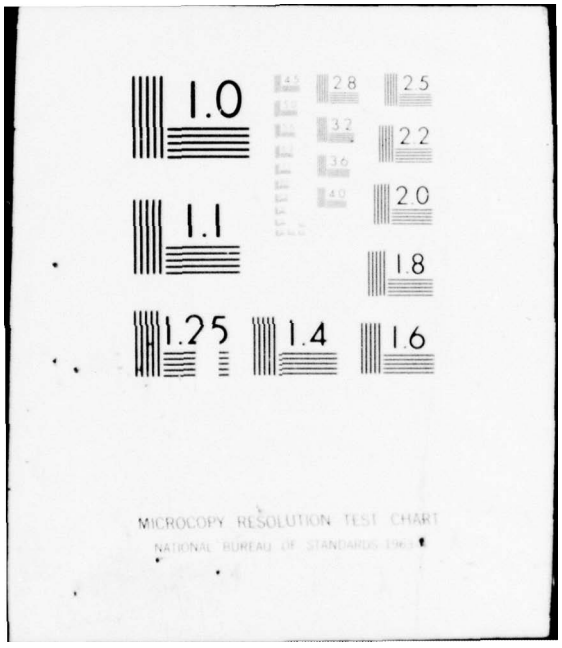
AD-A038 892 NAVAL SEA SYSTEMS COMMAND WASHINGTON D C F/G 17/1
NAVSEA OCEAN ENVIRONMENTAL ACOUSTIC DATA BANK - NAVDAR - IN SUP--ETC(U)
DEC 75

UNCLASSIFIED NAVSEA-06H1/036-EVA/MOST- NL

1 OF 2
AD A038892




1.0
1.1
1.25
1.4
1.6
2.8
2.5
3.2
2.2
3.6
2.0
4.0
1.8
MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS 1963

AD A038892

SEA 06H1/036-EVA/MOST-5

1 DECEMBER 1975

NAVSEA OCEAN ENVIRONMENTAL ACOUSTIC DATA BANK

—NAVDAB—

IN SUPPORT OF

MOBILE SONAR TECHNOLOGY DEVELOPMENT

VOLUME 4



NAVAL SEA SYSTEMS COMMAND
DEPARTMENT OF THE NAVY
WASHINGTON, D.C. 20362

AD No.
DDC FILE COPY

DDC
APR 28 1977
B

Approved for public release; distribution unlimited.

DISTRIBUTION STATEMENT A
Approved for public release;
Distribution Unlimited

SEA 06H1/036-EVA/MOST-5

1 DECEMBER 1975

NAVSEA OCEAN ENVIRONMENTAL ACOUSTIC DATA BANK

—NAVDAB—

IN SUPPORT OF

MOBILE SONAR TECHNOLOGY DEVELOPMENT

VOLUME 4

ACCESSION for
NTIS White Section
DDC Buff Section
UNANNOUNCED
JUSTIFICATION
BY
DISTRIBUTION/AVAILABILITY CODES
Dist. AVAIL. and/or SPECIAL
A

NAVAL SEA SYSTEMS COMMAND
DEPARTMENT OF THE NAVY
WASHINGTON, D.C. 20362

DDC
RECEIVED
APR 28 1977
A

Approved for public release; distribution unlimited.

ADMINISTRATIVE STATEMENT

The NAVSEA Ocean Environmental Acoustic Data Bank (NAVDAB) is a joint effort of the Naval Undersea Center (NUC), the Naval Underwater Systems Center (NUSC), the Naval Research Laboratory (NRL), and the Naval Oceanographic Office (NAVOCEANO), under the sponsorship of the Sonar Technology Office of the Naval Sea Systems Command (Task Area SF 52 552 601). Each of these organizations is represented in the NAVDAB Steering Group. The Steering Group has the responsibility for development of the data bank, which has been installed at NUC/San Diego, California, NUSC/New London, Connecticut, and NAVOCEANO/Washington, D.C.

The NAVDAB Steering Group is grateful to G. F. Anderson of Computer Sciences Corporation and G. E. Miller of Arthur D. Little, Inc., for their assistance on this project.

Released by:

L. A. Smothers (NUC)
NAVDAB Chairman

Sponsoring Authority:

C. D. Smith
NAVSEA

A. P. Franceschetti
NAVSEA

Steering Group

F. R. DiNapoli (NUSC)

K. V. Mackenzie (NAVOCEANO)

R. H. Ferris (NRL)

L. C. Maples (NUSC)

W. H. Geldes (NAVOCEANO)

S. R. Santaniello (NUSC)

R. F. Hosmer (NUC)

J. A. Whitney (NUC)

W. A. Kuperman (NRL)

(14) SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE	READ INSTRUCTIONS BEFORE COMPLETING FORM
1 REPORT NUMBER: NAVSEA 06H1/036-EVA/MOST-5	2 GOVT ACCESSION NO. / 3 RECIPIENT'S CATALOG NUMBER (9)
(6) 4 TITLE (and Subtitle): NAVSEA OCEAN ENVIRONMENTAL ACOUSTIC DATA BANK - NAVDAB - IN SUPPORT OF MOBILE SONAR TECHNOLOGY DEVELOPMENT VOLUME 4.	5 TYPE OF REPORT & PERIOD COVERED: Final Rept. / 6 PERFORMING ORG. REPORT NUMBER
7 AUTHOR(s): NAVDAB Steering Group	8 CONTRACT OR GRANT NUMBER(s): SF 52-552-601
9 PERFORMING ORGANIZATION NAME AND ADDRESS: Naval Undersea Center, Code 3073, San Diego, CA. 92132	10 PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS: Task No. 19324
11 CONTROLLING OFFICE NAME AND ADDRESS: Naval Sea Systems Command (NAVSEA 06H1-4), Washington, D.C. 20362	(11) 12 REPORT DATE: 1 December 1975 / 13 NUMBER OF PAGES
14 MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office): (12) 162 p.	15 SECURITY CLASS. (of this report): Unclassified / 15a DECLASSIFICATION/DOWNGRADING SCHEDULE

16 DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17 DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

(16) F52552 (17) SF52552601

18 SUPPLEMENTARY NOTES

Prepared in cooperation with the NAVDAB Steering Group.

19 KEY WORDS (Continue on reverse side if necessary and identify by block number)

Data Bank
Underwater Acoustics
Environmental data
Computer Programs

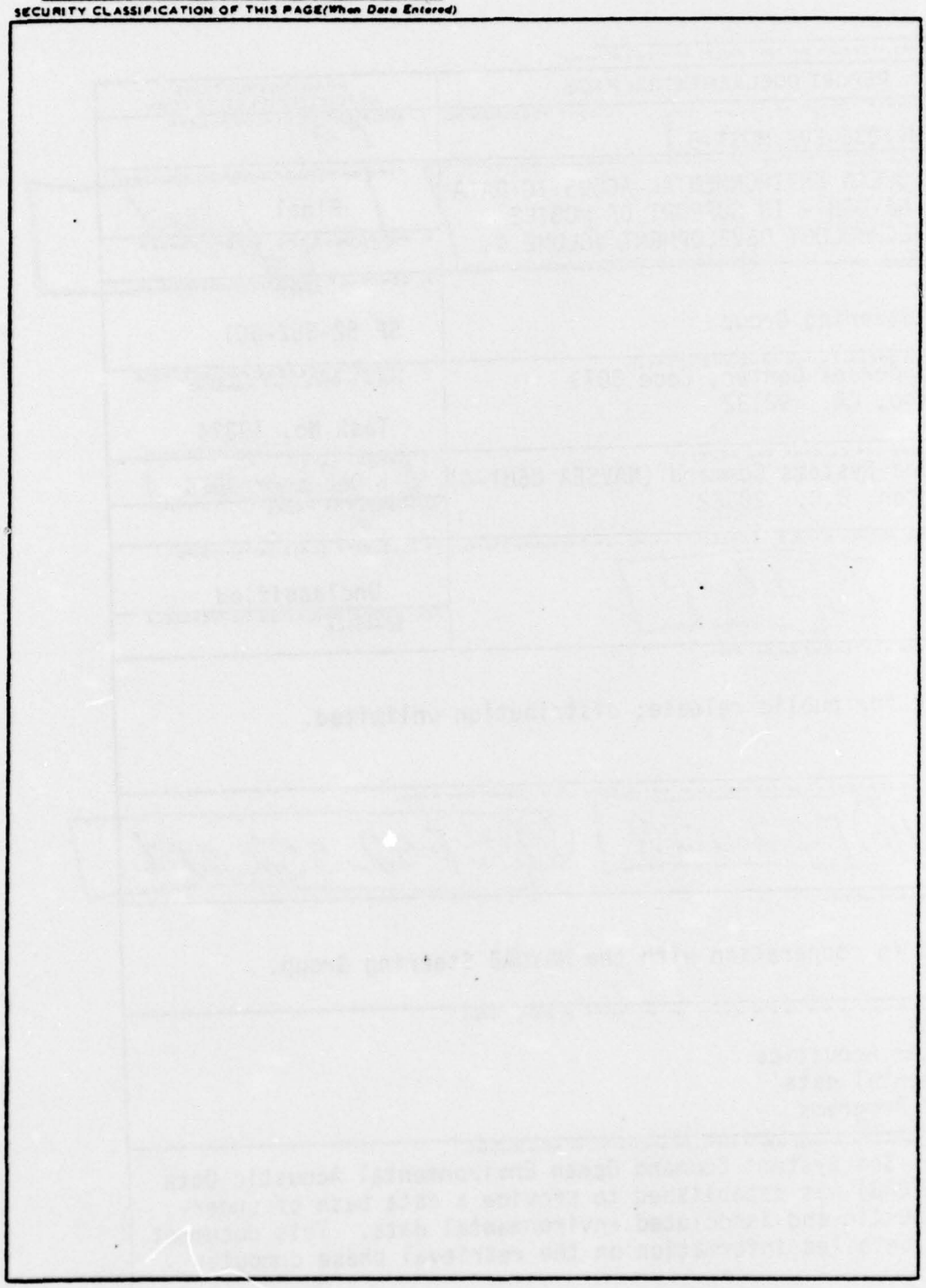
20 ABSTRACT (Continue on reverse side if necessary and identify by block number)

