)	NININS,NLINPS,NOTINS
NRETOS(NERRS)	NINTRS
NREWS(NRWNDS)	NCLOSS
NRM92S(NIERS)	NLOUTS,NOUTS
NRNLS(NLINPS)	PROFILE
NRSFS(NERRS)	NIOERS,NSTOPS
NRSXS(NIERS)	NLINPS,NLOUTS,NOBUFS
NRTRS(NIERS)	NFMTS,NININS,NINPTS,NLINPS
NR91S(NFMTS)	NOBUFS,NOUTS
NR92S(NFMTS)	NINPTS,NOUTS
NR93S(NFMTS)	NIERS,NINPTS,NOUTS
NSAOS(NERRS)	NIOERS
NSFS(NCNVTS)	NINPTS
NSLS(NRDCVS)	NLOUTS,NOUTS
NSTATS(NIERS)	NCNVTS,NFCHKS,NFMTS,NFTCHS,NINPTS,NIOERS,NLINPS
NSTOPS(NSTOPS)	PROFILE
NSTSVS(NIERS)	NCNVTS,NFMTS,NINPTS,NLINPS,NOTINS
NSWTCS(NSWTCS)	NWEFS
NS11S(NSTOPS)	NCLOSS,NERRS,NFCHKS,NIERS,NIOERS,NRWNDS,NWEFS
NTABS(NTABS)	NBSBLS,NCLOSS,NFCHKS,NFMTS,NFTCHS,NININS,NIOERS,NLINPS,NLOUTS,NOBUFS,NOTINS,NRBLKS,NRWNDS,NSWTCS,NUPDAS,NWBLKS NWEFS
NTBSZS(NTABS)	NCLOSS,NFCHKS
NTENDS(NFMTS)	NIOERS,NLOUTS,NOTINS
NTSTOS(NFMTS)	NOBUFS
NTIOS(NFMTS)	NINPTS,NOUTS
NVECS(NFTVS)	NFMTS,NINPTS,NOUTS
NWALKS(NERRS)	NCLOSS,NFCHKS,NIOERS,NOTINS,NRWNDS,NWEFS
NWDUS(NOBUFS)	PROFILE
NWEFS(NWEFS)	NCLOSS
NWNLS(NLOUTS)	PROFILE
NXVCS(NFTVS)	NFMTS
OPTS(ERUS)	NSTOPS
PACKTS(NIERS)	NFCHKS,NININS,NIOERS,NOBUFS,NOTINS,NWEFS
PLS(NBFOOS)	NFCHKS
PNCHAS(ERUS)	NOTINS,NWEFS
PPPS(NOUTS)	NOTINS
PRINTS(ERUS)	NCLOSS,NERRS,NFCHKS,NFMTS,NINTRS,NIOERS,NOTINS,NOUTS,NRWNDS,NSTOPS,NWEFS
PRNTAS(ERUS)	NOTINS,NOUTS
PUNCHS(ERUS)	NOUTS,NWEFS
RDBLKS(NRBLKS)	NFTCHS,NWEFS
READAS(ERUS)	NININS
READS(ERUS)	NINPTS
RESTS(NERRS)	NSTOPS
RS(ERUS)	NFTCHS,NRBLKS
SLUP(SLUP)	PROFILE
SNAPS(ERUS)	NSTOPS
SQRT(SQRTS)	PROFILE
STREGS(NERRS)	NCLOSS,NCNVTS,NFCHKS,NFMTS,NFTCHS,NINPTS,NINTRS,NIOERS,NLINPS,NRWNDS,NWEFS
TEMPS(NIERS)	NFCHKS,NININS,NOTINS
TINTLS(ERUS)	NRWNDS
TPCKS(NTABS)	NOUTS

TABLE 4B.-STAT COLLECTION AND SUBROUTINE CROSS-REFERENCE - PROFILE (contd)

```
ENTRY/BLOCK(ELEMENT)    **** REFERENCED BY ELEMENT ****
TSWAPS(ERUS)            NIOERS
UNITS(NIERS)            NCLOSS,NFCHKS,NFTCHS,NININS,NIOERS,NLINPS,NOTINS,NRBLKS,NWRLKS,NWEFS
UPDDAS(NUPDAS)          NBSBLS,NFCHKS,NFTCHS,NOTINS,NRBLKS,NWRLKS,NWEFS
WAITS(ERUS)             NBSBLS,NCLOSS,NFCHKS,NFTCHS,NORUFS,NOTINS,NRBLKS,NRWADS,NUPDAS,NWBLKS,NWEFS
WRBLKS(NWBLKS)          NFCHKS,NSWTCS
WS(ERUS)                NCLOSS,NFCHKS,NWRLKS
XFORS(NFMTS)            NOUTS
```

APPENDIX A

GLOSSARY

DSPA	Design Synthesis/Performance Analysis
I-V	Current-Voltage
NOAA	National Oceanic and Atmospheric Administration

APPENDIX B

PROGRAM LISTING

ACOMONS

```
ALLCMN• PROC
      PARAMETER  NIVB=9, NPCDG=51, NPSG=160, NSL=20
      PARAMETER  NQB=21, NTB=6, NESG=NIVB+1
      PARAMETER  NDIF=NPCDG+NPSG
      REAL  MSAPWR
      COMMON/DSPA/ ACTRL,APPSC,ATMEXC,AVGRL,BCUR,CB,CCM,DATEM,
     •       DATE,DECL,DIFIV(NDIF,2),DL,ET,HOURT,MFINAL,MSAPWR,
     •       NAPSG,ND,PBATT,QB,QBATT(NQB),QDT,RL,SDF,SI(NPSG),
     •       SPGR,SRT,SST,SV(NPSG),TBATT(NTB),TBFRZ,TCZ,THETLA,
     •       TIMEH,TRESI(NESG),TRESLT(NESG,2),TRESV(NESG),TSAC,
     •       TSAF,TTAMB,TTESG,TTPCD,VBAT,VBATT(NIVB,NQB,NTB),
     •       VBUS,VINCIV,VVOC,V2(NPSG),XI(NPCDG),XIB(NIVB),
     •       XIBATT(NIVB),XICHMX,XIISC,XI2(NPSG),XN,XV(NPCDG),
     •       XVB(NIVB),ZI(NSL),ZRFI,ZRFV,ZV(NSL)
 END
COMON• PROC
      PARAMETER  NBETAB=16, NETA=21, NTCZIV=10
      PARAMETER  NVCHI=10, NVLB=10, NXIHI=10, NZDIMP=10
      DIMENSION  TLL1(16), TLO(16,15)
      DATA  PI/3.14159265/
      DATA  DEGRAD/0.01745329/          @ DEGRAD = PI / 180.0
      DATA  OMEGA/0.0172028/            @ OMEGA = 2.0 • PI / 365.242
      COMMON/INPUT1/ AA(NETA,5,5),ACSTD,BETAB(NBETAB,8),BI(5),
     •       BTEMP(NBETAB),CLSIT(6),CLST(6,7),CLSTT(7),FA(7,5),MERGE,
     •       NBTEMP,NCDEG,NCLSI,NCLST,NCTEMP,NDTEMP,NRD,NROE,NRSCEL,
     •       NSOC,NSUNMW,NVDEG,NWRT,PO(3,2),P1(3,2),P2(3,2),P3(3,2),
     •       ROE(23),RSCELL(26),SADEGC(36,2),SADEGV(36,2),SOC(NETA),
     •       SUNLIT(23),SUNMW(8),TCSTD,TEMTAB(26),TL(16,16),TP(5),
     •       VV(30),XCSTD,XII(30)
      COMMON/INPUT2/ ACELL,CDEGA,CDEGB,CLR,CN,CURZ(10,2),
     •       DEBUG,DTAMBI(25,2),DTTAI(366,2),DTTESG,DTTPCD,
     •       DTTPSG,NDER,NDZMX,ICHRT,IFTYPE,IPSG,ISH,ITAPE,
     •       NBATT,NCURZ,NDTAMB,NDTTA,NESP,NP,NS,NTCZT,NTCZV,
     •       NVCHIS,NVCHIO,NVCHT,NVCHV,NVLBT,NVLBV,NVRISA,
     •       NVRIO,NXIHT,NXIHV,NZDT,NZDV,NZRA,NZRS,NZS,NZSH,
     •       NZTC,PHIAAD,PHIAID,QOFF,QON,REFLH,RLL,SPECOR,
     •       TCZIV(10,10),TCZT(10),TCZV(10),THELAD,THELOD,
     •       TSHREF,TTAVE,TZBR,TZN,VCHIST(10,2),VCHIT(10,10),
     •       VCHIOT(10,2),VCHTT(10),VCHVT(10),VBUSMN,VDEGA,
     •       VLBT(10,10),VLBTT(10),VLBVT(10),VMAXIV,VMINIV,
     •       VLR,VRISAT(10,2),VRIOT(10,2),VSAINC,VSHTOR,VZBR,
     •       XIHIT(10,10),XIHTT(10),XIHVT(10),ZDIMP(10,10),
     •       ZDIMPT(10),ZDIMPV(10),ZRAT(10,2),ZRST(10,2),
     •       ZSHTAB(10,2),ZTCOEF(10,2)
      EQUIVALENCE  (TLL1(1),TL(1,1)), (TLO(1,1),TL(1,2))
 END
DSCMN• PROC
      PARAMETER  NDCDCH=10, NDCDE=10, NDCDNZ=10, NDCDPS=10
      COMMON/DS/ BRCEST,BRCHMX,BRDEST,BRDSTD,CBAVAL(30),CBMAX,CELPAC,
     •       DCDAT(10,2),DCDCHT(10,10),DCDCNT(10),DCDCPT(10),DCDCT(10),
     •       DCDET(10,10),DCDNNT(10,2),DCDNPT(10,2),DCDNZT(10,10,2),
     •       DCDNT(10),DCDPNT(10),DCDPPT(10),DCDPST(10,10),DODT(10,2),
     •       DURAM,DWDAT(10,2),DWDCHT(10,2),DWDET(10,2),DWDPST(10,2),
     •       DWDNZT(10,2,2),FRCELL,INDFLS,NBATP,NDCDA,NDCDC,NDCDCN,
     •       NDCDCP,NDCDN,NDCDNN,NDCDNP,NDCDPN,NDCDPP,NDOD,NDWDA,NDWDE,
     •       NDWDCH,NDWDNZ,NDWDPS,NPREQ,NSAP,QBRES,SARES,TBDSTD
```

```
 END
PACMN• PROC
       REAL  MARSA
       COMMON/PA/ AD1(15,2),AD2(8,2),CSH,DAYSST,MARSA,NAD1,NAD2,NPLT,
      •        NSPGR,NTBFRZ,NZCHRA,NZCHRS,PESG,PPCD,PPSG,PSA,PSL,RSA,
      •        SPGR1(10,2),TBFRZ1(10,2),VSA,XIEC,XIPCD,XIPSG,XISA,XITT,
      •        XIZ,XPLT(10),YEAR,ZCHRAT(10,2),ZCHRST(10,2)
 END
SUMCMN• PROC
       INTEGER  SFILE
       COMMON/CMNSUM/ SFILE,ISIZE,XLN,YLN,
      •        IYEAR,IDAY,TIME,PRT1(8),PRT2(11),PRT3(11)
 END
```

BLKDTA

```
      COMPILER  (DATA=IBM)
      BLOCK DATA
      INCLUDE   ALLCMN,LIST
      INCLUDE   COMON,LIST
      INCLUDE   DSCMN,LIST
      INCLUDE   PACMN,LIST
      INCLUDE   SUMCMN,LIST
      DATA  AA(1,1,1)/1.000,0.998,0.995,0.992,0.988,0.984,0.976,
     .                0.968,0.960,0.950,0.930,0.920,0.910,0.900,
     .                0.890,0.880,0.870,0.850,0.830,0.787,0.000/
      DATA  AA(1,2,1)/1.000,0.998,0.995,0.992,0.988,0.984,0.976,
     .                0.968,0.960,0.950,0.930,0.920,0.910,0.900,
     .                0.890,0.880,0.870,0.850,0.830,0.787,0.000/
      DATA  AA(1,3,1)/1.000,0.998,0.995,0.992,0.988,0.984,0.976,
     .                0.968,0.960,0.950,0.930,0.920,0.910,0.900,
     .                0.890,0.880,0.870,0.850,0.830,0.787,0.000/
      DATA  AA(1,4,1)/1.000,0.976,0.956,0.940,0.930,0.920,0.910,
     .                0.900,0.880,0.860,0.833,0.817,0.800,0.782,
     .                0.765,0.747,0.730,0.685,0.640,0.523,0.000/
      DATA  AA(1,5,1)/1.000,0.835,0.670,0.610,0.585,0.560,0.540,
     .                0.525,0.510,0.490,0.470,0.460,0.435,0.422,
     .                0.410,0.396,0.375,0.354,0.300,0.208,0.000/
      DATA  AA(1,1,2)/1.000,0.998,0.995,0.992,0.988,0.984,0.976,
     .                0.968,0.960,0.950,0.930,0.920,0.910,0.900,
     .                0.890,0.880,0.870,0.850,0.830,0.787,0.000/
      DATA  AA(1,2,2)/1.000,0.998,0.995,0.992,0.988,0.984,0.976,
     .                0.968,0.960,0.950,0.930,0.920,0.910,0.900,
     .                0.890,0.880,0.870,0.850,0.830,0.787,0.000/
      DATA  AA(1,3,2)/1.000,0.998,0.995,0.992,0.988,0.984,0.976,
     .                0.968,0.960,0.950,0.930,0.920,0.910,0.900,
     .                0.890,0.880,0.870,0.850,0.830,0.787,0.000/
      DATA  AA(1,4,2)/1.000,0.976,0.956,0.940,0.930,0.920,0.910,
     .                0.900,0.880,0.860,0.833,0.817,0.800,0.782,
     .                0.765,0.747,0.730,0.685,0.640,0.523,0.000/
      DATA  AA(1,5,2)/1.000,0.835,0.670,0.610,0.585,0.560,0.540,
     .                0.525,0.510,0.490,0.470,0.460,0.435,0.422,
     .                0.410,0.396,0.375,0.354,0.300,0.208,0.000/
      DATA  AA(1,1,3)/1.000,0.998,0.995,0.992,0.988,0.984,0.976,
     .                0.968,0.960,0.950,0.930,0.920,0.910,0.900,
     .                0.890,0.880,0.870,0.850,0.830,0.787,0.000/
      DATA  AA(1,2,3)/1.000,0.998,0.995,0.992,0.988,0.984,0.976,
     .                0.968,0.960,0.950,0.930,0.920,0.910,0.900,
     .                0.890,0.880,0.870,0.850,0.830,0.787,0.000/
      DATA  AA(1,3,3)/1.000,0.998,0.995,0.992,0.988,0.984,0.976,
     .                0.968,0.960,0.950,0.930,0.920,0.910,0.900,
     .                0.890,0.880,0.870,0.850,0.830,0.787,0.000/
      DATA  AA(1,4,3)/1.000,0.976,0.956,0.940,0.930,0.920,0.910,
     .                0.900,0.880,0.860,0.833,0.817,0.800,0.782,
     .                0.765,0.747,0.730,0.685,0.640,0.523,0.000/
      DATA  AA(1,5,3)/1.000,0.835,0.670,0.610,0.585,0.560,0.540,
     .                0.525,0.510,0.490,0.470,0.460,0.435,0.422,
     .                0.410,0.396,0.375,0.354,0.300,0.208,0.000/
      DATA  AA(1,1,4)/1.000,0.998,0.995,0.992,0.988,0.984,0.976,
     .                0.968,0.960,0.950,0.930,0.920,0.910,0.900,
     .                0.890,0.880,0.870,0.850,0.830,0.787,0.000/
      DATA  AA(1,2,4)/1.000,0.998,0.995,0.992,0.988,0.984,0.976,
```

```
     .                0.968,0.960,0.950,0.930,0.920,0.910,0.900,
     .                0.890,0.880,0.870,0.850,0.830,0.787,0.000/
      DATA   AA(1,3,4)/1.000,0.998,0.995,0.992,0.988,0.984,0.976,
     .                0.968,0.960,0.950,0.930,0.920,0.910,0.900,
     .                0.890,0.880,0.870,0.850,0.830,0.787,0.000/
      DATA   AA(1,4,4)/1.000,0.976,0.956,0.940,0.930,0.920,0.910,
     .                0.900,0.880,0.860,0.833,0.817,0.800,0.782,
     .                0.765,0.747,0.730,0.685,0.640,0.523,0.000/
      DATA   AA(1,5,4)/1.000,0.835,0.670,0.610,0.585,0.560,0.540,
     .                0.525,0.510,0.490,0.470,0.460,0.435,0.422,
     .                0.410,0.396,0.375,0.354,0.300,0.208,0.000/
      DATA   AA(1,1,5)/1.000,0.998,0.995,0.992,0.988,0.984,0.976,
     .                0.968,0.960,0.950,0.930,0.920,0.910,0.900,
     .                0.890,0.880,0.870,0.850,0.830,0.787,0.000/
      DATA   AA(1,2,5)/1.000,0.998,0.995,0.992,0.988,0.984,0.976,
     .                0.968,0.960,0.950,0.930,0.920,0.910,0.900,
     .                0.890,0.880,0.870,0.850,0.830,0.787,0.000/
      DATA   AA(1,3,5)/1.000,0.998,0.995,0.992,0.988,0.984,0.976,
     .                0.968,0.960,0.950,0.930,0.920,0.910,0.900,
     .                0.890,0.880,0.870,0.850,0.830,0.787,0.000/
      DATA   AA(1,4,5)/1.000,0.976,0.956,0.940,0.930,0.920,0.910,
     .                0.900,0.880,0.860,0.833,0.817,0.800,0.782,
     .                0.765,0.747,0.730,0.685,0.640,0.523,0.000/
      DATA   AA(1,5,5)/1.000,0.835,0.670,0.610,0.585,0.560,0.540,
     .                0.525,0.510,0.490,0.470,0.460,0.435,0.422,
     .                0.410,0.396,0.375,0.354,0.300,0.208,0.000/
      DATA   ACSTD/4.0/
      DATA   AD1/0.58,0.6,0.65,0.7,0.75,0.775,0.8,0.825,
     1          0.85,0.875,0.9,0.925,0.95,0.975,1.0,
     2          0.0,0.05,0.2,0.4,0.88,1.18,1.5,1.91,
     2          2.4,2.97,3.70,4.35,5.3,6.2,7.7/
      DATA   AD2/0.75,0.775,0.8,0.825,0.85,0.9,0.95,1.0,
     .          0.0,2.0,4.0,6.3,8.8,14.4,22.3,30.0/
      DATA   BETAB(1,1)/2.235,2.209,2.199,2.198,2.202,2.216,2.230,2.244,
     .                  2.258,2.272,2.286,2.300,2.314,2.328,2.342,2.356/
      DATA   BETAB(1,2)/2.245,2.219,2.200,2.192,2.196,2.210,2.224,2.238,
     .                  2.252,2.266,2.280,2.294,2.308,2.322,2.336,2.350/
      DATA   BETAB(1,3)/2.158,2.185,2.211,2.229,2.234,2.231,2.219,2.196,
     .                  2.164,2.123,2.081,2.058,1.875,1.680,1.490,1.300/
      DATA   BETAB(1,4)/2.295,2.318,2.338,2.353,2.356,2.344,2.309,2.273,
     .                  2.237,2.201,2.165,2.142,1.977,1.800,1.635,1.463/
      DATA   BETAB(1,5)/2.338,2.360,2.378,2.392,2.395,2.382,2.340,2.303,
     .                  2.270,2.243,2.222,2.199,2.034,1.871,1.709,1.544/
      DATA   BETAB(1,6)/2.375,2.400,2.420,2.440,2.445,2.431,2.392,2.355,
     .                  2.327,2.305,2.283,2.260,2.095,1.932,1.770,1.605/
      DATA   BETAB(1,7)/2.406,2.428,2.450,2.462,2.475,2.456,2.423,2.394,
     .                  2.367,2.341,2.316,2.283,2.120,1.958,1.795,1.630/
      DATA   BETAB(1,8)/2.928,2.950,2.971,2.992,3.000,2.986,2.960,2.931,
     .                  2.898,2.863,2.826,2.786,2.620,2.440,2.290,2.140/
      DATA   BI/0.0001,0.001,0.005,0.5,1.0/
      DATA   BTEMP/160.0,140.0,120.0,100.0,80.0,60.0,40.0,20.0,0.0,
     .            -20.0,-40.0,-60.0,-80.0,-100.0,-120.0,-140.0/
      DATA   CLSIT/0.25,0.55,0.77,1.17,2.03,3.05/
      DATA   CLST/1.112,1.162,1.190,1.235,1.360,1.430,
     .            1.084,1.129,1.161,1.200,1.291,1.361,
     .            1.072,1.100,1.130,1.157,1.232,1.282,
     .            1.064,1.084,1.116,1.139,1.207,1.249,
```

```
     .           1.032,1.051,1.060,1.078,1.099,1.121,
     .           1.016,1.025,1.030,1.043,1.049,1.062,
     .           1.008,1.016,1.019,1.026,1.034,1.039/
      DATA  CLSTT/0.3,0.4,0.5,0.6,1.0,2.0,3.0/
      DATA  FA/0.3020,-22.930,-0.229,-0.2430,3.85100,0.0020,-0.550,
     .        0.0000,0.00700,-0.050,-0.0015,-0.1220,-0.156,-0.0050,
     .        368.44,24.5200,-1.140,-1.0900,0.58000,-0.180,0.28000,
     .        0.1717,-0.0344,0.0032,0.00240,-0.0043,0.0000,-0.0008,
     .        0.0905,-0.0410,0.0073,0.00150,-0.0034,0.0004,-0.0006/
      DATA  MERGE/12/
      DATA  NAD1/15/, NAD2/8/, NBATT/1/
      DATA  NBTEMP/16/, NCLS1/6/, NCLST/7/
      DATA  NROE/23/, NRSCEL/26/
      DATA  NSOC/21/, NSUNMW/8/
      DATA  NRD/5/, NWRT/6/
      DATA  P0/0.598,0.908,0.849,1.01,0.724,0.959/
      DATA  P1/0.00026,-0.03214,-0.01277,-0.01394,-0.00652,-0.02304/
      DATA  P2/0.0021,0.0102,0.0036,0.00553,0.00191,0.00787/
      DATA  P3/-0.00035,-0.00114,-0.00059,-0.00068,-0.00047,-0.00091/
      DATA  QBATT/0.00,0.03,0.05,0.10,0.15,0.20,0.25,
     .          0.35,0.40,0.45,0.475,0.50,0.70,0.75,
     .          0.80,0.86,0.875,0.89,0.92,0.94,1.00/
      DATA  ROE/0.07600,0.05400,0.04030,0.03130,0.02550,0.02120,
     .        0.01100,0.00740,0.00570,0.00480,0.00381,0.00335,
     .        0.00264,0.00223,0.00200,0.00177,0.00168,0.00148,
     .        0.00132,0.00124,0.00120,0.00122,0.001241/
      DATA  RSCELL/0.1889,0.1816,0.1742,0.1668,0.1596,0.1524,
     .          0.1509,0.1458,0.1400,0.1350,0.1311,0.1283,
     .          0.1262,0.1240,0.1213,0.1176,0.1130,0.1075,
     .          0.1016,0.0957,0.0904,0.0860,0.0823,0.0785,
     .          0.0748,0.0709/
      DATA  SFILC/28/
      DATA  SOC/0.00,0.03,0.06,0.10,0.15,0.20,0.30,0.40,0.50,0.60,0.70,
     .        0.75,0.80,0.825,0.85,0.875,0.90,0.925,0.95,0.975,1.00/
      DATA  SUNMW/540.0,394.0,253.0,139.6,100.0,50.0,25.0,5.0/
      DATA  SUNLIT/5.0,6.0,7.0,8.0,9.0,10.0,15.0,20.0,25.0,
     .          30.0,40.0,50.0,75.0,100.0,120.0,140.0,
     .          160.0,200.0,253.0,300.0,394.0,500.0,540.0/
      DATA  TBATT/-40.0,-20.0,50.0,70.0,90.0,120.0/
      DATA  TCSTD/60.0/
      DATA  TEMTAB/-140.0,-120.0,-100.0,-80.0,-60.0,-40.0,
     .          -30.0,-20.0,-10.0,0.0,10.0,20.0,30.0,
     .          40.0,50.0,60.0,70.0,80.0,90.0,100.0,
     .          110.0,120.0,130.0,140.0,150.0,160.0/
      DATA  TL/16*0.0,0.3,0.7,14*0.0,0.5,2.0,14*0.0,0.4,3.6,14*0.0,
     .        1.0,5.0,14*0.0,3.0,3.0,14*0.0,3.0,1.0,14*0.0,0.3,0.7,
     .        0.3,0.7,0.3,0.7,0.3,0.7,0.3,0.7,0.3,0.6,4*0.0,0.3,0.6,
     .        0.4,3.6,12*0.0,0.4,0.6,2.0,5.0,12*0.0,1.0,95*0.0/
      DATA  TP/-40.0,-20.0,50.0,90.0,120.0/
      DATA  VBATT(1, 1,1)/1.9618,1.9808,1.9997,2.1850,2.2923,
     .                    2.3417,2.3608,2.3750,2.3797/
      DATA  VBATT(1, 2,1)/1.9808,1.9997,2.0187,2.1992,2.3085,
     .                    2.3531,2.3702,2.3845,2.3892/
      DATA  VBATT(1, 3,1)/1.9997,2.0187,2.0378,2.2230,2.3151,
     .                    2.3626,2.3797,2.3940,2.3988/
      DATA  VBATT(1, 4,1)/2.0187,2.0378,2.0567,2.2467,2.3275,
     .                    2.3702,2.3892,2.4035,2.4083/
```

```
      DATA  VBATT(1, 5,1)/2.0473,2.0662,2.0871,2.2562,2.3370,
     .                    2.3769,2.3940,2.4083,2.4130/
      DATA  VBATT(1, 6,1)/2.0757,2.0948,2.1090,2.2705,2.3465,
     .                    2.3864,2.4073,2.4215,2.4263/
      DATA  VBATT(1, 7,1)/2.1137,2.1328,2.1660,2.2781,2.3541,
     .                    2.3921,2.4177,2.4320,2.4367/
      DATA  VBATT(1, 8,1)/2.1517,2.1707,2.2050,2.2895,2.3579,
     .                    2.3988,2.4272,2.4415,2.4462/
      DATA  VBATT(1, 9,1)/2.1802,2.1992,2.2325,2.3123,2.3731,
     .                    2.4111,2.4415,2.4557,2.4605/
      DATA  VBATT(1,10,1)/2.2078,2.2268,2.2610,2.3341,2.3873,
     .                    2.4225,2.4557,2.4700,2.4747/
      DATA  VBATT(1,11,1)/2.2372,2.2562,2.2914,2.3550,2.4006,
     .                    2.4320,2.4700,2.4843,2.4890/
      DATA  VBATT(1,12,1)/2.2657,2.2847,2.3218,2.3750,2.4130,
     .                    2.4405,2.4843,2.4985,2.5032/
      DATA  VBATT(1,13,1)/2.2752,2.2942,2.3322,2.3864,2.4292,
     .                    2.4662,2.5127,2.5270,2.5317/
      DATA  VBATT(1,14,1)/2.2942,2.3133,2.3417,2.3988,2.4386,
     .                    2.4814,2.5412,2.5555,2.5602/
      DATA  VBATT(1,15,1)/2.2990,2.3180,2.3465,2.4035,2.4519,
     .                    2.5270,2.6410,2.6552,2.6600/
      DATA  VBATT(1,16,1)/2.3038,2.3227,2.3512,2.4083,2.4652,
     .                    2.5726,2.7407,2.7550,2.7598/
      DATA  VBATT(1,17,1)/2.3133,2.3322,2.3589,2.4130,2.4710,
     .                    2.5944,2.7882,2.8025,2.8072/
      DATA  VBATT(1,18,1)/2.3227,2.3417,2.3655,2.4177,2.4766,
     .                    2.6172,2.8358,2.8500,2.8547/
      DATA  VBATT(1,19,1)/2.3417,2.3608,2.3797,2.4254,2.4938,
     .                    2.6344,2.8547,2.8690,2.8738/
      DATA  VBATT(1,20,1)/2.3474,2.3664,2.3864,2.4301,2.4994,
     .                    2.6495,2.8642,2.8785,2.8832/
      DATA  VBATT(1,21,1)/2.3608,2.3797,2.4006,2.4425,2.5127,
     .                    2.6714,2.8738,2.8880,2.8927/
      DATA  VBATT(1, 1,2)/1.8953,1.9142,1.9332,2.1185,2.2258,
     .                    2.2752,2.2942,2.3085,2.3133/
      DATA  VBATT(1, 2,2)/1.9142,1.9332,1.9523,2.1328,2.2420,
     .                    2.2867,2.3038,2.3180,2.3227/
      DATA  VBATT(1, 3,2)/1.9332,1.9523,1.9712,2.1565,2.2487,
     .                    2.2961,2.3133,2.3275,2.3322/
      DATA  VBATT(1, 4,2)/1.9523,1.9712,1.9902,2.1802,2.2610,
     .                    2.3038,2.3227,2.3370,2.3417/
      DATA  VBATT(1, 5,2)/1.9808,1.9997,2.0206,2.1897,2.2705,
     .                    2.3104,2.3275,2.3417,2.3465/
      DATA  VBATT(1, 6,2)/2.0093,2.0282,2.0425,2.2040,2.2800,
     .                    2.3199,2.3408,2.3550,2.3598/
      DATA  VBATT(1, 7,2)/2.0473,2.0662,2.0995,2.2116,2.2876,
     .                    2.3256,2.3512,2.3655,2.3702/
      DATA  VBATT(1, 8,2)/2.0852,2.1042,2.1384,2.2230,2.2914,
     .                    2.3322,2.3608,2.3750,2.3797/
      DATA  VBATT(1, 9,2)/2,1137,2.1328,2.1660,2.2458,2.3066,
     .                    2.3446,2.3750,2.3892,2.3940/
      DATA  VBATT(1,10,2)/2.1413,2.1603,2.1945,2.2676,2.3208,
     .                    2,3560,2.3892,2.4035,2.4083/
      DATA  VBATT(1,11,2)/2.1707,2.1897,2.2249,2.2885,2.3341,
     .                    2.3655,2.4035,2.4177,2.4225/
      DATA  VBATT(1,12,2)/2.1992,2.2183,2.2553,2.3085,2.3465,
```

```
     .                    2.3740,2.4177,2.4320,2.4367/
      DATA  VBATT(1,13,2)/2.2087,2.2278,2.2657,2.3199,2.3626,
     .                    2.3997,2.4462,2.4605,2.4652/
      DATA  VBATT(1,14,2)/2.2278,2.2467,2.2752,2.3322,2.3722,
     .                    2.4149,2.4747,2.4890,2.4938/
      DATA  VBATT(1,15,2)/2.2325,2.2515,2.2800,2.3370,2.3855,
     .                    2.4605,2.5745,2.5887,2.5935/
      DATA  VBATT(1,16,2)/2.2372,2.2562,2.2847,2.3417,2.3988,
     .                    2.5061,2.6742,2.6885,2.6933/
      DATA  VBATT(1,17,2)/2.2467,2.2657,2.2923,2.3465,2.4044,
     .                    2.5280,2.7217,2.7360,2.7407/
      DATA  VBATT(1,18,2)/2.2562,2.2752,2.2990,2.3512,2.4101,
     .                    2.5507,2.7692,2.7835,2.7882/
      DATA  VBATT(1,19,2)/2.2752,2.2942,2.3133,2.3589,2.4272,
     .                    2.5678,2.7882,2.8025,2.8072/
      DATA  VBATT(1,20,2)/2.2809,2.3000,2.3199,2.3636,2.4329,
     .                    2.5831,2.7977,2.8120,2.8167/
      DATA  VBATT(1,21,2)/2.2942,2.3133,2.3341,2.3759,2.4462,
     .                    2.6049,2.8072,2.8215,2.8262/
      DATA  VBATT(1, 1,3)/1.6625,1.6815,1.7005,1.8857,1.9931,
     .                    2.0425,2.0615,2.0757,2.0805/
      DATA  VBATT(1, 2,3)/1.6815,1.7005,1.7195,1.9000,2.0093,
     .                    2.0539,2.0710,2.0852,2.0900/
      DATA  VBATT(1, 3,3)/1.7005,1.7195,1.7385,1.9237,2.0159,
     .                    2.0634,2.0805,2.0948,2.0995/
      DATA  VBATT(1, 4,3)/1.7195,1.7385,1.7575,1.9475,2.0282,
     .                    2.0710,2.0900,2.1042,2.1090/
      DATA  VBATT(1, 5,3)/1.7480,1.7670,1.7879,1.9570,2.0378,
     .                    2.0777,2.0948,2.1090,2.1137/
      DATA  VBATT(1, 6,3)/1.7765,1.7955,1.8098,1.9712,2.0473,
     .                    2.0871,2.1080,2.1223,2.1270/
      DATA  VBATT(1, 7,3)/1.8145,1.8335,1.6668,1.9789,2.0548,
     .                    2.0928,2.1185,2.1328,2.1375/
      DATA  VBATT(1, 8,3)/1.8525,1.8715,1.9057,1.9902,2.0586,
     .                    2.0995,2.1280,2.1423,2.1470/
      DATA  VBATT(1, 9,3)/1.8810,1.9000,1.9332,2.0130,2.0738,
     .                    2.1118,2.1423,2.1565,2.1612/
      DATA  VBATT(1,10,3)/1.9086,1.9276,1.9618,2.0349,2.0881,
     .                    2.1233,2.1565,2.1707,2.1755/
      DATA  VBATT(1,11,3)/1.9380,1.9570,1.9922,2.0558,2.1014,
     .                    2.1328,2.1707,2.1850,2.1897/
      DATA  VBATT(1,12,3)/1.9665,1.9855,2.0225,2.0757,2.1137,
     .                    2.1413,2.1850,2.1992,2.2040/
      DATA  VBATT(1,13,3)/1.9760,1.9950,2.0330,2.0871,2.1299,
     .                    2.1669,2.2135,2.2278,2.2325/
      DATA  VBATT(1,14,3)/1.9950,2.0140,2.0425,2.0995,2.1394,
     .                    2.1821,2.2420,2.2562,2.2610/
      DATA  VBATT(1,15,3)/1.9997,2.0187,2.0473,2.1042,2.1527,
     .                    2.2278,2.3417,2.3560,2.3608/
      DATA  VBATT(1,16,3)/2.0045,2.0235,2.0520,2.1090,2.1660,
     .                    2.2734,2.4415,2.4557,2.4605/
      DATA  VBATT(1,17,3)/2.0140,2.0330,2.0596,2.1137,2.1717,
     .                    2.2952,2.4890,2.5032,2.5080/
      DATA  VBATT(1,18,3)/2.0235,2.0425,2.0662,2.1185,2.1774,
     .                    2.3180,2.5365,2.5507,2.5555/
      DATA  VBATT(1,19,3)/2.0425,2.0615,2.0805,2.1261,2.1945,
     .                    2.3351,2.5555,2.5698,2.5745/
```

```
      DATA  VBATT(1,20,3)/2.0482,2.0672,2.0871,2.1308,2.2002,
     .                    2.3503,2.5650,2.5793,2.5840/
      DATA  VBATT(1,21,3)/2.0615,2.0805,2.1014,2.1432,2.2135,
     .                    2.3722,2.5745,2.5887,2.5935/
      DATA  VBATT(1, 1,4)/1.5960,1.6150,1.6340,1.8193,1.9266,
     .                    1.9760,1.9950,2.0093,2.0140/
      DATA  VBATT(1, 2,4)/1.6150,1.6340,1.6530,1.8335,1.9427,
     .                    1.9874,2.0045,2.0187,2.0235/
      DATA  VBATT(1, 3,4)/1.6340,1.6530,1.6720,1.8572,1.9494,
     .                    1.9969,2.0140,2.0282,2.0330/
      DATA  VBATT(1, 4,4)/1.6530,1.6720,1.6910,1.8810,1.9618,
     .                    2.0045,2.0235,2.0378,2.0425/
      DATA  VBATT(1, 5,4)/1.6815,1.7005,1.7214,1.8905,1.9712,
     .                    2.0112,2.0282,2.0425,2.0473/
      DATA  VBATT(1, 6,4)/1.7100,1.7290,1.7432,1.9047,1.9808,
     .                    2.0206,2.0415,2.0558,2.0606/
      DATA  VBATT(1, 7,4)/1.7480,1.7670,1.8002,1.9124,1.9883,
     .                    2.0263,2.0520,2.0662,2.0710/
      DATA  VBATT(1, 8,4)/1.7860,1.8050,1.8392,1.9237,1.9922,
     .                    2.0330,2.0615,2.0757,2.0805/
      DATA  VBATT(1, 9,4)/1.8145,1.8335,1.8668,1.9465,2.0073,
     .                    2.0453,2.0757,2.0900,2.0948/
      DATA  VBATT(1,10,4)/1.8421,1.8611,1.8953,1.9684,2.0216,
     .                    2.0567,2.0900,2.1042,2.1090/
      DATA  VBATT(1,11,4)/1.8715,1.8905,1.9257,1.9893,2.0349,
     .                    2.0662,2.1042,2.1185,2.1233/
      DATA  VBATT(1,12,4)/1.9000,1.9190,1.9560,2.0093,2.0473,
     .                    2.0748,2.1185,2.1328,2.1375/
      DATA  VBATT(1,13,4)/1.9095,1.9285,1.9665,2.0206,2.0634,
     .                    2.1004,2.1470,2.1612,2.1660/
      DATA  VBATT(1,14,4)/1.9285,1.9475,1.9760,2.0330,2.0729,
     .                    2.1156,2.1755,2.1897,2.1945/
      DATA  VBATT(1,15,4)/1.9332,1.9523,1.9808,2.0378,2.0862,
     .                    2.1612,2.2752,2.2895,2.2942/
      DATA  VBATT(1,16,4)/1.9380,1.9570,1.9855,2.0425,2.0995,
     .                    2.2068,2.3750,2.3892,2.3940/
      DATA  VBATT(1,17,4)/1.9475,1.9665,1.9931,2.0473,2.1052,
     .                    2.2287,2.4225,2.4367,2.4415/
      DATA  VBATT(1,18,4)/1.9570,1.9760,1.9997,2.0520,2.1109,
     .                    2.2515,2.4700,2.4843,2.4890/
      DATA  VBATT(1,19,4)/1.9760,1.9950,2.0140,2.0596,2.1280,
     .                    2.2686,2.4890,2.5032,2.5080/
      DATA  VBATT(1,20,4)/1.9817,2.0007,2.0206,2.0644,2.1337,
     .                    2.2838,2.4985,2.5127,2.5175/
      DATA  VBATT(1,21,4)/1.9950,2.0140,2.0349,2.0767,2.1470,
     .                    2.3056,2.5080,2.5222,2.5270/
      DATA  VBATT(1, 1,5)/1.5295,1.5485,1.5675,1.7527,1.8601,
     .                    1.9095,1.9285,1.9427,1.9475/
      DATA  VBATT(1, 2,5)/1.5485,1.5675,1.5865,1.7670,1.8762,
     .                    1.9209,1.9380,1.9523,1.9570/
      DATA  VBATT(1, 3,5)/1.5675,1.5865,1.6055,1.7907,1.8829,
     .                    1.9304,1.9475,1.9618,1.9665/
      DATA  VBATT(1, 4,5)/1.5865,1.6055,1.6245,1.8145,1.8953,
     .                    1.9380,1.9570,1.9712,1.9760/
      DATA  VBATT(1, 5,5)/1.6150,1.6340,1.6549,1.8240,1.9047,
     .                    1.9446,1.9618,1.9760,1.9808/
      DATA  VBATT(1, 6,5)/1.6435,1.6625,1.6768,1.8382,1.9142,
```

```
     .                    1.9542,1.9751,1.9893,1.9941/
      DATA  VBATT(1, 7,5)/1.6815,1.7005,1.7338,1.8459,1.9218,
     .                    1.9598,1.9855,1.9997,2.0045/
      DATA  VBATT(1, 8,5)/1.7195,1.7385,1.7727,1.8572,1.9257,
     .                    1.9665,1.9950,2.0093,2.0140/
      DATA  VBATT(1, 9,5)/1.7480,1.7670,1.8002,1.8801,1.9409,
     .                    1.9789,2.0093,2.0235,2.0282/
      DATA  VBATT(1,10,5)/1.7756,1.7946,1.8287,1.9019,1.9551,
     .                    1.9902,2.0235,2.0378,2.0425/
      DATA  VBATT(1,11,5)/1.8050,1.8240,1.8592,1.9228,1.9684,
     .                    1.9997,2.0378,2.0520,2.0567/
      DATA  VBATT(1,12,5)/1.8335,1.8525,1.8895,1.9427,1.9808,
     .                    2.0083,2.0520,2.0662,2.0710/
      DATA  VBATT(1,13,5)/1.8430,1.8620,1.9000,1.9542,1.9969,
     .                    2.0340,2.0805,2.0948,2.0995/
      DATA  VBATT(1,14,5)/1.8620,1.8810,1.9095,1.9665,2.0064,
     .                    2.0491,2.1090,2.1233,2.1280/
      DATA  VBATT(1,15,5)/1.8668,1.8857,1.9142,1.9712,2.0197,
     .                    2.0948,2.2087,2.2230,2.2278/
      DATA  VBATT(1,16,5)/1.8715,1.8905,1.9190,1.9760,2.0330,
     .                    2.1403,2.3085,2.3227,2.3275/
      DATA  VBATT(1,17,5)/1.8810,1.9000,1.9266,1.9808,2.0387,
     .                    2.1622,2.3560,2.3702,2.3750/
      DATA  VBATT(1,18,5)/1.8905,1.9095,1.9332,1.9855,2.0444,
     .                    2.1850,2.4035,2.4177,2.4225/
      DATA  VBATT(1,19,5)/1.9095,1.9285,1.9475,1.9931,2.0615,
     .                    2.2021,2.4225,2.4367,2.4415/
      DATA  VBATT(1,20,5)/1.9152,1.9342,1.9542,1.9978,2.0672,
     .                    2.2173,2.4320,2.4462,2.4510/
      DATA  VBATT(1,21,5)/1.9285,1.9475,1.9684,2.0102,2.0805,
     .                    2.2391,2.4415,2.4557,2.4605/
      DATA  VBATT(1, 1,6)/1.4297,1.4488,1.4677,1.6530,1.7604,
     .                    1.8098,1.8287,1.8430,1.8477/
      DATA  VBATT(1, 2,6)/1.4488,1.4677,1.4867,1.6672,1.7765,
     .                    1.8211,1.8382,1.8525,1.8572/
      DATA  VBATT(1, 3,6)/1.4677,1.4867,1.5057,1.6910,1.7832,
     .                    1.8307,1.8477,1.8620,1.8668/
      DATA  VBATT(1, 4,6)/1.4867,1.5057,1.5247,1.7148,1.7955,
     .                    1.8382,1.8572,1.8715,1.8762/
      DATA  VBATT(1, 5,6)/1.5152,1.5342,1.5551,1.7242,1.8050,
     .                    1.8449,1.8620,1.8762,1.8810/
      DATA  VBATT(1, 6,6)/1.5438,1.5627,1.5770,1.7385,1.8145,
     .                    1.8544,1.8753,1.8895,1.8943/
      DATA  VBATT(1, 7,6)/1.5818,1.6007,1.6340,1.7461,1.8221,
     .                    1.8601,1.8857,1.9000,1.9047/
      DATA  VBATT(1, 8,6)/1.6197,1.6387,1.6729,1.7575,1.8259,
     .                    1.8668,1.8953,1.9095,1.9142/
      DATA  VBATT(1, 9,6)/1.6482,1.6672,1.7005,1.7803,1.8411,
     .                    1.8791,1.9095,1.9237,1.9285/
      DATA  VBATT(1,10,6)/1.6758,1.6948,1.7290,1.8021,1.8553,
     .                    1.8905,1.9237,1.9380,1.9427/
      DATA  VBATT(1,11,6)/1.7052,1.7242,1.7594,1.8230,1.8686,
     .                    1.9000,1.9380,1.9523,1.9570/
      DATA  VBATT(1,12,6)/1.7338,1.7527,1.7898,1.8430,1.8810,
     .                    1.9086,1.9523,1.9665,1.9712/
      DATA  VBATT(1,13,6)/1.7432,1.7623,1.8002,1.8544,1.8971,
     .                    1.9342,1.9808,1.9950,1.9997/
```

```
      DATA  VBATT(1,14,6)/1.7623,1.7812,1.8098,1.8668,1.9066,
     .                    1.9494,2.0093,2.0235,2.0282/
      DATA  VBATT(1,15,6)/1.7670,1.7860,1.8145,1.8715,1.9199,
     .                    1.9950,2.1090,2.1233,2.1280/
      DATA  VBATT(1,16,6)/1.7717,1.7907,1.8193,1.8762,1.9332,
     .                    2.0406,2.2087,2.2230,2.2278/
      DATA  VBATT(1,17,6)/1.7812,1.8002,1.8288,1.8810,1.9390,
     .                    2.0624,2.2562,2.2705,2.2752/
      DATA  VBATT(1,18,6)/1.7907,1.8098,1.8335,1.8857,1.9446,
     .                    2.0852,2.3038,2.3180,2.3227/
      DATA  VBATT(1,19,6)/1.8098,1.8287,1.8477,1.8934,1.9618,
     .                    2.1023,2.3227,2.3370,2.3417/
      DATA  VBATT(1,20,6)/1.8154,1.8344,1.8544,1.8981,1.9675,
     .                    2.1175,2.3322,2.3465,2.3512/
      DATA  VBATT(1,21,6)/1.8287,1.8477,1.8686,1.9105,1.9808,
     .                    2.1394,2.3417,2.3560,2.3608/
      DATA  VV/0.59540,0.57200,0.54040,0.52260,0.50900,
     .         0,48590,0,47308,0,46026,0,44743,0,43461,
     .         0.42179,0.40897,0.39743,0.38461,0.37179,
     .         0.35897,0.34615,0.33333,0.32051,0.30769,
     .         0.29487,0.23077,0.16667,0.10256,0.03846,
     .         0.0,-0.06410,-0.12820,-0.19231,-0.25641/
      DATA  XCSTD/145.0/
      DATA  XIBATT/-1.0,-0.1,-0.05,0,0,0.05,0.1,0.2,0.5,1.0/
      DATA  XII/-0.30900,-0.293698,-0.106502,-0.042848,0.0,
     .          0.045521,0.068053,0.085984,0.099321,0.109896,
     .          0.118172,0.124156,0.128286,0.131510,0.134271,
     .          0.136104,0.137669,0.138772,0.139606,0.140152,
     .          0.140564,0.141491,0.141851,0.142129,0.142407,
     .          0.142593,0.142881,0.143159,0.143448,0.143726/
      END
```

BLKDTA/NI-CD

```
AA(1,1,1)=1.000,1.000,1.000,0.997,0.984,0.962,0.943,0.925,
          0.907,0.890,0.660,0.815,0.731,0.522,0.261,0.000,5*0.0,
AA(1,2,1)=1.000,1.000,1.000,1.000,0.997,0.987,0.973,0.955,
          0.940,0.925,0.902,0.861,0.784,0.605,0.303,0.000,5*0.0,
AA(1,3,1)=1.000,1.000,1.000,1.000,1.000,1.000,0.992,0.979,
          0.967,0.956,0.935,0.906,0.856,0.717,0.400,0.000,5*0.0,
AA(1,4,1)=1.000,1.000,1.000,1.000,1.000,1.000,1.000,0.996,
          0.989,0.981,0.965,0.942,0.900,0.780,0.390,0.000,5*0.0,
AA(1,5,1)=1.000,1.000,1.000,1.000,1.000,1.000,1.000,1.000,
          1.000,0.997,0.984,0.967,0.929,0.840,0.531,0.000,5*0.0,
AA(1,1,2)=1.000,1.000,0.996,0.985,0.968,0.939,0.921,0.895,
          0.874,0.853,0.816,0.779,0.681,0.583,0.292,0.000,5*0.0,
AA(1,2,2)=1.000,1.000,1.000,1.000,0.995,0.979,0.963,0.941,
          0.922,0.904,0.869,0.835,0.765,0.634,0.364,0.000,5*0.0,
AA(1,3,2)=1.000,1.000,1.000,1.000,1.000,0.998,0.990,0.973,
          0.960,0.945,0.921,0.884,0.829,0.713,0.406,0.000,5*0.0,
AA(1,4,2)=1.000,1.000,1.000,1.000,1.000,1.000,1.000,0.995,
          0.985,0.972,0.955,0.925,0.874,0.762,0.475,0.000,5*0.0,
AA(1,5,2)=1.000,1.000,1.000,1.000,1.000,1.000,1.000,1.000,
          1.000,0.997,0.984,0.959,0.915,0.805,0.472,0.000,5*0.0,
AA(1,1,3)=0.983,0.980,0.976,0.967,0.943,0.907,0.875,0.829,
          0.796,0.752,0.702,0.633,0.544,0.455,0.228,0.000,5*0.0,
AA(1,2,3)=1.000,1.000,0.999,0.993,0.980,0.949,0.923,0.881,
          0.848,0.816,0.758,0.700,0.602,0.504,0.291,0.000,5*0.0,
AA(1,3,3)=1.000,1.000,1.000,1.000,0.997,0.980,0.959,0.924,
          0.894,0.864,0.816,0.744,0.666,0.555,0.317,0.000,5*0.0,
AA(1,4,3)=1.000,1.000,1.000,1.000,1.000,1.000,0.989,0.962,
          0.936,0.910,0.863,0.807,0.722,0.620,0.417,0.000,5*0.0,
AA(1,5,3)=1.000,1.000,1.000,1.000,1.000,1.000,1.000,0.985,
          0.968,0.943,0.903,0.843,0.759,0.635,0.417,0.000,5*0.0,
AA(1,1,4)=0.940,0.927,0.920,0.902,0.866,0.801,0.755,0.694,
          0.659,0.610,0.562,0.496,0.425,0.319,0.160,0.000,5*0.0,
AA(1,2,4)=1.000,0.990,0.980,0.957,0.920,0.857,0.811,0.757,
          0.714,0.671,0.624,0.558,0.486,0.389,0.219,0.000,5*0.0,
AA(1,3,4)=1.000,1.000,0.998,0.986,0.960,0.910,0.869,0.810,
          0.775,0.727,0.675,0.605,0.529,0.428,0.241,0.000,5*0.0,
AA(1,4,4)=1.000,1.000,1.000,1.000,0.987,0.950,0.912,0.863,
          0.827,0.785,0.743,0.655,0.567,0.477,0.338,0.000,5*0.0,
AA(1,5,4)=1.000,1.000,1.000,1.000,1.000,0.976,0.945,0.900,
          0.857,0.815,0.750,0.685,0.596,0.471,0.261,0.000,5*0.0,
AA(1,1,5)=0.860,0.860,0.853,0.839,0.810,0.759,0.720,0.670,
          0.632,0.595,0.545,0.487,0.421,0.326,0.208,0.000,5*0.0,
AA(1,2,5)=0.933,0.929,0.918,0.903,0.874,0.825,0.790,0.743,
          0.704,0.665,0.621,0.557,0.485,0.382,0.250,0.000,5*0.0,
AA(1,3,5)=0.992,0.981,0.972,0.951,0.920,0.869,0.828,0.775,
          0.737,0.700,0.658,0.598,0.532,0.439,0.251,0.000,5*0.0,
AA(1,4,5)=1.000,0.998,0.991,0.976,0.951,0.909,0.870,0.819,
          0.786,0.745,0.689,0.633,0.552,0.470,0.235,0.000,5*0.0,
AA(1,5,5)=1.000,1.000,1.000,0.998,0.978,0.936,0.899,0.851,
          0.817,0.774,0.725,0.666,0.593,0.482,0.296,0.000,5*0.0,
BI=0.025,0.05,0.1,0.25,0.5,
NSOC=16,
QBATT=0.001,0.01,0.03,0.05,0.1,0.15,0.2,0.25,0.35,0.5,0.7,
      0.83,0.86,0.89,0.92,0.94,0.96,0.97,0.98,0.99,1.0,
SOC=0.00,0.30,0.40,0.50,0.60,0.70,0.75,0.80,0.825,
    0.85,0.875,0.90,0.925,0.95,0.975,1.00,5*0.0,
```

```
TBATT=40.0,50.0,60.0,70.0,80.0,95.0,
TP=40.0,50.0,70.0,80.0,95.0,
VBATT(1, 1,1)=1.137,1.142,1.174,1.187,1.234,
              1.282,1.300,1.327,1.335,
VBATT(1, 2,1)=1.153,1.158,1.190,1.203,1.254,
              1.306,1.324,1.351,1.359,
VBATT(1, 3,1)=1.168,1.173,1.205,1.218,1.271,
              1.324,1.342,1.369,1.377,
VBATT(1, 4,1)=1.176,1.181,1.213,1.226,1.281,
              1.335,1.353,1.380,1.388,
VBATT(1, 5,1)=1.193,1.198,1.230,1.243,1.298,
              1.353,1.371,1.398,1.406,
VBATT(1, 6,1)=1.204,1.209,1.241,1.254,1.309,
              1.364,1.382,1.409,1.417,
VBATT(1, 7,1)=1.213,1.218,1.250,1.263,1.316,
              1.370,1.388,1.415,1.423,
VBATT(1, 8,1)=1.223,1.228,1.260,1.273,1.323,
              1.374,1.392,1.419,1.427,
VBATT(1, 9,1)=1.236,1.241,1.273,1.286,1.333,
              1.380,1.398,1.425,1.433,
VBATT(1,10,1)=1.247,1.252,1.284,1.297,1.342,
              1.388,1.406,1.433,1.441,
VBATT(1,11,1)=1.260,1.265,1.297,1.310,1.355,
              1.400,1.418,1.445,1.453,
VBATT(1,12,1)=1.268,1.273,1.305,1.318,1.365,
              1.413,1.431,1.458,1.468,
VBATT(1,13,1)=1.273,1.278,1.310,1.323,1.370,
              1.418,1.436,1.463,1.473,
VBATT(1,14,1)=1.281,1.286,1.318,1.331,1.378,
              1.423,1.441,1.468,1.478,
VBATT(1,15,1)=1.289,1.294,1.326,1.339,1.383,
              1.427,1.445,1.472,1.482,
VBATT(1,16,1)=1.307,1.312,1.344,1.357,1.395,
              1.433,1.451,1.478,1.438,
VBATT(1,17,1)=1.326,1.331,1.363,1.376,1.408,
              1.441,1.459,1.486,1.496,
VBATT(1,18,1)=1.334,1.339,1.371,1.384,1.414,
              1.445,1.463,1.490,1.500,
VBATT(1,19,1)=1.345,1.350,1.382,1.395,1.424,
              1.453,1.471,1.498,1.508,
VBATT(1,20,1)=1.354,1.359,1.391,1.404,1.432,
              1.461,1.479,1.506,1.516,
VBATT(1,21,1)=1.361,1.366,1.398,1.411,1.438,
              1.464,1.482,1.509,1.519,
VBATT(1, 1,2)=1.126,1.133,1.148,1.168,1.215,
              1.262,1.273,1.295,1.304,
VBATT(1, 2,2)=1.151,1.158,1.173,1.193,1.240,
              1.287,1.298,1.320,1.329,
VBATT(1, 3,2)=1.167,1.174,1.189,1.209,1.257,
              1.305,1.316,1.338,1.347,
VBATT(1, 4,2)=1.174,1.181,1.196,1.216,1.266,
              1.315,1.326,1.348,1.357,
VBATT(1, 5,2)=1.193,1.200,1.215,1.235,1.287,
              1.339,1.350,1.372,1.381,
VBATT(1, 6,2)=1.205,1.212,1.227,1.247,1.298,
              1.350,1.361,1.383,1.392,
VBATT(1, 7,2)=1.216,1.223,1.238,1.258,1.308,
```

```
                1.359,1.370,1.392,1.401,
VBATT(1, 8,2)=1.226,1.233,1.248,1.268,1.317,
                1.365,1.376,1.398,1.407,
VBATT(1, 9,2)=1.236,1.243,1.258,1.278,1.324,
                1.371,1.385,1.404,1.413,
VBATT(1,10,2)=1.246,1.253,1.268,1.288,1.334,
                1.380,1.393,1.413,1.422,
VBATT(1,11,2)=1.256,1.265,1.280,1.300,1.344,
                1.388,1.404,1.418,1.427,
VBATT(1,12,2)=1.268,1.275,1.290,1.310,1.356,
                1.402,1.413,1.430,1.439,
VBATT(1,13,2)=1.272,1.279,1.294,1.314,1.364,
                1.413,1.422,1.446,1.455,
VBATT(1,14,2)=1.280,1.287,1.302,1.322,1.369,
                1.417,1.426,1.450,1.459,
VBATT(1,15,2)=1.289,1.296,1.311,1.331,1.377,
                1.423,1.432,1.456,1.465,
VBATT(1,16,2)=1.300,1.307,1.322,1.342,1.385,
                1.427,1.436,1.460,1.469,
VBATT(1,17,2)=1.315,1.322,1.337,1.357,1.395,
                1.434,1.443,1.467,1.476,
VBATT(1,18,2)=1.324,1.333,1.348,1.368,1.403,
                1.439,1.448,1.472,1.481,
VBATT(1,19,2)=1.333,1.340,1.355,1.375,1.411,
                1.447,1.456,1.480,1.489,
VBATT(1,20,2)=1.344,1.351,1.366,1.386,1.421,
                1.457,1.466,1.490,1.499,
VBATT(1,21,2)=1.355,1.362,1.377,1.397,1.430,
                1.463,1.472,1.496,1.505,
VBATT(1, 1,3)=1.115,1.124,1.141,1.150,1.195,
                1.241,1.246,1.263,1.272,
VBATT(1, 2,3)=1.130,1.139,1.156,1.165,1.216,
                1.267,1.272,1.289,1.298,
VBATT(1, 3,3)=1.147,1.156,1.173,1.182,1.233,
                1.285,1.290,1.307,1.316,
VBATT(1, 4,3)=1.156,1.165,1.182,1.191,1.242,
                1.293,1.298,1.315,1.322,
VBATT(1, 5,3)=1.174,1.183,1.200,1.209,1.264,
                1.320,1.325,1.342,1.349,
VBATT(1, 6,3)=1.187,1.196,1.213,1.222,1.279,
                1.335,1.340,1.357,1.364,
VBATT(1, 7,3)=1.201,1.210,1.227,1.236,1.297,
                1.347,1.352,1.369,1.376,
VBATT(1, 8,3)=1.208,1.217,1.234,1.243,1.299,
                1.355,1.360,1.377,1.384,
VBATT(1, 9,3)=1.217,1.226,1.243,1.252,1.310,
                1.367,1.372,1.389,1.396,
VBATT(1,10,3)=1.227,1.236,1.253,1.262,1.320,
                1.377,1.382,1.399,1.406,
VBATT(1,11,3)=1.236,1.245,1.262,1.271,1.331,
                1.390,1.395,1.412,1.419,
VBATT(1,12,3)=1.249,1.258,1.275,1.284,1.344,
                1.404,1.409,1.426,1.433,
VBATT(1,13,3)=1.252,1.261,1.278,1.287,1.347,
                1.408,1.413,1.430,1.437,
VBATT(1,14,3)=1.259,1.268,1.285,1.294,1.352,
                1.410,1.415,1.432,1.439,
```

```
VBATT(1,15,3)=1.265,1.274,1.291,1.300,1.358,
              1.415,1.420,1.437,1.444,
VBATT(1,16,3)=1.273,1.282,1.299,1.308,1.364,
              1.419,1.424,1.441,1.448,
VBATT(1,17,3)=1.287,1.296,1.313,1.322,1.373,
              1.423,1.428,1.445,1.452,
VBATT(1,18,3)=1.295,1.304,1.321,1.330,1.379,
              1.428,1.433,1.450,1.457,
VBATT(1,19,3)=1.305,1.314,1.331,1.340,1.389,
              1.437,1.442,1.459,1.466,
VBATT(1,20,3)=1.315,1.324,1.341,1.350,1.399,
              1.447,1.452,1.469,1.476,
VBATT(1,21,3)=1.328,1.337,1.354,1.363,1.408,
              1.452,1.457,1.474,1.481,
VBATT(1, 1,4)=1.106,1.113,1.128,1.144,1.186,
              1.228,1.237,1.254,1.262,
VBATT(1, 2,4)=1.122,1.129,1.144,1.160,1.206,
              1.251,1.260,1.277,1.285,
VBATT(1, 3,4)=1.140,1.147,1.162,1.178,1.224,
              1.269,1.278,1.295,1.303,
VBATT(1, 4,4)=1.150,1.157,1.172,1.188,1.246,
              1.283,1.292,1.309,1.317,
VBATT(1, 5,4)=1.169,1.176,1.191,1.207,1.257,
              1.306,1.315,1.332,1.340,
VBATT(1, 6,4)=1.180,1.187,1.202,1.218,1.271,
              1.323,1.332,1.349,1.357,
VBATT(1, 7,4)=1.192,1.199,1.214,1.230,1.283,
              1.335,1.344,1.361,1.369,
VBATT(1, 8,4)=1.202,1.209,1.224,1.240,1.292,
              1.343,1.352,1.369,1.377,
VBATT(1, 9,4)=1.212,1.219,1.234,1.250,1.301,
              1.351,1.360,1.377,1.385,
VBATT(1,10,4)=1.227,1.234,1.249,1.265,1.315,
              1.365,1.374,1.391,1.399,
VBATT(1,11,4)=1.239,1.246,1.261,1.277,1.327,
              1.377,1.386,1.403,1.411,
VBATT(1,12,4)=1.252,1.259,1.274,1.290,1.340,
              1.389,1.398,1.415,1.423,
VBATT(1,13,4)=1.254,1.261,1.276,1.292,1.342,
              1.392,1.401,1.418,1.426,
VBATT(1,14,4)=1.260,1.267,1.282,1.298,1.347,
              1.396,1.405,1.422,1.430,
VBATT(1,15,4)=1.268,1.275,1.290,1.306,1.352,
              1.400,1.409,1.426,1.434,
VBATT(1,16,4)=1.276,1.283,1.298,1.314,1.359,
              1.403,1.412,1.429,1.437,
VBATT(1,17,4)=1.286,1.293,1.308,1.324,1.365,
              1.406,1.415,1.432,1.440,
VBATT(1,18,4)=1.295,1.302,1.317,1.332,1.370,
              1.409,1.418,1.435,1.443,
VBATT(1,19,4)=1.302,1.309,1.324,1.339,1.377,
              1.415,1.424,1.441,1.449,
VBATT(1,20,4)=1.308,1.315,1.330,1.345,1.383,
              1.421,1.430,1.447,1.455,
VBATT(1,21,4)=1.318,1.325,1.340,1.355,1.390,
              1.425,1.434,1.451,1.459,
VBATT(1, 1,5)=1.098,1.102,1.114,1.138,1.177,
```

```
                1.215,1.228,1.244,1.251,
VBATT(1, 2,5)=1.114,1.118,1.132,1.154,1.195,
                1.236,1.249,1.265,1.272,
VBATT(1, 3,5)=1.134,1.136,1.152,1.174,1.214,
                1.254,1.267,1.283,1.290,
VBATT(1, 4,5)=1.147,1.151,1.165,1.187,1.227,
                1.267,1.280,1.296,1.303,
VBATT(1, 5,5)=1.164,1.168,1.182,1.204,1.253,
                1.292,1.305,1.321,1.328,
VBATT(1, 6,5)=1.174,1.178,1.192,1.214,1.262,
                1.310,1.323,1.339,1.346,
VBATT(1, 7,5)=1.184,1.188,1.202,1.224,1.272,
                1.320,1.333,1.349,1.356,
VBATT(1, 8,5)=1.193,1.197,1.211,1.233,1.280,
                1.327,1.340,1.356,1.363,
VBATT(1, 9,5)=1.208,1.212,1.226,1.248,1.302,
                1.336,1.349,1.365,1.372,
VBATT(1,10,5)=1.227,1.231,1.245,1.267,1.307,
                1.347,1.360,1.376,1.383,
VBATT(1,11,5)=1.242,1.246,1.260,1.282,1.322,
                1.364,1.377,1.393,1.400,
VBATT(1,12,5)=1.254,1.258,1.272,1.294,1.336,
                1.378,1.391,1.407,1.414,
VBATT(1,13,5)=1.257,1.261,1.275,1.297,1.339,
                1.381,1.394,1.410,1.417,
VBATT(1,14,5)=1.264,1.268,1.282,1.304,1.344,
                1.385,1.397,1.414,1.421,
VBATT(1,15,5)=1.270,1.274,1.288,1.310,1.349,
                1.388,1.400,1.417,1.424,
VBATT(1,16,5)=1.278,1.282,1.296,1.318,1.354,
                1.390,1.402,1.419,1.426,
VBATT(1,17,5)=1.284,1.288,1.302,1.324,1.357,
                1.391,1.403,1.420,1.427,
VBATT(1,18,5)=1.294,1.298,1.312,1.334,1.364,
                1.393,1.405,1.422,1.429,
VBATT(1,19,5)=1.299,1.303,1.317,1.339,1.367,
                1.395,1.407,1.424,1.431,
VBATT(1,20,5)=1.306,1.310,1.324,1.346,1.370,
                1.397,1.409,1.426,1.433,
VBATT(1,21,5)=1.313,1.317,1.331,1.353,1.375,
                1.398,1.410,1.427,1.434,
VBATT(1, 1,6)=1.086,1.088,1.107,1.129,1.162,
                1.195,1.215,1.229,1.235,
VBATT(1, 2,6)=1.093,1.095,1.114,1.136,1.175,
                1.213,1.233,1.247,1.253,
VBATT(1, 3,6)=1.116,1.118,1.137,1.159,1.195,
                1.231,1.251,1.265,1.271,
VBATT(1, 4,6)=1.134,1.136,1.155,1.177,1.212,
                1.247,1.267,1.281,1.287,
VBATT(1, 5,6)=1.147,1.149,1.168,1.190,1.230,
                1.270,1.290,1.304,1.310,
VBATT(1, 6,6)=1.156,1.158,1.177,1.199,1.236,
                1.274,1.317,1.331,1.337,
VBATT(1, 7,6)=1.163,1.165,1.184,1.206,1.246,
                1.287,1.330,1.344,1.350,
VBATT(1, 8,6)=1.170,1.172,1.191,1.213,1.256,
                1.299,1.342,1.356,1.362,
```

```
VBATT(1, 9,6)=1.185,1.187,1.206,1.228,1.271,
              1.319,1.357,1.371,1.377,
VBATT(1,10,6)=1.207,1.209,1.228,1.250,1.285,
              1.325,1.364,1.378,1.384,
VBATT(1,11,6)=1.232,1.234,1.253,1.271,1.303,
              1.340,1.372,1.386,1.392,
VBATT(1,12,6)=1.248,1.250,1.269,1.284,1.316,
              1.351,1.382,1.396,1.402,
VBATT(1,13,6)=1.252,1.254,1.273,1.291,1.318,
              1.352,1.383,1.397,1.403,
VBATT(1,14,6)=1.258,1.260,1.279,1.297,1.319,
              1.353,1.385,1.399,1.405,
VBATT(1,15,6)=1.264,1.266,1.285,1.303,1.326,
              1.356,1.386,1.400,1.406,
VBATT(1,16,6)=1.272,1.274,1.293,1.310,1.328,
              1.358,1.387,1.401,1.407,
VBATT(1,17,6)=1.279,1.281,1.300,1.316,1.336,
              1.362,1.388,1.402,1.408,
VBATT(1,18,6)=1.284,1.286,1.305,1.320,1.340,
              1.364,1.389,1.403,1.409,
VBATT(1,19,6)=1.288,1.290,1.309,1.323,1.343,
              1.368,1.392,1.406,1.412,
VBATT(1,20,6)=1.291,1.293,1.312,1.325,1.346,
              1.371,1.394,1.408,1.414,
VBATT(1,21,6)=1.297,1.299,1.318,1.330,1.350,
              1.374,1.395,1.409,1.415,
VV=0.59540,0.57200,0.54040,0.52260,0.50900,
   0.48590,0.47308,0.46026,0.44743,0.43461,
   0.42179,0.40897,0.39743,0.38461,0.37179,
   0.35897,0.34615,0.33333,0.32051,0.30769,
   0.29487,0.23077,0.16667,0.10256,0.03846,
   0.0,-0.06410,-0.12820,-0.19231,-0.25641,
XII=-0.30900,-0.293698,-0.106502,-0.042848,0.0,
    0.045521,0.068053,0.085984,0.099321,0.109896,
    0.118172,0.124156,0.128286,0.131510,0.134271,
    0.136104,0.137669,0.138772,0.139606,0.140152,
    0.140564,0.141491,0.141851,0.142129,0.142407,
    0.142593,0.142881,0.143159,0.143448,0.143726,
```

CENTER

```
C      CENTER - MAIN DS/PA DRIVER PROGRAM
C       CONTROLS EXECUTION OF DESIGN SYNTHESIS
C       AND/OR PERFORMANCE ANALYSIS SUBPROGRAMS
C
       INCLUDE  ALLCMN,LIST
       INCLUDE  COMON,LIST
       INCLUDE  DSCMN,LIST
       INCLUDE  PACMN,LIST
       INCLUDE  SUMCMN,LIST
C      INTEGER  IBUF(1000)
C
       COMMON/STATS/ ZALPHA,ZPRCNT
       NAMELIST/INPT/ AA,ACELL,ACSTD,AD1,AD2,BETAB,BI,BRCEST,BRCHMX,
     .  BRDEST,BRDSTD,BTEMP,CB,CBAVAL,CBMAX,CDEGA,CDEGB,CELPAC,CLR,
     .  CLSIT,CLST,CLSTT,CN,CSH,CURZ,DCDAT,DCDCHT,DCDCNT,DCDCPT,DCDCT,
     .  DCDET,DCDNNT,DCDNPT,DCDNT,DCDNZT,DCDPNT,DCDPPT,DCDPST,DODT,
     .  DTAMBI,DTTAI,DTTESG,DTTPCD,DTTPSG,DURAM,DWDAT,DWDCHT,DWDET,
     .  DWDNZT,DWDPST,FA,FRCELL,HDER,HDZMX,ICHRT,IFTYPE,INDFLS,IPSG,
     .  ISH,NAD1,NAD2,NBATP,NBATT,NBTEMP,NCDEG,NCURZ,NDCDA,NDCDC,
     .  NDCDCN,NDCDCP,NDCDN,NDCDNN,NDCDNP,NDCDPN,NDCDPP,NDOD,NDTAMB,
     .  NDTTA,NDWDA,NDWDCH,NDWDE,NDWDNZ,NDWDPS,NESP,NP,NPLT,NPREQ,
     .  NROE,NRSCEL,NS,NSAP,NSOC,NSPGR,NSUNMW,NTBFRZ,NTCZT,NTCZV,
     .  NVCHIS,NVCHIO,NVCHT,NVCHV,NVDEG,NVLBT,NVLBV,NVRISA,NVRIO,
     .  NXIHT,NXIHV,NZCHRA,NZCHRS,NZDT,NZDV,NZRA,NZRS,NZS,NZSH,NZTC,
     .  PHIAAD,PHIAID,P0,P1,P2,P3,QB,QBATT,QBRES,QOFF,QON,REFLH,RL,
     .  RLL,ROE,RSA,FSCELL,SADEGC,SADEGV,SARES,SOC,SPECOR,SPGRI,
     .  SUNLIT,SUNMW,TBATT,TBFRZI,TBDSTD,TCSTD,TCZIV,TCZT,TCZV,TEMTAB,
     .  THELAD,THELOD,TLLI,TLO,TP,TSHREF,TTAVE,TZBR,TZN,VBATT,VBUS,
     .  VBUSMN,VCHIST,VCHIT,VCHIOT,VCHTT,VCHVT,VDEGA,VLBT,VLBTT,VLBVT,
     .  VLR,VMAXIV,VMINIV,VRISAT,VRIOT,VSAINC,VSHTOR,VV,VZBR,XCSTD,
     .  XIBATT,XICHMX,XIHIT,XIHTT,XIHVT,XII,XN,XPLT,ZALPHA,ZCHRAT,
     .  ZCHRST,ZDIMP,ZDIMPT,ZDIMPV,ZPRCNT,ZRAT,ZRST,ZSHTAB,ZTCOEF/
C
       READ(NRD,10,ERR=50,END=80) IPRG,ITAPE,DEBUG,XLN,YLN
   10  FORMAT( )
       READ(NRD,INPT,ERR=100,END=150)   @READ NAMELIST INPUT DATA
       IF(IPRG.NE.1) CALL DSDRVR        @EXECUTE DESIGN SYNTHESIS
       IF(IPRG.EQ.0) GO TO 20           @DESIGN SYNTHESIS ONLY?
       IF(NPLT.EQ.0 .AND. XLN.LE.0.0) GO TO 12    @NO. PLOTTING REQUIRED?
       CALL PLOTS                       @YES. INITIALIZE PLOT PACKAGE
C      CALL PLOTS(IBUF,1000,6)          @YES. INITIALIZE PLOT PACKAGE
C      CALL PLOT(1.0,0.0,200)
C      CALL PLOT(0.5,0.5,-3)
   12  IF(ITAPE.NE.0) CALL NTRAN(MERGE,10,22)     @REWIND INPUT DATA FILE
       IF(IPRG.EQ.2)  READ(NRD,INPT,ERR=200,END=250)   @READ NEW INPUT
       CALL PADRVR                      @EXECUTE PERFORMANCE ANALYSIS
       WRITE(NWRT,15)
   15  FORMAT(/////'0 ARE SUMMARY OUTPUT TABLES DESIRED?')
       READ(NRD,18)  IPRG
   18  FORMAT(A6)
       IF(IPRG.EQ.'NO') GO TO 20
       CALL SUMARY                      @PRODUCE PA SUMMARY TABLES/PLOTS
       IF(NPLT.LE.1 .AND. XLN.LE.0.0) GO TO 20
       CALL PLOT(XLN+10.0,0.0,999)      @TERMINATE PLOT PACKAGE
   20  STOP DS/PA                       @TERMINATE DS/PA EXECUTION
C
```

```
   50 WRITE(NWRT,60)
   60 FORMAT('0***ERROR IN ATTEMT TO READ PROGRAM PARAMETERS')
      STOP
   80 WRITE(NWRT,90)
   90 FORMAT('0***PROGRAM PARAMETERS WERE NOT ENTERED')
      STOP
  100 WRITE(NWRT,110)
  110 FORMAT('0***ERROR IN NAMELIST INPUT'///)
      WRITE(NWRT,INPT)
      STOP
  150 WRITE(NWRT,160)
  160 FORMAT('0***END-OF-FILE ENCOUNTERED AT NAMELIST INPUT')
      STOP
  200 WRITE(NWRT,210)
  210 FORMAT('0***ERROR IN PERFORMANCE ANALYSIS NAMELIST INPUT'///)
      WRITE(NWRT,INPT)
      STOP
  250 WRITE(NWRT,260)
  260 FORMAT('0***END-OF-FILE ENCOUNTERED AT PERFORMANCE ANALYSIS ',
     .       'NAMELIST INPUT')
      STOP
C
      END
```