The Naval Sea Systems Command Ocean Environmental Acoustic Data Bank (NAVDAB) was established to provide a data base of underwater acoustic and associated environmental data. This document provides detailed information on the retrieval phase computer programs.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

391 345



DEPARTMENT OF THE NAVY
NAVAL SEA SYSTEMS COMMAND
WASHINGTON, D.C. 20362

FOREWORD

The Naval Sea Systems Command Ocean Environmental Acoustic Data Bank (NAVDAB) was established to provide a data base for the development and validation of environmental acoustic models for mobile sonar application. NAVDAB is unique in that it is primarily for environmental and acoustic data taken concurrently and it is the only data bank of this type operated for mobile sonar application.

A set of five reports have been prepared to describe how NAVDAB works and how to use it. This is volume 4 of the set. The individual reports cover the following topics:

Volume 1 User's Guide to Retrieval
Volume 2 Input Format Guide
Volume 3 Details of Creation Phase
Volume 4 Details of Retrieval Phase
Volume 5 Details of Miscellaneous Support Programs

Governmental and industrial activities desiring to submit or retrieve environmental acoustic data should contact:

Naval Undersea Center
Code 3073
San Diego, CA - 92132

Naval Underwater Systems Center
Code TALA
New London Laboratory
New London, CT - 06320

Naval Oceanographic Office
Code 3440
Washington, D.C. - 20373

Copies of the five reports can be obtained through the Defense Documentation Center, Defense Supply Agency, Cameron Station, Alexandria, VA - 22314.

C.D. Smith

C.D. Smith, Director
Sonar Technology Office,
06H1/036

CONTENTS

Section		Page
1.0	INTRODUCTION	1-1
2.0	DESCRIPTION OF THE DATA BASE	2-1
3.0	DESCRIPTION OF THE RETRIEVAL PROCESS	3-1

Appendices

A.	Computer Program Listings	A-1
B	Current NAVDAB Steering Group	B-1

LIST OF ILLUSTRATIONS

Figure		
2-1	NAVDAB Data Base Structure	2-2
3-1	Flowcharts for Retrieval Phase	3-2
3-2		3-3
3-3		3-4
A-1	Retrieval Program Cross-Reference Map	A-4

SECTION 1.0

INTRODUCTION

The Naval Sea Systems Command (NAVSEA) Ocean Environmental Acoustic Data Bank (NAVDAB) was established to provide a data base for the development and validation of environmental acoustic models for mobile sonar applications. The purpose of this document is to provide details of the computer programs which access the NAVDAB data base and retrieve data from it.

This is Volume 4 of the five-volume set covering the NAVDAB computer program documentation. The individual volumes cover the following topics:

Volume 1. User's Guide to Retrieval

Volume 2. Input Format Guide

Volume 3. Details of Creation Phase

Volume 4. Details of Retrieval Phase

Volume 5. Details of Miscellaneous Support Programs

Copies of these documents may be obtained from the Defense Documentation Center.

SECTION 2.0

DESCRIPTION OF THE DATA BASE

Many factors affect the design of a data base and retrieval system. In the case of NAVDAB a large number and variety of data parameters were considered. To meet this requirement an open-ended tabular format, with variable record length, was selected to accommodate the initial sets of parameters and data and allow for expansion.

Another primary consideration in the data base design was the need for rapid access to all elements of all data. The data base must be suited to a versatile, efficient, random-access, retrieval system. Random-access retrieval techniques dictate, to a great extent, how data are structured on a storage device.

Other requirements of the NAVDAB system include: safeguard of classified information; allowance for operation in either batch or time-sharing mode; ease of updating and maintenance; and use of a universally-accepted computer language, as well as sufficient modularity, for ready adaptation to the different types of computers on which the system may be implemented.

In view of all these considerations, the data organization shown in Figure 2-1 was developed. It is a multilevel, hierarchically-organized, partially circular, linked-list structure, with large-capacity disk for primary storage. Four levels are involved:

1. EXPERIMENT — The highest level, pertains to the overall program of measurements — such as AMOS, FASOR I (Forward Area Sonar Research)

2. CRUISE — The second level is applicable to natural subsets of an EXPERIMENT, such as individual AMOS cruises or measurements taken within a specified area

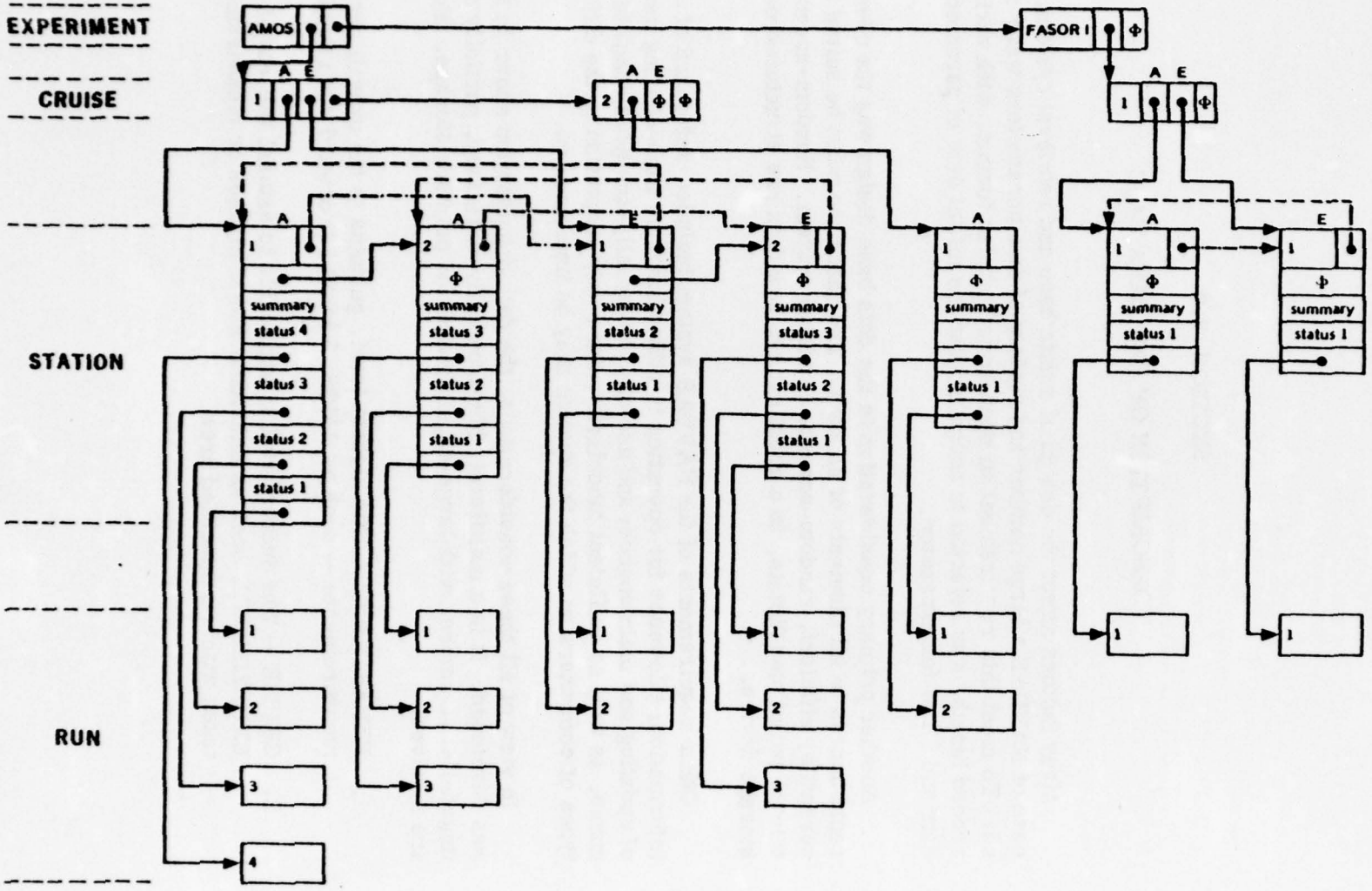


Figure 2-1. NAVDAB Data Base Structure.