DSDRVR

```
      SUBROUTINE  DSDRVR
C      MAIN DRIVER PROGRAM FOR DESIGN SYNTHESIS
C
      INCLUDE  ALLCMN,LIST
      INCLUDE  COMON,LIST
      INCLUDE  DSCMN,LIST
      DIMENSION  CT(7), EBX(52,3), EBDX(3), ETA(11), NMR(52), SALT(11)
      DIMENSION  QQSOLT(52), QSOL(11), QSOLC(11), QSOLM(11), QSOLMX(52)
      DIMENSION  TC(7), TCZI(NTCZIV), TIMEC(11), TJT(52), TKT(52)
      DIMENSION  TLT(52), TOFF(11), TON(11), VCDSTD(11), VCHIVT(10)
      REAL  IFOFF, IFON, IZRF25, MAXI, MAXV, MSABDP
      DEFINE  TCNVRT(T) = 5.0 * (T + 459.67) / 9.0 - 273.15
C
      NAMELIST/DSOUT1/ TTABMX,TTABMN,DTTAMX,DTTAMN,DTABMX,DTABMN,
     .          IFOFF,IFON,PFOFF,PFON,XI,XV/
      NAMELIST/DSOUT2/ LWEEK,CT,TC,NWEEK,SST,SRT,DTIMEH,SALT,QSOLC,
     .          QSOL,TIMEC,QDT,NMR,TJT,TKT,TLT,QQSOLT,QSOLMX,QQSOL,
     .          TOFF,TON,TJTT,TKTT,TLTT,JOFFMX,JONMX/
      NAMELIST/DSOUT3/ EPCDJ,EPCDK,EPCDL,TTESMX,VCDMN,VCCMN,XIB,
     .          XVB,ETABVN,DQBBI,ETA,ETABQ,RATBAT,ETACHG,ETAD,
     .          VCHIO,VCHISA,TKTT,TLTT,PPSGAV,EPSG,ESA/
      NAMELIST/DSOUT4/ TSAFMX,QDTMX,PWRMXI,ETAEOM,MAXI,MAXV,MSAPWR,
     .          QQSTOT,ASCTNM,XNSCNM,VSCOC,NP,NS,NESP,NSCTOT,ASA,
     .          ASCTOT,WSA,ASATP,CSA,X12,V2/
      NAMELIST/DSOUT5/ TSAFMN,VSAOC,MAXI,MAXV,MSAPWR,MSABDP,
     .          PESBD,X12,V2/
      NAMELIST/DSOUT6/ BSTATE,EBTHMN,EBTHMX,QQSOLA,PPSGL,PPSGK,
     .          PBJ,PBK,PBL,EBX/
      NAMELIST/DSOUT7/ EBDTH,EESG1,NMRT,NCYCLE,DOD,EESG2,PPSGMX,
     .          PBCHMX,TTESMN,VCDMX,VCCMX,ETABVX,EESG3,EESG,XIB,XVB/
      NAMELIST/DSOUT8/ VBCHMX,VCHIO,VCHISA,VCCHMN,XNCBMX,VCCHMX,
     .          XNCBMN,XNCELL,NCELL,VCDSTD,VCDAVG,CBDSTD,JBTOT,
     .          JBMAX,XNBATT,NBATT,CBDT,CBD,EBDA,DODA,YICHMX,ZICHMX,
     .          WBATT,CBATT,XIB,XVB/
      NAMELIST/DSOUT9/ ICHRT,PCHG,WCHG,CCHG,NCHG,ISH,WSL,CSL,
     .          VSLOP,XISLOP,TSAC,PZRF25,XNZS,NZS,VZOP,XIZOP,
     .          IZRF25,VZRF25,VZBR,TZBR,NZT,NSLP,TSHREF,VSHTOR,
     .          WPWR,CPWR,VBUS,CB,XN,XICHMX/
C
      IF(VBUS.GE.VBUSMN) GO TO 10          @VBUS TOO LOW?
      VBUS = VBUSMN + 4.0                  @YES. REDEFINE VBUS
      WRITE(NWRT,5) VBUS                   @DISPLAY DIAGNOSTIC MESSAGE
    5 FORMAT('0***VBUS ADJUSTED TO ',F5.1,' VOLTS')
   10 IF(ITAPE.EQ.0) GO TO 20              @USE NOAA TAPE INPUT?
      CALL TMNMX                           @YES.
      TTABMX = TTAMB                       @SET MAXIMUM AMBIENT TEMPERATURE
      TTABMN = TSAF                        @SET MINIMUM AMBIENT TEMPERATURE
      DATEM = 365.242 * DURAM              @COMPUTE MISSION DURATION
      GO TO 50
   20 CALL SLUP(1.0,DTTAMX,FDOT,DTTA1(1,1),DTTA1(1,2),NDTTA,1)  @NO.
      DTTAMN = DTTAMX
      DO 25 I=2,365                        @SET DAILY TEMPERATURE
      DATE = FLOAT(I)                      @ INCREMENT EXTREMES
      CALL SLUP(DATE,DTTA,FDOT,DTTA1(1,1),DTTA1(1,2),NDTTA,1)
      DTTAMX = AMAX1(DTTAMX,DTTA)          @MAXIMUM INCREMENT
   25 DTTAMN = AMIN1(DTTAMN,DTTA)          @MINIMUM INCREMENT
```

```
   30 CALL SLUP(0.0,DTABMX,FDOT,DTAMB1(1,1),DTAMB1(1,2),NDTAMB,1)
      DTABMN = DTABMX
      DO 35 I=1,24                          @SET HOURLY TEMPERATURE
      TIMEH = FLOAT(I)                      @ INCREMENT EXTREMES
      CALL SLUP(TIMEH,DTAMB,FDOT,DTAMB1(1,1),DTAMB1(1,2),NDTAMB,1)
      DTABMX = AMAX1(DTABMX,DTAMB)          @MAXIMUM INCREMENT
   35 DTABMN = AMIN1(DTABMN,DTAMB)          @MINIMUM INCREMENT
   40 TTABMX = TTAVE + DTTAMX + DTABMX      @MAXIMUM AMBIENT TEMPERATURE
      TTABMN = TTAVE + DTTAMN + DTABMN      @MINIMUM AMBIENT TEMPERATURE
C
   50 TTAMB = TTABMX                        @SET AMBIENT TEMPERATURE
      CALL DSPCDG(1)                        @COMPUTE PCDG I-V CHARACTERISTICS
      CALL SLUP(VBUS,IFOFF,FDOT,XV,XI,NPCDG,1)   @SET LAMP-OFF CURRENT
      CALL DSPCDC(2)
      CALL SLUP(VBUS,IFON,FDOT,XV,XI,NPCDG,1)    @SET LAMP-ON CURRENT
   60 PFOFF = VBUS * IFOFF                  @COMPUTE PCDG LAMP-OFF LOAD
      PFON = VBUS * IFON                    @COMPUTE PCDG LAMP-FLASHING LOAD
      IF(DEBUG.NE.0.0)  WRITE(NWRT,DSOUT1)
C
   70 READ(NRD,75,ERR=8000,END=530)  LWEEK,(CT(I),TC(I),I=1,7)
   75 FORMAT( )
      IF(LWEEK.GT.52) GO TO 530             @END OF TEST YEAR?
      NWEEK = LWEEK                         @NO. STORE WEEK NUMBER
   80 DATE = 7.0 * NWEEK - 6.0              @COMPUTE DATE OF TEST
   90 CALL TERMC                            @GET TERMINATOR CHARACTERISTICS
  110 DTIMEH = (SST - SRT) / 10.0           @COMPUTE DAILY TIME INCREMENT
  130 DO 140 I=1,11,10                      @COMPUTE INITIAL/FINAL
      SALT(I) = 0.0                         @ SOLAR ALTITUDE
      QSOLC(I) = 0.0                        @ CLEAR-DAY SOLAR INSOLATION
  140 TIMEC(I) = SRT + (I - 1) * DTIMEH     @ TIME OF DAY
      TIMEH = SRT + DTIMEH
      DO 170 I=2,10
      IF(ITAPE.EQ.0) GO TO 150              @USE NOAA TAPE INPUT?
      CALL RDTAPE                           @YES.
      QSOL(I) = QDT                         @SET NOAA SOLAR INSOLATION
      GO TO 160
  150 CALL CDSI(SALT(I))                    @NO. COMPUTE CLEAR-DAY INSOLATION
  160 QSOLC(I) = QDT
      TIMEC(I) = TIMEH
  170 TIMEH = TIMEH + DTIMEH                @INCREMENT DAILY TIME
C
  190 NMR(NWEEK) = 0                        @INITIALIZE WEEKLY SUMMARY DATA
      TJT(NWEEK) = 0.0
      TKT(NWEEK) = 0.0
      TLT(NWEEK) = 0.0
      QQSOLT(NWEEK) = 0.0
      QSOLMX(NWEEK) = 0.0
      DO 520 L=1,7
      IF(ITAPE.NE.0) GO TO 230              @USE NOAA TAPE INPUT?
  200 J = 1 + IFIX(CT(L))                   @NO. SET CLOUD TYPE INDICATOR
      DO 220 I=1,11
  210 CCM = 1.0                             @INITIALIZE CLOUD COVER MODIFIER
      IF(TC(L).EQ.0.0) GO TO 220
      K = 1                                 @SET SOLAR ALTITUDE INDICATOR
      IF(SALT(I).GT.PI/9.0) K = 2
      CCM = P0(J,K) + P1(J,K) * TC(L) + P2(J,K) * TC(L)**2.0 +
     .      P3(J,K) * TC(L)**3.0            @COMPUTE CLOUD COVER MODIFIER
```

```
  220 QSOL(I) = CCM * QSOLC(I)          @COMPUTE INCIDENT RADIATION
  230 CALL INTEG(1,11,DTIMEH,QSOL,QQSOL)    @COMPUTE TOTAL RADIATION
C
  240 QSOLM(L) = QSOL(1)
      DO 245 I=2,11                     @COMPUTE MAXIMUM SOLAR RADIATION
  245 QSOLM(L) = AMAX1(QSOLM(L),QSOL(I))
  250 JTOFF = 1                         @INITIALIZE FLASHER LOAD COUNTERS
      JTON = 1
  260 DO 360 I=2,11
      IF(QSOL(I).LT.QOFF) GO TO 310
  270 IF(INDFLS.NE.1) GO TO 310         @SELECT FLASHER CONDITION
  280 CALL SLUP(QOFF,TOFF(JTOFF),FDOT,QSOL(I-1),TIMEC(I-1),2,1)
  290 INDFLS = 0                        @RESET FLASHER CONDITION FLAG
  300 JTOFF = JTOFF + 1                 @INCREMENT LOAD TURN-OFF COUNTER
      GO TO 360
  310 IF(QSOL(I).GT.QON) GO TO 360
  320 IF(INDFLS.NE.0) GO TO 360         @SELECT FLASHER CONDITION
  330 CALL SLUP(QON,TON(JTON),FDOT,QSOL(I-1),TIMEC(I-1),2,1)
  340 INDFLS = 1                        @RESET FLASHER CONDITION FLAG
  350 JTON = JTON + 1                   @INCREMENT LOAD TURN-ON COUNTER
  360 CONTINUE
C
  380 JOFFMX = JTOFF - 1                @SET FLASHER TURN-OFF COUNT
      JONMX = JTON - 1                  @SET FLASHER TURN-ON COUNT
      DNMR = JOFFMX + JONMX             @COMPUTE DAILY MODE REVERSALS
  390 TJTT = SRT + 24.0 - SST           @DURATION OF DAILY OCCULTATION
  400 TKTT = TOFF(1) - SRT              @DURATION OF DAILY SHARE MODE
      TLTT = 0.0                        @DURATION OF DAILY CHARGING
  410 JTOFF = 1                         @RESET FLASHER LOAD COUNTERS
      JTON = 1
  420 IF(JTOFF.NE.JTON) GO TO 460
  430 DTL = TON(JTON) - TOFF(JTOFF)     @SET CHARGING PERIOD INCREMENT
      TLTT = TLTT + DTL                 @INCREMENT CHARGING DURATION
  440 JTOFF = JTOFF + 1                 @INCREMENT LOAD TURN-OFF COUNTER
  450 IF(JTOFF.GT.JOFFMX) GO TO 480     @MAXIMUM NUMBER OF TURN-OFFS?
      GO TO 420                         @NO.
  460 DTK = TOFF(JTOFF) - TON(JTON)     @SET SHARE PERIOD INCREMENT
      TKTT = TKTT + DTK                 @INCREMENT SHARE MODE DURATION
  470 JTON = JTON + 1                   @INCREMENT LOAD TURN-ON COUNTER
      GO TO 420
C
  480 TJT(NWEEK) = TJT(NWEEK) + TJTT    @INCREMENT WEEKLY SUMMARY
      TKT(NWEEK) = TKT(NWEEK) + TKTT    & OPERATIONAL PERIODS
      TLT(NWEEK) = TLT(NWEEK) + TLTT
  490 NMR(NWEEK) = NMR(NWEEK) + DNMR    @INCREMENT WEEKLY MODE REVERSALS
  500 QQSOLT(NWEEK) = QQSOLT(NWEEK) + QQSOL      @INCREMENT WEEKLY SUM-
      QSOLMX(NWEEK) = AMAX1(QSOLMX(NWEEK),QSOLM(L))   @ MARY INSOLATIONS
  520 CONTINUE
      IF(DEBUG.NE.0.0)  WRITE(NWRT,DSOUT2)
      GO TO 70                          @GET NEXT INPUT
C
C*****PERFORM INITIAL POWER SUBSYSTEM COMPUTATIONS
C
  530 EPCDJ = 0.0                       @INITIALIZE YEARLY LOAD ENERGY
      EPCDK = 0.0
      EPCDL = 0.0
  540 DO 550 I=1,NWEEK                  @COMPUTE YEARLY
```

```
      EPCDJ = EPCDJ + PFON * TJT(I)      @ OCCULTATION LOAD ENERGY
      EPCDK = EPCDK + PFON * TKT(I)      @ SHARE MODE LOAD ENERGY
  550 EPCDL = EPCDL + PFOFF * TLT(I)     @ CHARGING MODE LOAD ENERGY
  560 TTESMX = TTABMX + DTTESG           @COMPUTE MAXIMUM ESG TEMPERATURE
  570 CALL DSESGC(QBRES,TTESMX)          @COMPUTE BATTERY I-V ARRAYS
  580 CALL SLUP(-ABS(BRDEST),VCDMN,FDOT,XIB,XVB,NIVB,1)
  590 CALL SLUP(BRCEST,VCCMN,FDOT,XIB,XVB,NIVB,1)
  600 ETABVN = VCDMN / VCCMN             @COMPUTE MINIMUM EFFICIENCY
  610 DQBB1 = (1.0 - QBRES) / 10.0       @SET STATE-OF-CHARGE INCREMENT
  620 QBB1 = QBRES                       @INITIALIZE STATE-OF-CHARGE
      DO 640 I=1,11
  630 CALL DSBEFC(BRCEST,ETA(I),QBB1,TTESMX)     @COMPUTE EFFICIENCY
  640 QBB1 = QBB1 + DQBB1                @INCREMENT STATE-OF-CHARGE
  660 CALL INTEG(1,11,DQBB1,ETA,ETABQ)       @COMPUTE AVERAGE EFFICIENCY
      ETABQ = ETABQ / (1.0 - QBRES)
  670 RATBAT = 1.0 / (ETABQ * ETABVN)    @COMPUTE CHARGE/DISCHARGE RATIO
C
  680 ETACHG = 1.0                       @INITIALIZE CHARGER EFFICIENCY
      ETAD = 1.0                         @ AND DISCHARGE LINE EFFICIENCY
      IF(ICHRT.EQ.0) GO TO 760           @CHARGER PRESENT? NO...
  690 ETAD = VBUS / (1.0 + VBUS)         @SET DISCHARGE LINE EFFICIENCY
  700 ETACHG = 0.0                       @REDEFINE CHARGER EFFICIENCY
      CALL SLUP(TTESMX,VCHIO,FDOT,VCHIOT(1,1),VCHIOT(1,2),NVCHIO,1)
      IF(VBUS.LE.VCHIO) GO TO 760
  710 CALL SLUP(TTESMX,VCHISA,FDOT,VCHIST(1,1),VCHIST(1,2),NVCHIS,1)
  720 ETACHG = 1.0 - VCHIO / VBUS        @REDEFINE CHARGER EFFICIENCY
      IF(VBUS.LE.VCHISA) GO TO 760
  730 ETACHG = (VCHISA - VCHIO) / VBUS       @REDEFINE CHARGER EFFICIENCY
C
  760 TKTT = 0.0
      TLTT = 0.0
      DO 770 I=1,NWEEK                   @COMPUTE DURATION OF YEARLY
      TKTT = TKTT + TKT(I)               @ SHARE MODE LOADS
  770 TLTT = TLTT + TLT(I)               @ CHARGE MODE LOADS
  780 PPSGAV = (RATBAT * (EPCDJ + EPCDK) + ETAD * ETACHG * EPCDL) /
     *          (RATBAT * TKTT + ETAD * ETACHG * TLTT)
  790 EPSG = PPSGAV * (TKTT + TLTT)      @COMPUTE PSG ENERGY REQUIREMENT
  800 ESA = EPSG / (VBUS / (1.0 + VBUS))     @S/A ENERGY REQUIREMENT
      IF(DEBUG.NE.0.0)  WRITE(NWRT,DSOUT3)
C
C*****COMPUTE SOLAR ARRAY WEIGHT AND COST
C
  810 TSAFMX = TTABMX + DTTPSG           @COMPUTE MAXIMUM S/A TEMPERATURE
  820 QDTMX = QSOLMX(1)
      DO 825 I=2,NWEEK                   @COMPUTE MAXIMUM INSTANTANEOUS
  825 QDTMX = AMAX1(QDTMX,QSOLMX(I))     @ SOLAR RADIATION
  830 TSAF = TSAFMX                      @SET SOLAR ARRAY PARAMETERS
      QDT = QDTMX
      DATEM = 365.242 * DURAM
      NS = 50
      NP = 50
      NESP = 1
      CALL SAEC(MAXI,MAXV)               @COMPUTE S/A I-V CHARACTERISTICS
C
      PWRMX1 = MSAPWR / (NESP * NP * NS)
  840 ETAEOM = PWRMX1 / (ACELL * 1.0E-4 * QDT)   @SOLAR CELL EFFICIENCY
  850 QQSTOT = 0.0
```

```
      DO 855 I=1,NWEEK                    @COMPUTE TOTAL YEARLY RADIATION
  855 QQSTOT = QQSTOT + QQSOLT(I)
  860 ASCTNM = ESA / (ETAEQM * QQSTOT * (1.0 - SARES))     @S/C AREA
  870 XNSCNM = ASCTNM / (ACELL * 1.0E-4)    @ESTIMATE TOTAL SOLAR CELLS
  880 VSCOC = V2(MFINAL-1)                  @COMPUTE OPEN CIRCUIT VOLTAGE
  890 NS = 1 + (1 + VBUS) / ((MAXV + VSCOC) / (2 * NS))
  900 NESP = 1 + XNSCNM / (NS * NPREQ)
  910 NSCTOT = NESP * NS * NPREQ            @TOTAL NO. OF SOLAR CELLS REQR'D
  920 ASCTOT = ACELL * NSCTOT * 1.0E-4      @TOTAL AREA OF SOLAR CELLS
  930 ASA = ASCTOT / (CELPAC * 9.29E-2)     @TOTAL SOLAR ARRAY AREA
  940 CALL SLUP(ASA,WSA,FDOT,DWDAT(1,1),DWDAT(1,2),NDWDA,1)
      WSA = ASA * WSA                       @COMPUTE SOLAR ARRAY WEIGHT
  950 ASATP = NSAP * ASA
      CALL SLUP(ASATP,CSA,FDOT,DCDAT(1,1),DCDAT(1,2),NDCDA,1)
      CSA = ASA * CSA                       @COMPUTE SOLAR ARRAY COST
      IF(DEBUG.NE.0.0)  WRITE(NWRT,DSOUT4)
C
  960 TSAFMN = TTABMN + DTTPSG              @COMPUTE MINIMUM S/A TEMPERATURE
  970 TSAF = TSAFMN                         @SET SOLAR ARRAY PARAMETERS
      QDT = QDTMX
      DATEM = 0.0
      NP = NPREQ
      CALL SAEC(MAXI,MAXV)                  @COMPUTE S/A I-V CHARACTERISTICS
  980 VSAOC = V2(MFINAL-1)                  @COMPUTE OPEN CIRCUIT VOLTAGE
  990 MSABDP = MSAPWR / (1.0 + VBUS)        @BLOCKING DIODE POWER LOSS
 1000 PESBD = MSABDP / NESP                 @S/A BLOCKING DIODE RATING
      IF(DEBUG.NE.0.0)  WRITE(NWRT,DSOUT5)
C
C*****COMPUTE BATTERY WEIGHT AND COST
C
 1010 BSTATE = 0.0
      EBTHMN = 0.0
      EBTHMX = 0.0
      DO 1120 I=1,NWEEK
 1030 QQSOLA = QQSTOT / NWEEK             @COMPUTE AVERAGE INSOLATION
 1040 PPSGL = PPSGAV * QQSOLT(I) / QQSOLA   @COMPUTE PSG POWER OUTPUT
      PPSGK = PPSGL * QOFF / QSOLMX(I)
 1050 PBJ = -PFON / ETAD                    @COMPUTE LOAD PROFILE POWER LEVELS
      PBK = (PPSGK - PFON) / ETAD
      PBL = ETACHG * (PPSGL - PFOFF) / RATBAT
 1060 IF(PBK.GT.0.0) PBK = 0.0
 1070 EBX(I,1) = PBJ * TJT(I)              @COMPUTE ENERGY PROFILE LEVELS
      EBX(I,2) = PBK * TKT(I)
      EBX(I,3) = PBL * TLT(I)
      EBDX(1) = EBX(I,1) / 7.0
      EBDX(2) = EBX(I,2) / 7.0
      EBDX(3) = EBX(I,3) / 7.0
      DO 1100 K=1,7
 1080 DO 1100 J=1,3                         @COMPUTE BATTERY ENERGY STATE
      BSTATE = BSTATE + EBDX(K)
      EBTHMN = AMIN1(BSTATE,EBTHMN)
 1100 EBTHMX = AMAX1(BSTATE,EBTHMX)
      IF(DEBUG.NE.0.0)  WRITE(NWRT,DSOUT6)
 1120 CONTINUE
C
 1130 EBDTH = EBTHMX - EBTHMN               @DISCHARGE ENERGY REQUIREMENT #1
 1140 EESG1 = EBDTH / (1.0 - QBRES)
```

```
 1150 NMRT = 0
      DO 1155 I=1,NWEEK                    @COMPUTE YEARLY BATTERY MODE
 1155 NMRT = NMRT + NMR(I)                 @ REVERSALS
 1160 NCYCLE = 1 + 26 * DURAM * NMRT / NWEEK     @TOTAL CYCLE REQUIREMEN
 1170 CALL SLUP(ALOG(NCYCLE),DOD,FDOT,DODT(1,1),DODT(1,2),NDOD,1)
 1180 IF(DOD.LT.0.0) DOD = 0.001
      IF(DOD.GT.1.0) DOD = 1.0
 1190 EESG2 = EBDTH / DOD                  @DISCHARGE ENERGY REQUIREMENT #2
 1200 PPSGMX = MSAPWR - MSABDP             @MAXIMUM PSG POWER OUTPUT
 1210 PBCHMX = ETACHG * (PPSGMX - PFOFF)     @MAXIMUM CHARGE POWER
C
 1220 TTESMN = TTABMN + DTTESG             @COMPUTE ESG MINIMUM TEMPERATURE
 1230 CALL DSESGC(1.0,TTESMN)              @COMPUTE BATTERY I-V ARRAYS
      CALL SLUP(-ABS(BRDEST),VCDMX,FDOT,XIB,XVB,NIVB,1)
 1240 CALL SLUP(BRCEST,VCCMX,FDOT,XIB,XVB,NIVB,1)
 1250 ETABVX = VCDMX / VCCMX               @MAXIMUM BATTERY EFFICIENCY
 1260 EESG3 = PBCHMX * ETABVX / BRCHMX       @DISCHARGE ENERGY #3
 1270 EESG = AMAX1(EESG1,EESG2)            @COMPUTE ACTUAL BATTERY ENERGY
      EESG = AMAX1(EESG,EESG3)             @ DISCHARGE CAPABILITY
      IF(DEBUG.NE.0.0)  WRITE(NWRT,DSOUT7)
C
 1290 VBCHMX = VSAOC - 1.0                 @INITIALIZE MAX. CHARGE VOLTAGE
      IF(ICHRT.GT.0) GO TO 1330            @SELECT CHARGER TYPE
 1320 IF(ISH.GT.0) VBCHMX = VBUS + 0.75 * (VBCHMX - VBUS)
      GO TO 1390
 1330 ISH = 0                              @RESET SHUNT LIMITER TYPE FLAG
 1340 CALL SLUP(TTESMN,VCHIO,FDOT,VCHIOT(1,1),VCHIOT(1,2),NVCHIO,1)
 1350 IF(VBCHMX.LE.VCHIO) GO TO 8500       @ERROR IN 'VBUS' INPUT VALUE
      CALL SLUP(TTESMN,VCHISA,FDOT,VCHIST(1,1),VCHIST(1,2),NVCHIS,1)
 1360 IF(VBCHMX.GT.VCHISA) GO TO 1380
 1370 VBCHMX = VBCHMX - VCHIO
      GO TO 1390
 1380 CALL TB2SET(VCHVT,NVCHV,2,1,VCHTT,NVCHT,2,1,VCHIT,NVCHI,IERR)
      DO 1385 I=1,NVCHV
 1385 VCHIVT(I) = TB2GET(VCHVT(I),TTESMN)
      CALL SLUP(VBCHMX,VEST,FDOT,VCHIVT,VCHVT,NVCHV,1)
      VBCHMX = VEST
C
 1390 CALL DSESGC(1.0,TTESMX)              @COMPUTE BATTERY I-V ARRAYS
      CALL SLUP(0.0,VCCHMN,FDOT,XIB,XVB,NIVB,1)
 1400 XNCBMX = VBCHMX / VCCHMN             @MAXIMUM NO. OF CELLS IN SERIES
 1410 CALL DSESGC(0.0,TTESMN)              @COMPUTE BATTERY I-V ARRAYS
      CALL SLUP(-ABS(BRDEST),VCCHMX,FDOT,XIB,XVB,NIVB,1)
 1420 XNCBMN = VBUSMN / VCCHMX             @MINIMUM NO. OF CELLS IN SERIES
 1430 XNCELL = XNCBMN + FRCELL * (XNCBMX - XNCBMN)
 1440 NCELL = IFIX(XNCELL)
      DNCELL = XNCELL - NCELL
      IF(DNCELL.GT.0.5) NCELL = NCELL + 1
 1450 QBS = 0.0                            @INITIALIZE STATE-OF-CHARGE
      DO 1480 I=1,11
 1470 CALL DSESGC(QBS,TBDSTD)              @COMPUTE BATTERY I-V ARRAYS
      CALL SLUP(-ABS(BRDSTD),VCDSTD(I),FDOT,XIB,XVB,NIVB,1)
 1480 QBS = QBS + 0.1                      @INCREMENT STATE-OF-CHARGE
C
 1490 CALL INTEG(1,11,0.1,VCDSTD,VCDAVG)     @AVERAGE CELL DISCHARGE VLTG
 1500 CBDSTD = EESG / (NCELL * VCDAVG)     @TOTAL ESG DISCHARGE CAPACITY
 1510 DO 1530 I=1,30                       @FIND CELL SIZE INDEX
```

```
      JBTOT = 1                                 @MAXIMUM NO. OF TABLE ENTRIES
 1530 IF(CBAVAL(1).LE.0.0) GO TO 1550
 1540 DO 1570 I=1,JBTOT                         @FIND DESIRED CAPACITY INDEX
      JBMAX = I - 1
 1570 IF(CBMAX.LE.CBAVAL(I)) GO TO 1590
      JBMAX = JBTOT
 1590 XNBATT = CBDSTD / CBAVAL(JBMAX)           @NUMBER OF BATTERIES IN PARALLEL
 1600 IF(XNBATT.LT.1.0) GO TO 1610
      JBB = JBMAX
      GO TO 1660
C
 1610 JB = JBMAX - 1                            @LESS THAN 1 BATTERY: RECOMPUTE
      DO 1650 I=1,JB                            @ NO. OF BATTERIES IN PARALLEL
 1630 JBB = JBMAX - I
 1640 XNBATT = CBDSTD / CBAVAL(JBB)
 1650 IF(XNBATT.GE.1.0) GO TO 1660
      NBATT = 1                                 @DEFAULT = 1 BATTERY
      CBDT = CBDSTD                             @SET DEFAULT STORAGE CAPACITY
      CBD = CBDT
      GO TO 1740
C
 1660 IF(XNBATT.LE.10.0) GO TO 1720             @MORE THAN 10 BATTERIES IN
 1670 JBMAX = JBMAX + 1                         @ PARALLEL: RECOMPUTE NO.
      JB = JBTOT - 1                            @ OF BATTERIES
      DO 1710 I=JBMAX,JB
 1690 JBB = I
 1700 XNBATT = CBDSTD / CBAVAL(JBB)
 1710 IF(XNBATT.LE.10.0) GO TO 1720
      NBATT = 10                                @DEFAULT = 10 BATTERIES
      CBDT = CBDSTD                             @SET DEFAULT STORAGE CAPACITY
      CBD = CBDSTD / NBATT
      GO TO 1740
C
 1720 NBATT = IFIX(XNBATT)                      @COMPUTE NO. OF BATTERIES
      DNBATT = XNBATT - NBATT
      IF(DNBATT.GE.0.5) NBATT = NBATT + 1
 1730 CBD = CBAVAL(JBP)                         @COMPUTE ESG STORAGE CAPACITY
      CBDT = NBATT * CBD
 1740 EBDA = CBDT * NCELL * VCDAVG              @COMPUTE TOTAL BATTERY ENERGY
      DODA = EBDTH / EBDA                       @SET MAXIMUM DEPTH-OF-DISCHARGE
      YICHMX = CBDT * BACHMX                    @MAXIMUM ESG CHARGING CURRENT
      ZICHMX = YICHMX / NBATT                   @MAXIMUM BATTERY CHARGING CURRENT
 1750 CALL SLUP(CBD,WBATT,FDOT,DWDET(1,1),DWDET(1,2),NDWDE,1)
      WBATT = EBDA * WBATT                      @TOTAL BATTERY WEIGHT
 1760 CALL TB2SET(DCDCT,NDCDC,2,1,DCDNT,NDCDN,2,1,DCDET,NDCDE,IERR)
      CBATT = EBDA * TB2GLT(CBD,NBATP)          @COMPUTE TOTAL BATTERY COST
      IF(DEBUG.NE.0.0)  WRITE(NWRT,DSOUTB)
C
C*****COMPUTE BATTERY CHARGER WEIGHT AND COST
C
 1770 PCHG = 0.0                                @INITIALIZE CHARGER LOAD
      WCHG = 0.0                                @INITIALIZE CHARGER WEIGHT
      CCHG = 0.0                                @INITIALIZE CHARGER COST
      IF(ICHRT.EQ.0) GO TO 1810                 @CONSTANT VOLTAGE CHARGER?
 1780 PCHG = PBCHMX / NBATT                     @YES, COMPUTE CHARGER LOAD
 1790 CALL SLUP(PCHG,WCHG,FDOT,DWDCHT(1,1),DWDCHT(1,2),NDWDCH,1)
      WCHG = NBATT * PCHG * WCHG                @COMPUTE CHARGER WEIGHT
```

```
 1800 CALL TB2SET(DCDCPT,NDCDCP,2,1,DCDCNT,NDCDCN,2,1,DCDCHT,NDCDCH,
     .             IERR)
      NCHG = NBATP
      CCHG = NBATT * PCHG * TB2GET(PCHG,NCHG)     @COMPUTE CHARGER COST
C
C*****COMPUTE SHUNT LIMITER WEIGHT AND COST
C
 1810 WSL = 0.0                                   @INITIALIZE SHUNT LIMITER WEIGHT
      CSL = 0.0                                   @INITIALIZE SHUNT LIMITER COST
      IF(ISH.EQ.0) GO TO 2150                     @SHUNT LIMITER PRESENT?
 1820 VSLOP = VBCHMX * NCELL / XNCBMX             @YES, SET OPERATING POINT I-V
      CALL SLOP(VSLOP,XISLOP,FDOT,V2,X12,MFINAL,1)
      TSAC = CONVRT(TSAFMN)
 1830 IF(ISH.GT.2) GO TO 2080                     @CHANGE TO CENTIGRADE TEMPERATURE
 1840 IPSG = 1                                    @YES, SET POWER SOURCE GROUP TYPE
 1850 PZRFZS = VDER * HDZMX                       @SET REFERENCE POWER LEVEL
 1860 XNZS = PPSGMX / (NESP * PZRFZS)             @COMPUTE NO. OF ZENERS IN STRING
      NZS = IFIX(XNZS)
      DNZS = XNZS - NZS
      IF(DNZS.GE.0.5) NZS = NZS + 1
 1880 TCZ = TSAC                                  @SET ZENER OPERATING TEMPERATURE
 1890 VZOP = VSLOP / NZS                          @COMPUTE ZENER DIODE
      XIZOP = XISLOP / NESP                       @ OPERATING POINT
 1900 IF(ISH.GT.1) GO TO 2010                     @ORDINARY ZENER DIODE?
C
 1920 VZB30 = VZOP                                @YES, SET BREAKDOWN VOLTAGE
      TI = (TCZ - 30.0) / 100.0
      CALL TB2SET(ZDIMPV,NZDV,2,1,ZDIMPT,NZDT,2,1,ZDIMP,NZDIMP,IERR)
 1930 DO 2000 I=1,25                              @COMPUTE BREAKDOWN VOLTAGE
 1940 CALL SLOP(VZB3C,TCO,FDOT,ZTCOEF(1,1),ZTCOEF(1,2),NTCZ,1)
 1960 VZBR = VZOP - XIZOP * TB2GET(VZB30,TCO)
 1970 VZB = VZBR * (1.0 - TCO * TI)
 1980 DVZB = ABS((VZB - VZB30) / VZB30)
 1990 IF(DVZB.LE.0.1) GO TO 2050                  @ACCEPTABLE BREAKDOWN VOLTAGE?
 2000 VZB30 = VZB                                 @NO, REDEFINE REFERENCE VOLTAGE
      GO TO 2050
C                                                 @TEMPERATURE-COMPENSATED ZENER
 2010 CALL SLOP(HDZMX,IZRFZS,FDOT,CURZ(1,1),CURZ(1,2),NCURZ,1)
 2020 VZRFZS = PZRFZS / IZRFZS
 2030 CALL TB2SET(TCZV,NTCZV,2,1,TCZT,NTCZT,2,1,TCZIV,NTCZIV,IERR)
      DO 2035 I=1,NTCZV
 2035 TCZI(I) = TB2GET(TCZV(1),TCZ)
      CALL SLOP(0.0,RATVB,FDOT,TCZI,TCZV,NTCZV,1)
 2040 VZBR = RATVB * VZRFZS                       @COMPUTE BREAKDOWN VOLTAGE
 2050 TZBR = TCZ                                  @COMPUTE BREAKDOWN TEMPERATURE
 2060 NZT = NESP * NZS
      CALL SLOP(PZRFZS,WSL,FDOT,DWDNZT(1,1,ISH),DWDNZT(1,2,ISH),
     .          NDWDNZ,1)
      WSL = NZT * WSL                             @COMPUTE ZENER DIODE WEIGHT
 2070 NSLP = NSAP * NZT
      CALL TB2SET(DCDNPT(1,ISH),NDCDNP,2,1,DCDNNT(1,ISH),NDCDNN,2,1,
     .            DCDNZT(1,1,ISH),NDCDNZ,IERR)
      CSL = NZT * TB2GET(PZRFZS,NSLP)             @COMPUTE ZENER DIODE COST
      GO TO 2150
C                                                 @ACTIVE SHUNT LIMITER
 2080 IPSG = 0                                    @SET POWER SOURCE GROUP TYPE
 2090 TSHREF = TSAC
```

```
 2100 CALL SLUP(TSHREF,ZSH,FDOT,ZSHTAB(1,1),ZSHTAB(1,2),NZSH,1)
 2110 VSHTOR = VSLOP - ZSH * XISLOP     @COMPUTE TURN-ON VOLTAGE
 2120 PSL = PPSGMX                      @COMPUTE LOAD
 2130 CALL SLUP(PSL,WSL,FDOT,DWDPST(1,1),DWDPST(1,2),NDWDPS,1)
      WSL = PSL * WSL                   @COMPUTE SHUNT LIMITER WEIGHT
 2140 NSLP = NSAP
      CALL TB2SET(DCDPPT,NDCDPP,2,1,DCDPNT,NDCDPN,2,1,DCDPST,NDCDPS,
     .            IERR)
      CSL = PSL * TB2GET(PSL,NSLP)      @COMPUTE SHUNT LIMITER COST
C
C*****COMPUTE TOTAL POWER SYSTEM WEIGHT AND COST
C
 2150 WPWR = WSA + WBATT + WCHG + WSL   @COMPUTE POWER SYSTEM WEIGHT
 2160 CPWR = CSA + CBATT + CCHG + CSL   @COMPUTE POWER SYSTEM COST
      CB = CRD                          @BATTERY DISCHARGE CAPACITY
      XN = NCELL                        @NO. OF STORAGE CELLS IN SERIES
      XICHMX = ZICHMX                   @MAXIMUM BATTERY CHARGING CURRENT
      IF(DEBUG.NE.0.0)  WRITE(NWRT,DSOUT9)
      CALL DSPRT                        @PRINT DESIGN SYNTHESIS OUTPUT
      RETURN
C
C*****ERROR MESSAGE EXITS
C
 8000 WRITE(NWRT,8100)
 8100 FORMAT('0***ERROR IN WEEK NUMBER INPUT')
      STOP
 8500 WRITE(NWRT,8600) VBUS,VBCHMX,VCHIO
 8600 FORMAT('0***VBUS TOO LOW TO TURN ON BATTERY CHARGER:',3(2X,F6.2))
      STOP
C
C**********************  INTERNAL SUBROUTINES  **********************
C
      SUBROUTINE  INTEG(N1,NN,D,A,AREA)
C      COMPUTES AREA UNDER ARRAY 'A' USING SIMPSON'S INTEGRATION RULE
C
      DIMENSION  A(NN)
      AR1 = A(N1) + A(NN)
      AR2 = 0.0
      AR3 = -A(NN)
      I1 = N1 + 1
      I2 = NN - 1
      DO 100 I=I1,I2,2
      AR2 = AR2 + A(I)
  100 AR3 = AR3 + A(I+1)
      AREA = D * (AR1 + 2.0 * AR2 + 4.0 * AR3) / 3.0
      RETURN
C
C
      SUBROUTINE  DSPRT
C      WRITES DESIGN SYNTHESIS OUTPUT TABLES TO PRINTER
C
C*****PRINT TABLE 1:  SYSTEM DESIGN CHARACTERISTICS
C
      WRITE(NWRT,3000)
 3000 FORMAT('1',T42,'NAVIGATION AID POWER SYSTEM DESIGN ',
     .       'CHARACTERISTICS',T122,'DS-PAGE 01'///)
      WRITE(NWRT,3010)  DURAM,THELAD
```

```
 3010 FORMAT(' MISSION DURATION =',T38,E9.4,' YEARS',
     .    T67,'BUOY LATITUDE =',T97,E9.4,' DEGREES')
      WRITE(NWRT,3020)  NWEEK,THELOD
 3020 FORMAT(' DESIGN PERIOD =',T45,I2,' WEEKS',
     .    T67,'BUOY LONGITUDE =',T97,E9.4,' DEGREES')
      WRITE(NWRT,3030)  VBUS,TZN
 3030 FORMAT(' NOMINAL OPERATING VOLTAGE =',T38,E9.4,' VOLTS',
     .    T67,'TIME ZONE NUMBER =',T97,E9.4)
      WRITE(NWRT,3040)  PHIAID,TTAVE
 3040 FORMAT(' SOLAR ARRAY SURFACE TILT ANGLE =',T38,E9.4,' DEGREES',
     .    T67,'AVERAGE YEARLY TEMPERATURE =  ',E9.4,' DEG. FAHRENHEIT')
      WRITE(NWRT,3050)  PHIAAD,TTABMX
 3050 FORMAT(' SOLAR ARRAY SURFACE AZIMUTH ANGLE = ',E9.4,' DEGREES',
     .    T67,'MAXIMUM AMBIENT TEMPERATURE = ',E9.4,' DEG. FAHRENHEIT')
      WRITE(NWRT,3060)  TTABMN
 3060 FORMAT(1X,T67,'MINIMUM AMBIENT TEMPERATURE = ',E9.4,
     .        ' DEG. FAHRENHEIT'//)
C
      WRITE(NWRT,3100)
 3100 FORMAT('0',T42,'DESIGN PERIOD LOAD ENERGY REQUIREMENTS ',
     .        '(WATT-HOURS)'//)
      WRITE(NWRT,3110)  EPCDJ,EPCDK,EPCDL
 3110 FORMAT(' FOR SOLAR OCCULTATION:  ',E9.4,
     .    T44,'FOR SHARE-MODE OPERATION:  ',E9.4,
     .    T90,'FOR BATTERY-CHARGING PERIODS:  ',E9.4//)
C
      WRITE(NWRT,3200)
 3200 FORMAT('0',T54,'USER SYSTEM REQUIREMENTS'//)
      WRITE(NWRT,3210)  IFTYPE
 3210 FORMAT(' FLASHER PATTERN TYPE = ',I2)
      DO 3220 I=1,16
 3220 IF(TL(I,IFTYPE+1).NE.0.0) N = I
      WRITE(NWRT,3230)  (TL(I,IFTYPE+1),I=1,N)
 3230 FORMAT(' FLASHER PATTERN =  ',16(F3.1,', '))
      WRITE(NWRT,3240)  QON
 3240 FORMAT(' SOLAR INSOLATION LEVEL FOR LAMP-FLASHER TURN-ON =',
     .    T67,E9.4,' WATTS/SQ.METER')
      WRITE(NWRT,3250)  QOFF
 3250 FORMAT(' SOLAR INSOLATION LEVEL FOR LAMP-FLASHER TURN-OFF =',
     .    T67,E9.4,' WATTS/SQ.METER')
      WRITE(NWRT,3260)  IFON
 3260 FORMAT(' POWER CONDITIONING AND DISTRIBUTION GROUP ',
     .        'LAMP-FLASHING CURRENT =',E9.4,' AMPERES')
      WRITE(NWRT,3270)  IFOFF
 3270 FORMAT(' POWER CONDITIONING AND DISTRIBUTION GROUP ',
     .        'LAMP-OFF CURRENT =',T67,E9.4,' AMPERES')
      WRITE(NWRT,3280)  PFON
 3280 FORMAT(' POWER CONDITIONING AND DISTRIBUTION GROUP ',
     .        'LAMP-FLASHING LOAD =',T67,E9.4,' WATTS')
      WRITE(NWRT,3290)  PFOFF
 3290 FORMAT(' POWER CONDITIONING AND DISTRIBUTION GROUP ',
     .        'LAMP-OFF LOAD =',T67,E9.4,' WATTS'//)
C
      WRITE(NWRT,3300)
 3300 FORMAT('0',T49,'INDIVIDUAL POWER SYSTEM CHARATERISTICS'/,
     .        T37,'NO. TO BE      WEIGHT',T66,'AREA',T81,'COST'/,
     .        ' SUBSYSTEM',T28,'TYPE      PROCURED      (POUNDS)      ',
```

```
     .       '(SQ. FEET)          ($)')
      WRITE(NWRT,3310)
 3310 FORMAT(1X,86('-'))
      WRITE(NWRT,3320)  IPSG
 3320 FORMAT(' POWER SOURCE GROUP',T29,I2)
      WRITE(NWRT,3330)  NSAP,WSA,ASA,CSA
 3330 FORMAT('     SOLAR ARRAY',T38,I6,T49,E9.4,T63,E9.4,T78,E9.4)
      WRITE(NWRT,3340)  ISH,NSLP,WSL,CSL
 3340 FORMAT('     SHUNT LIMITER',T29,I2,T38,I6,T49,E9.4,T78,E9.4)
      WRITE(NWRT,3350)
 3350 FORMAT(' ENERGY STORAGE GROUP')
      WRITE(NWRT,3360)  NBATP,WBATT,CBATT
 3360 FORMAT('     BATTERY',T38,I6,T49,E9.4,T78,E9.4)
      WRITE(NWRT,3370)  ICHRT,NCHG,WCHG,CCHG
 3370 FORMAT('     CHARGER',T29,I2,T38,I6,T49,E9.4,T78,E9.4)
      WRITE(NWRT,3310)
      WRITE(NWRT,3380)  WPWR,CPWR
 3380 FORMAT(' TOTALS',T49,E9.4,T78,E9.4)
C
C*****PRINT TABLE 2:  ENGINEERING DESIGN DATA
C
      WRITE(NWRT,4000)
 4000 FORMAT('1',T50,'SUMMARY OF ENGINEERING DESIGN DATA',T122,
     .       'DS-PAGE 02'/,1X,T51,'FOR NAVIGATION AID POWER SYSTEM'///)
      WRITE(NWRT,4010)  DURAM,EPSG
 4010 FORMAT(' MISSION DURATION =',T40,E9.4,' YEARS',T70,
     .   'POWER SOURCE GROUP ENERGY REQUIREMENT = ',E9.4,' WATT-HOURS')
      WRITE(NWRT,4020)  NWEEK,ESA
 4020 FORMAT(' DESIGN PERIOD =',T47,I2,' WEEKS',
     .   T70,'SOLAR ARRAY ENERGY REQUIREMENT =',T110,E9.4,' WATT-HOURS')
      WRITE(NWRT,4030)  QDTMX,MSAPWR
 4030 FORMAT(' MAXIMUM SOLAR RADIATION =',T40,E9.4,' WATTS/SQ.METER',
     .   T70,'MAXIMUM SOLAR ARRAY POWER =',T110,E9.4,' WATTS')
      WRITE(NWRT,4040)  QQSTOT,PPSGAV
 4040 FORMAT(' TOTAL DESIGN PERIOD SOLAR RADIATION = ',E9.4,
     .       ' WATT-HOURS/SQ.METER',
     .   T70,'AVERAGE POWER SOURCE GROUP POWER =',T110,E9.4,' WATTS'//)
C
      WRITE(NWRT,4100)
 4100 FORMAT('0',T59,'POWER SOURCE GROUP'//,
     .       ' SOLAR ARRAY:',T72,'SHUNT LIMITER:')
      WRITE(NWRT,4110)  ACELL,ISH
 4110 FORMAT('     AREA OF A SINGLE SOLAR CELL =',T47,E9.4,
     .       ' SQ.CENTIMETERS',
     .   T75,'TYPE OF SHUNT LIMITER =',T121,I2)
      IF(ISH.GT.0) GO TO 4200            @SHUNT LIMITER PRESENT?
      WRITE(NWRT,4120)  NP                 @NO.
 4120 FORMAT('     NO. OF SOLAR CELLS IN PARALLEL =',T50,I6)
      WRITE(NWRT,4130)  NS
 4130 FORMAT('     NO. OF SOLAR CELLS IN SERIES =',T50,I6)
      WRITE(NWRT,4140)  NESP
 4140 FORMAT('     NO. OF ELECTRICAL SECTIONS IN PARALLEL =',T50,I6)
      WRITE(NWRT,4150)  NSCTOT
 4150 FORMAT('     TOTAL NO. OF SOLAR CELLS =',T47,I9)
      WRITE(NWRT,4160)  SARES
 4160 FORMAT('     SOLAR ARRAY RESERVE FRACTION =',T47,E9.4)
      WRITE(NWRT,4170)  PESBD
```

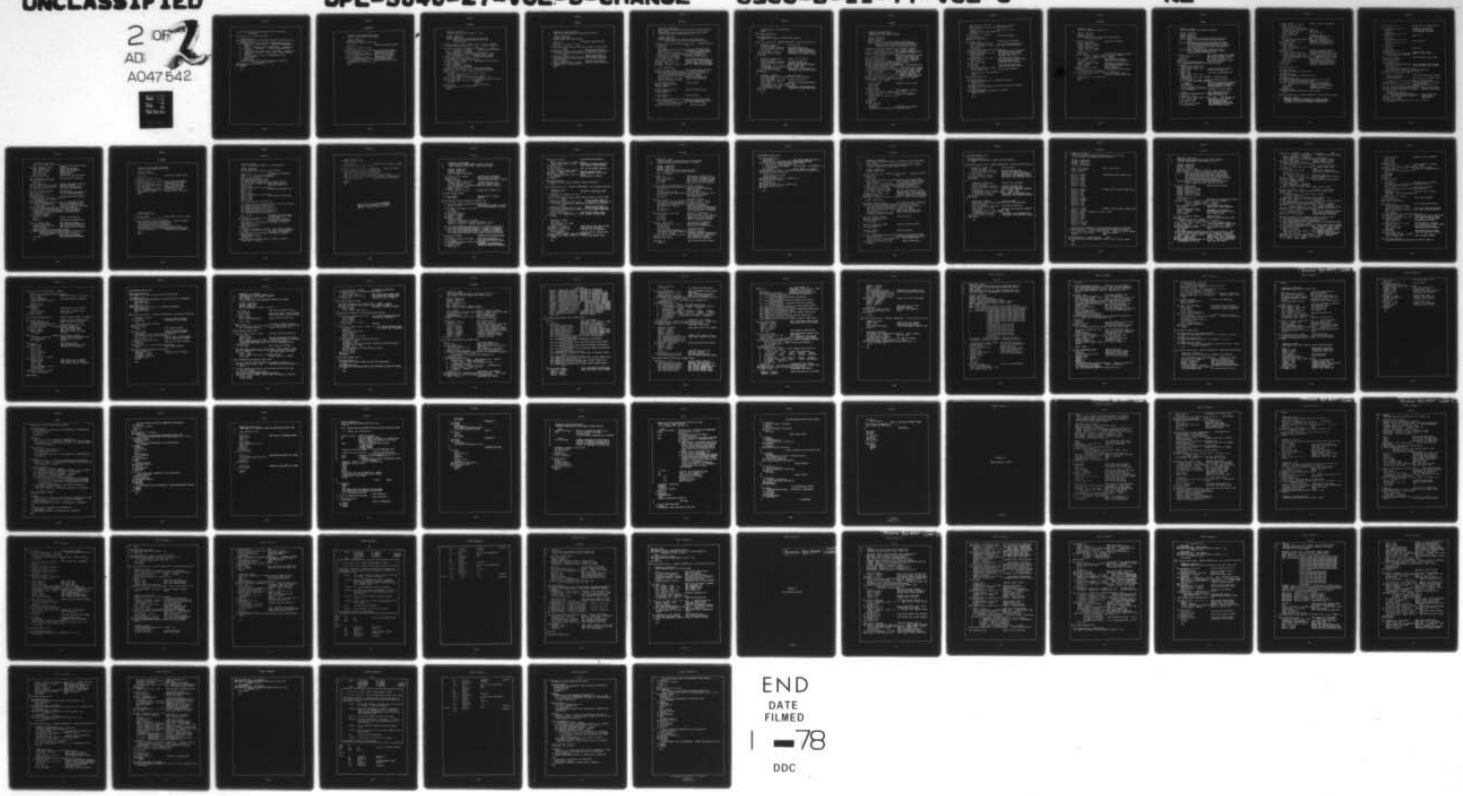
```
 4170 FORMAT('     ELECTRICAL SECTION BLOCKING DIODE RATING =',E9.4,
     .       ' WATTS'//)
      GO TO 4500
C
 4200 WRITE(NWRT,4210)  NP,VSLOP        @YES, SHUNT LIMITER PRESENT
 4210 FORMAT('     NO. OF SOLAR CELLS IN PARALLEL =',T50,I6,
     .   T75,'OPERATING VOLTAGE =',T115,E9.4,' VOLTS')
      WRITE(NWRT,4220)  NS,XISLOP
 4220 FORMAT('     NO. OF SOLAR CELLS IN SERIES =',T50,I6,
     .   T75,'OPERATING CURRENT =',T115,E9.4,' AMPERES')
      IF(ISH.LT.3) GO TO 4300           @ZENER DIODE?
      WRITE(NWRT,4230)  NESP,VSHTOR     @NO. ACTIVE SHUNT LIMITER
 4230 FORMAT('     NO. OF ELECTRICAL SECTIONS IN PARALLEL =',T50,I6,
     .   T75,'ACTIVE SHUNT LIMITER TURN-ON VOLTAGE =  ',E9.4,' VOLTS')
      WRITE(NWRT,4240)  NSCTOT,TSMREF
 4240 FORMAT('     TOTAL NO. OF SOLAR CELLS =',T47,I9,
     .   T75,'SHUNT LIMITER REFERENCE TEMPERATURE =   ',E9.4,' DEG. C')
      WRITE(NWRT,4250)  SARES,PSL
 4250 FORMAT('     SOLAR ARRAY RESERVE FRACTION =',T47,E9.4,
     .   T75,'ACTIVE SHUNT LIMITER LOAD =',T115,E9.4,' WATTS')
      WRITE(NWRT,4170)  PESBD
      GO TO 4500
C
 4300 WRITE(NWRT,4310)  NESP,VZOP       @ZENER DIODE
 4310 FORMAT('     NO. OF ELECTRICAL SECTIONS IN PARALLEL =',T50,I6,
     .   T75,'SINGLE ZENER DIODE OPERATING VOLTAGE =  ',E9.4,' VOLTS')
      WRITE(NWRT,4320)  NSCTOT,XIZOP
 4320 FORMAT('     TOTAL NO. OF SOLAR CELLS =',T47,I9,
     .   T75,'SINGLE ZENER DIODE OPERATING CURRENT =  ',E9.4,' AMPERES')
      WRITE(NWRT,4330)  SARES,HDZMX
 4330 FORMAT('     SOLAR ARRAY RESERVE FRACTION =',T47,E9.4,
     .   T75,'MAXIMUM ZENER DIODE HEAT DISSIPATION =  ',E9.4,' WATTS')
      WRITE(NWRT,4340)  PESBD,HDER
 4340 FORMAT('     ELECTRICAL SECTION BLOCKING DIODE RATING =',E9.4,
     .       ' WATTS',
     .   T75,'HEAT DISSIPATION DERATING FACTOR =',T115,E9.4)
      WRITE(NWRT,4350)  NZS
 4350 FORMAT(1X,T75,'NO. OF ZENER DIODES IN A STRING =',T118,I6)
      WRITE(NWRT,4360)  NZT
 4360 FORMAT(1X,T75,'TOTAL NO. OF ZENER DIODES =',T118,I6)
      WRITE(NWRT,4370)  VZBR
 4370 FORMAT(1X,T75,'ZENER DIODE BREAKDOWN VOLTAGE =',T115,E9.4,
     .       ' VOLTS')
      WRITE(NWRT,4380)  TZBR
 4380 FORMAT(1X,T75,'ZENER DIODE TEMPERATURE AT BREAKDOWN =  ',E9.4,
     .       ' DEG. C'//)
C
 4500 WRITE(NWRT,4510)
 4510 FORMAT('0',T58,'ENERGY STORAGE GROUP'//,
     .       ' BATTERY:',T72,'CHARGER:')
      WRITE(NWRT,4520)  NCELL,ICHRT
 4520 FORMAT('     NO. OF STORAGE CELLS IN SERIES =',T58,I3,
     .   T75,'TYPE OF CHARGER =',T122,I2)
      WRITE(NWRT,4530)  NBATT,PCHG
 4530 FORMAT('     NO. OF BATTERIES IN PARALLEL =',T58,I3,
     .   T75,'MAXIMUM LOAD FOR A SINGLE CHARGER =     ',E9.4,' WATTS')
      WRITE(NWRT,4540)  QBRES
```

```
 4540 FORMAT('     BATTERY RESERVE FRACTION =',T52,E9.4)
      WRITE(NWRT,4550)  CBD
 4550 FORMAT('     DISCHARGE CAPACITY FOR A SINGLE BATTERY =',T52,E9.4,
     .        ' AMP-HOURS')
      WRITE(NWRT,4560)  CBDT
 4560 FORMAT('     TOTAL DISCHARGE CAPACITY FOR ALL BATTERIES =   ',
     .  E9.4,' AMP-HOURS')
      WRITE(NWRT,4570)  ZICHMX
 4570 FORMAT('     MAXIMUM CHARGING CURRENT FOR A SINGLE BATTERY =',
     .  E9.4,' AMPERES')
      WRITE(NWRT,4580)  EBDA
 4580 FORMAT('     TOTAL BATTERY ENERGY =',T52,E9.4,' WATT-HOURS')
C
C*****PRINT TABLE 3:  BATTERY PERFORMANCE ANALYSIS
C
      WRITE(NWRT,5000)
 5000 FORMAT('1',T42,'POWER LOAD PROFILE AND BATTERY PERFORMANCE ',
     .       'ANALYSIS',T122,'DS-PAGE 03'///)
      WRITE(NWRT,5010)  QBRES
 5010 FORMAT(' BATTERY RESERVE FRACTION =',T59,E9.4)
      WRITE(NWRT,5020)  BRDSTD
 5020 FORMAT(' STANDARD NORMALIZED BATTERY DISCHARGE CURRENT =',
     .   T59,E9.4,' AMPERES')
      WRITE(NWRT,5030)  TBDSTD
 5030 FORMAT(' STANDARD BATTERY DISCHARGE TEMPERATURE =',T59,E9.4,
     .        ' DEG. FAHRENHEIT')
      WRITE(NWRT,5040)  VCHIO
 5040 FORMAT(' BATTERY CHARGER TURN-ON INPUT VOLTAGE =',T59,E9.4,
     .        ' VOLTS')
      WRITE(NWRT,5050)  VCHISA
 5050 FORMAT(' BATTERY CHARGER SATURATED-TO-ACTIVE INPUT VOLTAGE =',
     .   T59,E9.4,' VOLTS')
C
      WRITE(NWRT,5100)  NCYCLE
 5100 FORMAT('0TOTAL MISSION BATTERY CYCLE REQUIREMENTS =',T62,I6)
      WRITE(NWRT,5110)  DOD
 5110 FORMAT(' THEORETICAL DEPTH-OF-DISCHARGE =',T59,E9.4)
      WRITE(NWRT,5120)  ZICHMX
 5120 FORMAT(' MAXIMUM ALLOWABLE CHARGING CURRENT FOR A SINGLE ',
     .       'BATTERY =',E9.4,' AMPERES')
      WRITE(NWRT,5130)  DODA
 5130 FORMAT(' ACTUAL DEPTH-OF-DISCHARGE =',T59,E9.4)
      WRITE(NWRT,5140)  EBDA
 5140 FORMAT(' TOTAL BATTERY ENERGY =',T59,E9.4,' WATT-HOURS')
C
      WRITE(NWRT,5200)  EBDTH
 5200 FORMAT('0THEORETICAL DISCHARGE ENERGY REQUIREMENT =',T59,E9.4,
     .        ' WATT-HOURS')
      I = 1
      WRITE(NWRT,5210)  I,EESG1
 5210 FORMAT(' DISCHARGE ENERGY USING CRITERION NO. ',I1,' =',
     .   T59,E9.4,' WATT-HOURS')
      I = 2
      WRITE(NWRT,5210)  I,EESG2
      I = 3
      WRITE(NWRT,5210)  I,EESG3
      WRITE(NWRT,5220)  EESG
```

AD-A047 542 JET PROPULSION LAB PASADENA CALIF F/G 9/2
COMPUTER PROGRAM FOR DESIGN AND PERFORMANCE ANALYSIS OF NAVIGAT--ETC(U).
JUL 77 G GOLTZ, H WEINER

UNCLASSIFIED JPL-5040-27-VOL-3-CHANGE USCG-D-11-77-VOL-3 NL

2 OF 2
AD
A047 542



END
DATE
FILMED
1 -78
DDC

```
 5220 FORMAT(' SELECTED DISCHARGE ENERGY CAPACITY =',T59,E9.4,
     .       ' WATT-HOURS')
C
C*****PRINT TABLE 4:  POWER LOAD PROFILE
C
      WRITE(NWRT,6000)
 6000 FORMAT('1',T53,'POWER LOAD PROFILE ANALYSIS',T122,'DS-PAGE 04'///)
      WRITE(NWRT,6100)
 6100 FORMAT(9X,'NO. OF',8X,'******* WEEKLY DURATION OF ******',6X,
     1    'WEEKLY TOTAL  WEEKLY SOLAR      BATTERY DISCHARGE ENERGY ',
     1    'DURING'/,
     2    ' WEEK     MODE',8X,'SOLAR',7X,'SHARE-MODE     CHARGING',6X,
     2    'OF SOLAR    INSOLATION      SOLAR',7X,'SHARE-MODE',5X,
     2    'CHARGING'/,
     3    ' INDEX  REVERSALS  OCCULTATION    OPERATIONS     PERIODS',6X,
     3    'INSOLATION     MAXIMUM     OCCULTATION    OPERATIONS',5X,
     3    'PERIODS'/,
     4    14X,3(7X,'(HOURS)'),3X,2(11X,'(WT-HRS/SQ.M)'),3(2X,
     4    '(WATT-HOURS)')/,
     5    1X,128('-'))
      DO 6300 I=1,NWEEK
      WRITE(NWRT,6200)  I,NMR(I),TJT(I),TKT(I),TLT(I),QQSOLT(I),
     .                  QSOLMX(I),(EBX(I,J),J=1,3)
 6200 FORMAT(2X,I2,5X,I6,8(5X,E9.4))
 6300 CONTINUE
      RETURN
C
      END
```

DS-BATEFC

```
      SUBROUTINE  DSBEFC(BRR,ETA,QST,TTESGX)
C      COMPUTES BATTERY OPERATING EFFICIENCY
C
      INCLUDE  COMON,LIST
      DIMENSION  A(NETA,2)
      NAMELIST/BEFC/ BRR,QST,TTESGX,A,ETA/
C
      DO 10 I=2,4                          @DETERMINE TEMPERATURE INDEX
      ITP = I
   10 IF(TP(I).GT.TTESGX) GO TO 100
      ITP = 5
  100 CALL TB2SET(SOC,NSOC,2,1,BI,5,2,1,AA(1,1,ITP-1),NETA,IERR)
      DO 120 I=1,NSOC                      @BUILD EFFICIENCY TABLE AT
  120 A(I,1) = TB2GET(SOC(I),BRR)          @TEMPERATURE = TP(ITP-1)
      CALL TB2SET(SOC,NSOC,2,1,BI,5,2,1,AA(1,1,ITP),NETA,IERR)
      DO 150 I=1,NSOC                      @BUILD EFFICIENCY TABLE AT
  150 A(I,2) = TB2GET(SOC(I),BRR)          @TEMPERATURE = TP(ITP)
      CALL TB2SET(SOC,NSOC,2,1,TP(ITP-1),2,2,1,A,NETA,IERR)
      ETA = TB2GET(QST,TTESGX)             @COMPUTE BATTERY EFFICIENCY
      IF(DEBUG.NE.0.0)  WRITE(NWRT,BEFC)
      RETURN
      END
```

DS-CDSI

```
      SUBROUTINE  CDSI(SALT)
C      COMPUTES CLEAR-DAY SOLAR INSOLATION  (QDTC)
C
      INCLUDE  ALLCMN,LIST
      INCLUDE  COMON,LIST
      NAMELIST/CDSIQ/ BHOUR,COSTZS,COSTW,COSTS,SALT,SAZM,
     .          QDN,PHIAI,PHIAA,ETAA,ETAB,ETAC,COSTLT,BS,
     .          QDG,YV,QDS,QDT/
C
   20 BHOUR = DEGRAD * (15.0 * (TIMEH - 12.0 + TZN + ET) - THETOD)
   30 IF(ABS(BHOUR).GE.ABS(HOURT)) GO TO 230     @SOLAR OCCULTATION?
C
   40 COSTZS = COS(BHOUR) * COS(DECL) * COS(THETLA) +      @NO...
     .          SIN(DECL) * SIN(THETLA)      @COMPUTE DIRECTION COSINES
      COSTW = COS(DECL) * SIN(BHOUR)
      COSTS = SQRT(ABS(1.0 - COSTZS**2.0 - COSTW**2.0))
      IF(COS(BHOUR).LT.(TAN(DECL)/TAN(THETLA))) COSTS = -COSTS
   50 SALT = ASIN(COSTZS)                   @COMPUTE SOLAR ALTITUDE
   60 SAZM = ASIN(COSTW / COS(SALT))        @COMPUTE SOLAR AZIMUTH
      IF(COSTS.LT.0.0) SAZM = PI - SAZM
   90 QDN = APPSC * CN * EXP(-ATMEXC / COSTZS)
  110 PHIAI = DEGRAD * PHIAID               @SOLAR ARRAY POINTING ANGLES
      PHIAA = DEGRAD * PHIAAD
  120 ETAA = COS(PHIAI)
      ETAB = SIN(PHIAA) * SIN(PHIAI)
      ETAC = COS(PHIAA) * SIN(PHIAI)
  130 COSTLT = ETAA * COSTZS + ETAB * COSTW + ETAC * COSTS
  150 BS = SDF * QDN / CN**2.0              @COMPUTE SKY BRIGHTNESS
  160 QDG = REFLH * (BS + QDN * COSTZS) * (1.0 - ETAA) / 2.0
  200 YV = 0.45
      IF(COSTLT.GT.-0.2) YV = 0.55 + 0.437 * COSTLT +
     .   0.313 * COSTLT**2.0
      QDS = QDN * (SDF * YV + REFLH * (SDF + COSTZS) / 2.0)
  210 QDS = QDS + ABS(QDN * SDF - QDS) * COS(SALT)
  220 QDT = QDG + QDS                       @TOTAL INCIDENT SOLAR RADIATION
      IF(COSTLT.GT.0.0) QDT = QDT + QDN * COSTLT
      GO TO 500                             @RETURN
C
  230 QDT = 0.0                             @SOLAR OCCULTATION
  500 IF(DEBUG.GE.3.0)  WRITE(NWRT,CDSIQ)
      RETURN
      END
```

DS-ESGIV

```
      SUBROUTINE  DSESGC(QBX,TTESGX)
C      COMPUTES REQUIRED BATTERY I-V CHARACTERISTIC ARRAYS
C
      INCLUDE  ALLCMN,LIST
      INCLUDE  COMON,LIST
      NAMELIST/ESGIV/ QBX,TTESGX,XIB,XVB/
C
      DO 10 I=2,5                          @DETERMINE TEMPERATURE INDEX
      ITP = I
   10 IF(TBATT(I).GT.TTESGX) GO TO 100
      ITP = 6
  100 CALL TB2SET(XIBATT,NIVB,2,1,QBATT,NQB,2,1,VBATT(1,1,ITP-1),NIVB,
     .            IERR)
      DO 120 I=1,NIVB                      @BUILT VOLTAGE TABLE AT
  120 TRESLT(I,1) = TB2GET(XIBATT(I),QBX)  @ TEMPERATURE = TBATT(ITP-1)
      CALL TB2SET(XIBATT,NIVB,2,1,QBATT,NQB,2,1,VBATT(1,1,ITP),NIVB,
     .            IERR)
      DO 150 I=1,NIVB                      @BUILD VOLTAGE TABLE AT
  150 TRESLT(I,2) = TB2GET(XIBATT(I),QBX)  @TEMPERATURE = TBATT(ITP)
      CALL TB2SET(XIBATT,NIVB,2,1,TBATT(ITP-1),2,2,1,TRESLT,NESG,IERR)
      DO 180 I=1,NIVB
      XIB(I) = XIBATT(I)                   @SET CELL CURRENT ARRAY
  180 XVB(I) = TB2GET(XIBATT(I),TTESGX)    @COMPUTE CELL VOLTAGE ARRAY
      IF(DEBUG.NE.0.0)  WRITE(NWRT,ESGIV)
      RETURN
      END
```

DS-PCDGC

```
      SUBROUTINE  DSPCDG(KPCD)
C      INITIALIZES POWER CONDITIONING & DISTRIBUTION GROUP PARAMETERS
C     ENTRY  DSPCDC
C      COMPUTES POWER CONDITIONING AND DISTRIBUTION GROUP CURRENT-
C      VOLTAGE CHARACTERISTIC CURVE ARRAYS  (XI,XV)
C
      INCLUDE  ALLCMN,LIST
      INCLUDE  COMON,LIST
      NAMELIST/PCDGC/ KPCD,TTAMB,IFTYPE,TL,TLON,TLOFF,CLS,DL,
     .          ACTRL,AVGRL,VINCIV,TTPCD,VRIO,VRISA,ZRA,ZRS,XI,XV/
C
   10 IF(IFTYPE.LT.0 .OR. IFTYPE.GT.15) GO TO 800     @ILLEGAL PATTERN?
   40 TLON = 0.0                                      @NO.
      TLOFF = 0.0
      CLS = 0.0
      CALL TB2SET(CLSIT,NCLSI,2,1,CLSTT,NCLST,2,1,CLST,NCLSI,IERR)
      DO 45 J=1,15,2                                  @FOR SELECTED PATTERN, COMPUTE
      TLON = TLON + TL(J,IFTYPE+1)                    @ ILLUMINATION DURATION TOTAL,
      TLOFF = TLOFF + TL(J+1,IFTYPE+1)                @ SHUT-OFF DURATION TOTAL
   45 CLS = CLS + TL(J,IFTYPE+1) * TB2GET(CLR,TL(J,IFTYPE))
      IF(TLON.LE.0.0) GO TO 800                       @ILLEGAL FLASHER PATTERN?
      IF(TLOFF.LT.0.0) GO TO 800
   50 DL = TLON / (TLON + TLOFF)                      @NO, COMPUTE LAMP DUTY CYCLE
   60 CLS = CLS / TLON                                @COLD-FILAMENT SURGE COEFFICIENT
   80 ACTRL = VLR / (CLR * CLS)                       @COMPUTE ACTUAL LAMP RESISTANCE
  100 AVGRL = ACTRL / DL                              @COMPUTE AVERAGE LAMP RESISTANCE
  120 VINCIV = (VMAXIV - VMINIV) / (NPCDG - 1)        @SET VOLTAGE INCREMENT
C
      ENTRY  DSPCDC(KPCD)
C
  140 TTPCD = TTAMB + DTTPCD                          @SET PCDG EQUIPMENT TEMPERATURE
  150 CALL SLUP(TTPCD,VRIO,FDOT,VRIOT(1,1),VRIOT(1,2),NVRIO,1)
      CALL SLUP(TTPCD,VRISA,FDOT,VRISAT(1,1),VRISAT(1,2),NVRISA,1)
      CALL SLUP(TTPCD,ZRA,FDOT,ZRAT(1,1),ZRAT(1,2),NZRA,1)
      CALL SLUP(TTPCD,ZRS,FDOT,ZRST(1,1),ZRST(1,2),NZRS,1)
      XV(1) = VMINIV                                  @SET INITIAL VOLTAGE VALUE
      IF(VMINIV.GT.VRISA) GO TO 290                   @SELECT VOLTAGE POSITION
      IF(VMINIV.GT.VRIO) GO TO 240
C
  160 CALL VISUB1(J1,$320)                            @VMINIV.LE.VRIO
  200 CALL VISUB2(J1,J2,$320)
  220 CALL VISUB3(J2)
      GO TO 320
C
  240 CALL VISUB2(1,J2,$320)                          @VRIO.LT.VMINIV.LE.VRISA
  270 CALL VISUB3(J2)
      GO TO 320
C
  290 CALL VISUB3(1)                                  @VMINIV.GT.VRISA
      GO TO 320
C
  320 CALL TB2SET(XIHVT,NXIHV,2,1,XIHTT,NXIHT,2,1,XIHIT,NXIHI,IERR)
      DO 330 J=1,NPCDG                                @COMPUTE CHARACTERISTIC ARRAYS
      XI(J) = XI(J) + TB2GET(XV(J),TTPCD)             @PCDG CURRENT ARRAY
      IF(XI(J).LT.0.0) XI(J) = 0.0
  330 XV(J) = XV(J) + RLL * XI(J)                     @PCDG VOLTAGE ARRAY
```

```
      IF(DEBUG.NE.0.0)  WRITE(NWRT,PCDGC)
      RETURN
C
C*****ERROR MESSAGE EXITS
C
  800 WRITE(NWRT,810)
  810 FORMAT('0*** INCORRECT FLASHER PATTERN ENTRIES')
      STOP
C
C**********************  INTERNAL SUBROUTINES  ************************
C
      SUBROUTINE  VISUB1(J,*)              @XV(J).LE.VRIO
      DO 190 J=1,NPCDG
      IF(XV(J).GT.VMAXIV) RETURN 2         @MAXIMUM VOLTAGE? NO...
      IF(XV(J).GT.VRIO) GO TO 195          @SATURATION CONDITION? NO...
  170 XI(J) = 0.0                          @COMPUTE LAMP REGULATOR CURRENTS
      IF(J.EQ.NPCDG) RETURN 2              @PCDG ARRAY SIZE EXCEEDED? NO...
  190 XV(J+1) = XV(J) + VINCIV             @INCREMENT LAMP REGULATOR VOLTAGE
  195 RETURN
C
C
      SUBROUTINE  VISUB2(K,J,*)            @VRIO.LT.XV(J).LE.VRISA
      DO 210 J=K,NPCDG
      IF(XV(J).GT.VMAXIV) RETURN 3         @MAXIMUM VOLTAGE? NO...
      IF(XV(J).GT.VRISA) GO TO 215         @ACTIVE CONDITION? NO...
      XI(J) = 0.0                          @COMPUTE LAMP REGULATOR CURRENTS
      IF(KPCD.EQ.2) XI(J) = (XV(J) - VRIO) / (AVGRL + ZRS)
      IF(KPCD.EQ.3) XI(J) = (XV(J) - VRIO) / (ACTRL + ZRS)
      IF(J.EQ.NPCDG) RETURN 3              @PCDG ARRAY SIZE EXCEEDED? NO...
  210 XV(J+1) = XV(J) + VINCIV             @INCREMENT LAMP REGULATOR VOLTAGE
  215 RETURN
C
C
      SUBROUTINE  VISUB3(K)                @XV(J).GT.VRISA
      CALL TB2SET(VLBVT,NVLBV,2,1,VLBTT,NVLBT,2,1,VLBT,NVLB,IERR)
      DO 230 J=K,NPCDG
      IF(XV(J).GT.VMAXIV) GO TO 235        @MAXIMUM VOLTAGE? NO...
      VLB = TB2GET(XV(J),TTPCD)
      XI(J) = 0.0                          @COMPUTE LAMP REGULATOR CURRENTS
      IF(KPCD.EQ.2) XI(J) = VLB / (AVGRL + ZRA)
      IF(KPCD.EQ.3) XI(J) = VLB / (ACTRL + ZRA)
      IF(J.EQ.NPCDG) GO TO 235             @PCDG ARRAY SIZE EXCEEDED? NO...
  230 XV(J+1) = XV(J) + VINCIV             @INCREMENT LAMP REGULATOR VOLTAGE
  235 RETURN
C
      END
```

DS-SAGC

```
      SUBROUTINE  SAEC(MAXI,MAXV)
C      COMPUTES SOLAR ARRAY CURRENT-VOLTAGE
C      CHARACTERISTIC CURVE ARRAYS  (XI2,V2)
C
      INCLUDE  ALLCMN,LIST
      INCLUDE  COMON,LIST
      REAL  MAXI, MAXV
      DEFINE  TCNVRT(T) = 5.0 * (T + 459.67) / 9.0 - 273.15
      NAMELIST/SAIV/ TSAC,CDEG,VDEG,X,XX,ALPHA,BETA,RCELL,RHO,
     .          XISC,DISC,C3,C4,MFINAL,DMPPV,DXV,MAXI,MAXV,SAPWR,
     .          MSAPWR,NP,NS,NESP,XIISC,VVOC,NSV,NXX,SI,SV,XI2,V2/
C
   30 CALL SLUP(DATEM,CDEG,FDOT,SADEGC(1,1),SADEGC(1,2),NCDEG,1)
   40 CDEG = 1.0 - 1.0E-6 * (100.0 - CDEGA) * (100.0 - CDEGB) *
     .       (100.0 - CDEG)              @SET CURRENT DEGRADATION FACTOR
   60 CALL SLUP(DATEM,VDEG,FDOT,SADEGV(1,1),SADEGV(1,2),NVDEG,1)
   70 VDEG = 1.0 - 1.0E-4 * (100.0 - VDEGA) *
     .       (100.0 - VDEG)              @SET VOLTAGE DEGRADATION FACTOR
  100 X = SPECOR * QDT / 10.0            @EFFECTIVE SOLAR INSOLATION
  110 XX = X * (1.0 - CDEG)              @MODIFIED SOLAR INSOLATION
  130 TSAC = TCNVRT(TSAF)                @CONVERT TEMP. TO CENTIGRADE
  140 ALPHA = ACELL * XX * (7.428E-7 - 1.83E-9 * TSAC) / ACSTD
  150 CALL SLUP(TSAC,RCELL,FDOT,TEMTAB,RSCELL,NRSCEL,3)
  160 CALL SLUP(XX,RHO,FDOT,SUNLIT,ROE,NROE,3)
  170 CALL TB2SET(BTEMP,NBTEMP,3,1,SUNMW,NSUNMW,3,1,BETAB,NBETAB,IERR)
      BETA = TB2GET(TSAC,XX) / 1000.0
  180 CALL SLUP(0.0,XIISC,FDOT,VV,XII,30,1) @FIND SHORT CIRCUIT CURRENT
      CALL SLUP(0.0,VVOC,FDOT,XII,VV,30,1)  @FIND OPEN CIRCUIT VOLTAGE
  210 ALPHA = NP * ALPHA                 @SHORT CIRCUIT CURRENT FACTOR
      BETA = NS * BETA                   @OPEN CIRCUIT VOLTAGE FACTOR
      RCELL = NS * (0.114 + RCELL) / NP      @SERIES RESISTANCE
      RHO = NS * RHO / NP                @TEMPERATURE CORRECTION FACTOR
  220 XISC = NP * XIISC * (1.0 - CDEG)       @SHORT CIRCUIT CURRENT
  230 DISC = ALPHA * (TSAC - TCSTD) - XISC * (1.0 - X / XCSTD)
  240 C3 = BETA * (TSAC - TCSTD) + DISC * RCELL
      C4 = RHO * (TSAC - TCSTD)
C
      DO 250 I=1,NPSG                    @ZERO-FILL PSG I-V ARRAYS
      SI(I) = 0.0
      SV(I) = 0.0
      V2(I) = 0.0
  250 XI2(I) = 0.0
      DO 260 J=1,30                      @SET REFERENCE I-V ARRAYS
      XI2(J) = NP * (XII(J) - CDEG * XIISC) + DISC
  260 SV(J) = NS * (VV(J) - VDEG * VVOC) - C3 - C4 * XI2(J)
      DO 270 I=30,2,-1                   @CHECK 'SV' MONOTANICITY
      IF(SV(I)-SV(I-1).LT.0.0) GO TO 270
      NSV = I
      GO TO 280
  270 CONTINUE
      NSV = 1
  280 NXX = 31 - NSV
      V2(1) = 0.0
      J = NPSG - 1
      DO 285 L=1,J                       @REDEFINE PSG I-V ARRAYS
      CALL SLUP(V2(L),SI(L),FDOT,SV(NSV),XI2(NSV),NXX,1)
```

```
      IF(SI(L).GT.0.0) GO TO 285
      SI(L) = 0.0                          @SET FINAL I-V POINT
      CALL SLUP(0.0,V2(L),FDOT,XI2(NSV),SV(NSV),NXX,1)
      MFINAL = L + 1
      SI(MFINAL) = 0.0                     @SET FINAL I-V POINT
      V2(MFINAL) = 2.0 * AMAX1(VBUS,V2(L))
      GO TO 300
  285 V2(L+1) = V2(L) + VSAINC
      GO TO 800                            @PSGC RANGE EXCEEDED
  300 DO 305 L=1,MFINAL
  305 XI2(L) = NESP * SI(L)
C
  320 MSAPWR = 0.0                         @INITIALIZE MAXIMUM POWER
      IF(XI2(1).LE.0.0) GO TO 840
      MAXV = V2(MFINAL-1) / 3.0            @INITIALIZE MAX-POWER VOLTAGE
      DMPPV = (V2(MFINAL-1) - MAXV) / 50.0  @SET VOLTAGE INCREMENT
      DXV = DMPPV
      DO 360 L=1,60
  330 CALL SLUP(MAXV,MAXI,FDOT,V2,XI2,MFINAL-1,1)
      SAPWR = MAXI * MAXV                  @COMPUTE SOLAR ARRAY (PSG) POWER
  340 IF(SAPWR.LE.MSAPWR) GO TO 350        @PSG POWER.LE.MAXIMUM POWER?
      MSAPWR = SAPWR                       @NO. REDEFINE MAXIMUM POWER
      MAXV = MAXV + DXV                    @INCREMENT PSG VOLTAGE
      GO TO 360
  350 IF(DXV.LT.DMPPV) GO TO 370           @MAX-POWER-POINT LOCATED?
      MSAPWR = 0.0                         @NO. RE-INITIALIZE MAXIMUM POWER
      MAXV = MAXV - DXV                    @ AND MAX-POWER VOLTAGE
      DXV = DXV / 10.0                     @REDEFINE VOLTAGE INCREMENT
  360 CONTINUE
      GO TO 820                            @MAX-POWER-POINT NOT FOUND
  370 MAXV = MAXV - DXV                    @YES. SET MAX-POWER VOLTAGE
      IF(MAXV.LE.0.0) MAXV = 0.00001
      MAXI = MSAPWR / MAXV                 @SET MAX-POWER CURRENT
C
      IF(DEBUG.NE.0.0)  WRITE(NWRT,SAIV)
      RETURN
C
C*****ERROR MESSAGE EXITS
C
  800 WRITE(NWRT,810)
  810 FORMAT('0*** SOLAR ARRAY I-V DIMENSIONS EXCEEDED')
      WRITE(NWRT,SAIV)
      STOP
  820 WRITE(NWRT,830)
  830 FORMAT('0*** MAX-POWER-POINT CALCULATION FAILED TO CONVERGE')
      STOP
  840 WRITE(NWRT,850)
  850 FORMAT('0***ILLEGAL SOLAR ARRAY I-V CURVE')
      WRITE(NWRT,SAIV)
      STOP
C
      END
```

DS-TERMC

```
      SUBROUTINE  TERMC
C      COMPUTES TERMINATOR CHARACTERISTICS
C
      INCLUDE  ALLCMN,LIST
      INCLUDE  COMON,LIST
      DIMENSION  VAR(5)
      NAMELIST/TERM/ ALPHEQ,VAR,DECL,ET,APPSC,ATMEXC,SDF,
     .          THETLA,HOURT,SRT,SST/
C
   20 ALPHEQ = OMEGA * DATE                  @SOLAR VECTOR LOCATION
      COS1AQ = COS(ALPHEQ)
      COS2AQ = COS(2.0 * ALPHEQ)
      COS3AQ = COS(3.0 * ALPHEQ)
      SIN1AQ = SIN(ALPHEQ)
      SIN2AQ = SIN(2.0 * ALPHEQ)
      SIN3AQ = SIN(3.0 * ALPHEQ)
      J = 5
      IF(ITAPE.NE.0) J = 2
      DO 30 I=1,J                            @COMPUTE SOLAR RADIATION VARIABLS
   30 VAR(I) = FA(1,I) + FA(2,I) * COS1AQ + FA(3,I) * COS2AQ +
     .          FA(4,I) * COS3AQ + FA(5,I) * SIN1AQ +
     .          FA(6,I) * SIN2AQ + FA(7,I) * SIN3AQ
      DECL = DEGRAD * VAR(1)                 @SOLAR DECLINATION ANGLE
      ET = VAR(2)                            @EQUATION OF TIME DIFFERENCE
      IF(ITAPE.NE.0) GO TO 50
      APPSC = 3.1524808 * VAR(3)             @APPARENT SOLAR CONSTANT
      ATMEXC = VAR(4)                        @ATMOSPHERE EXTINCTION FACTOR
      SDF = VAR(5)                           @SKY DIFFUSE FACTOR
   50 THETLA = DEGRAD * THELAD               @BUOY LATITUDE
   60 HOURT = PI                             @COMPUTE TERMINATOR HOUR ANGLE
      IF(THETLA.LT.(0.5 * PI - DECL))
     .   HOURT = ACOS(-1.0 * TAN(THETLA) * TAN(DECL))
  100 SRT = 12.0 * (1.0 - HOURT / PI) - ET - TZN + THELOD / 15.0
      SST = 24.0 - SRT                       @COMPUTE SUNRISE & SUNSET TIMES
C
      IF(DEBUG.NE.0.0)  WRITE(NWRT,TERM)
      RETURN
      END
```

PADRVR

```
      SUBROUTINE  PADRVR
C      MAIN DRIVER PROGRAM FOR PERFORMANCE ANALYSIS
C
      INCLUDE  ALLCMN,LIST
      INCLUDE  COMON,LIST
      INCLUDE  PACMN,LIST
      INCLUDE  SUMCMN,LIST
      NAMELIST/PAOUT1/ ISIZE,NCTYPE,VBUS,YEAR,DATE,TIMEH,DATEH,
     .         ACCQB,CT,HLLA,TC,INDFLS,NTS,DURA,DURAH,DURAM,
     .         DAYSST/
      NAMELIST/PAOUT2/ACTRL,APPSC,ATMEXC,AVGRL,DECL,DL,ET,HOURI,
     .         HFINAL,MSAPWR,NAPSG,ND,QDT,SDF,SRT,SST,TCZ,THETLA,
     .         TSAC,TSAF,TTAMB,TTESG,TTPCD,VBUS,VINCIV,VVOC,XIISC,
     .         ZRFI,ZRFV,XI,XV,XI2,V2,ZI,ZV,SI,SV,XIB,XVB,DIFIV,
     .         TRESLT,TRESI,TRESV/
      NAMELIST/PAOUT3/ BCUR,VBAT,XIPCD,XIPSG,XIZ,XISA,XIEC,
     .         VDIODE,VSA,PBATT,PESG,PPCD,PPSG,PSL,PSA,MARSA,
     .         CHRN,ETA,QB,DQB,H,SPGR,TBFRZ,LNIS/
C
   10 ISIZE = 3 + 8 + 11 + 11          @SET PA PRINT RECORD SIZE (WORDS)
      NPLT = NPLT + 1                  @SET MAXIMUM NUMBER OF I-V PLOTS
      MXPLTS = 1                       @INITIALIZE I-V PLOTS COUNTER
   20 NCTYPE = 0                       @SET INITIAL RUN FLAG
  440 H = 0.0                          @INITIALIZE INTEGRATION INTERVAL
   30 READ(NRD,35,ERR=800,END=900)     @READ START-UP DATA
     .    YEAR,DATE,TIMEH,DATEH,ACCQB,CT,HLLA,TC,INDFLS
   35 FORMAT( )
      IF(ACCQB.LE.0.0) ACCQB = 0.01
      IF(HLLA.LE.0.0) HLLA = 1.0
   40 TIMEH = TIMEH + DATEH / 60.0     @CONVERT START TIME TO HOURS
      DATEH = 0.0                      @INITIALIZE ELAPSED TIME
      DAYSST = 0.0
   50 YEAR1 = YEAR                     @SET START TIME REFERENCE DATA
      DATE1 = DATE + TIMEH / 24.0
      DATEH1 = DATEH
      IF(DEBUG.GT.0.0 .AND. DATE1.GE.DEBUG)  WRITE(NWRT,PAOUT1)
      WRITE(NWRT,55)
   55 FORMAT('1 YEAR: DAY: HOUR    DAYSST',6X,'VBUS',4X,'XIPSG',
     .       4X,'XITT',3X,'XIPCD',3X,'QB'//)
      CALL ZENER                       @INITIALIZE ZENER DIODE PARAMETERS
      CALL PASUB($60)                  @INITIAL PERFORMANCE ANALYSIS
      CALL PRTPLT                      @STORE PERFORMANCE ANALYSIS DATA
      IF(XPLT(1).LE.0.0) GO TO 60      @'INSTANTANEOUS' I-V PLOTS?
      IF(DATE1.GE.XPLT(1)) CALL CRVPLT
      MXPLTS = MXPLTS + 1              @INCREMENT I-V PLOTS COUNTER
C
   60 NCTYPE = 1                       @SET TIME-VARIANT RUN FLAG
      READ(NRD,65,ERR=820,END=700)  NTS,DURA,DURAH,DURAM,CT,TC
   65 FORMAT( )                        @READ TIME-VARIANT DATA
      IF(NTS.LE.1) NTS = 1
   70 DURAH = 24.0 * DURA + DURAH + DURAM / 60.0      @SET RUN DURATION
      HINT = DURAH / NTS               @SET PRINTOUT TIME INTERVAL
      DO 660 LNTS=1,NTS                @PERFORM DURATION LOOP
   75 H2 = 0.0                         @SET REFERENCE PRINTOUT TIME
   80 H = AMIN1(HINT-H2,HLLA)          @SET INTEGRATION TIME INTERVAL
      DO 620 LNIS=1,20                 @PERFORM INTEGRATION LOOP
```

```
   90 DATEM = DATEM1 + H / 24.0          @COMPUTE 'TIME' OF CALCULATION
      DATE2 = DATE1 + H / 24.0
      DATE = AINT(DATE2)
      TIMEH = 24.0 * (DATE2 - DATE)
      IF(TIMEH.GE.24.0) TIMEH = TIMEH - 24.0
      IF(DATE.LT.366.0) GO TO 95         @NEW YEAR?
      DATE2 = DATE2 - 365.0              @YES. REVISE DAY
      DATE = AINT(DATE2)
      YEAR = YEAR1 + 1.0                 @INCREMENT YEAR
   95 KLL = 3                            @INITIALIZE LOAD SELECTOR
      H3 = H                             @SAVE TIME INCREMENT
      IF(DEBUG.GT.0.0 .AND. DATE2.GE.DEBUG)  WRITE(NWRT,PAOUT1)
      CALL PASUB($630)                   @COMPUTE I-V ARRAYS & NEW SOC
  620 IF(QB.LE.0.0) GO TO 860            @BATTERIES DISCHARGED? NO...
  630 DATEM1 = DATEM                     @REDEFINE TIME REFERENCE DATA
      DATE1 = DATE2
      DATE = AINT(DATE1)
      TIMEH = 24.0 * (DATE1 - DATE)
      YEAR1 = YEAR
  640 H2 = H2 + H3                       @INCREMENT PRINTOUT REFERENCE
  650 IF(H2.LT.HINT) GO TO 80            @PRINTOUT REQUIRED? YES...
      DAYSST = DAYSST + H2 / 24.0        @COMPUTE DAYS SINCE START OF TEST
      IF(MXPLTS.GE.NPLT) GO TO 660       @MAXIMUM NO. OF I-V PLOTS? NO...
      IF(XPLT(MXPLTS).LE.0.0) GO TO 660      @'INSTANTANEOUS' I-V PLOTS
      IF(DATE1.LT.XPLT(MXPLTS)) GO TO 660    @ REQUIRED AT CURRENT TIME?
      CALL CRVPLT                        @YES. PRODUCE I-V PLOTS
      MXPLTS = MXPLTS + 1                @INCREMENT I-V PLOTS COUNTER
  660 CALL PRTPLT                        @STORE PERFORMANCE ANALYSIS DATA
      GO TO 60                           @GET NEXT TIME-VARIANT INPUT
  700 RETURN                             @TERMINATE PERFORMANCE ANALYSIS
C
C*****ERROR MESSAGE EXITS
C
  800 WRITE(NWRT,810)
  810 FORMAT('0***ERROR IN START-UP DATA')
      STOP
  820 WRITE(NWRT,830)
  830 FORMAT('0***ERROR IN TIME-VARIANT DATA')
      STOP
  860 WRITE(NWRT,870) QB
  870 FORMAT('0***BATTERIES COMPLETELY DISCHARGED: QB= ',F6.3)
      WRITE(NWRT,880) YEAR,DATE,TIMEH,DAYSST
  880 FORMAT(4X,'TIME = ',F5.0,F4.0,F5.2,2X,'(',F8.3,' DAYS ',
     .       'SINCE START OF TEST)'/,' ***EXECUTION TERMINATED')
      CALL PRTPLT                        @PRODUCE FINAL SUMMARY ENTRY
      CALL CRVPLT                        @PRODUCE I-V PLOTS
      RETURN                             @END PERFORMANCE ANALYSIS
  900 WRITE(NWRT,910)
  910 FORMAT('0***EOF ENCOUNTERED AT START-UP DATA')
      STOP
C
C**********************  INTERNAL SUBROUTINES  **********************
C
      SUBROUTINE PASUB(*)
C      CONTROLS COMPUTATION OF EQUIPMENT I-V CHARACTERISTICS,
C      DIFFERENCE CURVE, CURVE INTERSECTION, OPERATING POINT
C      PARAMETERS, AND NEW STATE-OF-CHARGE FOR EACH BATTERY
```

```
C
  100 CALL PSGC(CT,TC)                    @COMPUTE PSG I-V CHARACTERISTICS
C
  110 IF(QDT.GT.0.0) GO TO 112            @DETERMINE LAMP FLASHER
      INDFLS = 1                          @ CONDITION INDICATOR AND
      KL = 2                              @ LOAD SELECTOR INDICATOR...
      GO TO 120
  112 IF(QDT.GT.QON) GO TO 114
      INDFLS = 1                          @LAMP ON...
      KL = 3
      GO TO 118
  114 IF(QDT.GE.QOFF) GO TO 116
      KL = 1                              @LAMP FLASHING...
      IF(INDFLS.EQ.0) GO TO 120
      KL = 3
      GO TO 118
  116 INDFLS = 0
      KL = 1                              @LAMP OFF...
      GO TO 120
  118 IF(NCTYPE.EQ.0) KL = 2
  120 KPCD = KL                           @SELECT PCDG I-V LOAD
  130 IF(NCTYPE.EQ.0) CALL PCDG(KPCD)     @COMPUTE PCDG I-V ARRAYS
      IF(NCTYPE.NE.0) CALL PCDGC(KPCD)
C
  140 DO 142 I=1,NPCDG                    @SET DIFFERENCE CURVE VOLTAGES
  142 DIFIV(I,1) = XV(I)
  143 DO 145 I=1,NAPSG
      J = NPCDG + I
  145 DIFIV(J,1) = SV(I)
      CALL SORT(DIFIV,NPCDG+NAPSG,ND)     @SORT DIFFERENCE CURVE VOLTAGES
      DO 155 I=1,ND                       @FIND DIFFERENCE CURVE CURRENTS
      DIFIV(I,2) = 0.0
      IF(DIFIV(I,1).GE.SV(NAPSG-1)) GO TO 150
      CALL SLUP(DIFIV(I,1),DIFIV(I,2),FDOT,SV,SI,NAPSG-1,1)
  150 CALL SLUP(DIFIV(I,1),DXI,FDOT,XV,XI,NPCDG,1)
  155 DIFIV(I,2) = DIFIV(I,2) - DXI
C
      VPSGMX = SV(NAPSG-1)                @SET MAXIMUM SOLAR ARRAY VOLTAGE
      IF(QDT.LE.0.0) VPSGMX = 0.0
  160 CALL ESGC(VPSGMX)                   @COMPUTE ESG I-V CHARACTERISTICS
C
      IF(DEBUG.GT.0.0 .AND. DATE2.GE.DEBUG)  WRITE(NWRT,PAOUT2)
  180 CALL INTER                          @COMPUTE OPERATING POINT VOLTAGE
  210 CALL SLUP(VBUS,XITT,FDOT,DIFIV(1,1),DIFIV(1,2),ND,1)
C                                         @COMPUTE OPERATING POINT I,V...
  230 CALL SLUP(VBUS,BCUR,FDOT,TRESV,TRESI,NESG,1)
  250 CALL SLUP(BCUR,VBAT,FDOT,XIB,XVB,NIVB,1)   @ENERGY STORAGE GROUP
  260 CALL SLUP(VBUS,XIPCD,FDOT,XV,XI,NPCDG,1)   @PWR COND & DIST GROUP
      XIPSG = 0.0
      IF(QDT.LE.0.0) GO TO 290
      IF(VBUS.GE.SV(NAPSG-1)) GO TO 290
  270 CALL SLUP(VBUS,XIPSG,FDOT,SV,SI,NAPSG-1,1) @POWER SOURCE GROUP
  290 CALL SLUP(VBUS,XIZ,FDOT,ZV,ZI,NSL,1)       @SHUNT LIMITER
  300 XISA = XIZ + XIPSG                         @SOLAR ARRAY
  310 XIEC = XISA / NESP
  320 CALL SLUP(XIEC,VDIODE,FDOT,ADI(1,2),ADI(1,1),NADI,1)
      VSA = 0.0
```

```
      IF(QDT.LE.0.0) GO TO 330
      VSA = VBUS + VDIODE + RSA * XIEC
C                                        @COMPUTE OPERATING POINT POWER...
  330 PBATT = ABS(BCUR * VBAT)           @BATTERY
      PESG = ABS(VBUS * XITT)            @ENERGY STORAGE GROUP
      PPCD = VBUS * XIPCD                @POWER COND & DIST GROUP
      PPSG = VBUS * XIPSG                @POWER SOURCE GROUP
      PSL = VBUS * XIZ                   @SHUNT LIMITER
      PSA = VSA * XISA                   @SOLAR ARRAY
  340 MARSA = MSAPWR - PSA               @COMPUTE EQUIPMENT POWER MARGIN
C
  360 IF(NCTYPE.EQ.0) ETA = 1.0          @COMPUTE BATTERY EFFICIENCY
  380 IF(BCUR.GE.0.0) GO TO 390
      ETA = 1.0
      GO TO 420
  390 CHRN = BCUR / CB                   @COMPUTE NORMALIZED CHARGE RATE
  400 CALL BATEFC(CHRN,ETA,QB,TTESG)     @COMPUTE EFFICIENCY
  410 IF(ETA.LE.0.0) ETA = 0.00001       @ELIMINATE NEGATIVE VALUES
  420 DQB = BCUR * ETA / CB              @SET RATE OF CHANGE OF SOC
C
  440 IF(NCTYPE.EQ.0) GO TO 490          @INITIAL RUN?
  450 IF(KL.NE.3) GO TO 490              @NO, NEW INTEGRATION INTERVAL IF
  460 H = DL * H3                        @ LAMP FLASHING (0<QDT<QOFF)
  470 IF(KLL.EQ.1) H = H3 - H
  490 DQB = H * DQB                      @SET STATE-OF-CHARGE INCREMENT
  510 IF(NCTYPE.EQ.0 .OR. KLL.NE.3) GO TO 600
      IF(DQB.EQ.0.0) DQB = 0.00001
  570 XQB = ABS(DQB)
  590 IF(H*ACCQB/XQB.GE.HINT-H2-0.01) GO TO 600  @MAXIMUM TIME STEP?
      IF(XQB.LT.0.7*ACCQB .OR.           @NO, IS SOC INCREMENT WITHIN
     .   XQB.GT.ACCQB) GO TO 625         @ SPECIFIED ACCURACY LIMITS?
  600 QB = QB + DQB                      @YES, COMPUTE NEW STATE-OF-CHARGE
      CALL SLUP(QB,SPGR,FDOT,SPGRI(1,1),SPGRI(1,2),NSPGR,1)
  610 CALL SLUP(SPGR,TBFRZ,FDOT,TBFRZI(1,1),TBFRZI(1,2),NTBFRZ,1)
      IF(DEBUG.GT.0.0 .AND. DATE2.GE.DEBUG)  WRITE(NWRT,PAOUT2)
      IF(DEBUG.GT.0.0 .AND. DATE2.GE.DEBUG)  WRITE(NWRT,PAOUT3)
      IF(NCTYPE.EQ.0) RETURN
  620 IF(KL.NE.3) RETURN 1
      IF(KLL.NE.3) RETURN 1
      KLL = 1                            @REDEFINE LOAD SELECTOR
      KPCD = 1
      GO TO 130                          @COMPUTE NEW CHARACTERISTICS
C
  625 H = H * ACCQB / XQB                @NO, REDEFINE INTERVAL
      IF(LNIS.LT.20) RETURN              @MAXIMUM NUMBER OF ITERATIONS?
      IF(XQB.LE.ACCQB) GO TO 600         @YES, ACCEPTABLE ACCURACY?
      LYEAR = IFIX(YEAR)                 @NO, CONVERT 'TIME' TO INTEGER
      LDAY = IFIX(DATE)
      WRITE(NWRT,627) LYEAR,LDAY,TIMEH,ACCQB,DQB,H
  627 FORMAT('0***COMPUTED STATE-OF-CHARGE INCREMENT IS OUTSIDE THE ',
     .       'SPECIFIED ACCURACY LIMITS:'/4X,'TIME = ',I4,':',I3,
     .       ':',F5.2,4X,'ACCQB = ',F7.5,4X,'DQB = ',F7.5,4X,
     .       'H = ',F10.8)             @PRINT DIAGNOSTIC MESSAGE
      GO TO 600                          @USE COMPUTED STATE-OF-CHARGE
C
      END
```

PA-BATEFC

```
      SUBROUTINE  BATEFC(CHRN,ETA,QST,TTESG)
C      COMPUTES BATTERY OPERATING EFFICIENCY .
C
      INCLUDE  COMON,LIST
      DIMENSION  A(NETA,2)
C
      DO 10 I=2,4                          @DETERMINE TEMPERATURE INDEX
      ITP = I
   10 IF(TP(I).GT.TTESG) GO TO 100
      ITP = 5
  100 CALL TB2SET(SOC,NSOC,2,1,BI,5,2,1,AA(1,1,ITP-1),NETA,IERR)
      DO 120 I=1,NSOC                      @BUILD EFFICIENCY TABLE AT
  120 A(I,1) = TB2GET(SOC(I),CHRN)         @TEMPERATURE = TP(ITP-1)
      CALL TB2SET(SOC,NSOC,2,1,BI,5,2,1,AA(1,1,ITP),NETA,IERR)
      DO 150 I=1,NSOC                      @BUILD EFFICIENCY TABLE AT
  150 A(I,1) = TB2GET(SOC(I),CHRN)         @TEMPERATURE = TP(ITP)
      CALL TB2SET(SOC,NSOC,2,1,TP(ITP-1),2,2,1,A,NETA,IERR)
      ETA = TB2GET(QST,TTESG)              @COMPUTE BATTERY EFFICIENCY
      RETURN
      END
```

```
      TRESLT(NTRES,2) = XX
   60 NTRES = NTRES - NTP + 1
C
      DO 90 I=3,NSL                        @MODIFY SHUNT LIMITER I-V ARRAY
   90 IF(ZI(I).GE.SI(1)) ZI(I) = SI(1)
C
  100 CALL QLINE(XV,XI,NPCDG,1,2,4,XASC,YASC)     @PLOT I-V CURVES
      CALL QLINE(V2,XI2,MFINAL,1,4,5,XASC,YASC)
      CALL QLINE(ZV,ZI,NSL,1,2,10,XASC,YASC)
      CALL QLINE(SV,SI,NAPSG,1,4,14,XASC,YASC)
      CALL QLINE(DIFIV(1,1),DIFIV(1,2),ND,1,4,24,XASC,YASC)
      CALL QLINE(TRESLT(NTP,1),TRESLT(NTP,2),NTRES,1,1,27,XASC,YASC)
      CALL PLOT(13.0,0.0,4)                @ESTABLISH NEW PLOT ORIGIN
C
      RETURN
      END
```

PA-CRVPLT

```
      SUBROUTINE  CRVPLT
C      PLOTS 'INSTANTANEOUS' EQUIPMENT I-V CHARACTERISTICS
C
      INCLUDE  ALLCMN,LIST
      INCLUDE  PACMN,LIST
      REAL  MNMX(4), OPR(2), XASC(2), YASC(2), ZERO(2)
C
      MNMX(1) = 0.0                          @DETERMINE PLOT BOUNDARIES
      MNMX(2) = 1.1 * AMAX1(V2(MFINAL),XV(NPCDG))
      MNMX(4) = 1.1 * AMAX1(XI2(1),XI(NPCDG))
      MNMX(3) = -2.0 * MNMX(4)
      CALL QSCALE(MNMX(1),10.0,2,1,XASC)
      CALL QSCALE(MNMX(3),9.0,2,1,YASC)
      CALL AXIS(0.0,0.0,'VOLTAGE',-7,10.0,0.0,XASC(1),XASC(2))
      CALL AXIS(0.0,0.0,'CURRENT',7,9.0,90.0,YASC(1),YASC(2))
      ZERO(1) = 0.0                          @DRAW ZERO-CURRENT LINE
      ZERO(2) = 0.0
      CALL QLINE(MNMX,ZERO,2,1,0,3,XASC,YASC)
      ZERO(1) = XITT                         @DRAW OPERATING CURRENT LINE
      ZERO(2) = XITT
      OPR(1) = 0.0
      OPR(2) = VBUS
      CALL QLINE(OPR,ZERO,2,1,0,3,XASC,YASC)
      ZERO(1) = VBUS                         @DRAW OPERATING VOLTAGE LINE
      ZERO(2) = VBUS
      OPR(1) = YASC(1)
      OPR(2) = XITT
      CALL QLINE(ZERO,OPR,2,1,0,3,XASC,YASC)
C
      CALL SYMBOL(1.0,10.0,0.21,'INSTANTANEOUS I-V PLOTS FOR:',0.0,28)
      CALL SYMBOL(1.5,9.5,0.14,'YEAR= ',0.0,6)
      CALL NUMBER(999.,999.,0.14,YEAR,0.0,0)
      CALL SYMBOL(999.,999.,0.14,'   DAYS ',0.0,8)
      CALL NUMBER(999.,999.,0.14,DATE,0.0,0)
      CALL SYMBOL(999.,999.,0.14,'   TIME= ',0.0,9)
      CALL NUMBER(999.,999.,0.14,TIMEH,0.0,3)
C
      DO 10 I=1,NESG                         @DETERMINE PORTION OF ENERGY
      NTP = I                                @ STORAGE GROUP CURVE TO BE
   10 IF(TRESLT(I,2).GT.MNMX(3)) GO TO 20    @ PLOTTED
   20 IF(NTP.LE.1) GO TO 30                  @TRUNCATION NOT REQUIRED
      NTP = NTP - 1                          @TRUNCATE TO MINIMUM CURRENT
      CALL SLUP(MNMX(3),XX,FDOT,TRESLT(1,2),TRESLT(1,1),NESG,1)
      TRESLT(NTP,1) = XX
      TRESLT(NTP,2) = MNMX(3)
   30 DO 40 I=NTP,NESG
      NTRES = I
      IF(TRESLT(I,1).GT.MNMX(2)) GO TO 55    @MAX. VOLTAGE EXCEEDED...
   40 IF(TRESLT(I,2).GT.MNMX(4)) GO TO 50    @MAX. CURRENT EXCEEDED...
      GO TO 60                               @TRUNCATION NOT REQUIRED...
   50 CALL SLUP(MNMX(4),XX,FDOT,TRESLT(1,2),TRESLT(1,1),NESG,1)
      TRESLT(NTRES,1) = XX
      TRESLT(NTRES,2) = MNMX(4)
      GO TO 60
   55 CALL SLUP(MNMX(2),XX,FDOT,TRESLT(1,1),TRESLT(1,2),NESG,1)
      TRESLT(NTRES,1) = MNMX(2)
```

```
      TRESLT(NTRES,2) = XX
   60 NTRES = NTRES - NTP + 1
C
      DO 90 I=3,NSL                          @MODIFY SHUNT LIMITER I-V ARRAY
   90 IF(ZI(I).GE.SI(I)) ZI(I) = SI(I)
C
  100 CALL GLINE(XV,XI,NPCDG,1,2,4,XASC,YASC)      @PLOT I-V CURVES
      CALL GLINE(V2,XI2,NFINAL,1,4,5,XASC,YASC)
      CALL GLINE(ZV,ZI,NSL,1,2,10,XASC,YASC)
      CALL GLINE(SV,SI,NAPSG,1,4,14,XASC,YASC)
      CALL GLINE(DIFIV(1,1),DIFIV(1,2),ND,1,4,24,XASC,YASC)
      CALL GLINE(TRESLT(NTP,1),TRESLT(NTP,2),NTRES,1,1,27,XASC,YASC)
      CALL PLOT(13.0,0.0,4)                  @ESTABLISH NEW PLOT ORIGIN
C
      RETURN
      END
```