3. STATION – A subdivision of a CRUISE: refers to measurements at or about a specific geographical location, such as FASOR II, Cruise 1, Station 1

4. RUN – The fourth and lowest level: designates a discrete set of measurements forming an element of a STATION, such as a propagation loss run or a hydrographic cast

Actual numerical data are stored only at the RUN level, which contains no other information. Key catalog information about the various Runs of a Station are stored at the STATION level, which contains, for all the Runs, chronological and geographical data and access category, and lists the acoustic and environmental parameters. The organization of the data is described for each Run, including the Run's storage location on the disk. Locations of associated Stations are also included. There are two types of Stations, acoustic and environmental, but the only environmental data currently approved by the Steering Group are those supporting accepted acoustic data.

Catalog information for the Stations of a Cruise is stored at the CRUISE level, which also contains storage locations of the first acoustic and environmental Stations and the next Cruise of the Experiment. In addition, there is an alphanumeric description of the features of the particular Cruise. These notes may be printed out in the retrieval process as background information.

The EXPERIMENT level contains summary information for the Experiment and directs flow down through the other levels. This flexible pointer structure allows additions or deletions of data without changing the programs that access the data.

SECTION 3.0

DESCRIPTION OF THE RETRIEVAL PROCESS

The retrieval process consists of two operations, a data selection pass and an application pass. The two-pass technique was selected to allow flexibility in the type of request for data in terms of area, season, frequency, water depth, range, propagation mode, etc.

The data selection pass reads and analyzes the request and, starting at the EXPERIMENT level, examines the summary data stored at each level to determine whether there is justification to continue searching at the lower levels or to go "across" to the next entry at the same level. The examination continues to the STATION level, where the final decision is made as to whether any of the Runs of the Station meet the acoustic data requirements of the potential user.

When a suitable acoustic Run is found, the pointer to the associated environmental Station is followed to determine whether environmental requirements have been met. If both acoustic and environmental criteria have been met, the pointers to data locations are stored in a temporary file for processing by the application pass which reads in the pointers and retrieves the data. A standard set of application programs is planned to output the data in a format selected by the user. The user may also receive the output with conversion to units of his choice.

Flowcharts for the retrieval process are shown in Figure 3-1 through 3-3. Complete computer program listings are contained in Appendix A.

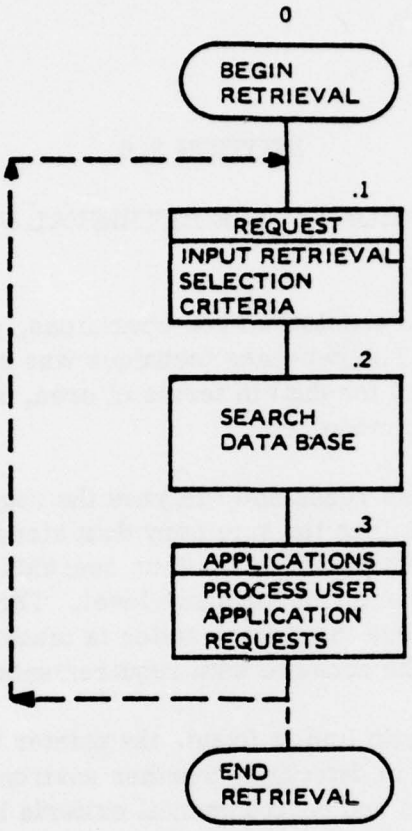

0
BEGIN RETRIEVAL
.1
REQUEST
INPUT RETRIEVAL SELECTION CRITERIA
.2
SEARCH DATA BASE
.3
APPLICATIONS
PROCESS USER APPLICATION REQUESTS
END RETRIEVAL

Figure 3-1.

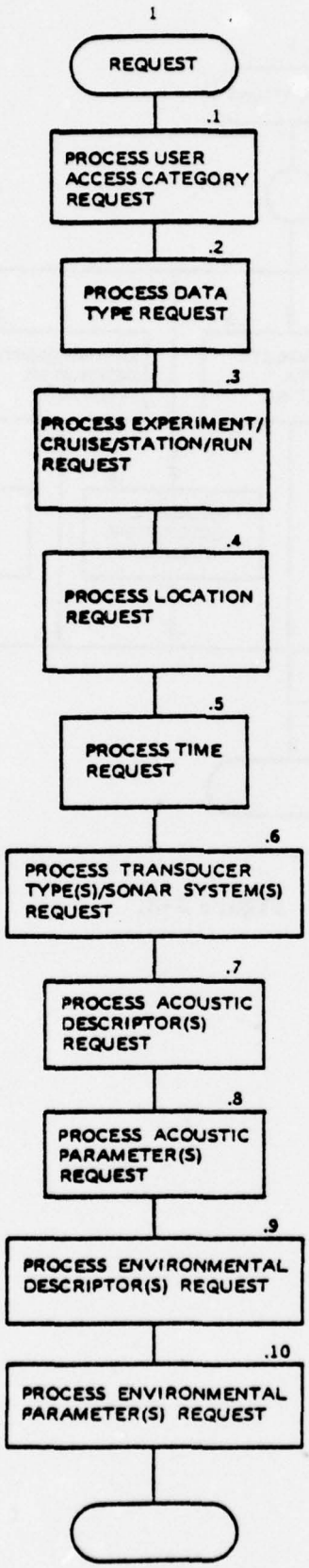

1
REQUEST
.1
PROCESS USER ACCESS CATEGORY REQUEST
.2
PROCESS DATA TYPE REQUEST
.3
PROCESS EXPERIMENT/ CRUISE/STATION/RUN REQUEST
.4
PROCESS LOCATION REQUEST
.5
PROCESS TIME REQUEST
.6
PROCESS TRANSDUCER TYPE(S)/SONAR SYSTEM(S) REQUEST
.7
PROCESS ACOUSTIC DESCRIPTOR(S) REQUEST
.8
PROCESS ACOUSTIC PARAMETER(S) REQUEST
.9
PROCESS ENVIRONMENTAL DESCRIPTOR(S) REQUEST
.10
PROCESS ENVIRONMENTAL PARAMETER(S) REQUEST

Figure 3-2.

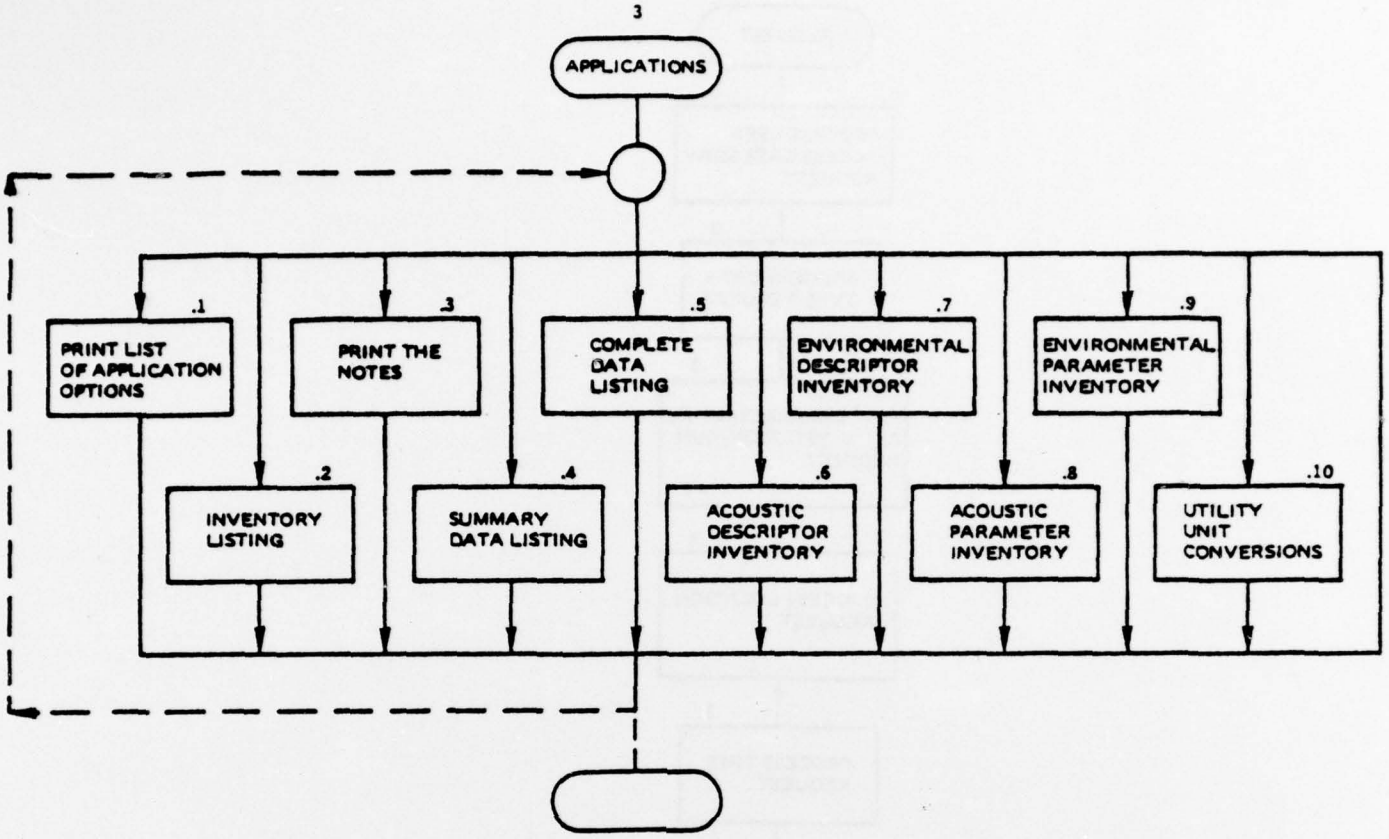

3
APPLICATIONS
.1
PRINT LIST OF APPLICATION OPTIONS
.2
INVENTORY LISTING
.3
PRINT THE NOTES
.4
SUMMARY DATA LISTING
.5
COMPLETE DATA LISTING
.6
ACOUSTIC DESCRIPTOR INVENTORY
.7
ENVIRONMENTAL DESCRIPTOR INVENTORY
.8
ACOUSTIC PARAMETER INVENTORY
.9
ENVIRONMENTAL PARAMETER INVENTORY
.10
UTILITY UNIT CONVERSIONS

Figure 3-3.

APPENDIX A

COMPUTER PROGRAM LISTINGS

ELEMENT TABLE

NAME	PAGE	TYPE	DATE	TIME	SEQ #	SIZE-PRE, TEXT	(CYCLE	WORD)	PSRMODE	LOCATION
ACPRES	A-5	FOR SYMB	22 AUG 74	10:23:46	1	12	5	0	1	1792
ACPRES		RELOCATABLE	16 DEC 75	16:07:04	2	1 8				1804
ANGLES	A-7	FOR SYMB	01 NOV 74	12:02:12	3	13	5	0	1	1813
ANGLES		RELOCATABLE	16 DEC 75	16:07:01	4	1 10				1826
AREACK	A-9	ELT SYMB	07 NOV 75	15:41:24	5	3	5	0	1	1837
AREACK		RELOCATABLE	16 DEC 75	16:07:12	6	1 5				1840
ASUNPK	A-10	FOR SYMB	16 DEC 75	10:16:40	7	7	5	0	1	1846
ASUNPK		RELOCATABLE	16 DEC 75	16:07:13	8	1 4				1853
AUNPCK	A-11	FOR SYMB	16 DEC 75	10:17:48	9	32	5	0	1	1858
AUNPCK		RELOCATABLE	16 DEC 75	16:07:24	10	2 17				1890
CHECK	A-14	FOR SYMB	10 SEP 75	11:51:04	11	9	5	0	1	1909
CHECK		RELOCATABLE	16 DEC 75	16:07:37	12	1 4				1918
CLASS	A-15	FOR SYMB	16 DEC 75	09:48:12	13	58	5	0	1	1923
CLASS		RELOCATABLE	16 DEC 75	16:07:49	14	2 54				1981
CONVRT	A-21	FOR SYMB	31 OCT 75	15:40:50	15	27	5	0	1	2037
CONVRT		RELOCATABLE	16 DEC 75	16:08:01	16	3 25				2064
CVTPRO	A-24	FOR SYMB	21 OCT 74	10:18:36	17	38	5	0	1	2092
CVTPRO		RELOCATABLE	16 DEC 75	16:08:04	18	3 28				2130
CYCLES	A-28	FOR SYMB	24 JUN 74	14:33:34	19	12	5	0	1	2161
CYCLES		RELOCATABLE	16 DEC 75	16:08:01	20	1 8				2173
DATECK	A-30	FOR SYMB	16 DEC 75	10:24:48	21	10	5	0	1	2182
DATECK		RELOCATABLE	16 DEC 75	16:08:05	22	1 12				2192
DATLST	A-31	FOR SYMB	04 NOV 75	17:21:46	23	83	5	0	1	2205
DATLST		RELOCATABLE	16 DEC 75	16:08:13	24	3 79				2288
DBLOSS	A-38	FOR SYMB	06 SEP 74	09:20:35	25	15	5	0	1	2370
DBLOSS		RELOCATABLE	16 DEC 75	16:08:17	26	1 9				2385
DEEBEE	A-40	FOR SYMB	30 OCT 75	08:46:09	27	19	5	0	1	2395
DEEBEE		RELOCATABLE	16 DEC 75	16:08:28	28	1 15				2414
DENSTY	A-42	FOR SYMB	28 JUN 74	13:34:17	29	12	5	0	1	2430
DENSTY		RELOCATABLE	16 DEC 75	16:08:25	30	1 7				2442
DESCTB	A-44	FOR SYMB	02 OCT 75	09:10:28	31	5	5	0	1	2450
DESCTB		RELOCATABLE	16 DEC 75	16:08:28	32	1 5				2455
DSKREQ	A-45	FOR SYMB	16 DEC 75	10:24:49	33	3	5	0	1	2461
DSKREQ		RELOCATABLE	16 DEC 75	16:08:37	34	1 4				2464
EAST	A-46	ELT SYMB	04 NOV 75	17:05:22	35	2	5	0	1	2469
EAST		RELOCATABLE	16 DEC 75	16:08:36	36	1 3				2471
ENERGY	A-47	FOR SYMB	24 JUN 74	14:33:40	37	13	5	0	1	2475
ENTRGY		RELOCATABLE	16 DEC 75	16:08:40	38	1 8				2488
FREEFD	A-49	FOR SYMB	16 DEC 75	10:30:34	39	17	5	0	1	2497
FREEFD		RELOCATABLE	16 DEC 75	16:08:48	40	1 16				2514
GETINT	A-51	FOR SYMB	04 NOV 75	17:26:29	41	30	5	0	1	2531
GETINT		RELOCATABLE	16 DEC 75	16:09:04	42	3 16				2561
IDNAME	A-55	SYMBOLIC	16 OCT 75	10:43:27	43	10	5	0	1	2580
IDNAME		RELOCATABLE	16 DEC 75	16:09:05	44	1 10				2590
INPUT	A-56	FOR SYMB	16 DEC 75	10:26:04	45	180	5	0	1	2601
INPUT		RELOCATABLE	16 DEC 75	16:09:28	46	2 160				2781
INTENS	A-73	FOR SYMB	24 JUN 74	14:34:29	47	12	5	0	1	2943
INTENS		RELOCATABLE	16 DEC 75	16:09:28	48	1 7				2955
INVNTY	A-75	FOR SYMB	14 AUG 75	10:17:53	49	12	5	0	1	2963
INVNTY		RELOCATABLE	16 DEC 75	16:09:40	50	2 10				2975
LINEAR	A-76	FOR SYMB	30 AUG 74	13:34:51	51	15	5	0	1	2987
LINEAR		RELOCATABLE	16 DEC 75	16:09:41	52	1 11				3002
LNGTHE	A-78	FOR SYMB	24 JUN 74	14:32:36	53	12	5	0	1	3014
LNGTHE		RELOCATABLE	16 DEC 75	16:09:53	54	1 8				3026