BEST AVAILABLE COPY

PA-ESGC

```
      SUBROUTINE  ESGC(VPSGMX)
C      COMPUTES ENERGY STORAGE GROUP (BATTERY) UNIT AND
C      GROUP CURRENT-VOLTAGE ARRAYS (TRESI,TRESV,TRESLT)
C
      INCLUDE  ALLCMN,LIST
      INCLUDE  COMON,LIST
      INCLUDE  PACMN,LIST
      REAL  VBM(NESG), XIBM(NESG)
C
   40 TTESG = TTAMB + DTTESG              @COMPUTE ESG TEMPERATURE
   60 CALL ESGIV                          @GET BATTERY I-V ARRAYS
   70 DO 75 I=1,NIVB                      @COMPUTE MODIFIED ESG I-V ARRAYS
      VBM(I) = XVB(I) + RL * XIB(I)
      IF(ICHRT.EQ.0) GO TO 75             @CHARGER PRESENT? YES...
      IF(XIB(I).GE.0.0) GO TO 75          @POSITIVE CURRENT? NO...
      CALL SLUP(ABS(XIB(I)),VDROP,FDOT,AD2(1,2),AD2(1,1),NAD2,1)
      VBM(I) = VBM(I) - VDROP
   75 XIBM(I) = XIB(I)
      IF(ICHRT.NE.0) GO TO 170            @SELECT ESG CHARGER TYPE
C
C.....NO CHARGER PRESENT
C
   80 CALL ESGSUB                         @COMPUTE ESG I-V ARRAYS
      GO TO 500                           @RETURN
C
C.....CONSTANT VOLTAGE CHARGER WITH CURRENT LIMIT
C
  170 CALL SLUP(XICHMX,VICHMX,FDOT,XIBM,VBM,NIVB,1)
      DO 175 I=1,NIVB                     @POSITION MAX-CURRENT I-V POINT
      K = I                               @ IN PROPER SEQUENCE IN ARRAYS
  175 IF(XIBM(I).GT.XICHMX) GO TO 180
      K = NESG
      GO TO 190
  180 DO 185 I=NIVB,K,-1
      XIBM(I+1) = XIBM(I)
  185 VBM(I+1) = VBM(I)
  190 XIBM(K) = XICHMX
      VBM(K) = VICHMX
      IF(XIBM(K-1).LT.XICHMX) GO TO 200
      XIBM(K-1) = XIBM(K-2) + (XICHMX - XIBM(K-2)) / 2.0
      VBM(K-1) = VBM(K-2) + (VICHMX - VBM(K-2)) / 2.0
C
  200 CALL SLUP(TTESG,VCHISA,FDOT,VCHIST(1,1),VCHIST(1,2),NVCHIS,1)
      CALL SLUP(TTESG,VCHIO,FDOT,VCHIOT(1,1),VCHIOT(1,2),NVCHIO,1)
  210 CALL SLUP(TTESG,ZCHRA,FDOT,ZCHRAT(1,1),ZCHRAT(1,2),NZCHRA,1)
      CALL SLUP(TTESG,ZCHRS,FDOT,ZCHRST(1,1),ZCHRST(1,2),NZCHRS,1)
      CALL TB2SET(VCHVT,NVCHV,2,1,VCHTT,NVCHT,2,1,VCHIT,NVCHI,IERR)
C
  220 DO 240 I=1,NESG                     @COMPUTE ESG DISCHARGE ARRAYS
      L = I
  240 IF(XIBM(I).GE.0.0) GO TO 250        @POSITIVE CURRENT? NO...
      GO TO 800                           @ALL NEGATIVE CURRENT VALUES
  250 IF(VPSGMX.GT.(VCHIO + VBM(L))) GO TO 280
      DO 260 I=L,NESG                     @VOLTAGE LE PSG MAXIMUM...
  260 XIBM(I) = 0.0                       @ZERO-FILL REMAINING CURRENTS
      GO TO 320
```

```
  280 DO 290 I=L,NESG                    @VOLTAGE.GT.PSG MAXIMUM...
      K = I
      VEST = VCHIO + VBM(I) + ZCHRS * XIBM(I)
      IF(VEST.GE.VCHISA) GO TO 300       @SATURATED CHARGER CONDITION?
  290 VBM(I) = VEST                      @YES, REVISE VOLTAGE VALUES
      GO TO 320
  300 DO 310 I=K,NESG                    @NO, ACTIVE CHARGER CONDITION
      VEST = VBM(I) + ZCHRA * XIBM(I)
  310 VBM(I) = TB2GET(VEST,TTESG)        @REVISE VOLTAGE VALUES
  320 DO 325 I=1,NESG                    @SET CURRENTS.LE.LIMIT VALUE
  325 IF(XIBM(I).GT.XICHMX) XIBM(I) = XICHMX
      CALL ESGSUB                        @COMPUTE ESG I-V ARRAYS
  500 RETURN
C
  800 WRITE(NWRT,810)
  810 FORMAT('0***ALL BATTERY CURRENT VALUES ARE NEGATIVE')
      STOP
C
C********************  INTERNAL SUBROUTINES  ***********************
C
      SUBROUTINE  ESGIV
      DO 10 I=2,5                        @DETERMINE TEMPERATURE INDEX
      ITP = I
   10 IF(TBATT(I).GT.TTESG) GO TO 100
      ITP = 6
  100 CALL TB2SET(XIBATT,NIVB,2,1,QBATT,NQB,2,1,VBATT(1,1,ITP-1),NIVB,
     *            IERR)
      DO 120 I=1,NIVB                    @BUILT VOLTAGE TABLE AT
  120 TRESLT(I,1) = TB2GET(XIBATT(I),QB) @ TEMPERATURE = TBATT(ITP-1)
      CALL TB2SET(XIBATT,NIVB,2,1,QBATT,NQB,2,1,VBATT(1,1,ITP),NIVB,
     *            IERR)
      DO 150 I=1,NIVB                    @BUILD VOLTAGE TABLE AT
  150 TRESLT(I,2) = TB2GET(XIBATT(I),QB) @TEMPERATURE = TBATT(ITP)
      CALL TB2SET(XIBATT,NIVB,2,1,TBATT(ITP-1),2,2,1,TRESLT,NESG,IERR)
      DO 180 I=1,NIVB
      XIB(I) = CB * XIBATT(I)            @SET BATTERY CURRENT ARRAY
  180 XVB(I) = XN * TB2GET(XIBATT(I),TTESG) @COMPUTE VOLTAGE ARRAY
      RETURN
C
C
      SUBROUTINE  ESGSUB
      NPX = NIVB
      IF(ICHRT.NE.0) NPX = NESG
      DO 140 I=1,NPX
   90 TRESLT(I,1) = VBM(I)               @FORM SPECIAL ESG VOLTAGE ARRAY
  130 TRESV(I) = TRESLT(I,1)             @SET CELL VOLTAGE ARRAY
  140 VBM(I) = VBM(I) / XN               @COMPUTE CELL VOLTAGE ARRAY
      DO 160 I=1,NESG
      VCM = TRESV(I) / XN                @COMPUTE BATTERY CURRENT ARRAY
  150 CALL SLUP(VCM,TRESI(I),FDOT,VBM,XIBM,NPX,1)
  160 TRESLT(I,2) = NBATT * TRESI(I)     @COMPUTE ESG CURRENT ARRAY
      RETURN
C
      END
```

PA-INTER

```
      SUBROUTINE  INTER
C      LOCATES 'STABLE' INTERSECTION FOR DIFFERENCE AND
C      ENERGY STORAGE GROUP CHARACTERISTIC I-V CURVES
C
      INCLUDE  ALLCMN,LIST
      INCLUDE  COMON,LIST
      INCLUDE  PACMN,LIST
      NAMELIST/INTR/ DIFIV,TRESLT,VBUS,VMAX,VMIN,
     .         IFLG,D,DV,DVX,V1,V3,V4,I,XID,XIT/
      DATA  EPS/0.0001/
C
   10 IFLG = 0                              @INITIALIZE LIMITS FLAG = 0
      VMAX = DIFIV(ND,1)                    @SET MAXIMUM ALLOWABLE VOLTAGE
      VMIN = DIFIV(1,1)                     @SET MINIMUM ALLOWABLE VOLTAGE
      V1 = VBUS                             @SET INITIAL VOLTAGE ESTIMATE
      DV = 0.1                              @INITIALIZE VOLTAGE INCREMENT
C
  100 CALL SLUP(V1,XID,FDOT,DIFIV(1,1),DIFIV(1,2),ND,1)
      CALL SLUP(V1,XIT,FDOT,TRESLT(1,1),TRESLT(1,2),NESG,1)
      D = XID - XIT                         @COMPUTE CURRENT DIFFERENCE
      V3 = V1                               @SAVE VOLTAGE
      IF(D.NE.0.0) GO TO 200                @AT INTERSECTION?
      IF(V1.LT.VMIN .OR. V1.GT.VMAX) GO TO 300   @YES, VOLTAGE LIMIT?
      V1 = V1 - DV                          @NO, DECRMENT VOLTAGE
      GO TO 100                             @CONTINUE SEARCH
C
  200 V1 = V1 + DV                          @INCREMENT VOLTAGE
      IF(V1.LT.VMIN .OR. V1.GT.VMAX) GO TO 300   @VOLTAGE LIMIT?
      CALL SLUP(V1,XID,FDOT,DIFIV(1,1),DIFIV(1,2),ND,1)    @NO.
      CALL SLUP(V1,XIT,FDOT,TRESLT(1,1),TRESLT(1,2),NESG,1)
      IF(D*(XID-XIT)) 400,200,              @CROSSING ENCOUNTERED?
      V3 = V1                               @NO, SAVE PRESENT VOLTAGE VALUE
      D = XID - XIT                         @SAVE PRESENT CURVE DIFFERENCE
      GO TO 200                             @CONTINUE SEARCH FOR CROSSPOINT
C
  300 IF(IFLG.NE.0) GO TO 800               @CURVE SCAN COMPLETED?
      IFLG = 1                              @NO, SET VOLTAGE LIMIT FLAG = 1
      V1 = VBUS                             @RE-INITIALIZE VOLTAGE ESTIMATE
      DV = -DV                              @REDEFINE VOLTAGE INCREMENT
      GO TO 100                             @SCAN IN OPPOSITE DIRECTION
C
  400 V4 = AMIN1(V1,V3)                     @INTERSECTION ENCOUNTERED
      DVX = ABS(V3 - V1) / 10.0             @SET VOLTAGE INCREMENT
      DO 450 I=1,20                         @SEARCH FOR INTERSECTION...
      DO 420 J=1,10
      V4 = V4 + DVX                         @INCREMENT VOLTAGE
      CALL SLUP(V4,XID,FDOT,DIFIV(1,1),DIFIV(1,2),ND,1)
      CALL SLUP(V4,XIT,FDOT,TRESLT(1,1),TRESLT(1,2),NESG,1)
      IF(XID-XIT) 440,500                   @PASSED/AT/BEFORE INTERSECTION?
  420 IF((XID-XIT).LT.EPS) GO TO 500        @ACCEPTABLE CURVE DIFFERENCE?
  440 V4 = V4 - DVX                         @NO, DECREMENT VOLTAGE
  450 DVX = DVX / 10.0                      @REVISE VOLTAGE INCREMENT
C
  500 VBUS = V4                             @SET OPERATING POINT VOLTAGE
      RETURN
C
```

```
C*****ERROR MESSAGE EXITS
C
  800 WRITE(NWRT,INTR)
      CALL CRVPLT                          @NO STABLE OPERATING POINT...
      WRITE(NWRT,810) YEAR,DATE,TIMEH      @WRITE ERROR MESSAGE
  810 FORMAT('0***NO STABLE OPERATING POINT FOUND'/,
     .   4X,'TIME = ',F5.0,F5.0,F5.2///,7X,
     .   'ESG VOLTAGE   ESG CURRENT       DIF VOLTAGE   DIF CURRENT'//)
      N = MIND(ND,NESG)
      DO 820 I=1,N                         @WRITE ESG/DIF I-V ARRAYS
  820 WRITE(NWRT,830) TRESLT(I,1),TRESLT(I,2),DIFIV(I,1),DIFIV(I,2)
  830 FORMAT(7X,2(2(F11.5,3X),3X))
      N = N + 1
      IF(ND.GT.NESG) GO TO 860
      DO 840 I=N,NESG
  840 WRITE(NWRT,850) TRESLT(I,1),TRESLT(I,2)
  850 FORMAT(7X,2(F11.5,3X))
      GO TO 900
  860 DO 870 I=N,ND
  870 WRITE(NWRT,880) DIFIV(I,1),DIFIV(I,2)
  880 FORMAT(38X,2(F11.5,3X))
  900 STOP
C
      END
```

PA-PCDGC

```
      SUBROUTINE  PCDG(KPCD)
C      INITIALIZES POWER CONDITIONING & DISTRIBUTION GROUP PARAMETERS
C     ENTRY  PCDGC
C      COMPUTES POWER CONDITIONING AND DISTRIBUTION GROUP CURRENT-
C      VOLTAGE CHARACTERISTIC CURVE ARRAYS  (XI,XV)
C
      INCLUDE  ALLCMN.LIST
      INCLUDE  COMON.LIST
C
   10 IF(IFTYPE.LT.0 .OR. IFTYPE.GT.15) GO TO 800     @ILLEGAL PATTERN?
   40 TLON = 0.0                                     @NO.
      TLOFF = 0.0
      CLS = 0.0
      CALL TB2SET(CLSIT,NCLSI,2,1,CLSTT,NCLST,2,1,CLST,NCLSI,IERR)
      DO 45 J=1,15,2                                 @FOR SELECTED PATTERN, COMPUTE
      TLON = TLON + TL(J,IFTYPE+1)                   @ ILLUMINATION DURATION TOTAL,
      TLOFF = TLOFF + TL(J+1,IFTYPE+1)               @ SHUT-OFF DURATION TOTAL
   45 CLS = CLS + TL(J,IFTYPE+1) * TB2GET(CLR,TL(J,IFTYPE+1))
      IF(TLON.LE.0.0) GO TO 800                      @ILLEGAL FLASHER PATTERN?
      IF(TLOFF.LT.0.0) GO TO 800
   50 DL = TLON / (TLON + TLOFF)                     @NO. COMPUTE LAMP DUTY CYCLE
      CLS = CLS / TLON                               @COLD-FILAMENT SURGE COEFFICIENT
   80 ACTRL = VLR / (CLR * CLS)                      @COMPUTE ACTUAL LAMP RESISTANCE
  100 AVGRL = ACTRL / DL                             @COMPUTE AVERAGE LAMP RESISTANCE
  120 VINCIV = (VMAXIV - VMINIV) / (NPCDG - 1)       @SET VOLTAGE INCREMENT
C
      ENTRY  PCDGC(KPCD)
C
  140 TTPCD = TTAMB + DTTPCD                         @SET PCDG EQUIPMENT TEMPERATURE
  150 CALL SLUP(TTPCD,VRIO,FDOT,VRIOT(1,1),VRIOT(1,2),NVRIO,1)
      CALL SLUP(TTPCD,VRISA,FDOT,VRISAT(1,1),VRISAT(1,2),NVRISA,1)
      CALL SLUP(TTPCD,ZRA,FDOT,ZRAT(1,1),ZRAT(1,2),NZRA,1)
      CALL SLUP(TTPCD,ZRS,FDOT,ZRST(1,1),ZRST(1,2),NZRS,1)
      XV(1) = VMINIV                                 @SET INITIAL VOLTAGE VALUE
      IF(VMINIV.GT.VRISA) GO TO 290                  @SELECT VOLTAGE POSITION
      IF(VMINIV.GT.VRIO) GO TO 240
C
  160 CALL VISUB1(J1,$320)                           @VMINIV.LE.VRIO
  200 CALL VISUB2(J1,J2,$320)
  220 CALL VISUB3(J2)
      GO TO 320
C
  240 CALL VISUB2(1,J2,$320)                         @VRIO.LT.VMINIV.LE.VRISA
  270 CALL VISUB3(J2)
      GO TO 320
C
  290 CALL VISUB3(1)                                 @VMINIV.GT.VRISA
      GO TO 320
C
  320 CALL TB2SET(XIHVT,NXIHV,2,1,XIHTT,NXIHT,2,1,XIHIT,NXIHI,IERR)
      DO 330 J=1,NPCDG                               @COMPUTE CHARACTERISTIC ARRAYS
      XI(J) = XI(J) + TB2GET(XV(J),TTPCD)            @PCDG CURRENT ARRAY
      IF(XI(J).LT.0.0) XI(J) = 0.0
  330 XV(J) = XV(J) + RLL * XI(J)                    @PCDG VOLTAGE ARRAY
      RETURN
C
```

```
C*****ERROR MESSAGE EXITS
C
  800 WRITE(NWRT,810)
  810 FORMAT('0*** INCORRECT FLASHER PATTERN ENTRIES')
      STOP
C
C*********************  INTERNAL SUBROUTINES  **********************
C
      SUBROUTINE  VISUB1(J,*)              @XV(J).LE.VRIO
      DO 190 J=1,NPCDG
      IF(XV(J).GT.VMAXIV) RETURN 2         @MAXIMUM VOLTAGE? NO...
      IF(XV(J).GT.VRIO) GO TO 195          @SATURATION CONDITION? NO...
  170 XI(J) = 0.0                          @COMPUTE LAMP REGULATOR CURRENTS
      IF(J.EQ.NPCDG) RETURN 2              @PCDG ARRAY SIZE EXCEEDED? NO...
  190 XV(J+1) = XV(J) + VINCIV             @INCREMENT LAMP REGULATOR VOLTAGE
  195 RETURN
C
C
      SUBROUTINE  VISUB2(K,J,*)            @VRIO.LT.XV(J).LE.VRISA
      DO 210 J=K,NPCDG
      IF(XV(J).GT.VMAXIV) RETURN 3         @MAXIMUM VOLTAGE? NO...
      IF(XV(J).GT.VRISA) GO TO 215         @ACTIVE CONDITION? NO...
      XI(J) = 0.0                          @COMPUTE LAMP REGULATOR CURRENTS
      IF(KPCD.EQ.2) XI(J) = (XV(J) - VRIO) / (AVGRL + ZRS)
      IF(KPCD.EQ.3) XI(J) = (XV(J) - VRIO) / (ACTRL + ZRS)
      IF(J.EQ.NPCDG) RETURN 3              @PCDG ARRAY SIZE EXCEEDED? NO...
  210 XV(J+1) = XV(J) + VINCIV             @INCREMENT LAMP REGULATOR VOLTAGE
  215 RETURN
C
C
      SUBROUTINE  VISUB3(K)                @XV(J).GT.VRISA
      CALL TB2SET(VLBVT,NVLBV,2,1,VLBTT,NVLBT,2,1,VLBT,NVLB,IERR)
      DO 230 J=K,NPCDG
      IF(XV(J).GT.VMAXIV) GO TO 235        @MAXIMUM VOLTAGE? NO...
      VLB = TB2GET(XV(J),TTPCD)
      XI(J) = 0.0                          @COMPUTE LAMP REGULATOR CURRENTS
      IF(KPCD.EQ.2) XI(J) = VLB / (AVGRL + ZRA)
      IF(KPCD.EQ.3) XI(J) = VLB / (ACTRL + ZRA)
      IF(J.EQ.NPCDG) GO TO 235             @PCDG ARRAY SIZE EXCEEDED? NO...
  230 XV(J+1) = XV(J) + VINCIV             @INCREMENT LAMP REGULATOR VOLTAGE
  235 RETURN
C
      END
```

PA-PRTPLT

```
      SUBROUTINE  PRTPLT
C      WRITES PERFORMANCE ANALYSIS SUMMARY DATA TO SFILE FOR
C      LATER PRINTOUT AND TREND-ANALYSIS PLOTTING
C
      INCLUDE  ALLCMN,LIST
      INCLUDE  COMON,LIST
      INCLUDE  PACMN,LIST
      INCLUDE  SUMCMN,LIST
C
      IYEAR = IFIX(YEAR)                    @SET TIME OF TEST
      IDAY = IFIX(DATE)
      TIME = TIMEH
C
      PRT1(1) = DAYSST                      @UNREGULATED BUS SUMMARY DATA
      PRT1(2) = VBUS
      PRT1(3) = PPSG
      PRT1(4) = XIPSG
      PRT1(5) = PESG
      PRT1(6) = XITT
      PRT1(7) = PPCD
      PRT1(8) = XIPCD
C
      PRT2(1) = DAYSST                      @POWER SOURCE GROUP SUMMARY DATA
      PRT2(2) = TSAF
      PRT2(3) = QDT
      PRT2(4) = VSA
      PRT2(5) = XISA
      PRT2(6) = PSA
      PRT2(7) = MSAPWR
      PRT2(8) = MARSA
      PRT2(9) = VBUS
      PRT2(10) = XIZ
      PRT2(11) = PSL
C
      PRT3(1) = DAYSST                      @ENERGY STORAGE GROUP SUMMARY DATA
      PRT3(2) = TTESG
      PRT3(3) = PBATT + ABS(BCUR * (VBUS - VBAT))
      PRT3(4) = VBUS
      PRT3(5) = PBATT
      PRT3(6) = BCUR
      PRT3(7) = VBAT
      PRT3(8) = QB
      PRT3(9) = CB * QB
      PRT3(10) = SPGR
      PRT3(11) = TBFRZ
C
      WRITE(NWRT,20) YEAR,DATE,TIMEH,DAYSST,VBUS,XIPSG,XITT,XIPCD,QB
   20 FORMAT(1X,F5.0,':',F4.0,':',F5.2,2X,F8.3,3X,F8.3,4(2X,F6.3))
      CALL NTRAN(SFILE,1,ISIZE,IYEAR,ISTAT,22)   @WRITE PA OUTPUT RECORD
      IF(ISTAT.LT.1) GO TO 80               @BAD I/O STATUS?
      RETURN                                @NO.
C
   80 WRITE(NWRT,90)  IYEAR,IDAY,TIME       @YES,
   90 FORMAT('0***NTRAN WRITE ERROR AT DATE = ',I4,':',I3,':',F6.3)
      STOP
C
      END
```

PA-PSGC

```
      SUBROUTINE  PSGC(CT,TC)
C      COMPUTES POWER SOURCE GROUP (SOLAR ARRAY) CURRENT-
C      VOLTAGE CHARACTERISTIC CURVE ARRAYS  (SI,SV)
C
      INCLUDE  ALLCMN,LIST
      INCLUDE  COMON,LIST
      INCLUDE  PACMN,LIST
      REAL  MAXI, MAXV, VAR(5)
      DEFINE  TCNVRT(T) = 5.0 * (T + 459.67) / 9.0 - 273.15
      NAMELIST/PSG/ CT,TC,ALPHEQ,VAR,DECL,ET,APPSC,ATMEXC,SDF,
     .          THETLA,HOURT,SRT,SST,BHOUR,DTTA,DTTAMB,TTAMB,
     .          TSAF,TSAC,COSTZS,COSTW,COSTS,SALT,SAZM,CCM,QDN,
     .          PHIAI,PHIAA,ETAA,ETAB,ETAC,COSTLT,BS,QDG,YV,QDS,
     .          QDT,CDEG,VDEG,X,XX,ALPHA,BETA,RCELL,RHO,XISC,DISC,
     .          C3,C4,MSAPWR,DMPPV,DXV,MAXI,MAXV,SAPWR/
      NAMELIST/PSGOUT/ YEAR,DATE,TIMEH,VZSB,SASCC,SAOCV,VZINC,
     .          MFINAL,NAPSG,XI2,V2,ZRFI,ZRFV,ZI,ZV,SI,SV/
C
   80 ALPHEQ = OMEGA * DATE              @SOLAR VECTOR LOCATION
      COS1AQ = COS(ALPHEQ)
      COS2AQ = COS(2.0 * ALPHEQ)
      COS3AQ = COS(3.0 * ALPHEQ)
      SIN1AQ = SIN(ALPHEQ)
      SIN2AQ = SIN(2.0 * ALPHEQ)
      SIN3AQ = SIN(3.0 * ALPHEQ)
      J = 5
      IF(ITAPE.NE.0) J = 2
      DO 90 I=1,J                        @COMPUTE SOLAR RADIATION VARIABLS
   90 VAR(I) = FA(1,I) + FA(2,I) * COS1AQ + FA(3,I) * COS2AQ +
     .          FA(4,I) * COS3AQ + FA(5,I) * SIN1AQ +
     .          FA(6,I) * SIN2AQ + FA(7,I) * SIN3AQ
      DECL = DEGRAD * VAR(1)             @SOLAR DECLINATION ANGLE
      ET = VAR(2)                        @EQUATION OF TIME DIFFERENCE
      IF(ITAPE.NE.0) GO TO 110
      APPSC = 3.1524808 * VAR(3)         @APPARENT SOLAR CONSTANT
      ATMEXC = VAR(4)                    @ATMOSPHERE EXTINCTION FACTOR
      SDF = VAR(5)                       @SKY DIFFUSE FACTOR
C
  110 THETLA = DEGRAD * THELAD           @BUOY LATITUDE
  120 HOURT = PI                         @COMPUTE TERMINATOR HOUR ANGLE
      IF(THETLA.LT.(0.5 * PI - DECL))
     .    HOURT = ACOS(-1.0 * TAN(THETLA) * TAN(DECL))
  160 SRT = 12.0 * (1.0 - HOURT / PI) - ET - TZN + THELOD / 15.0
      SST = 24.0 - SRT                   @COMPUTE SUNRISE & SUNSET TIMES
  170 BHOUR = DEGRAD * (15.0 * (TIMEH - 12.0 + TZN + ET) - THELOD)
   10 IF(ITAPE.EQ.0) GO TO 30            @NO. USE NOAA TAPE INPUT?
      CALL RDTAPE                        @YES.
      GO TO 60
   30 CALL SLUP(DATE,DTTA,FDOT,DTTA1((1,1),DTTA1((1,2),NDTTA,1)   @NO.
   40 CALL SLUP(TIMEH,DTTAMB,FDOT,DTAMB1((1,1),DTAMB1((1,2),NDTAMB,1)
      TTAMB = TTAVE + DTTA + DTTAMB      @COMPUTE AMBIENT TEMPERATURE
   60 TSAF = TTAMB + DTTPSG              @SOLAR ARRAY TEMP, (FAHRENHEIT)
   70 TSAC = TCNVRT(TSAF)                @SOLAR ARRAY TEMP. (CENTIGRADE)
  180 IF(ABS(BHOUR).GE.ABS(HOURT)) GO TO 1180    @SOLAR OCCULTATION?
      IF(ITAPE.NE.0) GO TO 410           @USE NOAA TAPE INPUT?
C
```

```
  190 COSTZS = COS(BHOUR) * COS(DECL) * COS(THETLA) +      @NO...
     .         SIN(DECL) * SIN(THETLA)       @COMPUTE DIRECTION COSINES
      COSTW = COS(DECL) * SIN(BHOUR)
      COSTS = SQRT(ABS(1.0 - COSTZS**2.0 - COSTW**2.0))
      IF(COS(BHOUR).LT.(TAN(DECL)/TAN(THETLA))) COSTS = -COSTS
  200 SALT = ASIN(COSTZS)                   @COMPUTE SOLAR ALTITUDE
  210 SAZM = ASIN(COSTW / COS(SALT))        @COMPUTE SOLAR AZIMUTH
      IF(COSTS.LT.0.0) SAZM = PI - SAZM
  220 I = 1 + IFIX(CT)                      @SET CLOUD TYPE INDICATOR
      IF(I.LT.1 .OR. I.GT.3) GO TO 8000     @VALID CLOUD TYPE?
  230 CCM = 1.0                             @YES.
      IF(TC.EQ.0.0) GO TO 250               @ZERO CLOUD COVER? NO...
      J = 1                                 @SET SOLAR ALTITUDE INDICATOR
      IF(SALT.GT.PI/4.0) J = 2
      CCM = P0(I,J) + P1(I,J) * TC + P2(I,J) * TC**2.0 +
     .      P3(I,J) * TC**3.0               @COMPUTE CLOUD COVER MODIFIER
  250 QDN = APPSC * CN * CCM * EXP(-ATMEXC / COSTZS)
  270 PHIAI = DEGRAD * PHIAID               @SOLAR ARRAY POINTING ANGLES
      PHIAA = DEGRAD * PHIAAD
  280 ETAA = COS(PHIAI)
      ETAB = SIN(PHIAA) * SIN(PHIAI)
      ETAC = COS(PHIAA) * SIN(PHIAI)
  290 COSTLT = ETAA * COSTZS + ETAB * COSTW + ETAC * COSTS
C
  310 BS = SDF * QDN / CN**2.0              @COMPUTE SKY BRIGHTNESS
  340 QDG = REFLH * (BS + QDN * COSTZS) * (1.0 - ETAA) / 2.0
  360 YV = 0.45
      IF(COSTLT.GT.-0.2) YV = 0.55 + 0.437 * COSTLT +
     .   0.313 * COSTLT**2.0
      QDS = QDN * (SDF * YV + REFLH * (SDF + COSTZS) / 2.0)
  370 QDS = QDS + ABS(QDN * SDF - QDS) * COS(SALT)
  380 QDT = QDG + QDS                       @TOTAL INCIDENT SOLAR RADIATION
      IF(COSTLT.GT.0.0) QDT = QDT + QDN * COSTLT
  410 CALL SLUP(DATEM,CDEG,FDOT,SADEGC(1,1),SADEGC(1,2),NCDEG,1)
  420 CDEG = 1.0 - 1.0E-6 * (100.0 - CDEGA) * (100.0 - CDEGB) *
     .       (100.0 - CDEG)               @SET CURRENT DEGRADATION FACTOR
  440 CALL SLUP(DATEM,VDEG,FDOT,SADEGV(1,1),SADEGV(1,2),NVDEG,1)
  450 VDEG = 1.0 - 1.0E-4 * (100.0 - VDEGA) *
     .       (100.0 - VDEG)               @SET VOLTAGE DEGRADATION FACTOR
  470 X = SPECOR * QDT / 10.0               @EFFECTIVE SOLAR INSOLATION
  480 XX = X * (1.0 - CDEG)                 @MODIFIED SOLAR INSOLATION
C
  500 ALPHA = ACELL * XX * (7.428E-7 - 1.83E-9 * TSAC) / ACSTD
  510 CALL SLUP(TSAC,RCELL,FDOT,TEMTAB,RSCELL,NRSCEL,3)
      CALL SLUP(XX,RHO,FDOT,SUNLIT,ROE,NROE,3)
  520 CALL TB2SET(BTEMP,NBTEMP,3,1,SUNMW,NSUNMW,3,1,BETAB,NBETAB,IERR)
      BETA = TB2GET(TSAC,XX) / 1000.0
  530 CALL SLUP(0.0,XIISC,FDOT,VV,XII,30,1) @FIND SHORT CIRCUIT CURRENT
      CALL SLUP(0.0,VVOC,FDOT,XII,VV,30,1)  @FIND OPEN CIRCUIT VOLTAGE
  560 ALPHA = NP * ALPHA                    @SHORT CIRCUIT CURRENT FACTOR
      BETA = NS * BETA                      @OPEN CIRCUIT VOLTAGE FACTOR
      RCELL = NS * (0.114 + RCELL) / NP     @SERIES RESISTANCE
      RHO = NS * RHO / NP                   @TEMPERATURE CORRECTION FACTOR
  570 XISC = NP * XIISC * (1.0 - CDEG)      @SHORT CIRCUIT CURRENT
  580 DISC = ALPHA * (TSAC - TCSTD) - XISC * (1.0 - X / XCSTD)
  590 C3 = BETA * (TSAC - TCSTD) + DISC * RCELL
      C4 = RHO * (TSAC - TCSTD)
```

```
C
      DO 600 J=1,NPSG                        @ZERO-FILL PSG I-V ARRAYS
      SI(J) = 0.0
      SV(J) = 0.0
      V2(J) = 0.0
  600 XI2(J) = 0.0
      DO 610 J=1,30                          @SET REFERENCE I-V ARRAYS
      XI2(J) = NP * (XII(J) - CDEG * XIISC) + DISC
  610 SV(J) = NS * (VV(J) - VDEG * VVOC) - C3 - C4 * XI2(J)
      DO 620 I=30,2,-1                       @CHECK 'SV' FOR MONOTANICITY
      IF(SV(I)-SV(I-1).LT.0.0) GO TO 620
      NSV = I
      GO TO 630
  620 CONTINUE
      NSV = 1
  630 NXX = 31 - NSV
      V2(1) = 0.0
      J = NPSG - 1
      DO 635 L=1,J                           @REDEFINE PSG I-V ARRAYS
      CALL SLUP(V2(L),SI(L),FDOT,SV(NSV),XI2(NSV),NXX,1)
      IF(SI(L).GT.0.0) GO TO 635
      SI(L) = 0.0                            @SET FINAL I-V POINT
      CALL SLUP(0.0,V2(L),FDOT,XI2(NSV),SV(NSV),NXX,1)
      MFINAL = L + 1
      SI(MFINAL) = 0.0                       @SET FINAL S/A I-V POINT
      V2(MFINAL) = 2.0 * AMAX1(VBUS,V2(L))
      GO TO 650
  635 V2(L+1) = V2(L) + VSAINC
      GO TO 8200                             @PSGC RANGE EXCEEDED
  650 DO 655 L=1,MFINAL
  655 XI2(L) = NESP * SI(L)
C
  670 MSAPWR = 0.0                           @INITIALIZE MAXIMUM POWER
      IF(MFINAL.LE.2) GO TO 1200
      IF(XI2(1).LE.0.0) GO TO 1200
      MAXV = V2(MFINAL-1) / 3.0              @INITIALIZE MAX-POWER VOLTAGE
      DMPPV = (V2(MFINAL-1) - MAXV) / 50.0   @SET VOLTAGE INCREMENT
      DXV = DMPPV
      DO 710 L=1,60
  680 CALL SLUP(MAXV,MAXI,FDOT,V2,XI2,MFINAL-1,1)
      SAPWR = MAXI * MAXV                    @COMPUTE SOLAR ARRAY (PSG) POWER
  690 IF(SAPWR.LE.MSAPWR) GO TO 700          @PSG POWER.LE.MAXIMUM POWER?
      MSAPWR = SAPWR                         @NO, REDEFINE MAXIMUM POWER
      MAXV = MAXV + DXV                      @INCREMENT PSG VOLTAGE
      GO TO 710
  700 IF(DXV.LT.DMPPV) GO TO 720             @MAX-POWER-POINT LOCATED?
      MSAPWR = 0.0                           @NO, RE-INITIALIZE MAXIMUM POWER
      MAXV = MAXV - DXV                      @ AND MAX-POWER VOLTAGE
      DXV = DXV / 10.0                       @REDEFINE VOLTAGE INCREMENT
  710 CONTINUE
      GO TO 8100                             @MAX-POWER-POINT NOT FOUND
  720 MAXV = MAXV - DXV                      @YES, SET MAX-POWER VOLTAGE
      IF(MAXV.LE.0.0) MAXV = 0.00001
      MAXI = MSAPWR / MAXV                   @SET MAX-POWER CURRENT
C
  740 DO 745 L=1,MFINAL
      CALL SLUP(SI(L),VSHIFT,FDOT,AD1(1,2),AD1(1,1),NAD1,1)
```

```
  745 V2(L) = V2(L) - RSA * SI(L) - VSHIFT
      L = 1
      CALL SLUP(0.0,SASCC,FDOT,V2,XI2,MFINAL,1)
      V2(1) = 0.0                          @REDEFINE INITIAL S/A I-V POINT
      XI2(1) = SASCC
      DO 750 I=2,MFINAL                    @ELIMINATE NEGATIVE VOLTAGES
      IF(V2(I).LE.0.0) GO TO 750
      L = L + 1
      V2(L) = V2(I)
      XI2(L) = XI2(I)
  750 CONTINUE
      MFINAL = L                           @SET SIZE OF S/A I-V ARRAYS
      L = L + 1
      DO 755 I=L,NPSG                      @ZERO-FILL REMAINDER OF S/A
      V2(I) = 0.0                          @ I-V ARRAYS
  755 XI2(I) = 0.0
C
  760 CALL SLIVC(V2(MFINAL-1))             @COMPUTE SHUNT-LIMITER I-V ARRAYS
C
  780 VZSB = ZV(2)                         @SHUNT LIMITER TURN-ON VOLTAGE
      SAOCV = V2(MFINAL-1)                 @PSG OPEN CIRCUIT VOLTAGE
      IF(VZSB.GE.SAOCV) GO TO 790
      IF(ISH.GT.0) GO TO 800               @SELECT SHUNT LIMITER TYPE
  790 DO 795 L=1,MFINAL          @NO SHUNT LIMITER PRESENT...
      SI(L) = XI2(L)                       @SET PSG CURRENT ARRAY
  795 SV(L) = V2(L)                        @SET PSG VOLTAGE ARRAY
      NAPSG = MFINAL                       @SET SIZE OF PSG I-V ARRAYS
      GO TO 5000                           @RETURN
C
  800 N = 1                          @ZENER DIODE OR SHUNT LIMITER...
      IF(IPSG.NE.0) N = NESP               @SELECT POWER SOURCE GROUP TYPE
      CALL PSGIV(*8200)                    @COMPUTE PSG I-V ARRAYS
      GO TO 5000                           @RETURN
C
 1180 QDT = 0.0                            @SOLAR OCCULTATION
      MSAPWR = 0.0                         @SET MAX-POWER-POINT = ZERO
 1200 SI(1) = 0.0                          @COMPUTE PSG I-V ARRAYS
      SV(1) = 0.0
      V2(1) = 0.0
      XI2(1) = 0.0
      DO 1210 L=2,NPSG
      SI(L) = 0.0
      XI2(L) = 0.0
      SV(L) = SV(L-1) + VSAINC
 1210 V2(L) = SV(L)
      MFINAL = NPSG                        @SET SIZE OF S/A I-V ARRAYS
      NAPSG = NPSG                         @SET SIZE OF PSG I-V ARRAYS
      ZI(1) = 0.0                          @FILL SHUNT LIMITER I-V ARRAYS
      ZV(1) = 0.0
      VZINC = SV(NPSG) / (NSL - 1)
      DO 1220 I=2,NSL
      ZI(I) = 0.0
 1220 ZV(I) = ZV(I-1) + VZINC
      ZV(NSL) = SV(NPSG)
C
 5000 RETURN
C
```

```
C*****ERROR MESSAGE EXITS
C
 8000 WRITE(NWRT,8010)
 8010 FORMAT('0*** INVALID CLOUD COVER TYPE SPECIFIED')
      STOP
 8100 WRITE(NWRT,8110)
 8110 FORMAT('0*** MAX-POWER-POINT CALCULATION FAILED TO CONVERGE')
      WRITE(NWRT,PSG)
      WRITE(NWRT,PSGOUT)
      STOP
 8200 WRITE(NWRT,8210)
 8210 FORMAT('0*** SOLAR ARRAY I-V DIMENSIONS EXCEEDED')
      WRITE(NWRT,PSG)
      WRITE(NWRT,PSGOUT)
      STOP
C
C*********************  INTERNAL SUBROUTINES  ************************
C
      SUBROUTINE  PSGIV(*)
  810 J = NPSG - 1                          @FOR PSG VOLTAGE.LT.VZSB,
      DO 840 L=1,J                          @ COMPUTE PSG I-V ARRAYS
      I = L
      IF(V2(L).GE.VZSB) GO TO 850
  830 SI(L) = XI2(L)
  840 SV(L) = V2(L)
      RETURN 1                              @PSGC RANGE EXCEEDED
C
  850 SV(I) = VZSB                          @FOR PSG VOLTAGE.GE.VZSB,
      DO 870 L=I,J                          @ COMPUTE PSG I-V ARRAYS
      CALL SLUP(SV(L),SI(L),FDOT,V2,XI2,MFINAL-1,1)
      CALL SLUP(SV(L),XZI,FDOT,ZV,ZI,NSL,1)
      XZI = N * XZI                         @REVISE SHUNT LIMITER CURRENT
      SI(L) = SI(L) - XZI                   @REVISE SOLAR ARRAY CURRENT
  860 IF(SI(L).GT.0.0) GO TO 870
      NAPSG = L                             @SET SIZE OF PSG I-V ARRAYS
      GO TO 880
  870 SV(L+1) = SV(L) + AMIN1(VSAINC,ZV(4)-ZV(3))
      RETURN 1                              @PSGC RANGE EXCEEDED
C
  880 CALL SLUP(0.0,VZCR,FDOT,SI,SV,NAPSG,1)
      NAPSG = NAPSG + 1
      SV(NAPSG-1) = VZCR                    @COMPLETE PSG I-V ARRAYS
      SI(NAPSG-1) = 0.0
      SV(NAPSG) = V2(MFINAL)
      SI(NAPSG) = 0.0
      RETURN
C
      END
```

PA-SLIVC

```
      SUBROUTINE  SLIVC(VSAMX)
C      COMPUTES SHUNT LIMITER CURRENT-VOLTAGE
C      CHARACTERISTIC CURVE ARRAYS  (ZI,ZV)
C     ENTRY  ZENER
C      INITIALIZES ZENER DIODE REFERENCE CURRENT AND VOLTAGE
C
      INCLUDE  ALLCMN.LIST
      INCLUDE  COMON.LIST
      INCLUDE  PACMN.LIST
      REAL  TCZI(NTCZIV), XZI(NSL), XZV(NSL)
C
   10 I = ISH + 1                          @SET SHUNT LIMITER TYPE FLAG
      DO 30 J=1,NSL
      ZI(J) = 0.0                          @INITIALIZE SHUNT LIMITER CURRENT
   30 ZV(J) = 0.0                          @INITIALIZE SHUNT LIMITER VOLTAGE
      ZV(2) = VSAMX
      GO TO (700,470,550,600),I            @SELECT SHUNT LIMITER TYPE
C
      ENTRY  ZENER
C
  100 I = ISH + 1                          @SET SHUNT LIMITER TYPE FLAG
  120 IF(I.LT.1 .OR. I.GT.4) GO TO 800      @INVALID SHUNT LIMITER TYPE?
      GO TO (780,400,530,780),I            @SELECT ZENER DIODE TYPE
C
C.....ORDINARY ZENER DIODE
C
  400 T1 = (TZBR - 30.0) / 100.0           @INITIAL LOAD LINE ANALYSIS...
      TCZ = 0.0
      VZSB = 1.0E30
      DO 420 J=1,15                        @COMPUTE REFERENCE ZENER DIODE
      ZRFV = VZBR * (1.0 - T1 * TCZ)       @ BREAKDOWN VOLTAGE & TEMPERATURE
      CALL SLUP(ZRFV,TCZ,FDOT,ZTCOEF(1,1),ZTCOEF(1,2),NZTC,1)
      IF(ABS(ZRFV)-ABS(VZSB).LT.0.1) GO TO 780   @REFERENCE FOUND?
  420 VZSB = ZRFV
      GO TO 820                            @NO. FAILED TO CONVERGE
C
  470 VZSB = NZS * ZRFV * (1.0 + TCZ * (TSAC - 30.0) / 100.0)   @YES.
  480 CALL TB2SET(ZDIMPV,NZDV,2,1,ZDIMPT,NZDT,2,1,ZDIMP,NZDIMP,IERR)
  490 ZV(2) = VZSB                         @SET ZENER DIODE I-V ARRAYS
      ZI(3) = 100.0
      ZV(3) = VZSB + NZS * ZI(3) * TB2GET(ZRFV,TSAC)
      GO TO 700                            @SCALE AND FILL I-V ARRAYS
C
C.....TEMPERATURE-COMPENSATED ZENER DIODE
C
  530 CALL SLUP(HDZMX,ZRFI,FDOT,CURZ(1,1),CURZ(1,2),NCURZ,1)
  540 ZRFV = HDER * HDZMX / ZRFI           @COMPUTE REFERENCE BREAKDOWN I,V
      RETURN
C
  550 CALL TB2SET(TCZV,NTCZV,2,1,TCZT,NTCZT,2,1,TCZIV,NTCZIV,IERR)
      DO 560 J=1,NTCZV
  560 TCZI(J) = TB2GET(TCZV(J),TSAC)
  570 CALL SLUP(0.0,RATVB,FDOT,TCZI,TCZV,NTCZV,1)
  580 RTVINC = 1.05 - RATVB / (NSL-2)      @SET VOLTAGE-RATIO INCREMENT
      ZV(2) = RATVB
      DO 590 J=3,NSL
```

```
      ZV(J) = ZV(J-1) + RTVINC          @INCREMENT VOLTAGE RATIO
  590 CALL SLUP(ZV(J),ZI(J),FDOT,TCZV,TCZI,NTCZV,1)
      DO 595 J=1,NSL
      ZI(J) = ZRFI * ZI(J)              @SET ZENER DIODE CURRENT ARRAY
  595 ZV(J) = NZS * ZRFV * ZV(J)        @SET ZENER DIODE VOLTAGE ARRAY
      GO TO 700                         @SCALE AND FILL I-V ARRAYS
C
C*****ACTIVE SHUNT LIMITER
C
  600 VSHTO = VSHTOR * (1.0 + CSH * (TSAC - TSHREF) / 100.0)
  640 CALL SLUP(TSAC,ZSH,FDOT,ZSHTAB(1,1),ZSHTAB(1,2),NZSH,1)
  650 ZV(2) = VSHTO                     @SET SHUNT LIMITER I-V ARRAYS
      ZI(3) = 100.0
      ZV(3) = VSHTO + ZSH * ZI(3)
C
C*****SCALE I-V ARRAYS TO MAXIMUM SOLAR ARRAY VOLTAGE
C
  700 IF(ZV(2).LT.VSAMX) GO TO 730      @IF TURN-ON VOLTAGE.GE.MAXIMUM
      ZV(2) = VSAMX                     @ SOLAR ARRAY VOLTAGE...
      DO 710 I=3,NSL
      ZI(I) = 0.0
  710 ZV(I) = ZV(I-1) + 0.001
      GO TO 780                         @RETURN
C
  730 VZINC = (VSAMX - ZV(2)) / (NSL - 3)   @IF TURN-ON VOLTAGE.LT.MAXI-
      N = NSL-2                             @ MUM SOLAR ARRAY VOLTAGE...
      NSH = 3
      IF(ISH.EQ.2) NSH = NSL
      XZV(2) = ZV(2)
      DO 740 I=3,N
      XZV(I) = XZV(I-1) + VZINC
      XZI(I) = 0.0
  740 CALL SLUP(XZV(I),XZI(I),FDOT,ZV,ZI,NSH,1)
      XZV(NSL-1) = VSAMX
      XZI(NSL-1) = 0.0
      IF(ISH.NE.0) CALL SLUP(XZV(NSL-1),XZI(NSL-1),FDOT,ZV,ZI,NSH,1)
      XZV(NSL) = 1.1 * VSAMX
      XZI(NSL) = 0.0
      CALL SLUP(XZV(NSL),XZI(NSL),FDOT,ZV,ZI,NSH,1)
      DO 750 I=3,NSL
      ZI(I) = XZI(I)
  750 ZV(I) = XZV(I)
  780 RETURN
C
C*****ERROR MESSAGE EXITS
C
  800 WRITE(NWRT,810)
  810 FORMAT('0*** INVALID SHUNT LIMITER TYPE SPECIFIED')
      STOP
  820 WRITE(NWRT,830)
  830 FORMAT('0*** REFERENCE ZENER DIODE BREAKDOWN VOLTAGE NOT FOUND')
      STOP
C
      END
```

PA-SUMARY

```
      SUBROUTINE  SUMARY
C      READS PERFORMANCE ANALYSIS OUTPUT DATA FROM 'SFILE'
C      AND PRODUCES TABULAR PRINTOUT AND SUMMARY PLOTS
C
      INCLUDE  COMCN.LIST
      INCLUDE  SUMCMN.LIST
      INTEGER  PGCNT
      DATA  A/0.0/, H/0.14/
      REAL  DAYSST(4)
      REAL  TSAF(4), VSA(4), XISA(4), XIZ(4)
      REAL  BCUR(4), CB(4), TTES3(4), VBAT(4)
C
      IYEAR = 9999                          @BUILD END-OF-FILE RECORD
      CALL NTRAN(SFILE,1,ISIZE,IYEAR,ISTAT,22)   @WRITE END-OF-FILE MARK
      IF(ISTAT.GT.0) GO TO 50               @BAD I/O STATUS? YES...
      WRITE(NWRT,10)                        @WRITE ERROR MESSAGE
   10 FORMAT('0***NTRAN WRITE ERROR, END-OF-FILE RECORD')
      STOP                                  @TERMINATE THE PROGRAM
C
   50 DAYSST(1) = 0.0                       @SET INITIAL TEST DAY
      DAYSST(2) = PRT1(1)                   @SET FINAL TEST DAY
      DO 60 I=1,2                           @INITIALIZE MIN./MAX. VALUES FOR:
      TSAF(I) = PRT2(2)                     @ SOLAR ARRAY TEMPERATURE
      VSA(I) = PRT2(4)                      @ SOLAR ARRAY OPERATING VOLTAGE
      XISA(I) = PRT2(5)                     @ SOLAR ARRAY OPERATING CURRENT
      XIZ(I) = PRT2(10)                     @ SHUNT LIMITER OPERATING CURRENT
      TTES3(I) = PRT3(2)                    @ BATTERY TEMPERATURE
      BCUR(I) = PRT3(6)                     @ BATTERY OPERATING CURRENT
      VBAT(I) = PRT3(7)                     @ BATTERY OPERATING VOLTAGE
   60 CB(I) = PRT3(9)                       @ BATTERY CAPACITY
C
C*****PRINT UNREGULATED BUS SUMMARY TABLE
C
  100 NT = 1                                @SET TABLE NUMBER = 1
      CALL INIT_                            @INITIALIZE INPUT/OUTPUT
  110 CALL READPA(%200)                     @READ PERFORMANCE ANALYSIS DATA
      IF(LNCNT.GE.54) GO TO 150             @NEW PAGE?
      PGCNT = PGCNT + 1                     @YES. INCREMENT PAGE COUNTER
      WRITE(NWRT,120)  NT,PGCNT             @PRINT TITLE AND HEADERS
  120 FORMAT('1',T39,'NAVIGATION AID POWER SYSTEM PERFORMANCE ANALYSIS',
     .    T117,'PAGE ',I2,'-',I4//,
     .    1X,T46,'TABLE 1:  UNREGULATED BUS SUMMARY'////)
      WRITE(NWRT,130)
  130 FORMAT(1X,T28,'POWER',/,
     2    1X,T19,'TIME    SYSTEM    POWER SOURCE GROUP   ENERGY ',
     2    'STORAGE GROUP   POWER CONDITIONING',/,
     3    '  DATE OF TEST    SINCE  OPERATING',T79,'AND ',
     3    'DISTRIBUTION GROUP',/,
     4    1X,T19,'START   VOLTAGE',4X,3('POWER     CURRENT',5X)/,
     5    ' YEAR:DAY:HOUR    (DAYS)  (VOLTS)   ',
     5    3('(WATTS)   (AMPERES)   ')/,
     6    1X,99('-'))
      LNCNT = 12                            @INITIALIZE LINE COUNTER
  150 WRITE(NWRT,160)  IYEAR,IDAY,TIME,(PRT1(I),I=1,8)     @WRITE DATA
  160 FORMAT(1X,I4,':',I3,':',F5.2,2(2X,F7.2),6(2X,E9.4))
      LNCNT = LNCNT + 1                     @INCREMENT LINE COUNTER
```

```
C                                       DETERMINE...
      TSAF(1) = AMIN1(TSAF(1),PRT2(2))  @MINIMUM S/A TEMPERATURE
      TSAF(2) = AMAX1(TSAF(2),PRT2(2))  @MAXIMUM S/A TEMPERATURE
      VSA(1) = AMIN1(VSA(1),PRT2(4))    @MINIMUM S/A OPERATING VOLTAGE
      VSA(2) = AMAX1(VSA(2),PRT2(4))    @MAXIMUM S/A OPERATING VOLTAGE
      XISA(1) = AMIN1(XISA(1),PRT2(5))  @MINIMUM S/A OPERATING CURRENT
      XISA(2) = AMAX1(XISA(2),PRT2(5))  @MAXIMUM S/A OPERATING CURRENT
      XIZ(1) = AMIN1(XIZ(1),PRT2(10))   @MINIMUM S/L OPERATING CURRENT
      XIZ(2) = AMAX1(XIZ(2),PRT2(10))   @MAXIMUM S/L OPERATING CURRENT
      TTESG(1) = AMIN1(TTESG(1),PRT3(2))     @MINIMUM BATTERY TEMPERATURE
      TTESG(2) = AMAX1(TTESG(2),PRT3(2))     @MAXIMUM BATTERY TEMPERATURE
      BCUR(1) = AMIN1(BCUR(1),PRT3(6))       @MIN. BATT OPERATING CURRENT
      BCUR(2) = AMAX1(BCUR(2),PRT3(6))       @MAX. BATT OPERATING CURRENT
      VBAT(1) = AMIN1(VBAT(1),PRT3(7))       @MIN. BATT OPERATING VOLTAGE
      VBAT(2) = AMAX1(VBAT(2),PRT3(7))       @MAX. BATT OPERATING VOLTAGE
      CB(1) = AMIN1(CB(1),PRT3(9))      @MINIMUM BATTERY CAPACITY
      CB(2) = AMAX1(CB(2),PRT3(9))      @MAXIMUM BATTERY CAPACITY
      GO TO 110                         @GET NEXT RECORD
C
C.....PRINT POWER SOURCE GROUP SUMMARY TABLE
C
  200 NT = 2                            @SET TABLE NUMBER = 2
      L = 1                             @INITIALIZE PLOT LINE TYPE = NONE
      CALL INITL                        @INITIALIZE INPUT/OUTPUT
      IF(X_N.LE.0.0) GO TO 205          @SUMMARY PLOTS DESIRED? YES...
      CALL PSCALE(DAYSST,XLN,2,1)       @SET TEST DAY PLOT SCALE
      CALL PSCALE(TSAF,Y_N,2,1)         @SET S/A TEMPERATURE SCALE
      CALL PSCALE(VSA,YLN,2,1)          @SET S/A OPERATING VOLTAGE SCALE
      CALL PSCALE(XISA,Y_N,2,1)         @SET S/A OPERATING CURRENT SCALE
      CALL PSCALE(XIZ,YLN,2,1)          @SET S/L OPERATING CURRENT SCALE
      CALL PLOT(10.0,0.0,-3)            @ESTABLISH NEW PLOT ORIGIN
      CALL AXIS(0.0,0.0,'DAYS SINCE START OF TEST',-24,XLN,0.0,
     .          DAYSST(3),DAYSST(4))
      CALL AXIS(0.0,0.0,'SOLAR ARRAY TEMPERATURE',23,YLN,90.0,
     .          TSAF(3),TSAF(4))
      CALL AXIS(-1.0,0.0,'SOLAR ARRAY OPERATING VOLTAGE',29,YLN,90.0,
     .          VSA(3),VSA(4))
      CALL AXIS(-2.0,0.0,'SOLAR ARRAY OPERATING CURRENT',29,YLN,90.0,
     .          XISA(3),XISA(4))
      CALL AXIS(-3.0,0.0,'SHUNT LIMITER OPERATING CURRENT',31,YLN,90.0,
     .          XIZ(3),XIZ(4))
      CALL SYMBOL(XLN-3.3,YLN+0.6,.14,0.0,-1)
      CALL SYMBOL(XLN-3.3,YLN+0.6,.14,'= SOLAR ARRAY TEMPERATURE CURVE',
     .            0.0,31)
      CALL SYMBOL(XLN-3.3,YLN+0.3,.14,1.0,-1)
      CALL SYMBOL(XLN-3.0,YLN+0.3,.14,'= SOLAR ARRAY VOLTAGE CURVE',0.0,27)
      CALL SYMBOL(XLN-3.3,YLN,.14,3.0,-1)
      CALL SYMBOL(XLN-3.0,YLN,.14,'= SOLAR ARRAY CURRENT CURVE',0.0,27)
      CALL SYMBOL(XLN-3.3,YLN-0.3,.14,10.0,-1)
      CALL SYMBOL(XLN-3.0,YLN-0.3,.14,'= SHUNT LIMITER CURRENT CURVE',
     .            0.0,29)
C
  205 CALL READPA(&210)                 @READ PERFORMANCE ANALYSIS DATA
      DAYSST(1) = PRT2(1)               @MOVE INPUT DATA TO PLOT ARRAYS
      TSAF(1) = PRT2(2)
      VSA(1) = PRT2(4)
      XISA(1) = PRT2(5)
```

```
      XIZ(1) = PRT2(10)
      CALL INIT_                          @RE-INITIALIZE INPUT/OUTPUT
C
  210 CALL READPA(&270)                   @READ PERFORMANCE ANALYSIS DATA
      IF(LNCNT.LE.54) GO TO 250           @NEW PAGE?
      PGCNT = PGCNT + 1                   @YES. INCREMENT PAGE COUNTER
      WRITE(NWRT,220)  NT,PGCNT           @PRINT TITLE AND HEADERS
  220 FORMAT('1',T39,'NAVIGATION AID POWER SYSTEM PERFORMANCE ANALYSIS',
     .    T117,'PA',J2,'-PAGE  ',J4//,
     .    1X,T45,'TABLE 2:  POWER SOURCE GROUP SUMMARY'////)
      WRITE(NWRT,230)
  230 FORMAT(1X,T29,'POWER'/,
     2    1X,T19,'TIME      SOURCE   INCIDENT',4X,18('*'),
     2    ' SOLAR ARRAY ',18('*'),5X,'****** SHUNT LIMITER ******'/,
     3    '  DATE OF TEST    SINCE    GROUP     SOLAR',
     3    T80,'MAXIMUM     POWER'/,
     4    1X,T19,'START    TEMP.   RADIATION   VOLTAGE    CURRENT',
     4    5X,2('POWER',6X),'MARGIN    VOLTAGE    CURRENT     POWER'/,
     5    ' YEAR:DAY:HOUR    (DAYS) (DEG. F) (WATTS/SQ.M) (VOLTS)',
     5    3X,'(AMPERES)   ',3('(WATTS)',4X),'(VOLTS)   (AMPERES)',
     5    3X,'(WATTS)'/,
     6    1X,131('-'))
      LNCNT = 12                          @INITIALIZE LINE COUNTER
  250 WRITE(NWRT,260)  IYEAR,IDAY,TIME,(PRT2(I),I=1,11)    @WRITE DATA
  260 FORMAT(1X,I4,':',I3,':',F5.2,2(2X,F7.2),9(2X,E9.4))
      LNCNT = LNCNT + 1                   @INCREMENT LINE COUNTER
C
      DAYSST(2) = PRT2(1)                 @MOVE INPUT DATA TO PLOT ARRAYS
      TSAF(2) = PRT2(2)
      VSA(2) = PRT2(4)
      XISA(2) = PRT2(5)
      XIZ(2) = PRT2(10)
      IF(XLN._E.0.0) GO TO 265            @SUMMARY PLOTS DESIRED? YES...
      CALL LINE(DAYSST,TSAF,2,1,L,0)      @PLOT ARRAY PAIRS VS. DAY PAIR
      CALL LINE(DAYSST,VSA,2,1,L,1)
      CALL LINE(DAYSST,XISA,2,1,L,3)
      CALL LINE(DAYSST,XIZ,2,1,L,13)
  265 DAYSST(1) = DAYSST(2)               @STORE LAST INPUT DATA POINTS
      TSAF(1) = TSAF(2)
      VSA(1) = VSA(2)
      XISA(1) = XISA(2)
      XIZ(1) = XIZ(2)
      L = 1                               @REDEFINE PLOT LINE TYPE
      GO TO 210                           @GET NEXT RECORD
  270 IF(XPLT._E.0.0) GO TO 300           @SUMMARY PLOTS DESIRED?
      CALL PLOT(XLN+10.0,0.0,-3)          @YES. ESTABLISH NEW PLOT ORIGIN
C
C*****PRINT ENERGY STORAGE UNIT SUMMARY TABLES
C
  300 DAYSST(1) = 0.0                     @RE-SET INITIAL TEST DAY
      IF(XLN._E.0.0) GO TO 303            @SUMMARY PLOTS DESIRED? YES...
      CALL PSCALE(DAYSST,XLN,2,1)         @SET TEST DAY PLOT SCALE
      CALL PSCALE(TTES3,YLN,2,1)          @SET BATTERY TEMPERATURE SCALE
      CALL PSCALE(BCUR,YLN,2,1)           @SET BATTERY CURRENT SCALE
      CALL PSCALE(VBAT,YLN,2,1)           @SET BATTERY VOLTAGE SCALE
      CALL PSCALE(CB,YLN,2,1)             @SET BATTERY CAPACITY SCALE
C
```

```
  303 NT = 3                                  @SET TABLE NUMBER = 3
      L = 3                                   @INITIALIZE PLOT LINE TYPE = NONE
      CALL INITL                              @INITIALIZE INPUT/OUTPUT
      IF(XLN.LE.0.0) GO TO 305                @SUMMARY PLOTS DESIRED? YES...
      CALL AXIS(3.3,0.0,'DAYS SINCE START OF TEST',-24,XLN,0.0,
     .          DAYSST(3),DAYSST(4))
      CALL AXIS(3.3,0.0,'BATTERY TEMPERATURE',19,YLN,90.0,
     .          TTESG(3),TTESG(4))
      CALL AXIS(-1.0,0.0,'BATTERY OPERATING CURRENT',25,YLN,90.0,
     .          BCUR(3),BCUR(4))
      CALL AXIS(-2.0,0.0,'BATTERY OPERATING VOLTAGE',25,YLN,30.0,
     .          VBAT(3),VBAT(4))
      CALL AXIS(-3.0,0.0,'BATTERY CAPACITY',16,YLN,90.0,CB(3),CB(4))
      CALL SYMBOL(XLN-3.3,YLN+0.6,H,1,A,-1)
      CALL SYMBOL(XLN-3.0,YLN+0.6,H,'= BATTERY TEMPERATURE CURVE',A,27)
      CALL SYMBOL(XLN-3.3,YLN+0.3,H,4,A,-1)
      CALL SYMBOL(XLN-3.0,YLN+0.3,H,'= BATTERY CURRENT CURVE',A,23)
      CALL SYMBOL(XLN-3.3,YLN,H,5,A,-1)
      CALL SYMBOL(XLN-3.0,YLN,H,'= BATTERY VOLTAGE CURVE',A,23)
      CALL SYMBOL(XLN-3.3,YLN-0.3,H,24,A,-1)
      CALL SYMBOL(XLN-3.0,YLN-0.3,H,'= BATTERY CAPACITY CURVE',A,24)
C
  305 CALL READPA($400)                       @READ PERFORMANCE ANALYSIS DATA
      DAYSST(1) = PRT3(1)                     @MOVE INPUT DATA TO PLOT ARRAYS
      TTESG(1) = PRT3(2)
      BCUR(1) = PRT3(6)
      VBAT(1) = PRT3(7)
      CB(1) = PRT3(9)
      CALL INITL                              @RE-INITIALIZE INPUT/OUTPUT
C
  310 CALL READPA($400)                       @READ PERFORMANCE ANALYSIS DATA
      IF(LNCNT.LE.54) GO TO 350               @NEW PAGE?
      PGCNT = PGCNT + 1                       @YES. INCREMENT PAGE COUNTER
      WRITE(NWRT,320)  NT,PGCNT,NT,NBATT      @PRINT TITLE AND HEADERS
  320 FORMAT('1',T39,'NAVIGATION AID POWER SYSTEM PERFORMANCE ANALYSIS',
     .    T117,'PA',J2,'-PAGE  ',J4//,
     .    1X,T36,'TABLE ',J2,':  ENERGY STORAGE UNIT SUMMARY, ',
     .    I2,' BATTERIES'////)
      WRITE(NWRT,330)
  330 FORMAT(1X,T28,'ENERGY'/,
     2    1X,T19,'TIME    STORAGE  ENERGY STORAGE UNIT',5X,30('*'),
     2    ' BATTERY ',30('*')/,
     3    '  DATE OF TEST    SINCE    GROUP',T125,'FREEZING'/,
     4    1X,T19,'START    TEMP.      POWER      VOLTAGE      POWER',
     4    5X,'CURRENT    VOLTAGE    STATE OF   CAPACITY   SPECIFIC',
     4    5X,'TEMP.'/,
     5    ' YEAR:DAY:HOUR    (DAYS) (DEG. F)   (WATTS)     (VOLTS)',
     5    4X,'(WATTS)   (AMPERES)   (VOLTS)     CHARGE  (AMP-HOURS)',
     5    2X,'GRAVITY    (DEG. F)'/,
     6    1X,131('-'))
      LNCNT = 12                              @INITIALIZE LINE COUNTER
  350 WRITE(NWRT,360)  IYEAR,IDAY,TIME,(PRT3(I),I=1,11)    @WRITE DATA
  360 FORMAT(1X,I4,':',I3,':',F5.2,2(2X,F7.2),9(2X,E9.4))
      LNCNT = LNCNT + 1                       @INCREMENT LINE COUNTER
C
      DAYSST(2) = PRT3(1)                     @MOVE INPUT DATA TO PLOT ARRAYS
      TTESG(2) = PRT3(2)
```

```
      BCUR(2) = PRT3(5)
      VBAT(2) = PRT3(7)
      CB(2) = PRT3(9)
      IF(XLN.LE.0.0) GO TO 365          @SUMMARY PLOTS DESIRED? YES...
      CALL LINE(DAYSST,TTESG,2,1,L,1)   @PLOT ARRAY PAIRS VS. DAY PAIR
      CALL LINE(DAYSST,BCUR,2,1,L,4)
      CALL LINE(DAYSST,VBAT,2,1,L,5)
      CALL LINE(DAYSST,CB,2,1,L,24)
  365 DAYSST(1) = DAYSST(2)             @STORE LAST INPUT DATA POINTS
      TTESG(1) = TTESG(2)
      BCUR(1) = BCUR(2)
      VBAT(1) = VBAT(2)
      CB(1) = CB(2)
      L = 1                             @REDEFINE PLOT LINE TYPE
      GO TO 310                         @GET NEXT RECORD
  400 IF(XPLT.LE.0.0) GO TO 500         @SUMMARY PLOTS DESIRED?
      CALL PLOT(XLN+10.0,0.0,-3)        @YES.
  500 CONTINUE
C
      RETURN
C
C*********************  INTERVAL SUBROUTINES  **********************
C
      SUBROUTINE  INITL
      LNCNT = 1000                      @INITIALIZE LINE COUNTER
      PGCNT = 0                         @INITIALIZE PAGE COUNTER
      CALL NTRAN(SFILE,10,22)           @REWIND PERFORMANCE ANALYSIS FILE
      RETURN
C
C
      SUBROUTINE  READPA($)
      CALL NTRAN(SFILE,2,ISIZE,IYEAR,ISTAT,22)   @READ A RECORD
      IF(ISTAT.EQ.-2)  RETURN 1         @END OF FILE?
      IF(IYEAR.EQ.9999)  RETURN 1
      IF(ISTAT.GE.1)  RETURN            @NO. BAD I/O STATUS?
      WRITE(NWRT,800)  IYEAR,IDAY,TIME      @YES.
  800 FORMAT('0***NTRAN READ ERROR AT RECORD AFTER DATE = ',
     .    I4,':',I3,':',F5.2)
      STOP
C
      END
```

READTAPE

```
      SUBROUTINE  RDTAPE
C      OBTAINS AMBIENT TEMPERATURE, SOLAR INSOLATION, AND WIND
C      VELOCITY FROM NOAA OR STATISTICAL WEATHER TAPE INPUT
C     ENTRY  TMNMX
C      OBTAINS MINIMUM AND MAXIMUM AMBIENT TEMPERATURES OVER A
C      ONE-YEAR PERIOD OF TIME DEFINED BY THE VALUE OF 'ITAPE'
C
      INCLUDE  ALLCMN,LIST
      INCLUDE  COMON,LIST
      COMMON/STATS/ ZALPHA,ZPRCNT
      INTEGER  LOC(4)/1000,-1000,2000,-2000/
      INTEGER  SECEND, SECNOW, SECNXT, SECONE, SECTOR
      LOGICAL  FIRST/.TRUE./, STATAP, SUNRIZ, SUNSET
      REAL  OUTBUF(28,3)
      REAL  DB(24), SOLAR(24), WV(24), X(24), X1(26), Y1(26)
      DATA  X/1.0,2.0,3.0,4.0,5.0,6.0,7.0,8.0,9.0,10.0,11.0,12.0,13.0,
     .        14.0,15.0,16.0,17.0,18.0,19.0,20.0,21.0,22.0,23.0,24.0/
      REAL  ZTABLE(50,2)/0.5000,0.5398,0.5793,0.5987,0.6179,0.6368,
     .                   0.6554,0.6736,0.6915,0.7088,0.7258,0.7422,
     .                   0.7580,0.7734,0.7881,0.8023,0.8159,0.8289,
     .                   0.8413,0.8531,0.8643,0.8749,0.8849,0.8944,
     .                   0.9032,0.9115,0.9192,0.9265,0.9332,0.9394,
     .                   0.9452,0.9505,0.9554,0.9599,0.9641,0.9678,
     .                   0.9713,0.9773,0.9821,0.9861,0.9893,0.9918,
     .                   0.9938,0.9960,0.9970,0.9978,0.9987,0.9990,
     .                   1.0000,1.0000,
     .                   0.00,0.10,0.20,0.25,0.30,0.35,
     .                   0.40,0.45,0.50,0.55,0.60,0.65,
     .                   0.70,0.75,0.80,0.85,0.90,0.95,
     .                   1.00,1.05,1.10,1.15,1.20,1.25,
     .                   1.30,1.35,1.40,1.45,1.50,1.55,
     .                   1.60,1.65,1.70,1.75,1.80,1.85,
     .                   1.90,2.00,2.10,2.20,2.30,2.40,
     .                   2.50,2.65,2.75,2.85,3.00,3.60,
     .                   4.00,1000.0/
      EQUIVALENCE  (OUTBUF(2,1),IHEAD2), (OUTBUF(5,1),DB(1))
      EQUIVALENCE  (OUTBUF(5,2),WV(1)), (OUTBUF(5,3),SOLAR(1)).
C
      IF(DATEM.LE.0.0) FIRST = .TRUE.  @SET INITIAL CALL FLAG,
      IF(.NOT.FIRST) GO TO 30          @INITIAL CALL TO READTAPE?
      FIRST = .FALSE.                  @YES. RESET CALL FLAG
      CALL NTRAN(MERGE,10,22)          @REWIND DATA FILE
      SECTOR = 0                       @INITIALIZE SECTOR NUMBER
      OLDATE = DATE                    @STORE REQUESTED DATE
      IF(ITAPE.GE.0) GO TO 10          @STATISTICAL DATA FILE?
      STATAP = .TRUE.                  @YES. SET STAT PARAMETERS
      SECEND = 366
      SECNOW = 1
      SECONE = 1
      DAY = 1.0
      IYR = 0
      GO TO 50
   10 STATAP = .FALSE.                 @NO. SET NOAA PARAMETERS
      CALL FILCHK($820,$800)
      IYR = 1000 * IFIX(ITAPE / 1000)
      DAY = FLOAT(ITAPE - IYR)
```

```
      GO TO 50
C
   30 IF(DATE.NE.OLDATE) GO TO 50       @SAME DATE AS LAST REQUEST?
      CALL SLUP(TIMEH,QDT,FDOT,X1,Y1,26,1)  @YES. GET SOLAR INSOLATION
      CALL SLUP(TIMEH,TTAMB,FDOT,X,DB,24,1) @GET AMBIENT TEMPERATURE
      CALL SLUP(TIMEH,WIND,FDOT,X,WV,24,1)  @GET WIND VELOCITY
      RETURN
C
   50 OLDATE = DATE                     @NO. STORE REQUESTED DATE
     * SECNXT = IYR + IFIX(DATE)        @SET RECORD POINTER
      IF(DATE.LT.DAY) SECNXT = SECNXT + 1000     @SET POINTR TO NXT YEAR
      CALL POINTR($800,$820)            @READ NEXT INPUT DATA RECORD
      CALL SLUP(TIMEH,TTAMB,FDOT,X,DB,24,1) @GET AMBIENT TEMPERATURE
      CALL SLUP(TIMEH,WIND,FDOT,X,WV,24,1)  @GET WIND VELOCITY
      IF(SOLAR(12).GE.0.0) GO TO 70     @VALID SOLAR INSOLATION DATA?
      NXTSEC = SECNXT                   @NO. SAVE RECORD POINTER
      IF(STATAP) GO TO 840              @IN STATISTICAL DATA FILE?
      DO 60 I=1,4                       @NO. SCAN SURROUNDING 4 YEARS FOR
      SECNXT = NXTSEC + LOC(I)          @ VALID SOLAR INSOLATION DATA
      CALL POINTR($60,$820)
      IF(SOLAR(12).LT.0.0) GO TO 60
      WRITE(NWRT,55) NXTSEC,SECNXT
   55 FORMAT('0***SOLAR INSOLATION DATA NOT AVAILABLE FOR REQUESTED ',
     .       'DATE (',I5,')! OBTAINED FROM ',I5)
      GO TO 65
   60 CONTINUE
      GO TO 840                         @NO VALID INSOLATION DATA FOUND
   65 SECNXT = NXTSEC                   @RESET RECORD POINTER
   70 CALL T26HRS                       @BUILD SOLAR INSOLATION TABLES
      CALL SLUP(TIMEH,QDT,FDOT,X1,Y1,26,1)  @GET SOLAR INSOLATION
C
      IF(SECNOW.LT.SECEND) GO TO 80     @'DATE' BEYOND END OF YEAR?
      SECTOR = 0                        @YES. RE-INITIALIZE SECTOR NUMBER
      SECNOW = SECONE                   @RE-INITIALIZE RECORD NUMBER
      CALL NTRAN(MERGE,10,22)           @REWIND DATA FILE
   80 IYEAR = 1000 * IFIX(SECNOW / 1000)    @SET REQUESTED YEAR
      IDAY = SECNOW - IYEAR             @SET REQUESTED DAY
      SECTOR = SECTOR + 3               @INCREMENT SECTOR NUMBER
      SECNOW = SECNOW + 1               @INCREMENT POINTER
      IF(IDAY.GE.366) SECNOW = IDAY - 365 + 1000 * (IYEAR + 1)
      RETURN
C
      ENTRY  TMNMX
C
  100 CALL NTRAN(MERGE,10,22)           @REWIND DATA FILE
      CCM = 1.0                         @SET CLOUD-COVER MODIFIER
      SECTOR = 0                        @INITIALIZE SECTOR NUMBER
      IF(ITAPE.GE.0) GO TO 110          @STATISTICAL DATA FILE?
      STATAP = .TRUE.                   @YES. SET STAT PARAMETERS
      SECEND = 366
      SECNOW = 1
      SECNXT = 1
      SECONE = 1
      CALL NTRAN(MERGE,2,84,OUTBUF,ISTAT,22)     @READ 1ST 'STAT' REC
      IF(ISTAT.LT.0)  GO TO 820         @BAD I/O STATUS? NO...
      Z = AMAX1(ZALPHA,1.0-ZALPHA)
      CALL SLUP(Z,ZA,FDOT,ZTABLE(1,1),ZTABLE(1,2),50,1)
```

```
      IF(ZALPHA.GT.0.50)  ZA = -ZA
      Z = AMAX1(ZPRCNT,1.0-ZPRCNT)
      CALL SLOP(Z,ZP,FDOT,ZTABLE(1,1),ZTABLE(1,2),50,1)
      IF(ZPRCNT.LT.0.50)  ZP = -ZP
      AL = 1.0 - ZA**2.0 / (2.0 * (OUTBUF(1,3) - 1.0))
      BL = ZP**2.0 - ZA**2.0 / OUTBUF(1,3)
      CL = (ZP + SQRT(ZP**2.0 - AL * BL)) / AL
      TSAF = OUTBUF(1,2) - CL * OUTBUF(2,2)       @COMPUTE MINIMUM TEMP.
      TTAMB = OUTBUF(3,2) + CL * OUTBUF(4,2)      @COMPUTE MAXIMUM TEMP.
      RETURN
C
  110 STATAP = .FALSE.                            @NO. SET NOAA PARAMETERS
      CALL FILCHK($820,$800)
      SECNXT = ITAPE
C
  120 CALL POINTR($800,$820)                      @READ NEXT INPUT DATA RECORD
      TSAF = DB(1)                                @INITIALIZE MINIMUM TEMPERATURE
      TTAMB = DB(1)                               @INITIALIZE MAXIMUM TEMPERATURE
      DO 130 J=2,24
      TSAF = AMIN1(TSAF,DB(J))
  130 TTAMB = AMAX1(TTAMB,DB(J))
      DO 150 I=2,366                              @COMPUTE YEARLY MINIMUM AND
      CALL NTRAN(MERGE,2,84,OUTBUF,ISTAT,22)      @ MAXIMUM TEMPERATURES
      IF(ISTAT.LT.0) GO TO 820
      IF(IHEAD2.GE.99999) GO TO 800
      DO 140 J=1,24
      TSAF = AMIN1(TSAF,DB(J))
  140 TTAMB = AMAX1(TTAMB,DB(J))
  150 CONTINUE
      RETURN
C
C*****ERROR MESSAGE EXITS
C
  800 WRITE(NWRT,810) SECNXT,SECONE,SECEND
  810 FORMAT('0***REQUESTED DATE (',I5,') IS OUT OF RANGE:',2(2X,I5))
      STOP
  820 WRITE(NWRT,830) SECNXT
  830 FORMAT('0***NTRAN READ ERROR AT REQUESTED DATE = ',I5)
      STOP
  840 WRITE(NWRT,850) NXTSEC
  850 FORMAT('0***NO SOLAR INSOLATION DATA AVAILABLE FOR DATE = ',I5/,
     .       '    PROGRAM TERMINATED')
      STOP
C
C**********************  INTERNAL SUBROUTINES  **************************
C
      SUBROUTINE  FILCHK($,$)
C      READS FIRST DATA FILE RECORD AND STORES FILE PARAMETERS
C
      CALL NTRAN(MERGE,2,84,OUTBUF,ISTAT,22)      @READ FIRST RECORD
      IF(ISTAT.LT.0)  RETURN 1                    @BAD I/O STATUS? NO...
      SECONE = OUTBUF(2,1)                        @SET FIRST RECORD POINTER
      SECEND = OUTBUF(4,1)                        @SET LAST RECORD POINTER
      SECNOW = SECONE                             @SET PRESENT RECORD POINTER
      IF(ITAPE.LT.SECONE)  RETURN 2               @REQUESTED DATE OUTSIDE LIMITS?
      IF(SECEND-ITAPE.LT.365)  RETURN 2
      CALL NTRAN(MERGE,10,22)                     @NO. REWIND INPUT DATA FILE
```

Preceding Page BLANK - FILMED


```
      RETURN
C
C
      SUBROUTINE  POINTR(S,S)
C      READS REQUESTED DATA FILE INPUT RECORD
C
      LYR = 0                              @INITIALIZE YEARS-TO-SKIP
      CALL NTRAN(MERGE,20,ISTAT)           @WAIT AND UNSTACK
      IF(ISTAT.LT.0)  RETURN 2             @BAD I/O STATUS? NO...
      IF(SECNXT.LT.SECONE)  RETURN 1       @IS REQUESTED DATE OUTSIDE THE
      IF(SECNXT.GT.SECEND)  RETURN 1       @ RANGE OF THE DATA FILE? NO...
      ISIGN = 1                            @INITIALZE DATE FLAG
      IF(SECNXT-SECNOW) 10,50,20           @REQUESTED DATE.LT/EQ/GT.PRESENT
C
   10 ISIGN = -1                           @SET DATE FLAG PRIOR TO PRESENT
   20 IYRS = SECNXT / 1000                 @SET REQUESTED YEAR
      LDAY = SECNXT - 1000 * IYRS          @SET REQUESTED DAY
      IYR1 = SECNOW / 1000                 @SET PRESENT YEAR
      LDY = SECNOW - 1000 * IYR1           @SET PRESENT DAY
      IYRS = ABS(IYRS - IYR1)
      LDAY = ISIGN * (LDAY - LDY)          @REDEFINE REQUESTED DAY
      IF(IYRS.EQ.0) LDAY = ABS(LDAY)
      IF(IYRS.NE.0) LDAY = ABS(366 + LDAY)
      IF(IYRS.NE.0) LYR = IYRS - 1         @REDEFINE YEARS-TO-SKIP
C
   30 NEXT = 3 * ISIGN * (366 * LYR + LDAY)      @REDEFINE POINTER
      SECTOR = SECTOR + NEXT               @SET NEW SECTOR NUMBER
      SECNOW = SECNXT                      @REDEFINE PRESENT DAY
      CALL NTRAN(MERGE,6,NEXT)             @POSITION TO REQUESTED RECORD
   50 CALL NTRAN(MERGE,2,84,OUTBUF,ISTAT,22)     @READ NEXT 3 SECTORS
      IF(ISTAT.LT.0)  RETURN 2             @BAD I/O STATUS? NO...
      IF(IHEAD2.NE.SECNXT) GO TO 100       @FILE DATE SAME AS REQUESTED?
      RETURN                               @YES.
C
  100 IF(SECNXT.LT.1000)  RETURN           @STATISTICAL FILE?
      WRITE(NWRT,120) SECNXT,IHEAD2        @NO. WRITE ERROR MESSAGE
  120 FORMAT('0***REQUESTED RECORD NOT FOUND IN FILE:',2(2X,I5))
      STOP
C
C
      SUBROUTINE  T26HRS
C      ADDS SUNRISE AND SUNSET TIMES TO SOLAR INSOLATION ARRAY
C
      NSOL = 1                             @INITIALIZE TIME INDEX
      SUNRIZ = .FALSE.                     @INITIALIZE SUNRISE FLAG
      SUNSET = .FALSE.                     @INITIALIZE SUNSET FLAG
      DO 50 I=1,24
      TIME = FLOAT(I)                      @SET TIME OF DAY
      IF(SUNRIZ) GO TO 20                  @AFTER SUNRISE?
      IF(SRT.GT.TIME) GO TO 10
      IF(SRT.EQ.TIME) TIME = TIME + 0.00001
      SUNRIZ = .TRUE.                      @YES. RESET SUNRISE FLAG
      X1(NSOL) = SRT                       @INSERT SUNRISE TIME
      Y1(NSOL) = 0.0                       @INSERT SUNRISE SOLAR INSOLATION
      NSOL = NSOL + 1                      @INCREMENT INDEX NUMBER
   10 X1(NSOL) = TIME                      @INSERT TIME OF DAY
      Y1(NSOL) = 0.0                       @INSERT SOLAR INSOLATION
```

```
      NSOL = NSOL + 1                          @INCREMENT INDEX NUMBER
      GO TO 50
   20 IF(SUNSET) GO TO 40                      @AFTER SUNSET?
      IF(SST.GT.TIME) GO TO 30
      IF(SST.EQ.TIME) TIME = TIME + 0.00001
      SUNSET = .TRUE.                          @YES. RESET SUNSET FLAG
      X1(NSOL) = SST                           @INSERT SUNSET TIME
      Y1(NSOL) = 0.0                           @INSERT SUNSET SOLAR INSOLATIO
      NSOL = NSOL + 1                          @INCREMENT INDEX NUMBER
      GO TO 40
   30 X1(NSOL) = TIME                          @INSERT TIME OF DAY
      Y1(NSOL) = SOLAR(I)                      @INSERT SOLAR INSOLATION
      NSOL = NSOL + 1                          @INCREMENT INDEX NUMBER
      GO TO 50
   40 X1(NSOL) = TIME                          @INSERT TIME OF DAY
      Y1(NSOL) = 0.0                           @INSERT SOLAR INSOLATION
      NSOL = NSOL + 1                          @INCREMENT INDEX NUMBER
   50 CONTINUE
      RETURN
C
      END
```

SLUP

```
      SUBROUTINE  SLUP(X,Y,YDOT,XS,YS,LN,MORD)
C
C     IDENTIFICATION...
C        SINGLE PRECISION SUBROUTINE  SLUP/LAGRANGIAN INTERPOLATION
C        BUD POULSON (JPL)
C        M. MILANE (JPL)
C        FORTRAN IV
C
C     PURPOSE...
C        SINGLE LOOK UP AND LAGRANGIAN INTERPOLATION.
C        GIVEN A VALUE OF AN INDEPENDENT VARIABLE, X, FIND F(X) FROM
C          A GIVEN TABULATED FUNCTION OF X(I) VS. Y(I), AND OPTIONALLY,
C          THE FIRST DERIVATIVE.
C
C     RESTRICTIONS...
C        NO EXPLICIT RESTRICTIONS, I.E.,
C          ANY ORDER INTERPOLATION
C          ANY SIZE TABLE
C        THE INDEPENDENT VARIABLE MAY BE MONOTONICALLY INCREASING OR
C          DECREASING
C
C     METHOD...
C        LAGRANGE'S FORMULA IS USED WITH EXTRAPOLATION FOR POINTS
C        OUTSIDE THE TABLE.  A BINARY SEARCH IS USED, WHICH MAY BE THE
C          SOLE PURPOSE OF USING SLUP.
C
C     CALLING SEQUENCE...
C        X, INDEPENDENT VARIABLE  (INPUT)
C        Y, INTERPOLATED VALUE OF DEPENDENT VARIABLE, Y(X)  (OUTPUT)
C        YDOT, CALCULATED FIRST DERIVATIVE, Y'(X)  (OUTPUT)
C        XS, TABULATED INDEPENDENT VARIABLE   (INPUT)
C        YS, TABULATED DEPENDENT VARIABLE   (INPUT)
C        LN, NUMBER OF TABULATED POINTS  (INPUT)
C          IF ONE DESIRES TO FIND K IN THE SEQUENCE X(1),X(2),X(3),
C             ....,X(K),X(K+1),....,X(LN) WHEN THE GIVEN X IS BETWEEN
C             X(K) AND X(K+1), THEN MAKE LN NEGATIVE AND THE VALUE
C             K COMES BACK IN LN.
C        MORD, ORDER OF INTERPOLATION, USING MORD+1 POINTS IN
C          LAGRANGE'S FORMULA  (INPUT),
C          MAKE MORD NEGATIVE IF THE FIRST DERIVIATIVE IS DESIRED.
C
      DATA  ZERO/0.0/, ONE/1.0/
      DIMENSION  XS(2), YS(2)
C
      N=IABS(LN)
C        INSURE THAT THE NUMBER OF POINTS USED TO INTERPOLATE IS LESS
C        THAN OR EQUAL TO THE NUMBER OF DATA POINTS SUPPLIED
      M=MIND(IABS(MORD)+1,N)
C        ASSUME INDEPENDENT VARIABLE IS MONOTONICALLY DECREASING
      K=1
      L=N
C        MONOTONICALLY INCREASING OR DECREASING?
      IF(XS(1)-XS(2)) 5,5,10
C        INDEPENDENT VARIABLE IS MONOTONICALLY INCREASING
    5 K=N
      L=1
```

```
C         NOW BEGIN BINARY SEARCH FOR APPROPRIATE SUB-INTERVAL
   10 J=(K+L)/2
      IF(X-XS(J)) 15,35,20
   15 K=J
      GO TO 25
   20 L=J
   25 IF(IABS(K-L)-1) 10,30,10
   30 J=MAXO(K,L)
C         SET INDICES OF SUB-INTERVAL CONTAINING POINT GIVEN
C         (ALSO INSURE INDICES ARE WITHIN THOSE ALLOWED, I.E., 1 TO N)
   35 NN=MAXO(1,J-M/2)
      MM=MINO(N,NN+M-1)
      NN=MM-M+1
C         IS INTERPOLATION REQUESTED AT THE GIVEN POINT?
      IF(LN) 95,95,40
   40 Y=ZERO
      YDOT=ZERO
      DO 90 J=NN,MM
      T=YS(J)
      K=0
      DO 60 I=NN,MM
      IF(I-J) 45,60,45
   45 R=X-XS(I)
      IF(R) 55,50,55
   50 R=ONE
      K=1
      IF(MORD) 55,55,90
   55 T=T*R/(XS(J)-XS(I))
   60 CONTINUE
      IF(K) 70,70,65
   65 YDOT=YDOT+T
      GO TO 90
   70 Y=Y+T
C         IS THE DERIVATIVE REQUESTED AT THE GIVEN POINT?
      IF(MORD) 75,75,90
   75 DO 85 I=NN,MM
      IF(I-J) 80,85,80
   80 YDOT=YDOT+T/(X-XS(I))
   85 CONTINUE
   90 CONTINUE
      GO TO 100
C         BINARY SEARCH ONLY WAS REQUESTED.  RETURN SUB-INTERVAL INDICES
   95 LN=NN
      MORD=MM
  100 RETURN
C
      END
```

SORT

```
      SUBROUTINE  SORT(X,N,M)
C      SORTS ARRAY X IN ASCENDING ORDER AND REMOVES DUPLICATE ITEMS
C
      REAL  EPS/0.01/, X(N)
C
      M = N - 1
      DO 20 I=1,M                          @SORT ARRAY IN ASCENDING ORDER
      XMIN = X(I)
      IMIN = I
      K = I + 1
      DO 10 J=K,N
      IF(X(J).GT.XMIN) GO TO 10
      XMIN = X(J)
      IMIN = J
   10 CONTINUE
      X(IMIN) = X(I)
   20 X(I) = XMIN
C
      M = 1
      DO 50 I=2,N                          @ELIMINATE DUPLICATE X(I) TERMS
      IF(X(I)-X(M).LT.EPS) GO TO 50
      M = M + 1
      X(M) = X(I)
   50 CONTINUE
C
      J = M + 1
      DO 60 I=J,N                          @ZERO-FILL REMAINING X(I) TERMS
   60 X(I) = 0.0
C
      RETURN
      END
```

TB2GET

```
      FUNCTION TB2GET(X,Y)
C      TABLE LOOKUP AND INTERPOLATION FOR F(X,Y)
      GO TO 10
C
      ENTRY  TB2SET(XTAB,NX,MX,MODEX,YTAB,NY,MY,MODEY,FTAB,IDIMF,IERR)
C
C      C. L. LAWSON, JPL, 1968 JULY 30
C
C
C     (XTAB(I),I=1,NX)    TABLE OF X VALUES
C     NX                  NO. OF ENTRIES IN XTAB
C     MX                  NO. OF X POINTS TO BE USED BY INTERPOLATION
C                         FORMULA. (DEGREE OF FORMULA WILL BE MX-1)
C                         PERMIT 1.LE.MX.LE.10
C     MODEX               =1,2,OR 3 TO SELECT SEARCH METHOD
C                         1 LINEAR SEARCH
C                         2 BINARY SEARCH
C                         3 RECALL SEARCH
C     (YTAB(J),J=1,NY),NY,MY,MODEY       SIMILAR PARAMETERS FOR Y
C
C     ((FTAB(I,J),I=1,NX),J=1,NY)        TABLE OF FUNCTION VALUES
C                                  F(I,J) IS VALUE OF F AT (X(I),Y(J))
C     IDIMF               DIMENSION OF FIRST SUBSCRIPT OF F( , )
C     IERR                =1 GOOD
C                         =2 ARGUMENTS OUT OF RANGE, EXTRAPOLATED VALUE
C                            WILL BE COMPUTED.
      DIMENSION  XTAB(NX), YTAB(NY), FTAB(IDIMF,2)
      DIMENSION  CX(10), CY(10)
      IERR=1
      IXX=0
      JYY=0
      MX1=MX
      MY1=MY
      IF(MX1.LE.0 .OR. MX1.GT.MIN0(10,NX)) IERR=2
      IF(MY1.LE.0 .OR. MY1.GT.MIN0(10,NY)) IERR=2
      IF(IERR.EQ.2)  STOP TB2SET
      RETURN
C
C                                          ENTRY..      TB2GET
C
   10 CONTINUE
      IERR=1
      X1=X
      Y1=Y
      CALL TBSR(XTAB,NX,MX1,MODEX,X1,IBX,IXX,KGOX)
      CALL TBSR(YTAB,NY,MY1,MODEY,Y1,JBY,JYY,KGOY)
      GO TO (20,70,60,60,60),KGOY
C                                          EXACT MATCH ON Y
   20 GO TO (30,50,40,40,40),KGOX
C                                          EXACT MATCH ON X
   30 F1=FTAB(IXX,JYY)
      GO TO 140
C                                          EXACT Y, NON-EXACT X
   40 IERR=2
   50 J1=JYY
      J2=JYY
```

```
      CY(1)=1.0
      GO TO 100
C                                          NON-EXACT Y
   60 IERR=2
   70 J1=JBY
      J2=JBY+MY1-1
      CALL LCOEF(YTAB(JBY),MY1,Y1,CY)
      GO TO (80,100,90,90,90),KGOX
C                                          EXACT X
   80 I1=IXX
      I2=IXX
      CX(1)=1.0
      GO TO 110
C                                          NON-EXACT X
   90 IERR=2
  100 I1=IBX
      I2=IBX+MX1-1
      CALL LCOEF(XTAB(IBX),MX1,X1,CX)
C
C                                          INTERPOLATION LOOP
C
  110 F1=0.
      JC=0
      DO 130 J=J1,J2
      JC=JC+1
      XSUM=0.0
      IC=0
      DO 120 I=I1,I2
      IC=IC+1
  120 XSUM=XSUM+CX(IC)*FTAB(I,J)
  130 F1=F1+CY(JC)*XSUM
  140 TB2GET=F1
      RETURN
      END
```