	PAGE										
MSQCVT	A-80	ELT SYMB	07 NOV 75	15:41:25	55		12	5	0	1	3035
MSQCVT		RELOCATABLE	16 DEC 75	16:10:01	56	1	7				3047
NAMEIT	A-82	FOR SYMB	03 SEP 75	11:54:31	57		69	5	0	1	3055
NAMEIT		RELOCATABLE	16 DEC 75	16:10:05	58	1	27				3124
NAMGET	A-87	FOR SYMB	02 SEP 75	15:11:05	59		13	5	0	1	3152
NAMGET		RELOCATABLE	16 DEC 75	16:10:04	60	1	15				3165
NOTE	A-88	FOR SYMB	16 DEC 75	10:27:01	61		17	5	0	1	3181
NOTE		RELOCATABLE	16 DEC 75	16:10:29	62	2	14				3198
PARATB	A-90	FOR SYMB	02 OCT 75	10:15:31	63		7	5	0	1	3214
PARATB		RELOCATABLE	16 DEC 75	16:10:36	64	1	6				3221
PDP	A-91	FOR PROC	29 OCT 75	12:08:58	65		14	1	0	1	3228
PERCNT	A-92	FOR SYMB	17 JUL 75	13:52:12	66		12	5	0	1	3242
PERCNT		RELOCATABLE	16 DEC 75	16:10:41	67	1	8				3254
PLANAR	A-94	FOR SYMB	03 SEP 75	09:15:53	68		13	5	0	1	3263
PLANAR		RELOCATABLE	16 DEC 75	16:10:48	69	1	9				3276
PO-OWR	A-96	FOR SYMB	24 JUN 74	14:34:03	70		12	5	0	1	3286
PO-OWR		RELOCATABLE	16 DEC 75	16:10:51	71	1	7				3298
PRESSR	A-98	FOR SYMB	22 AUG 74	10:25:08	72		14	5	0	1	3306
PRESSR		RELOCATABLE	16 DEC 75	16:11:00	73	1	10				3320
PRSLVL	A-101	FOR SYMB	22 AUG 74	11:09:28	74		12	5	0	1	3331
PRSLVL		RELOCATABLE	16 DEC 75	16:11:01	75	1	7				3343
RANGE	A-103	FOR SYMB	16 DEC 75	10:27:24	76		37	5	0	1	3351
RANGE		RELOCATABLE	16 DEC 75	16:11:04	77	2	23				3388
SELECT	A-106	FOR SYMB	16 DEC 75	10:27:49	78		139	5	0	1	3413
SELECT		RELOCATABLE	16 DEC 75	16:11:15	79	5	92				3552
SHORT	A-120	FOR SYMB	21 AUG 75	15:07:12	80		11	5	0	1	3649
SHORT		RELOCATABLE	16 DEC 75	16:11:24	81	2	9				3660
SHRTME	A-121	FOR SYMB	24 JUN 74	14:32:29	82		12	5	0	1	3671
SHRTME		RELOCATABLE	16 DEC 75	16:11:24	83	1	9				3683
SORGET	A-123	FOR SYMB	22 AUG 74	11:07:41	84		16	5	0	1	3693
SORGET		RELOCATABLE	16 DEC 75	16:11:29	85	1	9				3709
SPEEDS	A-125	FOR SYMB	26 JUN 74	21:22:11	86		13	5	0	1	3719
SPEEDS		RELOCATABLE	16 DEC 75	16:11:40	87	1	10				3732
STATCK	A-127	ELT SYMB	04 NOV 75	17:05:13	88		3	5	0	1	3743
STATCK		RELOCATABLE	16 DEC 75	16:11:40	89	1	3				3746
STORE	A-128	SYMBOLIC	10 OCT 75	15:17:14	90		8	5	0	1	3750
STORE		RELOCATABLE	16 DEC 75	16:11:51	91	2	6				3758
TMPTUR	A-129	FOR SYMB	17 JUL 75	13:45:33	92		13	5	0	1	3766
TMPTUR		RELOCATABLE	16 DEC 75	16:12:01	93	1	11				3779
UNPKCR	A-131	ELT SYMB	04 NOV 75	17:05:15	94		10	5	0	1	3791
UNPKCR		RELOCATABLE	16 DEC 75	16:12:03	95	1	6				3801
UNPKXP	A-132	ELT SYMB	07 NOV 75	15:41:20	96		7	5	0	1	3808
UNPKXP		RELOCATABLE	16 DEC 75	16:12:12	97	1	5				3815
UNTCVT	A-133	FOR SYMB	31 OCT 75	17:04:53	98		11	5	0	1	3821
UNTCVT		RELOCATABLE	16 DEC 75	16:12:25	99	2	8				3832
UPXREA	A-134	ELT SYMB	07 NOV 75	15:41:22	100		15	5	0	1	3842
UPXREA		RELOCATABLE	16 DEC 75	16:12:28	101	1	8				3857
VOLUME	A-137	FOR SYMB	24 JUN 74	14:32:21	102		13	5	0	1	3866
VOLUME		RELOCATABLE	16 DEC 75	16:12:39	103	1	8				3879
WEST	A-139	ELT SYMB	04 NOV 75	17:06:04	104		2	5	0	1	3888
WEST		RELOCATABLE	16 DEC 75	16:12:38	105	1	3				3890
											3894