TBLCOEF

```
      SUBROUTINE LCOEF(XTAB,MPTS,X,C)
C      LAGRANGE INTERPOLATION COEFFICIENTS (TB2GET ROUTINE)
C
C        INPUT..
C     (XTAB(I),I=1,MPTS)      TABLE OF X VALUES, ANY ORDER IS OK
C     MPTS                    NO. OF X POINTS TO BE USED,
C                             1.LE.MPTS.LE.15
C     X                       X VALUE AT WHICH INTERPOLATION IS DESIRED
C
C        OUTPUT..
C     (C(I),I=1,MPTS)         LAGRANGE INTERPOLATION COEFFICIENTS,
C                             VALUE OF A DEPENDENT VARIABLE CAN BE
C                             COMPUTED BY COMPUTING DOT PRODUCT OF
C                             C ARRAY AND DEPENDENT VARAIBLE ARRAY.
C
      DIMENSION  XTAB(15), C(15), D(15)
      M= MIND (15,MPTS)
      DO 10 I=1,M
      C(I)=1.
   10 D(I)=X-XTAB(I)
      IF(M-1) 40,40,20
C
   20 DO 30 I=2,M
      IM1=I-1
      DO 30 J=1,IM1
      U=XTAB(I)-XTAB(J)
      C(I)=C(I)*(D(J)/U)
   30 C(J)=-C(J)*(D(I)/U)
   40 RETURN
      END
```

TBSR

```
      SUBROUTINE TBSR(XTAB,NXTAB,NWANT,MODE,X,IB,IX,KGO)
C      TABLE SEARCH (TBGET ROUTINE)
C        C. L. LAWSON, JPL, 1968 JULY 30
C
C        INPUT..
C     (XTAB(I),I=1,NXTAB)     MONOTONE TABLE, INCREASING OR DECREASING
C     NXTAB                   NO. OF ENTRIES IN XTAB( ).
C                             NTAB.GE.1
C     NWANT                   NO. OF BRACKETING POINTS WANTED.
C                             1.LE.NWANT.LE.NXTAB
C     MODE                    =1,2,OR 3. RESULT WILL BE SAME IN ANY MODE
C                             BUT MORE EFFICIENT IF APPROPRIATE MODE
C                             IS SELECTED.
C                             =1 LINEAR SEARCH. RECOMMENDED WHEN XTAB IS
C                                EQUALLY SPACED OR NEARLY SO. TBSR WILL
C                                COMPUTE FIRST INDEX BY LINEAR INTERP-
C                                OLATION BETWEEN BEGINNING AND END OF
C                                XTAB, SEARCH WILL BE SEQUENTIAL FROM
C                                THERE.
C                             =2 BINARY SEARCH. RECOMMENDED WHEN
C                                SPACING IS VERY NONLINEAR.
C                             =3 RECALL SEARCH. RECOMMENDED WHEN SPACING
C                                IS VERY NONLINEAR AND SUCESSIVE REQUEST
C                                ARGUMENTS ARE EXPECTED TO BE IN SAME
C                                REGION OF TABLE. IN THIS MODE THE INPUT
C                                VALUE OF IX IS RELEVANT.
C                                IF 1.LE.IX.LE.NXTAB, A SEQUENTIAL
C                                SEARCH WILL START AT IX.  OTHERWISE A
C                                BINARY SEARCH WILL BE USED.
C     X                       INPUT ARGUMENT
C
C        OUTPUT..
C     IB                      INDEX OF FIRST BRACKETING POINT
C     IX                      INDEX OF POINT NEAREST TO X
C     KGO    = 1              EXACT MATCH (X=XTAB(IX))
C            = 2              BRACKETED BUT NOT MATCHED
C            = 3              OUTSIDE OF X(1)
C            = 4              OUTSIDE OF X(NTAB)
C            = 5              NWANT.GT.NTAB
C
      DIMENSION  XTAB(NXTAB)
      NW=NWANT
      NXT=NXTAB
      IF(NXT-NW) 10,20,20
   10 KGO1=5
      GO TO 310
   20 NWM1=NW-1
      IBMAX=NXT-NWM1
      X1=X
      S=SIGN(1.E0,XTAB(NXT)-XTAB(1))
C
C                               TEST MODE
C
      GO TO (130,90,30),MODE
   30 IX1=IX
      IF(IX1.GE.1 .AND. IX1.LE.NXT) GO TO 150
```

```
C
C                              TEST ENDS BEFORE STARTING BINARY SEARCH
C
   40 CONTINUE
      IF(S*(X1-XTAB(1))) 300,50,60
   50 KGO1=1
      GO TO 310
   60 IF(S*(X1-XTAB(NXT))) 80,70,330
   70 KGO1=1
      GO TO 340
C
C                                    BEGIN BINARY SEARCH
C
   80 NLOW=1
      NHIGH=NXT
   90 IX1=(NLOW+NHIGH)/2
      IF(S*(X1-XTAB(IX1))) 100,270,110
  100 NHIGH=IX1
      GO TO 120
  110 NLOW=IX1
  120 IF(NHIGH-NLOW-1) 230,230,90
C
C                                LINEAR INTERPOLATION FOR STARTING INDEX
C
  130 CONTINUE
      FNM1=NXT-1
      IX1=1.5E0+FNM1*(X1-XTAB(1))/(XTAB(NXT)-XTAB(1))
      IF(IX1-1)300, 140, 140
  140 IF(IX1-NXT) 150,150,330
  150 IF(S*(X1-XTAB(IX1))) 160,270,190
C
C                                          SEARCH BACKWARD
C
  160 IX1=IX1-1
      IF(IX1) 300,300,170
  170 IF(S*(X1-XTAB(IX1))) 160,270,180
  180 NLOW=IX1
      GO TO 220
C
C                                          SEARCH FORWARD
C
  190 IX1=IX1+1
      IF(NXT-IX1) 380,200,200
  200 IF(S*(X1-XTAB(IX1))) 210,270,190
  210 NLOW=IX1-1
  220 NHIGH=NLOW+1
C
C                                    X IS BRACKETED,NOT MATCHED
  230 KGO1=2
      IF(S*(XTAB(NHIGH)-2.E0*X1   +XTAB(NLOW))) 240,250,250
  240 IX1=NHIGH
      GO TO 260
  250 IX1=NLOW
  260 IB1=NLOW-NWM1/2
      GO TO 280
C
C                                                X IS MATCHED
```

```
  270 KGO1=1
      IB1=IX1-NWM1/2
C                                  ADJUST IB1 RELATIVE TO ENDS OF TABLE
  280 IF(IB1-1) 320,320,290
  290 IF(IBMAX-IB1) 350,360,360
C
C                                              TERMINATION
C
  300 KGO1=3
  310 IX1=1
  320 IB1=1
      GO TO 360
  330 KGO1=4
  340 IX1=NXT
  350 IB1=IBMAX
C
  360 IB=IB1
  370 IX=IX1
      KGO=KGO1
      RETURN
      END
```


NASA—JPL—Comm., L.A., Calif.

APPENDIX C

MERGE PROGRAMS LISTINGS

Preceding Page Blank FILMED

TDF14

```
C     TDF14
C      BUILDS INITIAL 'MERGE' FILE FROM USER-INPUT NOAA WEATHER
C      TAPE:  OUTPUT RECORDS CONTAIN THE STATION ID, JULIAN DATE,
C      SECTOR NUMBER, HOURLY TEMPERATURE, HOURLY WIND VELOCITY,
C      AND SPACE FOR HOURLY SOLAR RADIATION
C
      PARAMETER  NRD=3, NWRT=4
      REAL  OUTBUF(28,3), DB(24), SOLAR(24), WV(24)
      INTEGER  BUFR(83,4), ISTAT(4)
      INTEGER  JDATE(12)/0,31,59,90,120,151,181,212,243,273,304,334/
      INTEGER  MERGE(2)/'12',' '/, MODE/0110000000000/, MSTAT/0/
      INTEGER  TAPE(2)/'11',' '/, TEMP(6), TPDAY, TPMO, TPYR, WIND(6)
      EQUIVALENCE  (OUTBUF(1,1),IHD1), (OUTBUF(2,1),IHD2)
      EQUIVALENCE  (OUTBUF(3,1),IHD3), (OUTBUF(4,1),IHD4)
      EQUIVALENCE  (OUTBUF(5,1),DB(1)), (OUTBUF(5,2),WV(1))
      EQUIVALENCE  (OUTBUF(5,3),SOLAR(1))
C
      NTP=0                                  @INITIALIZE TAPE READ COUNT
      NFW=0                                  @INITIALIZE FILE WRITE COUNT
      READ(NRD,10)  IDSTN,IYR,LSTYR          @READ TAPE PARAMETERS
   10 FORMAT(A5,I3,I3)
      WRITE(NWRT,20) TAPE,MODE,IDSTN,IYR,LSTYR
   20 FORMAT('1TDF14 PROGRAM'/,
     .       '  TAPE UNIT = ',2A6,', EVEN PARITY, MODE = ',O14/,
     .       '  STATION NUMBER = ',A5/,
     .       '  FIRST YEAR = 19',I2,'  LAST YEAR = 19',I2///)
      ISTAT(1) = IOW(TAPE,34,1,MODE,0,ICOUNT)    @SET INPUT TAPE MODE
      IF(ISTAT(1).EQ.0) GO TO 30             @BAD I/O STATUS?
      WRITE(NWRT,25)  ISTAT(1)               @YES.
   25 FORMAT('0***MODE STATUS ERROR: ',I6)
      STOP MODE
C
   30 CALL RDCHK                             @NO. VERIFY TAPE STATION ID
      ISECTR = 0                             @INITIALIZE SECTOR NUMBER
      IBLK = 1                               @INITIALIZE INPUT BLOCK COUNT
      DO 40 I=1,24
      DB(I) = -1000.0                        @INITIALIZE AMBIENT TEMPERATURE
      WV(I) = -1000.0                        @INITIALIZE WIND VELOCITY
   40 SOLAR(I) = -1000.0                     @INITIALIZE SOLAR RADIATION
      IHD1 = IDSTN                           @MOVE STATION ID TO OUTPUT RECORD
      IHD4 = 366 * 1000 + LSTYR              @MOVE FINAL DATE TO OUTPUT RECORD
      LDAY = 0                               @INITIALIZE DAY INDEX
      LYR = IYR                              @INITIALIZE YEAR INDEX
      LDATE = 1000 * LYR                     @INITIALIZE JULIAN DATE
C
   50 DO 60 I=1,4                            @READ ONE DAY'S DATA FROM TDF14
      ISTAT(I) = IOW(TAPE,16,83,BUFR(1,I),0,ICOUNT)   @ INPUT TAPE
      NTP = NTP + 1                          @INCREMENT TAPE READ COUNT
      IF(ISTAT(I).EQ.1)  GO TO 500           @END OF TAPE FILE ENCOUNTERED?
      IF((ISTAT(I).EQ.4 .OR. ISTAT(I).EQ.0) .AND. ICOUNT.EQ.83)
     .     GO TO 60                          @NO. BAD I/O STATUS?
      WRITE(NWRT,55) ISTAT(I),IBLK,ISECTR,ICOUNT      @YES.
   55 FORMAT('0***TAPE READ ERROR: STATUS = ',A5/,
     .       ' AT BLOCK = ',I6,', SECTOR = ',I3,', ICOUNT = ',I6)
      ISTAT(I) = -1                          @SET ERROR STATUS FLAG
   60 CALL CHKDTS                            @CHECK INPUT RECORD DATES
```

```
      IBLK = IBLK + 1                    @INCREMENT INPUT BLOCK COUNT
      DO 70 I=1,4
   70 IF(ISTAT(I).EQ.4 .OR. ISTAT(I).EQ.0)  GO TO 80      @TAPE ERROR?
   75 WRITE(NWRT,76) IBLK,ISECTR         @YES.
   76 FORMAT('0***BAD DAY: BLOCK = ',I6,'  SECTOR = ',I6)
      IYR = LYR                          @SET YEAR INDEX
      IDAY = LDAY + 1                    @INCREMENT DAY INDEX
      IF(IDAY.LE.365)  GO TO 100         @END OF YEAR REACHED?
      IYR = IYR + 1                      @YES, INCREMENT YEAR INDEX
      IDAY = 1                           @REDEFINE DAY INDEX
      GO TO 100
C
   80 FLD(24,12,TPYR) = FLD(18,12,BUFR(2,I))      @TAPE READ OKAY...
      FLD(24,6,TPMO) = FLD(30,6,BUFR(2,I))
      FLD(30,6,TPMO) = FLD(0,6,BUFR(3,I))
      FLD(24,12,TPDAY) = FLD(6,12,BUFR(3,I))
      DECODE(6,90,TPYR) IYR              @CONVERT TAPE YEAR TO INTEGER
      INST = INSTAT(DUM)
      IF(INST.NE.0)  WRITE(NWRT,95) INST,IBLK,ISECTR
      DECODE(6,90,TPMO) IMO              @CONVERT TAPE MONTH TO INTEGER
      INST = INSTAT(DUM)
      IF(INST.NE.0)  WRITE(NWRT,95) INST,IBLK,ISECTR
      DECODE(6,90,TPDAY) IDAY            @CONVERT TAPE DAY TO INTEGER
      INST = INSTAT(DUM)
      IF(INST.NE.0)  WRITE(NWRT,95) INST,IBLK,ISECTR
   90 FORMAT(4X,I2)
   95 FORMAT('0***DECODE ERROR: STATUS = ',O14,'  AT BLOCK = ',I6,
     .       ', SECTOR = ',I6)
      IF(IMO.EQ.2 .AND. IDAY.GE.29)  GO TO 50    @LEAP DAY?
      IDAY = IDAY + JDATE(IMO)           @NO, COMPUTE JULIAN DAY
C
  100 IDATE = IDAY + 1000 * IYR          @COMPUTE JULIAN DATE
      IF(IDATE.GE.IMD4)  GO TO 500       @END OF 'MERGE' FILE REACHED?
      IF(IDATE.LT.LDATE)  STOP SEQNCE    @NO, INCORRECT DATE SEQUENCE?
      IF(IDATE.EQ.LDATE)  ISECTR = ISECTR - 3    @NO, SAME DAY AS LAST?
      IF(IYR.GT.LYR)  CALL FILLYR        @NO, FILL IN END OF YEAR?
      IF(IYR.GT.LYR+1)  CALL ADDYR       @NO, INSERT MISSING YEARS?
      IF(IDAY.GT.LDAY+1)  CALL ADDAY     @NO, INSERT MISSING DAYS?
      LYR = IYR                          @NO, REDEFINE YEAR INDEX
      LDAY = IDAY                        @REDEFINE DAY INDEX
      LDATE = IDATE                      @REDEFINE JULIAN DATE INDEX
C
      DO 200 I=1,4                       @FOR EACH QUARTER DAY...
      IF(ISTAT(I).EQ.4 .OR. ISTAT(I).EQ.0) GO TO 150     @TAPE ERROR?
      DO 120 J=1,6                       @YES.
      K = J + 6 * (I - 1)
      DB(K) = -1000.0                    @FLAG BAD TEMPERATURE DATA
  120 WV(K) = -1000.0                    @FLAG BAD WIND VELOCITY DATA
      GO TO 200
  150 WIND(1) = FLD(18,18,BUFR(5,I))     @NO, SAVE TAPE WIND VELOCITY DATA
      FLD(18,6,WIND(2)) = FLD(30,6,BUFR(18,I))
      FLD(24,12,WIND(2)) = FLD(0,12,BUFR(19,I))
      WIND(3) = FLD(6,18,BUFR(32,I))
      WIND(4) = FLD(18,18,BUFR(45,I))
      FLD(18,6,WIND(5)) = FLD(30,6,BUFR(58,I))
      FLD(24,12,WIND(5)) = FLD(0,12,BUFR(59,I))
      WIND(6) = FLD(6,18,BUFR(72,I))
```

PRECEDING PAGE BLANK - FILMED


```
      RETURN
C
C
      SUBROUTINE CHKDTS
C       CHECKS TDF14 INPUT RECORDS FOR DATE DISCREPENCIES
      IF(I.LT.2)  RETURN
      IF(BUFR(1,I-1).NE.BUFR(1,I))  GO TO 10      @SAME DATE?
      IF(BUFR(2,I-1).NE.BUFR(2,I))  GO TO 10
      IF(FLD(0,18,BUFR(3,I-1)) .NE. FLD(0,18,BUFR(3,I)))  GO TO 10
      RETURN                                      @YES.
   10 WRITE(NWRT,20) NTP,ISECTR                   @NO.
   20 FORMAT('0***DATE FOR SAME DAY DIFFERS AT RECORD = ',I6,
     .       ', SECTOR = ',I6)
      WRITE(NWRT,30) (BUFR(J,I-1),J=1,3),(BUFR(J,I),J=1,3)
   30 FORMAT(1X,2A6,A5,3X,2A6,A5)
      RETURN
C
C
      SUBROUTINE  FILLYR
C       BUILDS SKELETAL 'MERGE' RECORDS FOR MISSING DAYS AT END OF YEAR
      DO 10 I=1,24
      DB(I) = -1000.0                             @FLAG BAD TEMPERATURE DATA
   10 WV(I) = -1000.0                             @FLAG BAD WIND VELOCITY DATA
      LDAY = LDAY + 1                             @SET START DAY
      KDAY = 366                                  @SET FINAL DAY
      DO 50 I=LDAY,KDAY                           @FOR EACH MISSING DAY...
      IDATE = I + 1000 * LYR                      @COMPUTE JULIAN DATE
   50 CALL WRTRCD                                 @WRITE 'MERGE' OUTPUT RECORD
      LDAY = 0                                    @REDEFINE INDEX DAY
      IDATE = 1000 * (LYR + 1) + 1                @REDEFINE JULIAN DATE INDEX
      RETURN
C
C
      SUBROUTINE  RDCHK
C       READS INITIAL TDF14 TAPE RECORD AND VERIFIES THE STATION ID
      ISTAT(1) = IOW(TAPE,16,83,BUFR(1,1),0,ICOUNT)
      IF((ISTAT(1).EQ.4 .OR. ISTAT(1).EQ.0) .AND. ICOUNT.EQ.83)
     .     GO TO 20                               @BAD I/O STATUS?
      WRITE(NWRT,10) ISTAT(1)                     @YES.
   10 FORMAT('0***RDCHK READ ERROR: STATUS = ',I6)
      RETURN 0
   20 DECODE(9,30,BUFR(1,1)) ID                   @NO. CONVERT TAPE STATION ID TO
   30 FORMAT(4X,A5)                               @ HOLLERITH
      INST = INSTAT(DUM)
      IF(INST.NE.0)  WRITE(NWRT,40) INST,IBLK,ISECTR
   40 FORMAT('0***RDCHK DECODE ERROR: STATUS = ',O14,' AT BLOCK = ',I6,
     .       ', SECTOR = ',I6)
      IF(ID.NE.IDSTN)  STOP STNID                 @REQUESTED STATION TAPE?
      ISTAT(1) = IOW(TAPE,41,83,BUFR(1,1),0,ICOUNT)   @YES. REWIND TAPE
      IF(ISTAT(1).EQ.0)  RETURN                   @BAD I/O STATUS?
      WRITE(NWRT,50) ISTAT(1)                     @YES.
   50 FORMAT('0***RDCHK REWIND ERROR: STATUS = ',I6)
      RETURN 0
C
C
      SUBROUTINE  UNPACK(VAR,VALUE)
C       CONVERTS TAPE WEATHER DATA TO INTEGER FORMAT
```

Preceding Page BLank - FILMed

DECK280

```
C     DECK280
C      ADDS DECK-280 SOLAR INSOLATION DATA TO 'MERGE' TAPE
C
      PARAMETER  NRD=3, NWRT=4
      REAL  BUFR(134), OUTBUF(28,3), SOLAR(24)
      INTEGER  JDATE(12)/0,31,59,90,120,151,181,212,243,273,304,334/
      INTEGER  IBUF(8,10), MERGE(2)/'12',' '/, MODE/0110000000000/
      INTEGER  SECEND, SECNXT, SECONE, SECTOR
      INTEGER  TAPE(2)/'11',' '/, TPDAY, TPMO, TPYR
      LOGICAL  LCHG/.FALSE./, LGET/.FALSE./
      EQUIVALENCE  (OUTBUF(1,1),IHD1), (OUTBUF(2,1),IHD2)
      EQUIVALENCE  (OUTBUF(3,1),IHD3), (OUTBUF(4,1),IHD4)
      EQUIVALENCE  (OUTBUF(5,3),SOLAR(1))
C
      NERR = 0                              @INITIALIZE ERROR COUNT
      NTP=0                                 @INITIAILZE TAPE READ COUNT
      NFR=0                                 @INITIAILZE FILE READ COUNT
      NFW=0                                 @INITIAILZE FILE WRITE COUNT
      READ(NRD,10)  IDSTN,IDSTN1,IDSTN2     @READ FILE PARAMETERS
   10 FORMAT(3A6)
      WRITE(NWRT,15) TAPE,MODE,IDSTN,IDSTN1,IDSTN2
   15 FORMAT('1DECK280 PROGRAM'/,
     .       '  TAPE UNIT = ',2A6,', EVEN PARITY, MODE = ',O14/,
     .       '  STATION NUMBER = ',A5/,
     .       '  ALTERNATE STATION NUMBERS = ',2A7)
      IBLK = 0                              @INITIALIZE INPUT BLOCK COUNT
      IFLG = 0                              @INITIALIZE TAPE FILE FLAG
      SECTOR = 0                            @INITIALIZE SECTOR NUMBER
      CALL FILCHK                           @GET 'MERGE' FILE PARAMETERS
      LDATE = SECONE                        @STORE 'MERGE' START DATE
      LYR1 = SECONE / 1000                  @SET FIRST YEAR
      LDY1 = SECONE - 1000 * LYR1           @SET FIRST DAY
      ISTAT = IOW(TAPE,34,1,MODE,0,ICOUNT)  @SET INPUT TAPE MODE
      IF(ISTAT.NE.0) GO TO 820              @BAD I/O STATUS? NO...
C
   20 ISTAT = IOW(TAPE,16,134,BUFR,0,ICOUNT)     @READ DECK-280 RECORD
      IBLK = IBLK + 1                       @INCREMENT INPUT BLOCKS COUNTER
      NTP = NTP + 10                        @INCREMENT TAPE READ COUNT
      IF(ISTAT.EQ.1)  GO TO 600             @END OF DECK-280 TAPE?
      IF(ISTAT.EQ.0 .OR. ISTAT.EQ.4)  GO TO 30   @NO. BAD I/O STATUS?
      WRITE(NWRT,25) ISTAT,IBLK,ICOUNT      @YES.
   25 FORMAT('0***TAPE READ ERROR: STATUS = ',I6,'  AT BLOCK =',I6,
     .       ', ICOUNT = ',I6)
      NERR = NERR + 1                       @INCREMENT ERROR COUNT
      IF(NERR.LE.10)  GO TO 20              @TOO MANY READ ERRORS?
      CALL PUTREC                           @YES.
      STOP READ
C
   30 NERR = 0                              @RE-INITIALIZE ERROR COUNT
      DECODE(80,40,BUFR)  ((IBUF(I,J),I=1,8),J=1,10)  @DECODE INPUT
   40 FORMAT(A5,I2,I2,R1,R1,I2,R1,I3,63X)
      INST = INSTAT(DUM)
      IF(INST.EQ.0)  GO TO 50
      WRITE(NWRT,45)  INST,IBLK
   45 FORMAT('0***DECODE ERROR: STATUS = ',O14,' AT BLOCK = ',I6)
      GO TO 20                              @SKIP RECORD IF DECODE ERROR
```

```
C
   50 DO 300 J=1,10                          @FOR EACH INPUT HOUR...
      IF(IBUF(1,J).EQ.IDSTN)  GO TO 60       @CORRECT DECK-280 STATION ID?
      IF(IBUF(1,J).EQ.IDSTN1)  GO TO 60
      IF(IBUF(1,J).EQ.IDSTN2)  GO TO 60
      IF(IFLG.EQ.0 .AND. IBLK.EQ.1)  CALL SKIP($30)   @NO. FIRST RCD?
      GO TO 850                              @NO.
C
   60 IF(IBUF(4,J).LT.48)  GO TO 70          @YES. CONVERT DAY TO NUMERICS
      IBUF(4,J) = IBUF(4,J) - 48
      GO TO 80
   70 IBUF(4,J) = IBUF(4,J) - 14
      IF(IBUF(4,J).GT.9)  IBUF(4,J) = 0
      IBUF(6,J) = IBUF(6,J) + 1
   80 IF(IBUF(5,J).LT.48)  GO TO 90
      IBUF(5,J) = IBUF(5,J) - 48
      GO TO 100
   90 IBUF(5,J) = IBUF(5,J) - 14
      IF(IBUF(5,J).GT.9)  IBUF(5,J) = 0
      IBUF(6,J) = IBUF(6,J) - 1
C
  100 IHR = IBUF(6,J)                        @SET INPUT HOUR
      TPYR = IBUF(2,J)                       @SET INPUT YEAR
      TPMO = IBUF(3,J)                       @SET INPUT MONTH
      TPDAY = 10 * IBUF(4,J) + IBUF(5,J)     @SET INPUT DAY
      IF(TPMO.EQ.2 .AND. TPDAY.GE.29)  GO TO 300       @LEAP DAY?
      IDATE = 1000 * TPYR + JDATE(TPMO) + TPDAY  @NO. COMPUTE JULIAN DATE
      IF(IDATE.EQ.LDATE)  GO TO 200          @SAME DAY AS LAST REQUEST?
      CALL PUTREC                            @NO. OVERWRITE LAST FILE RECORD
      LDATE = IDATE                          @STORE REQUESTED DATE
      SECNXT = IDATE                         @SET 'MERGE' RECORD POINTER
      CALL POINTR($300,$800)                 @GET REQUESTED 'MERGE' RECORD
C
  200 SOL = IBUF(8,J) / 10.0                 @COMPUTE SOLAR RADIATION LANGLEYS
      IF(IBUF(7,J).LT.48)  GO TO 210
      IBUF(7,J) = IBUF(7,J) - 48
      GO TO 240
  210 IBUF(7,J) = IBUF(7,J) - 14
      IF(IBUF(7,J).GT.9)  IBUF(7,J) = 0
  240 SOL = 100.0 * IBUF(7,J) + SOL
      SOLAR(IHR) = 41.84 * SOL / 3.6         @CONVERT TO WATTS/SQ.METER
      LCHG = .TRUE.                          @SET CHANGE FLAG
  300 CONTINUE
      GO TO 20                               @GET NEXT DECK-280 INPUT RECORD
C
  600 CONTINUE                               @END OF DECK280 TAPE
      CALL PUTREC                            @OVERWRITE LAST FILE RECORD
      WRITE(NWRT,610) TAPE,IBLK,NTP,NFR,NFW
  610 FORMAT('0END OF TAPE ',2A6,', ',I6,' INPUT BLOCKS (NTP = ',I6,
     .       ', NFR = ',I6,', NFW = ',I6)
      STOP DCK280
C
C*****ERROR MESSAGE EXITS
C
  800 WRITE(NWRT,810) SECNXT
  810 FORMAT('0***IOW READ ERROR AT REQUESTED DATE = ',I5)
      CALL PUTREC
```

```
      STOP
C
  820 WRITE(NWRT,830) ISTAT
  830 FORMAT('0***MODE STATUS ERROR: ',I6)
      STOP MODE
C
  850 WRITE(NWRT,860)  IBUF(1,J),J,IBLK,NTP,NFR,NFW
  860 FORMAT('0***NO DECK-280 TAPE EOF: STATION ID = ',A5/,
     .  4X,'AT HOUR NUMBER = ',I2,' OF BLOCK ',I6,
     .  ', NTP=',I6,', NFR=',I6,', NFW=',I6)
      CALL PUTREC
      STOP XTPEOF
C
C**********************  INTERNAL SUBROUTINES  **********************
C
      SUBROUTINE  FILCHK
C      READS FIRST DATA FILE RECORD AND STORES FILE PARAMETERS
      ISTAT = IOW(MERGE,16,84,OUTBUF,0,ICOUNT)   @READ FIRST RECORD
      IF(ISTAT.EQ.0) GO TO 20                    @BAD I/O STATUS?
      WRITE(NWRT,10) ISTAT                       @YES.
   10 FORMAT('0***FILCKH ERROR: STATUS = ',I6)
      RETURN 0
   20 LGET = .TRUE.                              @SET VALID READ FLAG
      SECONE = IHD2                              @SET FIRST RECORD POINTER
      SECEND = IHD4                              @SET LAST RECORD POINTER
      WRITE(NWRT,30) IHD1,IHD2,IHD4
   30 FORMAT('  MERGE FILE STATION NUMBER = ',A6,/' LIMIT DATES = ',I5,
     .       ' TO ',I5)
      IF(IDSTN.NE.IHD1)  STOP STNID              @CORRECT 'MERGE' STATION?
      RETURN                                     @YES.
C
C
      SUBROUTINE  POINTR(*,*)
C      READS REQUESTED DATA FILE INPUT RECORD
      LYR = 0                                    @INITIALIZE YEARS-TO-SKIP
      IF(SECNXT.LT.SECONE)  RETURN 1             @IS REQUESTED DATE OUTSIDE THE
      IF(SECNXT.GT.SECEND)  RETURN 1             @ RANGE OF THE DATA FILE? NO...
      LYRS = SECNXT / 1000                       @SET REQUESTED YEAR
      LDYS = SECNXT - 1000 * LYRS                @SET REQUESTED DAY
      LYR = LYRS - LYR1                          @REDEFINE REQUESTED YEAR
      LDAY = LDYS - LDY1                         @REDEFINE REQUESTED DAY
      SECTOR = 3 * (366 * LYR + LDAY)            @SET NEW SECTOR NUMBER
   50 ISTAT = IOW(MERGE,16,84,OUTBUF,SECTOR,ICOUNT)   @READ NEXT RCD
      NFR = NFR + 1                              @INCREMENT FILE READ COUNT
      IF(ISTAT.NE.0)  RETURN 2                   @BAD I/O STATUS?
      LGET = .TRUE.                              @NO. SET VALID READ FLAG
      IF(IHD2.EQ.SECNXT)  RETURN                 @FILE DATE SAME AS REQUESTED?
  100 WRITE(NWRT,120) SECNXT,IHD2                @NO. WRITE ERROR MESSAGE
  120 FORMAT('0***REQUESTED RECORD NOT FOUND IN FILE:',2(2X,I5))
      RETURN 1
C
C
      SUBROUTINE PUTREC
C      WRITES SOLAR INSOLATION DATA TO 'MERGE' FILE
      INTEGER NPUTER/0/
      IF(.NOT.LCHG)  RETURN                      @CHANGE FLAG SET?
      IF(LGET) GO TO 20                          @YES. READ ERROR?
```

```
      WRITE(NWRT,10)                          @YES.
   10 FORMAT('0***PUTREC ERROR: LCHG = TRUE, LGET = FALSE')
      NPUTER = NPUTER + 1                     @INCREMENT ERROR COUNT
      IF(NPUTER.LE.10) RETURN                 @TOO MANY ERRORS?
      STOP PUTREC                             @YES.
   20 ISTAT = IOW(MERGE,8,84,OUTBUF,SECTOR,ICOUNT)    @RE-WRITE 'MERGE'
      NFW = NFW + 1                           @INCREMENT FILE WRITE COUNT
      IF(ISTAT.EQ.0) GO TO 40                 @BAD I/O STATUS?
      WRITE(NWRT,30) ISTAT                    @YES.
   30 FORMAT('0***FILE WRITE ERROR: STATUS = ',I6)
      STOP WRTRCD
   40 LCHG = .FALSE.                          @NO. RE-INITIALIZE CHANGE FLAG
      LGET = .FALSE.                          @RE-INITIALIZE VALID READ FLAG
      RETURN
C
C
      SUBROUTINE  SKIP(S)
C      SKIPS TAPE RECORDS UNTIL A CORRECT STATION NUMBER IS FOUND
      ISKIP = 1                               @INITIALIZE SKIP COUNTER
      KERR = 0                                @INITIALIZE ERROR COUNT
      WRITE(NWRT,10)
   10 FORMAT('0SKIP UNTIL CORRECT STATION FOUND')
   20 ISTAT = IOW(TAPE,16,134,BUFR,0,ICOUNT)      @READ DECK-280 RECORD
      IBLK = IBLK + 1                         @INCREMENT INPUT BLOCK COUNT
      NTP = NTP + 10                          @INCREMENT TAPE READ COUNT
      IF(ISTAT.EQ.0 .OR. ISTAT.EQ.4)  GO TO 50   @BAD I/O STATUS?
      WRITE(NWRT,30) ISTAT,IBLK               @YES.
   30 FORMAT('0***SKIP ERROR: STATUS = ',I6,'  AT BLOCK = ',I6)
      IF(ISTAT.EQ.1)  STOP TPEOF              @END OF TAPE?
      KERR = KERR + 1                         @NO. INCREMENT ERROR COUNT
      IF(KERR.LE.10)  GO TO 20                @TOO MANY ERRORS?
      STOP SKIP                               @YES.
   50 KERR = 0                                @NO. RE-INITIALIZE ERROR COUNT
      DECODE(5,60,BUFR) ID                    @DECODE TAPE STATION ID
   60 FORMAT(A5)
      IF(ID.EQ.IDSTN)  GO TO 70               @CORRECT DECK-280 STATION ID?
      IF(ID.EQ.IDSTN1)  GO TO 70
      IF(ID.EQ.IDSTN2)  GO TO 70
      ISKIP = ISKIP + 1                       @NO. INCREMENT SKIP COUNTER
      GO TO 20                                @READ NEXT DECK-280 INPUT RECORD
   70 IFLG = 1                                @YES. RESET TAPE FILE FLAG
      WRITE(NWRT,80)  ISKIP
   80 FORMAT('0***',I6,' RECORDS SKIPPED BEFORE CORRECT STATION FOUND')
      RETURN 1
C
C
      END
```

IOW

```
          AXR$.     121470-0001          R. DOCKEN.
.                   121970-0002          R. DOCKEN.                12-21-70
.                   122370-0003          R. DOCKEN.                12-23-70
.                   010771-0004          R. DOCKEN.                01-07-71
          CHAR      057,072.
. ************************************************************************
. *                                                                      *
. *    J = IOW (FILE,JFUNCT, LENGTH, BUFFER,JDRUM, ICOUNT)               *
. *                                                                      *
. *    J = IOW (FILE,JFUNCT, LENGTH, BUFFER,JDRUM, ICOUNT, ID, FIND)     *
. *                                                                      *
. * THIS PROGRAM PROVIDES AN INTERFACE FOR THE FORTRAN CALLER TO         *
. * EXEC REQUEST IOW$ TO FACILITATE RANDOM ACCESS ON THE DRUMS AND       *
. * FASTRANDS AND TAPES.                                                 *
. *                                                                      *
. *       <FILE>    THE 2-WORD INTERNAL FILE NAME (LUSF) OF THE FILE     *
. *                 TO BE USED.  IT MUST BE ASSIGNED.                    *
. *                                                                      *
. *       <JFUNCT>  ONE OF THE RECOGNIZED I/O CODES.  010=WRITE,         *
. *                 020=READ, AND 040=REWIND ARE THE MOST COMMON         *
. *                 ONES.  OTHERS ARE DESCRIBED IN THE PRM AND THE       *
. *                 EXEC REFERENCE CARD.                                 *
. *                                                                      *
. *       <LENGTH>  THE MAXIMUM NUMBER OF WORDS TO TRANSFER.             *
. *                                                                      *
. *       <BUFFER>  THE CORE BUFFER USED IN TRANSFERS.  IT MUST BE       *
. *                 SPECIFIED EVEN FOR NO-DATA OPERATIONS LIKE A         *
. *                 TAPE REWIND.                                         *
. *                                                                      *
. *       <JDRUM>   THE I/O STARTING ADDRESS FOR RANDOM ACCESS           *
. *                 FILES.                                               *
. *                                                                      *
. *       <ICOUNT>  THIS IS SET BY IOW TO THE ACTUAL NUMBER OF DATA      *
. *                 WORDS COPIED.                                        *
. *                                                                      *
. *       <ID>      THE SENTINEL FOR SEARCHES.                           *
. *                                                                      *
. *       <FIND>    THE LOCATION OF THE FIND FOR A SEARCH.               *
. *                                                                      *
. * THE FUNCTION RETURNS THE STATUS CODE.                                *
. ************************************************************************
.
$(2)      LIT.
SX11      +         0.                 SAVE X11 FOR DEBUG TRACING.
PKT       RES       8.
PK        EQU       PKT-1.
.
$(1).
.
IOW.      S         X11,SX11.          SAVE <X11>.
          DL        A0,*0,x11.         <FILE>.
          DS        A0,PK+1.
          SZ        PK+3.              CLEAR INTERRUPT CODES.
          L         A0,*1,x11.         <JFUNCT>.
          S,T1      A0,PK+4.
          LXI       A0,*2,x11.         <LENGTH>.
```

```
        LXM     A0,3,X11.       <BUFFER>.               12-19-70
        L       A1,@4,X11.      <JDRUM>.
        DS      A0,PK+5.
        TZ,M1   6,X11.          IS <ID, FIND> SPECIFIED?
        J       $+3.            NO!
        L       A0,@6,X11.      <ID>.
        S       A0,PK+7.
        L,U     A0,PKT.
        EM      10#9.
        L,M2    A0,PK+4.        <ICOUNT>.
        S       A0,@5,X11.
        TZ,M1   6,X11.          IS <ID, FIND> SPECIFIED?
        J       EXIT36.         NO!
        SZ,T1   PK+8.
        L       A0,PK+8.        <FIND>.
        S       A0,@7,X11.
        L,S1    A0,PK+4.        <J>.
        J       9,S1.                                   01-07-71
EXIT36  L,S1    A1,PK+4.        <J>.
        J       7,X11.                                  01-07-71
        END.
```

LISTMERGE

```
C      LISTMERGE
C       LISTS A SPECIFIED NUMBER OF DAYS OF MERGE FILE
C       WEATHER DATA BEGINNING AT SELECTED START DATES
C
       PARAMETER  NRD=3, NWRT=4
       REAL  OUTBUF(28,3)
       INTEGER  MERGE/12/
       INTEGER  SECEND, SECNOW, SECNXT, SECONE, SECTOR
       EQUIVALENCE  (OUTBUF(1,1),IHD1), (OUTBUF(2,1),IHD2)
       EQUIVALENCE  (OUTBUF(3,1),IHD3), (OUTBUF(4,1),IHD4)
C
       SECTOR = 0                          @INITIALIZE SECTOR NUMBER
       CALL NTRAN(MERGE,10,22)             @REWIND 'MERGE' DATA FILE
       CALL NTRAN(MERGE,2,84,OUTBUF,ISTAT,22)     @READ FIRST RECORD
       IF(ISTAT.LT.0)  STOP BADFIL         @BAD I/O STATUS?
       SECONE = IHD2                       @NO. SAVE FIRST RECORD DATE
       SECEND = IHD4                       @SAVE LAST RECORD DATE
       SECNOW = SECONE                     @SET PRESENT RECORD POINTER
       CALL NTRAN(MERGE,10,22)             @REWIND 'MERGE' DATA FILE
       WRITE(NWRT,10)  IHD1,IHD2,IHD4
   10  FORMAT('1',50X,'WEATHER DATA FOR STATION ',A5,/
      .       52X,'LIMIT DATES = ',I5,' TO ',I5)
C
   50  WRITE(NWRT,60)
   60  FORMAT('0ENTER START DATE (YYDDD) AND NO. OF RECORDS TO DISPLAY')
       READ(NRD,70,ERR=800,END=500)  IDATE,NRCDS  @READ USER REQUEST
   70  FORMAT( )
       IF(IDATE.LE.0)  IDATE = SECONE     @REQUEST TO PRINT ENTIRE FILE?
       IF(NRCDS.LE.0)  NRCDS = 366 * (1 + SECEND / 1000 - SECONE / 1000)
       SECNXT = IDATE                      @SET 'MERGE' RECORD POINTER
       CALL POINTR($820,$84C)              @GET REQUESTED 'MERGE' RECORD
C
       DO 200 N=1,NRCDS                    @FOR REQUESTED NUMBER OF DAYS..
       WRITE(NWRT,100)  IHD2,IHD3          @DISPLAY RECORD DATE AND SECTOR
  100  FORMAT('0DATE = ',I5,3X,'SECTOR = ',I5)
       WRITE(NWRT,110)  (OUTBUF(I,1),I=5,28)      @DISPLAY TEMP. DATA
  110  FORMAT(7X,'TEMP = ',12F9.2,/14X,12F9.2)
       WRITE(NWRT,120)  (OUTBUF(I,2),I=5,28)      @DISPLAY WIND DATA
  120  FORMAT(7X,'WIND = ',12F9.2,/14X,12F9.2)
       WRITE(NWRT,130)  (OUTBUF(I,3),I=5,28)      @DISPLAY INSOLATION
  130  FORMAT(7X,'QST =  ',12F9.2,/14X,12F9.2)
C
  150  CALL NTRAN(MERGE,2,84,OUTBUF,ISTAT,22)     @READ NEXT RECORD
       IF(ISTAT.EQ.-2)  GO TO 900         @END OF 'MERGE' FILE ENCOUNTERED?
       IF(ISTAT.LT.0)  GO TO 840          @NO. BAD I/O STATUS?
       IF(IHD2.GT.SECEND)  GO TO 900      @NO. ILLEGAL DATE REQUESTED?
       IYR = 1000 * IFIX(IHD2/1000)       @NO. DETERMINE RECORD YEAR
       IF(IHD2-IYR.GE.366)  GO TO 150     @SKIP RECORD IF DAY = 366
  200  CONTINUE
       SECTOR = IHD3 + 3                  @SET SECTOR NUMBER TO NEXT RECORD
       SECNOW = IHD2 + 1                  @SET 'MERGE' POINTER TO NEXT RCD
       GO TO 50                           @GET NEXT USER REQUEST
C
  500  STOP
C
C*****ERROR MESSAGE EXITS
```

```
C
  800 STOP INPUT
  820 WRITE(NWRT,830)  IDATE,SECONE,SECEND
  830 FORMAT('0***REQUESTED DATE (',I5,') IS OUTSIDE RANGE OF ',
     .       'MERGE FILE (',I5,',',I5,')')
      STOP
  840 WRITE(NWRT,850)  SECNXT
  850 FORMAT('0***NTRAN READ ERROR AT DATE = ',I5)
      STOP
  900 STOP EOF
C
C**********************  INTERNAL SUBROUTINES  **********************
C
      SUBROUTINE  POINTR(S,S)
C      READS REQUESTED DATA FILE INPUT RECORD
C
      LYR = 0                                 @INITIALIZE YEARS-TO-SKIP
      CALL NTRAN(MERGE,20,ISTAT)              @WAIT AND UNSTACK
      IF(ISTAT.LT.0)  RETURN 2                @BAD I/O STATUS? NO...
      IF(SECNXT.LT.SECONE)  RETURN 1          @IS REQUESTED DATE OUTSIDE THE
      IF(SECNXT.GT.SECEND)  RETURN 1          @ RANGE OF THE DATA FILE? NO...
      ISIGN = 1                               @INITIALZE DATE FLAG
      IF(SECNXT-SECNOW) 10,50,20              @REQUESTED DATE.LT/EQ/GT.PRESENT
C
   10 ISIGN = -1                              @SET DATE FLAG PRIOR TO PRESENT
   20 IYRS = SECNXT / 1000                    @SET REQUESTED YEAR
      LDAY = SECNXT - 1000 * IYRS             @SET REQUESTED DAY
      IYR1 = SECNOW / 1000                    @SET PRESENT YEAR
      LDY = SECNOW - 1000 * IYR1              @SET PRESENT DAY
      IYRS = ABS(IYRS - IYR1)
      LDAY = ISIGN * (LDAY - LDY)             @REDEFINE REQUESTED DAY
      IF(IYRS.EQ.0) LDAY = ABS(LDAY)
      IF(IYRS.NE.0) LDAY = ABS(366 - LDAY)
      IF(IYRS.NE.0) LYR = IYRS - 1            @REDEFINE YEARS-TO-SKIP
C
   30 NEXT = 3 * ISIGN * (366 * LYR + LDAY)       @REDEFINE POINTER
      SECTOR = SECTOR + NEXT                  @SET NEW SECTOR NUMBER
      SECNOW = SECNXT                         @REDEFINE PRESENT DAY
      CALL NTRAN(MERGE,6,NEXT)                @POSITION TO REQUESTED RECORD
   50 CALL NTRAN(MERGE,2,84,OUTBUF,ISTAT,22)      @READ NEXT 3 SECTORS
      IF(ISTAT.LT.0)  RETURN 2                @BAD I/O STATUS? NO...
      IF(IHD2.NE.SECNXT) GO TO 100            @FILE DATE SAME AS REQUESTED?
      RETURN                                  @YES.
C
  100 IF(SECNXT.LT.1000)  RETURN              @STATISTICAL FILE?
      WRITE(NWRT,120) SECNXT,IHD2             @NO. WRITE ERROR MESSAGE
  120 FORMAT('0***REQUESTED RECORD NOT FOUND IN FILE:',2(2X,I5))
      STOP
C
      END
```