CALLED ROUTINES

CALLING ROUTINES

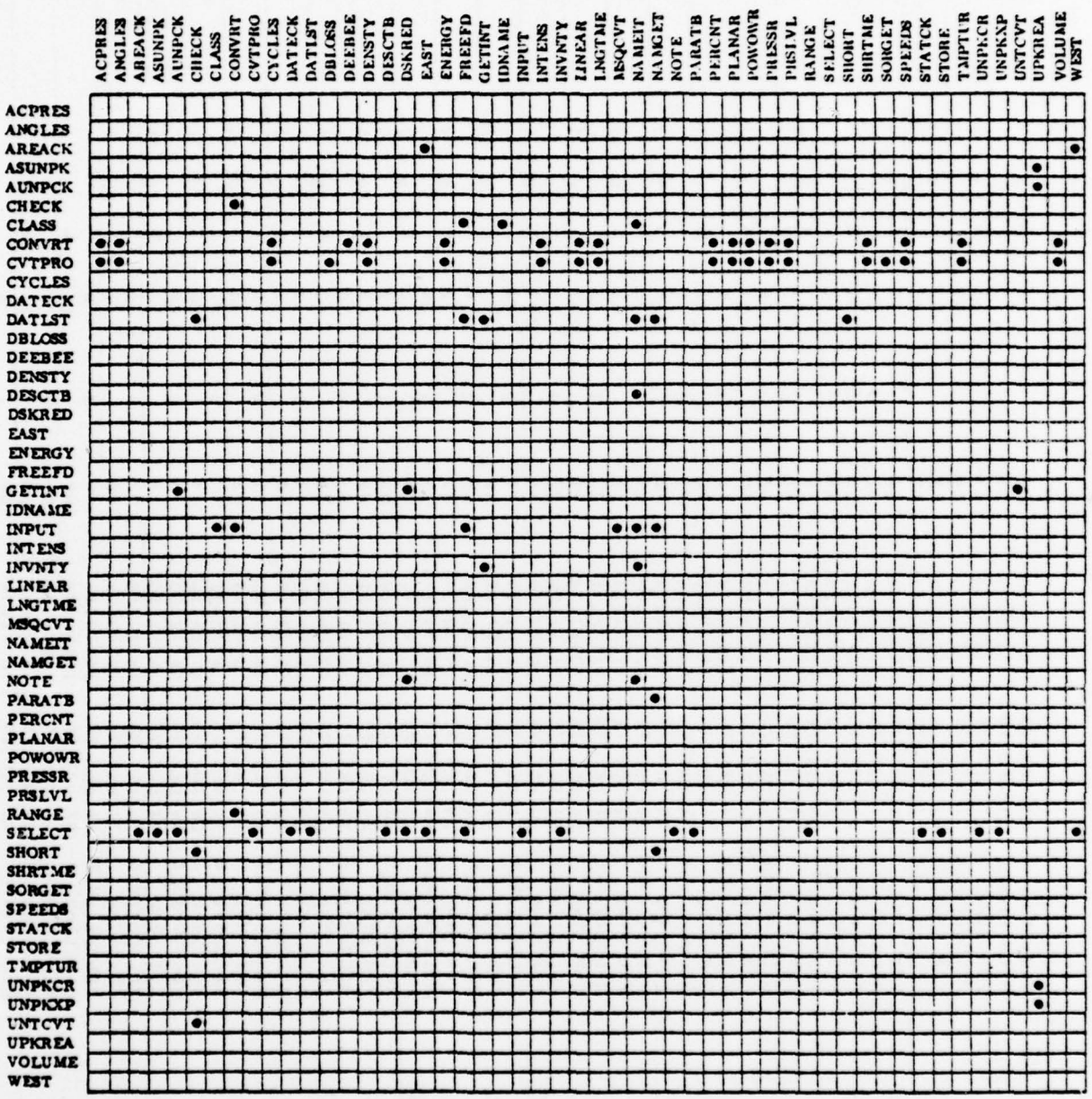


Figure A-1. Retrieval Program Cross-Reference Map.

```
@FOR,S# N.ACPRES,.ACPRES

      SUBROUTINE ACPRES(VALUE,UNICDE,UNITS)
C
C     TABLE OF ACPRES MEASURE CONVERSION FACTORS.
C
      INTEGER ENTRY(3,3),UNITS(9),UNICDE
C
C     FACTORS ARE DOUBLE PRECISION.
C
      DOUBLE PRECISION FACTOR(3)
C
C     PUT UNIT CODES INTO ENTRY.
C
      DATA (ENTRY(1,I),I=1,3)/48,49,50/
C
C     PUT IN ALPHA UNITS.
C
      DATA ((ENTRY(I,J),I=2,3),J=1,3)/'MICRO ','BARS  ',
     .                '*PASCAL','$     ','MICRO ','PAS   '/
C
C     ENTER INTERNAL UNITS.
C
      UNITS(3)='MICRO '
      UNITS(9)='PASCAL'
C
C     IF UNICDE=0, THEN SET CODE TO STANDARD UNITS.
C
      IF(UNICDE.EQ.0) UNICDE=-50
C
C     SET CONVERSION FACTORS.
C
      DATA (FACTOR(I),I=1,3)/ 1.D5,1.D3,100/
C
C     IF CODE IS IN TABLE, PERFORM CONVERSION. ALSO, ENTER UNITS ALPHA
C          CODE INTO 'UNITS':
C
      ICODE=IABS(UNICDE)
      DO 2 I=1,3
      IF(ICODE.NE.ENTRY(I,1)) GO TO 2
      IF(UNICDE.GE.0) VALUE=VALUE*FACTOR(I)
      IF(UNICDE.LT.0) VALUE=VALUE/FACTOR(I)
      DO 1 J=1,2
      UNITS(J)=ENTRY(J+1,I)
    1 CONTINUE
C
C     AT THIS POINT, CONVERSION IS COMPLETE.
C
      RETURN
C
C     ELSE, CHECK REST OF TABLE.
C
    2 CONTINUE
C
C     IF CODE NOT FOUND, RETURN WITH 'UNITS ERROR' IN 'UNITS'.
C
```

```
      UNITS(1)='UNITS '
      UNITS(2)='ERROR '
      PRINT 101,ICODE
  101 FORMAT(' ERROR--UNIT CODE',I3,' NOT IN ACOUSTIC PRESSURE TABLE.')
      RETURN
      END
```

```
@FOR,S H.ANGLES,.ANGLES

      SUBROUTINE ANGLES(VALUE,UNICDE,UNITS)
C
C     TABLE OF ANGLES MEASURE CONVERSION FACTORS.
C
      INTEGER ENTRY(3,3),UNITS(4),UNICDE
C
C     FACTORS ARE DOUBLE PRECISION.
C
      DOUBLE PRECISION FACTOR(3)
C
C     PUT UNIT CODES INTO ENTRY.
C
      DATA (ENTRY(I,1),I=1,3)/37,38,85/
C
C     PUT IN ALPHA UNITS.
C
      DATA ((ENTRY(I,J),I=2,3),J=1,3)/'RADIAN','S   ',
     *          'DEGREE','S   ','FTHMS/','NTCLML'/
C
C     SET CONVERSION FACTORS.
C
      DATA (FACTOR(I),I=1,2)/1.D00,.0174532917D0/
C
C     ENTER INTERNAL DEGREE UNITS.
C
      UNITS(3)='DEGREE'
      UNITS(4)='S   '
C
C     IF UNICDE=0, THEN SET CODE TO STANDARD UNITS.
C
      IF(UNICDE.EQ.0) UNICDE=-37
C
C     IF CODE IS IN TABLE, PERFORM CONVERSION. ALSO, ENTER UNITS ALPHA
C          CODE INTO 'UNITS'.
C
      ICODE=IABS(UNICDE)
      DO 2 I=1,2
      IF(ICODE.NE.ENTRY(1,I)) GO TO 2
      IF(UNICDE.GE.0) VALUE=VALUE*FACTOR(I)
      IF(UNICDE.LT.0) VALUE=VALUE/FACTOR(I)
      DO 1 J=1,2
      UNITS(J)=ENTRY(J+1,I)
    1 CONTINUE
C
C     AT THIS POINT, CONVERSION IS COMPLETE.
C
      RETURN
C
C     ELSE, CHECK REST OF TABLE.
C
    2 CONTINUE
      IF(ICODE.NE.ENTRY(1,3)) GO TO 3
      IF(UNICDE.GE.0) VALUE=ATAN(VALUE/1000.)/.0174533
      IF(UNICDE.LT.0) VALUE=TAN(VALUE*.0174533)*1000.
```

```
      UNITS(1)=ENTRY(2,3)
      UNITS(2)=ENTRY(3,3)
      RETURN
C
C     IF CODE NOT FOUND, RETURN WITH 'UNITS ERROR' IN 'UNITS'.
    3 UNITS(1)='UNITS '
      UNITS(2)='ERROR '
      PRINT 101,ICODE
  101 FORMAT(' ERROR--UNIT CODE ',I3,' NOT IN ANGLES TABLE.')
      RETURN
      END
```

```
@LLT,L      N.AREACK,.AREACK

      1.            SUBROUTINE AREACK (NLAT,ELON,SLAT,WLON,ILAT,ILON,FLAT,FLON,*)
      2.      C
      3.      C
      4.            IMPLICIT INTEGER (A-Z)
      5.      C
      6.            IF (NLAT .EQ. 0 .AND. SLAT .EQ. 0 .AND.
      7.           *    ELON .EQ. 0 .AND. WLON .EQ. 0) RETURN
      8.            IF (ILAT .LT. SLAT .OR. FLAT .GT. NLAT) RETURN 9
      9.            IF (WEST(WLON,ILON) .EQ. ILON .OR.
     10.           *    EAST(ELON,FLON) .EQ. FLON)      RETURN 9
     11.            RETURN
     12.            END
```

```
@ELT,L     H.ASUNPK,,ASUNPK

      1.            SUBROUTINE ASUNPK(STABFR,TYPE)
      2.            IMPLICIT INTEGER(A-Z)
      3.      C
      4.      C
      5.      C
      6.            COMMON /ASTATH/SANO,SANRUN,SAILAT,SAILON,SAIDAT,SAFLAT,SAFLON,
      7.           *              SAFDAT,SASTAT(2),PNXTAS,PENVS,SAIFLG
      8.      C
      9.      C
     10.      C
     11.            DIMENSION STABFR(1)
     12.      C
     13.      C     UNPACK TYPE, STATION NUMBER, AND NUMBER OF RUNS.
     14.      C
     15.            TYPE=FLD(15,3,STABFR(1))
     16.            SANO=FLD(6,18,STABFR(2))
     17.            SANRUN=FLD(18,18,STABFR(2))
     18.      C
     19.      C     UNPACK STATION INITIAL AND FINAL POINT BOUNDS.
     20.      C
     21.            CALL UPKREA(' ',SAILAT,SAILON,SAIDAT,0,' ',
     22.           *              SAFLAT,SAFLON,SAFDAT,0,' ',STABFR(3))
     23.      C
     24.      C     UNPACK STATION STATUS BITS.
     25.      C
     26.            SASTAT(1)=STABFR(7)
     27.            SASTAT(2)=STABFR(8)
     28.      C
     29.      C
     30.            PNXTAS=STABFR(9)
     31.      C
     32.      C
     33.            PENVS=STABFR(10)
     34.            RETURN
     35.            END
```

```
DELT,L     N,AUNPCK,,AUNPCK

      SUBROUTINE AUNPCK (BFR,IPTR)
C
C
C
      IMPLICIT INTEGER (A-Z)
C
C
      COMMON /RUN   /PORA,RILAT,RINS,RFLAT,RFNS,RILON,RIEW,RFLON,RFEW,
     *               RINO,RIDAY,RIYR,RITIM,RIZON,RFNO,RFDAY,RFYR,RFTIM,
     *               RFZON,NAVCOD,DTASRC,SYSSRC,SYSRCR,SRCTYP,RCRTYP,
     *               CLAS,DESCR,NAVDIR,NAVHT,HTUNIT,NAVPRD,PRDUNT,
     *               SEASTA,NDDIR,NDVEL,VELUNT,SWLDIR,SWLHT,
     *               SHTUNT,SWLPRD,SPDUNT,WEATHR,BOTDPH,DPHUNT,BTMSLP,
     *               BATHY,INFO(9),RSTAT(2),RUNNO
      REAL NAVDIR,NAVHT,NAVPRD,SEASTA,NDDIR,NDVEL,SWLDIR,
     *     SWLHT,SWLPRD,WEATHR,BOTDPH,BTMSLP
C
C
C