Preceding Page BLANK - NOT FILMed

APPENDIX D

STAT PROGRAMS LISTINGS

Preceding Page BLANK - FILMed

STATS

```
C     STATS
C      PERFORMS STATISTICAL ANALYSIS OF 'MERGE' FILE
C      WEATHER DATA AND OUTPUTS A 'STAT' DATA FILE
C
      PARAMETER  LMRG=3, LMYR=366*LMRG, LST=4, LSYR=366*LST
      PARAMETER  NRD=3, NWRT=4, NYRS=12, NYX2=2*NYRS
      REAL  BUFR(28,3), MBUF(28,4), OUTBUF(28,4)
      REAL  Q(24), T(24), V(24), TMAX(NYRS), TMIN(NYRS)
      INTEGER  IHB(4), IHM(4), IHO(4), ILMD(NYX2), JLMD(2)
      INTEGER  MDATE(12)/31,59,90,120,151,181,212,243,273,304,334,365/
      INTEGER  MERGE(2)/'12',' '/, MSTAT(2)/'11',' '/
      INTEGER  NLM(NYRS,3), NLMDH(24,3), NY(3)
      EQUIVALENCE  (BUFR(1,1),IHB(1)), (BUFR(5,1),T(1))
      EQUIVALENCE  (BUFR(5,2),V(1)), (BUFR(5,3),Q(1))
      EQUIVALENCE  (MBUF(1,1),IHM(1)), (OUTBUF(1,1),IHO(1))
C
      DO 10 I=1,NYRS
      TMAX(I) = -1000.0                    @INITIALIZE YEARLY MAXIMUM TEMP
   10 TMIN(I) = 1000.0                     @INITIALIZE YEARLY MINIMUM TEMP
  100 JSECTR = 0                           @INITIALIZE 'MERGE' SECTOR POINTR
      ISTAT = IOW(MERGE,16,84,BUFR,JSECTR,ICNT)  @READ 1ST 'MERGE' RCD
      IF(ISTAT.NE.0)  GO TO 900            @BAD I/O STATUS? NO...
      NYEARS = 1 + IHB(4)/1000 - IHB(2)/1000     @NUMBER OF 'MERGE' YRS
      WRITE(NWRT,110)  IHB(1),IHB(2),IHB(4),NYEARS
  110 FORMAT('1STATISTICAL ANALYSIS FOR STATION NUMBER ',A5,/
     .       '   LIMIT DATES = ',I5,' TO ',I5,/
     .       '   TIME SPAN = ',I2,' YEARS')
      M = 0                                @INITIALIZE MONTH COUNTER
      CALL MONTH(8,20)                     @INITIALIZE MONTHLY STATISTICS
      IHM(1) = IHB(1)                      @STORE 'STAT' STATION NUMBER
      IHO(1) = IHB(1)
      IHM(4) = 12                          @STORE 'STAT' LAST MONTH
      IHO(4) = 366                         @STORE 'STAT' LAST DAY
C
  120 DO 550 IDAY=1,366                    @FOR EACH DAY OF THE YEAR...
      ISECTR = IABS(LMRG * (IDAY - 1))         @SET PRIMARY 'MERGE' SECTOR
      DO 380 J=5,28
      DO 380 K=1,4
  380 OUTBUF(J,K) = 0.0                    @INITIALIZE HOURLY, DAILY STATS
      IHO(2) = IDAY                        @STORE 'STAT' DAY INDEX
      IHO(3) = IABS(LST * (IDAY - 1))          @STORE 'STAT' SECTOR NUMBER
      DO 390 J=1,24
      DO 390 K=1,3
  390 NLMDH(J,K) = 0                       @INITIALIZE HOURLY STATS COUNTER
      DO 480 J=1,NYX2
  480 ILMD(J) = 0                          @INITIALIZE DAILY STATS COUNTERS
      DO 540 J=1,2
  540 JLMD(J) = 0
C
  170 DO 340 J=1,NYEARS                    @FOR EACH 'MERGE' YEAR...
  200 JSECTR = IABS(ISECTR + LMYR * (J - 1))     @SET 'MERGE' RCD SECTOR
      ISTAT = IOW(MERGE,16,84,BUFR,JSECTR,ICNT)  @READ 'MERGE' RECORD
      IF(ISTAT.NE.0)  GO TO 900            @BAD I/O STATUS? NO...
      DO 340 K=1,24                        @FOR EACH HOUR OF THE DAY...
  210 IF(T(K).LE.-1000.0)  GO TO 260       @VALID TEMPERATURE? YES...
  220 TMAX(J) = AMAX1(TMAX(J),T(K))        @GET MAXIMUM YEARLY TEMPERATURE
```

```
  240 TMIN(J) = AMIN1(TMIN(J),T(K))     @GET MINIMUM YEARLY TEMPERATURE
  250 OUTBUF(K+4,1) = OUTBUF(K+4,1) + T(K)  @SUM HOURLY TEMPERATURES
      NLMDH(K,1) = NLMDH(K,1) + 1       @INCREMENT HOURLY TEMP COUNTER
      MBUF(J+4,1) = MBUF(J+4,1) + T(K)      @SUM MONTHLY TEMPERATURES
      NLM(J,1) = NLM(J,1) + 1           @INCREMENT MONTHLY TEMP COUNTER
  260 IF(V(K).LE.-1000.0)  GO TO 270    @VALID WIND VELOCITY? YES...
      OUTBUF(K+4,2) = OUTBUF(K+4,2) + V(K)  @SUM HOURLY VELOCITIES
      NLMDH(K,2) = NLMDH(K,2) + 1       @INCREMENT HOURLY WIND COUNTER
      OUTBUF(J+4,4) = OUTBUF(J+4,4) + V(K)  @SUM DAILY VELOCITIES
      OUTBUF(3,4) = OUTBUF(3,4) + V(K)
      ILMD(J) = ILMD(J) + 1             @INCREMENT DAILY WIND COUNTERS
      JLMD(1) = JLMD(1) + 1
      MBUF(J+4,2) = MBUF(J+4,2) + V(K)      @SUM MONTHLY VELOCITIES
      NLM(J,2) = NLM(J,2) + 1           @INCREMENT MONTHLY WIND COUNTER
  270 IF(Q(K).LE.-1000.0)  GO TO 340    @VALID SOLAR INSOLATION? YES...
      OUTBUF(K+4,3) = OUTBUF(K+4,3) + Q(K)  @SUM HOURLY INSOLATIONS
      NLMDH(K,3) = NLMDH(K,3) + 1       @INCREMENT HOURLY SOLAR COUNTER
      OUTBUF(J+NYRS+4,4) = OUTBUF(J+NYRS+4,4) + Q(K)  @SUM DAILY SOLAR
      OUTBUF(4,4) = OUTBUF(4,4) + Q(K)                @ INSOLATIONS
      ILMD(J+NYRS) = ILMD(J+NYRS) + 1   @INCREMENT DAILY SOLAR COUNTERS
      JLMD(2) = JLMD(2) + 1
      MBUF(J+4,3) = MBUF(J+4,3) + Q(K)      @SUM MONTHLY INSOLATIONS
      NLM(J,3) = NLM(J,3) + 1           @INCREMENT MONTHLY SOLAR COUNTER
  340 CONTINUE
C
      DO 360 J=1,24                     @COMPUTE HOURLY MEANS...
      DO 360 K=1,3
      IF(NLMDH(J,K).GT.0)  GO TO 360    @INVALID DATA? YES...
      OUTBUF(J+4,K) = -1000.0           @FLAG INVALID DATA
      NLMDH(J,K) = 1                    @REDEFINE HOURLY COUNTER
  360 OUTBUF(J+4,K) = OUTBUF(J+4,K) / NLMDH(J,K)      @COMPUTE MEAN
      DO 450 J=1,24                     @COMPUTE DAILY MEANS...
      IF(ILMD(J).GT.5)  GO TO 450       @INVALID DATA? YES...
      OUTBUF(J+4,4) = -1000.0           @FLAG INVALID DATA
      ILMD(J) = 1                       @REDEFINE DAILY COUNTER
  450 OUTBUF(J+4,4) = OUTBUF(J+4,4) / ILMD(J)    @COMPUTE MEAN
      DO 500 J=1,2
      IF(JLMD(J).GT.0)  GO TO 500       @INVALID DATA? YES...
      OUTBUF(J+2,4) = -1000.0           @FLAG INVALID DATA
      JLMD(J) = 1                       @REDEFINE DAILY COUNTER
  500 OUTBUF(J+2,4) = OUTBUF(J+2,4) / JLMD(J)    @COMPUTE MEAN
C
      NSECTR = IHO(3)                   @SET 'STAT' SECTOR NUMBER
      ISTAT = IOW(MSTAT,8,112,OUTBUF,NSECTR,ICNT)     @WRITE 'STAT' RCD
      IF(ISTAT.NE.0)  GO TO 920         @BAD I/O STATUS? NO...
      WRITE(NWRT,510)  (IHO(J),J=2,3),(OUTBUF(J,4),J=3,4),
     .     ((OUTBUF(J,K),J=5,28),K=1,4)     @DISPLAY HOURLY, DAILY STATS
  510 FORMAT('0DAY = ',I3,'   SECTOR = ',I4,/
     .       '     AVERAGE DAILY WIND VELOCITY = ',F9.2,' KNOTS;',
     .       '  AVERAGE DAILY SOLAR INSOLATION = ',F9.2,' WATTS/SQ.M',/
     .       7X,'TEMP = ',12F9.2,/14X,12F9.2,/
     .       7X,'WIND = ',12F9.2,/14X,12F9.2,/
     .       7X,'QDT =  ',12F9.2,/14X,12F9.2,/
     .       ' DAILY WIND = ',12F9.2,/' DAILY QDT =  ',12F9.2)
  550 IF(IDAY.EQ.MDATE(M))  CALL MONTH(*920)     @WRITE MONTHLY STATS
C
  600 WRITE(NWRT,610)                   @SKIP TO TOP OF NEW PAGE
```

```
  610 FORMAT('1')
      DO 650 I=1,12                          @FOR EACH MONTH...
      JSECTR = LSYR + LST * (I - 1)          @SET 'STAT' MONTH SECTOR
      ISTAT = IOW(MSTAT,16,112,MBUF,JSECTR,ICNT)      @READ MONTH RCD
      IF(ISTAT.NE.0)  GO TO 940              @BAD I/O STATUS? NO...
      WRITE(NWRT,630)  (IMM(J),J=2,3),((MBUF(J,K),J=5,18),K=1,3)
  630 FORMAT('0MONTH = ',I2,'    SECTOR = ',I4,/
     .       7X,'TEMP = ',12F8.2,5X,2F8.2,/
     .       7X,'WIND = ',12F8.2,5X,2F8.2,/
     .       7X,'QDT =  ',12F8.2,5X,2F8.2)
  650 CONTINUE
C
  750 JSECTR = 0                             @INITIALIZE 'STAT' SECTOR POINTER
      ISTAT = IOW(MSTAT,16,112,OUTBUF,JSECTR,ICNT)    @REREAD 1ST 'STAT'
      IF(ISTAT.NE.0) GO TO 940               @BAD I/O STATUS? NO...
      MN = 0                                 @INITIALIZE YEARLY COUNTERS
      MX = 0
      DO 760 I=1,4
  760 OUTBUF(I,2) = 0.0                      @INITIALIZE YEARLY STATISTICS
  770 DO 800 I=1,NYEARS                      @FOR EACH 'MERGE' YEAR...
  780 IF(TMIN(I).GE.1000.0)  GO TO 790       @VALID MINIMUM TEMPERATURE?
      OUTBUF(1,2) = OUTBUF(1,2) + TMIN(I)    @YES. SUM MIN. TEMPERATURES
      OUTBUF(2,2) = OUTBUF(2,2) + TMIN(I)**2.0   @SUM MIN. TEMP. SQUARES
      MN = MN + 1                            @INCREMENT YEARLY MIN. COUNTER
  790 IF(TMAX(I).LE.-1000.0)  GO TO 800      @VALID MAXIMUM TEMPERATURE?
      OUTBUF(3,2) = OUTBUF(3,2) + TMAX(I)    @YES. SUM MAX. TEMPERATURES
      OUTBUF(4,2) = OUTBUF(4,2) + TMAX(I)**2.0   @SUM MAX. TEMP. SQUARES
      MX = MX + 1                            @INCREMENT YEARLY MAX. COUNTER
  800 CONTINUE
      IF(MN.GT.0)  OUTBUF(1,2) = OUTBUF(1,2) / MN      @COMPUTE YEARLY
      IF(MX.GT.0)  OUTBUF(3,2) = OUTBUF(3,2) / MX      @TEMPERATURE MEANS
C
  810 IF(MN.GT.1)  OUTBUF(2,2) = SQRT((OUTBUF(2,2) - MN *
     .    OUTBUF(1,2)**2.0) / (MN - 1))       @COMPUTE STANDARD DEVIATION
      IF(MN.LE.1)  OUTBUF(2,2) = 0.0         @ FOR MINIMUM TEMPERATURE
      IF(MX.GT.1)  OUTBUF(4,2) = SQRT((OUTBUF(4,2) - MX *
     .    OUTBUF(3,2)**2.0) / (MX - 1))       @COMPUTE STANDARD DEVIATION
      IF(MX.LE.1)  OUTBUF(4,2) = 0.0         @ FOR MAXIMUM TEMPERATURE
  820 NYEARS = MIN0(MN,MX)                   @COMPUTE NUMBER OF VALID YEARS
      OUTBUF(1,3) = FLOAT(NYEARS)            @STORE 'STAT' VALID YEARS
      NSECTR = JSECTR                        @SET 'STAT' SECTOR NUMBER
      ISTAT = IOW(MSTAT,8,112,OUTBUF,NSECTR,ICNT)     @WRITE 'STAT' RCD
      IF(ISTAT.NE.0)  GO TO 920              @BAD I/O STATUS? NO...
      WRITE(NWRT,840)  NYEARS,(TMIN(I),I=1,NYRS),OUTBUF(1,2),
     .       OUTBUF(2,2),(TMAX(I),I=1,NYRS),OUTBUF(3,2),OUTBUF(4,2)
  840 FORMAT('1YEARLY STATISTICS:',/   @DISPLAY YEARLY TEMPERATURE STATS
     .       1X,I2,' VALID YEARS OF DATA FOUND',/
     .       7X,'TMIN = ',12F9.2,/
     .       7X,'MEAN TMIN =',F9.2,'    TMIN STANDARD DEVIATION  ',F9.2,/
     .       7X,'TMAX = ',12F9.2,/
     .       7X,'MEAN TMAX =',F9.2,'    TMAX STANDARD DEVIATION  ',F9.2)
      STOP STATS
C
C*****ERROR MESSAGE EXITS
C
  900 WRITE(NWRT,910)  JSECTR,ISTAT
  910 FORMAT('0***MERGE FILE READ ERROR AT SECTOR = ',I4,
```

```
     .       ':  STATUS = ',I2)
          STOP READ
  920 WRITE(NWRT,930)  NSECTR,ISTAT
  930 FORMAT('0***STAT FILE WRITE ERROR AT SECTOR = ',I6,
     .       ':  STATUS = ',I2)
          STOP WRITE
  940 WRITE(NWRT,950)  JSECTR,ISTAT
  950 FORMAT('0***STAT FILE REREAD ERROR AT SECTOR = ',I6,
     .       ':  STATUS = ',I2)
          STOP REREAD
C
C*********************  INTERNAL SUBROUTINES  **********************
C
          SUBROUTINE  MONTH(S)
C       COMPUTES MONTHLY STATISTICS AND WRITES THEM INTO 'STAT' FILE
C
  560 IF(M.LE.0)  GO TO 720                 @INITIALIZATION ONLY? NO...
          DO 700 L=1,3
          DO 590 N=1,NYEARS                 @FOR EACH 'MERGE' YEAR...
  570 IF(NLM(N,L).GT.0)  GO TO 580          @INVALID DATA?
          MBUF(N+4,L) = -1000.0             @YES, FLAG INVALID DATA
          GO TO 590
  580 MBUF(N+4,L) = MBUF(N+4,L) / NLM(N,L)   @NO, COMPUTE MEAN OF DATA
          MBUF(NYRS+5,L) = MBUF(NYRS+5,L) + MBUF(N+4,L)     @SUM YEARLY DATA
          MBUF(NYRS+6,L) = MBUF(NYRS+6,L) + MBUF(N+4,L)**2.0    @SUM SQUARES
          NY(L) = NY(L) + 1                 @INCREMENT YEARS COUNTER
  590 CONTINUE
  600 IF(NY(L).GT.0)  GO TO 620             @INVALID YEARLY SUM? YES...
          MBUF(NYRS+5,L) = -1000.0          @FLAG INVALID MONTHLY DATA
          MBUF(NYRS+6,L) = -1000.0
          NY(L) = 1                         @REDEFINE YEAR COUNTER
  620 MBUF(NYRS+5,L) = MBUF(NYRS+5,L) / NY(L)
          IF(NY(L).GT.1)  MBUF(NYRS+6,L) =
     .    SQRT((MBUF(NYRS+6,L) - NY(L) * MBUF(NYRS+5,L)**2.0) /
     .    (NY(L) - 1))                      @COMPUTE MEANS OVER ALL YEARS
  700 IF(NY(L).LE.1)  MBUF(NYRS+6,L) = 0.0
C
  710 JSECTR = LSYR + LST * (M - 1)         @SET 'STAT' MONTH SECTOR
          IMM(2) = M                        @STORE 'STAT' MONTH INDEX
          IMM(3) = JSECTR                   @STORE 'STAT' MONTH SECTOR
          ISTAT = IOW(MSTAT,8,.12,MBUF,JSECTR,ICNT)  @WRITE MONTHLY STATS
          IF(ISTAT.NE.0)  RETURN 1          @BAD I/O STATUS? NO...
C
  720 DO 740 L=1,3
          NY(L) = 0                         @INITIALIZE YEAR COUNTER
          DO 730 N=1,NYRS
  730 NLM(N,L) = 0                          @INITIALIZE MONTHLY COUNTER
          DO 740 N=5,28
  740 MBUF(N,L) = 0.0                       @INITIALIZE MONTHLY STATS
          M = M + 1                         @INCREMENT MONTH COUNTER
          RETURN
C
          END
```

PROFILE

```
C      PROFILE
C       MODIFIES 'STAT' FILE DATA ('MERGE' FILE STATISTICAL DATA)
C       PER USER REQUIREMENTS TO PRODUCE A 'PROFILE' DATA BASE
C       FOR USE WITH THE DSPA PROGRAM
C
       PARAMETER  LPR=3, LST=4, LSYR=366*LST, NRD=3, NWRT=4
       DOUBLE PRECISION  AWC(3), B(31), FCT, LAMBDA, PSUM, X
       REAL  BUFR(28,4), MBUF(28,4), OUTBUF(28,3), PWC(3)
       REAL  Q(24), QD(12), QM(2), T(24), TM(2), V(24), VD(12), VM(2)
       REAL  ZTABLE(50,2)/0.5000,0.5398,0.5793,0.5987,0.6179,0.6368,
      .                   0.6554,0.6736,0.6915,0.7088,0.7258,0.7422,
      .                   0.7580,0.7734,0.7881,0.8023,0.8159,0.8289,
      .                   0.8413,0.8531,0.8643,0.8749,0.8849,0.8944,
      .                   0.9032,0.9115,0.9192,0.9265,0.9332,0.9394,
      .                   0.9452,0.9505,0.9554,0.9599,0.9641,0.9678,
      .                   0.9713,0.9773,0.9821,0.9861,0.9893,0.9918,
      .                   0.9938,0.9960,0.9970,0.9978,0.9987,0.9999,
      .                   1.0000,1.0000,
      .                   0.00,0.10,0.20,0.25,0.30,0.35,
      .                   0.40,0.45,0.50,0.55,0.60,0.65,
      .                   0.70,0.75,0.80,0.85,0.90,0.95,
      .                   1.00,1.05,1.10,1.15,1.20,1.25,
      .                   1.30,1.35,1.40,1.45,1.50,1.55,
      .                   1.60,1.65,1.70,1.75,1.80,1.85,
      .                   1.90,2.00,2.10,2.20,2.30,2.40,
      .                   2.50,2.65,2.75,2.85,3.00,3.60,
      .                   4.00,1000.0/
       INTEGER  ID(3)/' ','WIND','QDT '/, IDIST(12,31,3), IHD(4), LH(6)
       INTEGER  MDATE(13)/1,32,60,91,121,152,182,213,244,274,305,335,366/
       INTEGER  MPROFL(2)/'12',' '/, MSTAT(2)/'11',' '/, NDAY(3)/14,9,19/
       NAMELIST/INPT/ ALPHAQ,ALPHAT,ALPHAV,ALPHHV,ALPHLQ,ALPHLV,
      .               LH,PHV,PLQ,PLV,PQ,PT,PV/
       EQUIVALENCE  (BUFR(1,1),IHD(1)), (BUFR(5,1),T(1))
       EQUIVALENCE  (BUFR(5,2),V(1)), (BUFR(5,3),Q(1))
       EQUIVALENCE  (BUFR(3,4),VDAY), (BUFR(4,4),QDAY)
       EQUIVALENCE  (BUFR(5,4),VD(1)), (BUFR(17,4),QD(1))
       EQUIVALENCE  (BUFR(1,1),OUTBUF(1,1)), (MBUF(17,1),TM(1))
       EQUIVALENCE  (MBUF(17,2),VM(1)), (MBUF(17,3),QM(1))
C
       JSECTR = 0                          @INITIALIZE INPUT SECTOR NUMBER
       M = 1                               @INITIALIZE MONTH COUNTER
       ISTAT = IOW(MSTAT,16,112,BUFR,JSECTR,ICNT)      @READ FIRST 'STAT'
       IF(ISTAT.NE.0)  GO TO 820           @BAD I/O STATUS?
       YEARS = BUFR(1,3)                   @NO. STORE NUMBER OF YEARS USED
       IYRS = IFIX(YEARS)
       WRITE(NWRT,20)  IHD(1),IYRS
   20  FORMAT('1STATISTICAL PROFILE FOR STATION ',A5,' (',I2,' YEARS)'//)
C
  100  DO 300 IDAY = 1,365                 @FOR EACH DAY OF THE YEAR...
       IF(IDAY.NE.MDATE(M))  GO TO 200          @FIRST DAY OF MONTH?
       IF(LWC.GT.0)  CALL WRSTCS           @YES, ADD 'WORST CASE' DATA
       READ(NRD,INPT,ERR=800,END=800)      @READ USER INPUT
       LWC = LH(4) + LH(5) + LH(6)         @SET 'WORST CASE' FLAG
       AWC(1) = ALPHLQ                     @STORE LOW RADIATION CONFIDENCE
       AWC(2) = ALPHLV                     @STORE LOW WIND CONFIDENCE LEVEL
       AWC(3) = ALPHHV                     @STORE HIGH WIND CONFIDENCE LEVEL
```

```
      PWC(1) = PLQ                          @STORE LOW RADIATION PROPORTION
      PWC(2) = PLV                          @STORE LOW WIND PROPORTION
      PWC(3) = PHV                          @STORE HIGH WIND PROPORTION
      CALL TOLER(ALPHAT,PT,CT)              @TEMPERATURE TOLERANCE FACTOR
      CALL TOLER(ALPHAV,PV,CV)              @WIND VELOCITY TOLERANCE FACTOR
      CALL TOLER(ALPHAQ,PQ,CQ)              @SOLAR RADIATION TOLERANCE FACTOR
      JSECTR = LSYR + LST * (M - 1)         @SET 'STAT' MONTH SECTOR
      ISTAT = IOW(MSTAT,16,112,MBUF,JSECTR,ICNT)      @READ 'STAT' MONTH
      IF(ISTAT.NE.0)  GO TO 840             @BAD I/O STATUS? NO...
      DLHT = LH(1) * CT * TM(2)             @SET LOW/HIGH DELTA/SCALE FACTORS
      RLHV = 1.0 + LH(2) * CV * VM(2) / VM(1)
      RLHQ = 1.0 + LH(3) * CQ * QM(2) / QM(1)
      WRITE(NWRT,120)  M,DLHT,RLHV,RLHQ     @DISPLAY MONTHLY PARAMETERS
  120 FORMAT('1***FOR MONTH ',I2,', DELTA/SCALE FACTORS =',/
     *       9X,'TEMP: ',F9.4,'   WIND: ',F6.4,'   QDT: ',F6.4)
      WRITE(NWRT,INPT)
      WRITE(NWRT,150)
  150 FORMAT(////)
      M = M + 1                             @INCREMENT MONTH COUNTER
C
  200 JSECTR = IABS(LST * (IDAY - 1))       @SET 'STAT' SECTOR NUMBER
      ISTAT = IOW(MSTAT,16,112,BUFR,JSECTR,ICNT)      @READ 'STAT' RCD
      IF(ISTAT.NE.0)  GO TO 820             @BAD I/O STATUS? NO...
      DO 210 I=1,24                         @FOR EACH HOUR OF THE DAY...
      IF(T(I).GT.-1000.0)  T(I) =           @MODIFY TEMPERATURE DATA
     *                     T(I) + DLHT
      IF(V(I).GE.0.0)  V(I) = RLHV * V(I)   @MODIFY WIND VELOCITY DATA
  210 IF(Q(I).GE.0.0)  Q(I) = RLHQ * Q(I)   @MODIFY INSOLATION DATA
      ISECTR = IABS(LPR * (IDAY - 1))       @SET 'PROFILE' SECTOR NUMBER
      IHD(3) = ISECTR                       @STORE OUTPUT SECTOR NUMBER
      ISTAT = IOW(MPROFL,8,84,OUTBUF,ISECTR,ICNT)     @WRITE 'PROFILE'
      IF(ISTAT.NE.0)  GO TO 860             @BAD I/O STATUS? NO...
      WRITE(NWRT,240)  IHD(2),IHD(3),((OUTBUF(I,J),I=5,28),J=1,3)
  240 FORMAT('0DAY = ',I3,'   SECTOR = ',I4,/
     *       7X,'TEMP = ',12F9.2,/14X,12F9.2,/
     *       7X,'WIND = ',12F9.2,/14X,12F9.2,/
     *       7X,'QDT =  ',12F9.2,/14X,12F9.2)
      IF(LWC.LE.0)  GO TO 300               @'WORST CASE' PROFILE REQUESTED?
      J = IDAY - MDATE(M-1) + 1             @YES, COMPUTE DAY OF MONTH
      DO 270 I=1,IYRS                       @CHECK 'WORST CASE' DATA...
      IF(QD(I).LT.0.0)  GO TO 260           @ERRONEOUS INSOLATION DATA? NO...
      QDY = PLQ * (IYRS * QDAY - QD(I)) / (IYRS - 1)
      IF(QD(I).LE.QDY)  IDIST(I,J,1) = 1    @LOW SOLAR RADIATION PERIOD
  260 IF(VD(I).LT.0.0)  GO TO 270           @ERRONEOUS WIND DATA? NO...
      VDY = PLV * (IYRS * VDAY - VD(I)) / (IYRS - 1)
      IF(VD(I).LE.VDY)  IDIST(I,J,2) = 1    @LOW WIND VELOCITY PERIOD
      VDY = PHV * (IYRS * VDAY - VD(I)) / (IYRS - 1)
      IF(VD(I).GE.VDY)  IDIST(I,J,3) = 1    @HIGH WIND VELOCITY PERIOD
  270 CONTINUE
  300 CONTINUE
C
  500 IF(LWC.GT.0)  CALL WRSTCS             @ADD 'WORST CASE' DATA
      JSECTR = LST * 365                    @SET LAST 'STAT' DAY SECTOR
      ISTAT = IOW(MSTAT,16,112,BUFR,JSECTR,ICNT)      @READ LAST 'STAT'
      IF(ISTAT.NE.0)  GO TO 820             @BAD I/O STATUS? NO...
      ISECTR = LPR * 365                    @SET LAST 'PROFILE' DAY SECTOR
      IHD(3) = ISECTR                       @STORE OUTPUT SECTOR NUMBER
```

```
      ISTAT = IOW(MPROFL,8,84,OUTBUF,ISECTR,ICNT)     @WRITE 'PROFILE'
      IF(ISTAT.NE.0)  GO TO 860                       @BAD I/O STATUS? NO...
      IMD(2) = 99999                                  @SET END-OF-FILE RECORD MARK
      ISECTR = LPR * 366                              @SET END-OF-FILE SECTOR
      IMD(3) = ISECTR                                 @STORE OUTPUT SECTOR NUMBER
      ISTAT = IOW(MPROFL,8,84,OUTBUF,ISECTR,ICNT)     @WRITE EOF
      IF(ISTAT.NE.0)  GO TO 860                       @BAD I/O STATUS?
      STOP PROFIL                                     @NO.
C
C*****ERROR MESSAGE EXITS
C
  800 WRITE(NWRT,810)  M
  810 FORMAT('0***ERROR IN PROFILE USER INPUT FOR MONTH ',I2)
      WRITE(NWRT,INPT)
      STOP INPUT
  820 WRITE(NWRT,830)  IDAY,JSECTR
  830 FORMAT('0***ERROR IN READING STAT DAY: DAY =',I4,', SECTOR =',I5)
      STOP STAT
  840 WRITE(NWRT,850)  M,JSECTR
  850 FORMAT('0***ERROR IN READING STAT MONTH: MONTH =',I3,
     .       ', SECTOR =',I5)
      STOP MONTH
  860 WRITE(NWRT,870)  IDAY,ISECTR
  870 FORMAT('0***ERROR IN WRITING PROFILE DAY: DAY =',I4,
     .       ', SECTOR =',I5)
      STOP WRITE
C
C**********************  INTERNAL SUBROUTINES  ************************
C
      SUBROUTINE  TOLER(A,P,CL)
C      COMPUTES INVERSE NORMAL ERROR FUNCTION AND
C      RETURNS TOLERANCE LIMITS FACTOR
      Z = AMAX1(A,1.0-A)                              @INVERSE ERROR FUNCTION FOR 1-A
      CALL SLUP(Z,ZA,FDOT,ZTABLE(1,1),ZTABLE(1,2),50,1)
      IF(A.GT.0.50)  ZA = -ZA
      Z = AMAX1(P,1.0-P)                              @INVERSE ERROR FUNCTION FOR P
      CALL SLUP(Z,ZP,FDOT,ZTABLE(1,1),ZTABLE(1,2),50,1)
      IF(P.LT.0.50)  ZP = -ZP
      AL = 1.0 - ZA**2.0 / (2.0 * (YEARS - 1.0))
      BL = ZP**2.0 - ZA**2.0 / YEARS
      CL = (ZP + SQRT(ZP**2.0 - AL * BL)) / AL    @TOLERENCE FACTOR
      RETURN
C
C
      SUBROUTINE  WRSTCS
C      ADDS 'WORST CASE' DATA TO PAST PROFILE MONTH
      IF(IDAY.LE.1)  RETURN                           @FIRST DAY OF YEAR? NO...
      N = M - 1                                       @SAVE PAST MONTH FLAG
      NDYS = MDATE(M) - MDATE(N)                      @GET NUMBER OF DAYS IN MONTH
      WRITE(NWRT,10)  N
   10 FORMAT('1***WORST CASE DATA CHANGES FOR MONTH ',I2)
      DO 500 K=1,3                                    @FOR EACH 'WORST CASE' VARIABLE..
      IF(LH(K+3).EQ.0)  GO TO 300                     @'WORST CASE' ANALYSIS REQUESTED?
      CNT = 0.0                                       @YES, INITIALIZE POISSON COUNTER
      SUM = 0.0                                       @INITIAILIZE POISSON SUM
      DO 60 I=1,IYRS                                  @FOR EACH YEAR...
      DO 60 J=1,NDYS                                  @FOR EACH DAY OF THE MONTH...
```

```
      SUM = SUM + IDIST(I,J,K)                    @COMPUTE POISSON SUM
      IF(IDIST(I,J,K).EQ.0)  GO TO 60             @GOOD DAY?
      IF(J.EQ.1)  GO TO 50                        @NO. FIRST DAY OF MONTH?
      IF(IDIST(I,J-1,K).EQ.1)  GO TO 60           @NO. BAD DAY AFTER BAD DAY?
   50 CNT = CNT + 1                               @NO. INCREMENT POISSON COUNTER
   60 CONTINUE
      LAMBDA = (SUM - CNT) / CNT                  @SET POISSON LIKELIHOOD ESTIMATE
C
  100 FCT = 1.0D0                                 @INITIALIZE FACTORIAL
      DO 120 J=1,NDYS                             @FOR EACH DAY OF THE MONTH...
      FCT = FCT * J                               @COMPUTE FACTORIAL
  120 B(J) = LAMBDA**J / FCT
      PSUM = DEXP(LAMBDA)                         @COMPUTE POISSON SUM
      X = 1.0D0                                   @INITIAILZE POISSON DETERMINATE
      IF(AWC(K).GE.1.0D0)  GO TO 160              @REQUEST FOR BAD MONTH? NO...
      IF(X/PSUM.GE.AWC(K))  GO TO 300             @ZERO BAD DAYS? NO...
      DO 150 J=1,NDYS                             @FOR EACH DAY OF THE MONTH...
      NR = J                                      @STORE NUMBER OF BAD DAYS
      X = X + B(J)                                @COMPUTE NEW DETERMINATE
  150 IF(X/PSUM.GE.AWC(K))  GO TO 200             @CORRECT NUMBER OF BAD DAYS?
  160 NR = NDYS                                   @NO. ENTIRE MONTH TO BE USED
C
  200 KDAY = MDATE(N) + NDAY(K) - (NR - 1) / 2
      KDAY = MAX0(KDAY,MDATE(N))                  @SET BAD DATA START DATE
      KX = 2                                      @SET VARIABLE SELECTOR
      IF(K.EQ.1)  KX = 3
      DO 270 I=1,NR                               @FOR EACH BAD DAY...
      JDAY = KDAY + I - 1                         @SET 'STAT'/'PROFILE' DATE
      JSECTR = IABS(LST + (JDAY - 1))             @SET 'STAT' SECTOR NUMBER
      ISTAT = IOW(MSTAT,16,112,MBUF,JSECTR,ICNT)      @READ 'STAT' RCD
      IF(ISTAT.NE.0)  GO TO 820                   @BAD I/O STATUS?
      ISECTR = IABS(LPR + (JDAY - 1))             @NO. SET 'PROFILE' SECTOR NUMBER
      ISTAT = IOW(MPROFL,16,84,OUTBUF,ISECTR,ICNT)    @READ 'PROFIL' RCD
      IF(ISTAT.NE.0)  GO TO 840                   @BAD I/O STATUS? NO...
      DO 240 J=5,28                               @FOR EACH HOUR OF THE DAY...
      IF(OUTBUF(J,KX).LE.-1000.0)  GO TO 240      @GOOD DATA? YES...
      IF(K.LT.3)  OUTBUF(J,KX) =                  @COMPUTE LOW 'PROFILE' DATA
     .            AMIN1(OUTBUF(J,KX),PWC(K)*MBUF(J,KX))
      IF(K.EQ.3)  OUTBUF(J,KX) =                  @COMPUTE HIGH 'PROFILE' DATA
     .            AMAX1(OUTBUF(J,KX),PWC(K)*MBUF(J,KX))
  240 CONTINUE
      ISTAT = IOW(MPROFL,8,84,OUTBUF,ISECTR,ICNT)     @WRITE 'PROFILE'
      IF(ISTAT.NE.0)  GO TO 860                   @BAD I/O STATUS? NO...
      WRITE(NWRT,260)  IMD(2),IMD(3),ID(KX),(OUTBUF(J,KX),J=5,28)
  260 FORMAT('0DAY = ',I3,'   SECTOR = ',I4,/
     .       7X,A4,' = ',12F9.2,/14X,12F9.2)
  270 CONTINUE
C
  300 DO 400 I=1,12
      DO 400 J=1,31
  400 IDIST(I,J,K) = 0                            @ZERO-FILL POISSON TABLE
  500 CONTINUE
      RETURN
C
  820 WRITE(NWRT,830)  JDAY,JSECTR
  830 FORMAT('0***ERROR IN READING STAT DAY: DAY =',I4,', SECTOR =',I5)
      STOP RRSTAT
```

```
  840 WRITE(NWRT,850)  JDAY,ISECTR
  850 FORMAT('0***ERROR IN READING PROFILE DAY: DAY =',I3,
     .        ', SECTOR =',I5)
      STOP REREAD
  860 WRITE(NWRT,870)  JDAY,ISECTR
  870 FORMAT('0***ERROR IN REWRITING PROFILE DAY: DAY =',I4,
     .        ', SECTOR =',I5)
      STOP REWRIT
C
      END
```

```
          AXRS.     121470-0001          R. DOCKEN.
.                   121970-0002          R. DOCKEN.                12-21-70
.                   122370-0003          R. DOCKEN.                12-23-70
.                   010771-0004          R. DOCKEN.                01-07-71
          CHAR      057,072.
. ************************************************************************
. *                                                                      *
. *    J = IOW (FILE,JFUNCT, LENGTH, BUFFER,JDRUM, ICOUNT)               *
. *                                                                      *
. *    J = IOW (FILE,JFUNCT, LENGTH, BUFFER,JDRUM, ICOUNT, ID, FIND)     *
. *                                                                      *
. * THIS PROGRAM PROVIDES AN INTERFACE FOR THE FORTRAN CALLER TO         *
. * EXEC REQUEST IOWS TO FACILITATE RANDOM ACCESS ON THE DRUMS AND       *
. * FASTRANDS AND TAPES.                                                 *
. *                                                                      *
. *       <FILE>    THE 2-WORD INTERNAL FILE NAME (LJSF) OF THE FILE     *
. *                 TO BE USED.  IT MUST BE ASSIGNED.                    *
. *                                                                      *
. *       <JFUNCT>  ONE OF THE RECOGNIZED I/O CODES.  010=WRITE,         *
. *                 020=READ, AND 040=REWIND ARE THE MOST COMMON         *
. *                 ONES.  OTHERS ARE DESCRIBED IN THE PRM AND THE       *
. *                 EXEC REFERENCE CARD.                                 *
. *                                                                      *
. *       <LENGTH>  THE MAXIMUM NUMBER OF WORDS TO TRANSFER.             *
. *                                                                      *
. *       <BUFFER>  THE CORE BUFFER USED IN TRANSFERS.  IT MUST BE       *
. *                 SPECIFIED EVEN FOR NO-DATA OPERATIONS LIKE A         *
. *                 TAPE REWIND.                                         *
. *                                                                      *
. *       <JDRUM>   THE I/O STARTING ADDRESS FOR RANDOM ACCESS           *
. *                 FILES.                                               *
. *                                                                      *
. *       <ICOUNT>  THIS IS SET BY IOW TO THE ACTUAL NUMBER OF DATA      *
. *                 WORDS COPIED.                                        *
. *                                                                      *
. *       <ID>      THE SENTINEL FOR SEARCHES.                           *
. *                                                                      *
. *       <FIND>    THE LOCATION OF THE FIND FOR A SEARCH.               *
. *                                                                      *
. * THE FUNCTION RETURNS THE STATUS CODE.                                *
. ************************************************************************
.
$(2)      LIT.
SX11      +         0.                  SAVE X11 FOR DEBUG TRACING.
PKT       RES       8.
PK        EQU       PKT-1.
.
$(1).
.
IOW.      S         X11,SX11.           SAVE <X11>.
          DL        A0,.0,X11.          <FILE>.
          DS        A0,PK+1.
          SZ        PK+3.               CLEAR INTERRUPT CODES.
          L         A0,.1,X11.          <JFUNCT>.
          S,TI      A0,PK+4.
          LXI       A0,.2,X11.          <LENGTH>.
```

```
        LXM     A0,3,X11.       <BUFFER>.                       12-19-70
        L       A1,=4,X11.      <JDRUM>.
        DS      A0,PK+5.
        TZ,H1   6,X11.          IS <ID, FIND> SPECIFIED?
        J       $+3.            NO!
        L       A0,=6,X11.      <ID>.
        S       A0,PK+7.
        L,U     A0,PKT.
        ER      IOW$.
        L,H2    A0,PK+4.        <ICOUNT>.
        S       A0,=5,X11.
        TZ,H1   6,X11.          IS <ID, FIND> SPECIFIED?
        J       EXIT$6.         NO!
        SZ,T1   PK+8.
        L       A0,PK+8.        <FIND>.
        S       A0,=7,X11.
        L,S1    A0,PK+4.        <J>.
        J       9,X11.                                          01-07-71
EXIT$6  L,S1    A0,PK+4.        <J>.
        J       7,X11.                                          01-07-71
        END.
```

SLUP

```
      SUBROUTINE  SLUP(X,Y,YDOT,XS,YS,LN,MORD)
C
C     IDENTIFICATION...
C        SINGLE PRECISION SUBROUTINE  SLUP/LAGRANGIAN INTERPOLATION
C        BUD POULSON (JPL)
C        M. MILANE (JPL)
C        FORTRAN IV
C
C     PURPOSE...
C        SINGLE LOOK UP AND LAGRANGIAN INTERPOLATION.
C        GIVEN A VALUE OF AN INDEPENDENT VARIABLE, X, FIND F(X) FROM
C          A GIVEN TABULATED FUNCTION OF X(I) VS. Y(I), AND OPTIONALLY,
C          THE FIRST DERIVATIVE.
C
C     RESTRICTIONS...
C        NO EXPLICIT RESTRICTIONS, I.E.,
C          ANY ORDER INTERPOLATION
C          ANY SIZE TABLE
C        THE INDEPENDENT VARIABLE MAY BE MONOTONICALLY INCREASING OR
C          DECREASING
C
C     METHOD...
C        LAGRANGE'S FORMULA IS USED WITH EXTRAPOLATION FOR POINTS
C        OUTSIDE THE TABLE.  A BINARY SEARCH IS USED, WHICH MAY BE THE
C          SOLE PURPOSE OF USING SLUP.
C
C     CALLING SEQUENCE...
C        X, INDEPENDENT VARIABLE  (INPUT)
C        Y, INTERPOLATED VALUE OF DEPENDENT VARIABLE, Y(X)  (OUTPUT)
C        YDOT, CALCULATED FIRST DERIVATIVE, Y'(X)  (OUTPUT)
C        XS, TABULATED INDEPENDENT VARIABLE   (INPUT)
C        YS, TABULATED DEPENDENT VARIABLE   (INPUT)
C        LN, NUMBER OF TABULATED POINTS  (INPUT)
C          IF ONE DESIRES TO FIND K IN THE SEQUENCE X(1),X(2),X(3),
C             ...,X(K),X(K+1),...,X(LN) WHEN THE GIVEN X IS BETWEEN
C             X(K) AND X(K+1), THEN MAKE LN NEGATIVE AND THE VALUE
C             K COMES BACK IN LN.
C        MORD, ORDER OF INTERPOLATION, USING MORD+1 POINTS IN
C          LAGRANGE'S FORMULA  (INPUT),
C          MAKE MORD NEGATIVE IF THE FIRST DERIVIATIVE IS DESIRED.
C
      DATA  ZERO/0.0/, ONE/1.0/
      DIMENSION  XS(2), YS(2)
C
      N=IABS(LN)
C        INSURE THAT THE NUMBER OF POINTS USED TO INTERPOLATE IS LESS
C        THAN OR EQUAL TO THE NUMBER OF DATA POINTS SUPPLIED
      M=MINO(IABS(MORD)+1,N)
C        ASSUME INDEPENDENT VARIABLE IS MONOTONICALLY DECREASING
      K=1
      L=N
C        MONOTONICALLY INCREASING OR DECREASING?
      IF(XS(1)-XS(2)) 5,5,10
C        INDEPENDENT VARIABLE IS MONOTONICALLY INCREASING
    5 K=N
      L=1
```

```
C         NOW BEGIN BINARY SEARCH FOR APPROPRIATE SUB-INTERVAL
   10 J=(K+L)/2
      IF(X-XS(J)) 15,35,20
   15 K=J
      GO TO 25
   20 L=J
   25 IF(IABS(K-L)-1) 10,30,10
   30 J=MAXO(K,L)
C         SET INDICES OF SUB-INTERVAL CONTAINING POINT GIVEN
C         (ALSO INSURE INDICES ARE WITHIN THOSE ALLOWED, I.E., 1 TO N)
   35 NN=MAXO(1,J-M/2)
      MM=MINO(N,NN+M-1)
      NN=MM-M+1
C         IS INTERPOLATION REQUESTED AT THE GIVEN POINT?
      IF(LN) 95,95,40
   40 Y=ZERO
      YDOT=ZERO
      DO 90 J=NN,MM
      T=YS(J)
      K=0
      DO 60 I=NN,MM
      IF(I-J) 45,60,45
   45 R=X-XS(I)
      IF(R) 55,50,55
   50 R=ONE
      K=1
      IF(MORD) 55,55,90
   55 T=T*R/(XS(J)-XS(I))
   60 CONTINUE
      IF(K) 70,70,65
   65 YDOT=YDOT+T
      GO TO 90
   70 Y=Y+T
C         IS THE DERIVATIVE REQUESTED AT THE GIVEN POINT?
      IF(MORD) 75,75,90
   75 DO 85 I=NN,MM
      IF(I-J) 80,85,80
   80 YDOT=YDOT+T/(X-XS(I))
   85 CONTINUE
   90 CONTINUE
      GO TO 100
C         BINARY SEARCH ONLY WAS REQUESTED.  RETURN SUB-INTERVAL INDICES
   95 LN=NN
      MORD=MM
  100 RETURN
C
      END
```

☆U.S. GOVERNMENT PRINTING OFFICE: 1977-701-783/212

NASA-JPL-Coml., L.A., Calif.