      COMMON /CPV   /NCON,CON(4,30),NPAR,PAR(30,30),NVAR,IVAR(3,30),
     *               NDATS,NROWS(50),NDATA,VALRNG(2,30,50),DATA(30000)
      DIMENSION ICON(4,30),IPAR(30,30),IVLRNG(2,30,50)
      EQUIVALENCE (CON(1,1),ICON(1,1)),(PAR(1,1),IPAR(1,1)),
     *            (VALRNG(1,1,1),IVLRNG(1,1,1))
      REAL CON,PAR,DATA,VALRNG
C
C
      DIMENSION BFR(1)
      REAL BFR
C
C
      ISUB = IPTR
C
C         UNPACK RUNNO
C
      RUNNO = FLD(0,18,BFR(ISUB))
C
C         UNPACK NAVCOD, DTASRC, CLAS, DESCR
C
      ISUB = ISUB +1
      NAVCOD = FLD( 0,9,BFR(ISUB))
      DTASRC = FLD( 9,9,BFR(ISUB))
      CLAS   = FLD(18,9,BFR(ISUB))
      DESCR  = FLD(27,9,BFR(ISUB))
C
C         UNPACK SYSSRC, SYSRCR, SRCTYP, RCRTYP
C
      ISUB = ISUB +1
      SYSSRC = FLD( 0,9,BFR(ISUB))
      SYSRCR = FLD( 9,9,BFR(ISUB))
      SRCTYP = FLD(18,9,BFR(ISUB))
      RCRTYP = FLD(27,9,BFR(ISUB))
C
C         UNPACK AREA/TIME
```

```
      C
            ISUB = ISUB +1
           . CALL UPKREA (PORA,RILAT,RILON,RIDAY,RITIM,RIZON,
           .                RFLAT,RFLON,RFDAY,RFTIM,RFZON,BFR(ISUB))
      C
      C          UNPACK STATUS BITS
      C
            ISUB = ISUB +9
            RSTAT(1) = FLD(0,36,BFR(ISUB  ))
            RSTAT(2) = FLD(0,36,BFR(ISUB+1))
      C
      C          THIS ENDS THE UNPACKING OF DATA INTO COMMON /RUN/. FROM HERE
      C          ON THE DATA WILL BE UNPACKED INTO /CPV/.
      C
      C               UNPACK NCON, NPAR, NVAR, NDATS
      C
            ISUB = ISUB +3
            NCON  = FLD( 0,9,BFR(ISUB))
            NPAR  = FLD( 9,9,BFR(ISUB))
            NVAR  = FLD(18,9,BFR(ISUB))
            NDATS = FLD(27,9,BFR(ISUB))
      C
      C          UNPACK NAME CODES & PROCESS CODES FOR ALL CONSTANTS
      C
            IF (NCON .EQ. 0) GO TO 45
            DO 30 I=1,NCON,2
            ISUB = ISUB +1
            ICON(1,I) = FLD( 0,8,BFR(ISUB))
            ICON(4,I) = FLD( 8,10,BFR(ISUB))
            J = I+1
            IF (J .GT. NCON) GO TO 30
            ICON(1,J) = FLD(18,8,BFR(ISUB))
            ICON(4,J) = FLD(26,10,BFR(ISUB))
         30 CONTINUE
      C
      C          UNPACK VALUES FOR CONSTANTS
      C
            DO 40 I=1,NCON
            ISUB = ISUB +1
            CON(3,I) = BFR(ISUB)
         40 CONTINUE
      C
      C          UNPACK DATA FOR ALL PARAMETERS
      C
         45 IF (NPAR .EQ. 0) GO TO 65
            DO 60 I=1,NPAR
      C
      C               UNPACK FOR PARAMETER(I)
      C
            ISUB = ISUB +1
            IPAR(1,I) = FLD( 0,9,BFR(ISUB))
            IPAR(2,I) = FLD( 9,8,BFR(ISUB))
            IPAR(4,I) = FLD(17,10,BFR(ISUB))
      C
      C               UNPACK ALL VALUES FOR PARAMETER(I)
      C
            JJ = IPAR(1,I)
```

```
                DO 50 J=1,JJ
                ISUB = ISUB +1
                PAN(Q,J,I) = BFR(ISUB)
   50           CONTINUE
C
   60 CONTINUE
C
C
   65 IF (NVAR .EQ. 0) GO TO 101
      DO 70 I=1,NVAR,2
      ISUB = ISUB +1
      IVAR(1,I) = FLD( 3,8,BFR(ISUB))
      IVAR(3,I) = FLD( 8,16,BFR(ISUB))
      J = I+1
      IF (J .GT. NVAR) GO TO 70
      IVAR(1,J) = FLD(18,8,BFR(ISUB))
      IVAR(3,J) = FLD(26,10,BFR(ISUB))
   70 CONTINUE
C
C
      DO 75 I=1,NDATS,2
      ISUB = ISUB +1
      NROWS(I) = FLD(0,18,BFR(ISUB))
      J = I+1
      IF (J .GT. NDATS) GO TO 75
      NROWS(J) = FLD(18,18,BFR(ISUB))
   75 CONTINUE
C
C          MOVE MAX & MIN VALUES FOR ALL DATA SETS
C
      DO 100 K=1,NDATS
C
          DO 90 J=1,NVAR
C
              DO 80 I=1,2
              ISUB = ISUB +1
              VALRNG(I,J,K) = BFR(ISUB)
   80         CONTINUE
C
   90     CONTINUE
C
  100 CONTINUE
C
C
  101 IPTR = ISUB + 1
C
C          DONE !
C
      RETURN
      END
```

```
PFOR,SA     N.CHECK,.CHECK

      SUBROUTINE CHECK (VALUE,NAME,UNITS)
C
C          CHECK CHECKS TO SEE IF THE NAME CODE 'NAME' HAS BEEN
C          ENTERED IN EITHER THE ACOUSTIC PARAMETER OR ENV PARAMETER
C          ON THE RETRIEVAL REQUEST.  IF IT HAS, IT CONVERTS THE VALUE
C          'VALUE' TO THE REQUESTED UNITS & PUTS THE NAME OF THE UNITS
C          IN 'UNITS'. IF NOT, IT DOES NOT CHANGE THE VALUE OF 'VALUE'
C          & PUTS THE NAME OF THE STORED UNITS IN 'UNITS'.
C
C
      IMPLICIT INTEGER (A-Z)
C
C
      INCLUDE RETNEV,LIST
C
C
      REAL VALUE
      DIMENSION UNITS(2)
C
C
C          CHECK ACOUSTICS
C
      TEMP=0
      IF (NACPAR .EQ. 0) GO TO 10
      DO 7 I=1,NACPAR
      IF (ACPARS(1,I) .NE. NAME) GO TO 7
      TEMP = -ACPARS(4,I)
      GO TO 30
    7 CONTINUE
C
C          CHECK ENVIRONMENTAL
C
   10 IF (NENPAR .EQ. 0) GO TO 30
      TEMP =0
      DO 20 I=1,NENPAR
      IF (ENPARS(1,I) .NE. NAME) GO TO 20
      TEMP = -ENPARS(4,I)
      GO TO 30
   20 CONTINUE
C
   30 CALL CONVRT (VALUE,NAME,TEMP,UNITS)
C
      RETURN
      END
```

```
@ELT,L      N.CLASS,,CLASS

      1.            SUBROUTINE CLASS (NOSEX,SEX,MASK,RQUEST,TNAME,TNBR,FIRST,BACKUP,
      2.           1CLSREQ)
      3.
      4.      C     THIS ROUTINE PROVIDES FOR SIGNING ON AND THE INPUT OF THE USER'S DESIRED
      5.      C     SECURITY LEVELS.
      6.      C
      7.            PARAMETER UNIT = 12
      8.            IMPLICIT INTEGER (A-Z)
      9.            LOGICAL PASS,SECURE,FIRST,BACKUP,CLSREQ,NUMCK,PASSCK
     10.            REAL WORDS (10)
     11.            DIMENSION SEX(1), NAME(5), TNAME(5), RQUEST(3), CHAR(81), MASK(3),
     12.           1LEVELS(3), REJECT(3), REJ(10)
     13.            DATA A/004050505050S/, Z/037050505050S/, CLASCK/0577777777777/,
     14.           1ALL/07777777777777/, ALL1/077777777774000/, NUMCK/.FALSE./,
     15.           2PASSCK/.FALSE./
     16.      C
     17.      C        IF BACKING UP, BRANCH TO APPROPRIATE QUESTION
     18.      C
     19.            IF (.NOT.BACKUP) GO TO 1
     20.            IF (CLSREQ) GO TO 220
     21.            IF (FIRST) GO TO 8
     22.      C
     23.      C        CLASSIFIED DATA REQUEST QUESTION
     24.      C
     25.      1     WRITE (6,2)
     26.      2     FORMAT ('0DO YOU INTEND TO RETRIEVE ANY CLASSIFIED DATA?')
     27.            READ (5,3,END=1,ERR=5) ANSWER
     28.      3     FORMAT (A3)
     29.            WRITE (6,4) ANSWER
     30.      4     FORMAT (3X,A3)
     31.            # = ' '
     32.            FLD (0,6,#) = FLD (0,6,ANSWER)
     33.            IF (#.EQ.'Y') GO TO 16
     34.            IF (#.EQ.' '.OR.#.EQ.'N') GO TO 7
     35.      C
     36.      C        ERROR
     37.      C
     38.      5     WRITE (6,6)
     39.      6     FORMAT (' *** INVALID RESPONSE TO YES OR NO QUESTION ***')
     40.            GO TO 1
     41.      C
     42.      C        UNCLASSIFIED REQUEST (ENTER NAME IF FIRST PASS)
     43.      C
     44.      7     CLSREQ = .FALSE.
     45.            IF (.NOT.FIRST) GO TO 14
     46.      8     WRITE (6,9)
     47.      9     FORMAT ('0ENTER USER NAME')
     48.            TNBR = -1
     49.            READ (5,10,END=1,ERR=12) (TNAME(I),I=1,5)
     50.      10    FORMAT (5A6)
     51.            WRITE (6,11) (TNAME(I),I=1,5)
     52.      11    FORMAT (3X,5A6)
     53.            GO TO 14
     54.      C
```

```
C          ERROR
C
12    WRITE (6,13)
13    FORMAT (' *** READ FORMAT ERROR ***')
      GO TO 8
C
C
14    NOSEX = 1
      SEX(1) = 1
      REQUEST(1) = 0
      REQUEST(2) = 0
      REQUEST(3) = 0
      FLD(1,1,REQUEST(1)) = 1
      RETURN
C
C
15    CLSREQ = .TRUE.
      IF (NUMCK) GO TO 220
16    COUNT = 0
18    COUNT = COUNT + 1
      IF (COUNT.LE.3) GO TO 30
      WRITE (6,20)
20    FORMAT ('0*** USER NAME AND/OR ID WAS ENTERED INCORRECTLY IN THREE
     1 ATTEMPTS,' / 5X,'PROGRAM TERMINATES ***')
      STOP ERROR
C
30    WRITE (6,40)
40    FORMAT ('0ENTER USER NAME, USER ID')
      READ (5,50,END=1,ERR=95) (CHAR(I),I=1,80)
50    FORMAT (80A1)
      WRITE (6,60) (CHAR(I),I=1,80)
60    FORMAT (3X,80A1)
      CALL IDNAME (CHAR,ID,NAME,FLAG)
      IF (FLAG.EQ.0) GO TO 120
      GO TO 30
C
C          ERROR
C
95    WRITE (6,100)
100   FORMAT (' *** ERROR ENTERING USER NAME OR ID ***')
      GO TO 30
C
C
120   REWIND UNIT
      N = 0
130   N = N + 1
      READ (UNIT,140,END=180,ERR=200) TNBR,(TNAME(I),I=1,5),
     1(LEVELS(I),I=1,3)
140   FORMAT (6A6,3012)
      IF (ID.NE.TNBR) GO TO 130
      DO 150 I = 1,5
150   IF (NAME(I).NE.TNAME(I)) GO TO 160
      GO TO 220
C
C          ERROR CONDITIONS
C
160   WRITE (6,170)
```

```
      170   FORMAT (' *** USER NAME DOES NOT MATCH NAME IN TABLE ***')
            GO TO 18
      180   WRITE (6,190) ID
      190   FORMAT (' *** USER ID ', A6, ' NOT FOUND IN TABLE ***')
            GO TO 18
      200   WRITE (6,210) N
      210   FORMAT (' *** ERROR READING RECORD', I6, ' OF MASTER USER FILE.'/
           1' CONTACT BRANCH MANAGER REGARDING ERROR ***')
            GO TO 130
C
C           ENTER SECURITY LEVELS
C
      220   WRITE (6,230)
      230   FORMAT (' ENTER SECURITY LEVEL(S)')
            RQUEST(1) = 0
            RQUEST(2) = 0
            RQUEST(3) = 0
            READ (5,50,END=255,ERR=240) (CHAR(I),I=1,80)
            CALL FREEFD (CHAR,WORDS,SEX,NOSEX,10,$240)
            GO TO 260
C
C           ERROR
C
      240   WRITE (6,250)
      250   FORMAT (' *** ERROR ENTERING SECURITY LEVELS ***')
            GO TO 220
C
C           BACKUP TO LAST QUESTION
C
      255   IF (.NOT.NUMCK) GO TO 16
            GO TO 1
C
      260   SECURE = .FALSE.
            IF (NOSEX.EQ.0) GO TO 280
            DO 270 I = 1,NOSEX
            ISEX = IABS(SEX(I))
            IF (ISEX.GT.99) GO TO 300
            IF (ISEX.NE.1) SECURE = .TRUE.
            WORD = ISEX/36 + 1
            BIT = MOD(ISEX,36)
            FLD (BIT,1,RQUEST(WORD)) = 1
            IF (SEX(I).LE.0) GO TO 270
            FLD (1,1,RQUEST(1)) = 1
            IF (SEX(I).EQ.3) FLD (2,1,RQUEST(1)) = 1
      270   CONTINUE
            NUMCK = .TRUE.
            IF (SECURE) GO TO 320
C
C           UNCLASSIFIED ONLY
C
      280   FLD (1,1,RQUEST(1)) = 1
            NUMCK = .TRUE.
            WRITE (6,290)
      290   FORMAT (' YOU HAVE REQUESTED UNCLASSIFIED DATA ONLY')
            GO TO 360
C
C           ERROR - LEVEL OUT OF RANGE
```

```
      C
  300 WRITE (6,310) SEX(I)
  310 FORMAT (' *** SECURITY LEVEL',I5,' OUT OF RANGE ***')
      GO TO 220
      C
      C
  320 IF (FLD(0,1,RQUEST(1)).NE.1) GO TO 340
      WRITE (6,330)
  330 FORMAT (/' YOUR REQUEST IS FOR ALL SECURITY CLASSIFICATIONS')
      FLD (0,36,RQUEST(1)) = FLD (0,36,ALL)
      FLD (0,36,RQUEST(2)) = FLD (0,36,ALL)
      FLD (0,36,RQUEST(3)) = FLD (0,36,ALL)
      GO TO 380
      C
      C        PRINT REQUESTED LEVELS
      C
  340 WRITE (6,350)
  350 FORMAT ('0YOUR REQUEST IS FOR THE FOLLOWING SECURITY LEVEL(S):'/)
      DO 370 I = 1,99
      WORD = I/36 + 1
      BIT = MOD(I,36)
      IF (FLD(BIT,1,RQUEST(WORD)).EQ.0) GO TO 370
      CALL NAME(I) (2,I,NAME,NR)
      WRITE (6,360) (NAME(J),J=1,NR)
  360 FORMAT (9X,6A6)
  370 CONTINUE
  380 IF (FLD(0,1,LEVELS(1)).NE.1) GO TO 390
      C
      C        STORE 1'S IN STATUS BITS
      C
      FLD (0,36,LEVELS(1)) = FLD (0,36,ALL)
      FLD (0,36,LEVELS(2)) = FLD (0,36,ALL)
      FLD (0,36,LEVELS(3)) = FLD (0,36,ALL)
      C
      C
  390 DO 400 I = 1,3
      MASK(I) = AND (RQUEST(I),LEVELS(I))
  400 REJECT(I) = XOR (RQUEST(I),MASK(I))
      C
      C        PRINT REJECTED LEVELS (CHECK IF NO AUTHORIZED LEVELS)
      C
      IF (REJECT(1).EQ.0.AND.REJECT(2).EQ.0.AND.REJECT(3).EQ.0) GOTO 440
      IF (MASK(1).NE.0.OR.MASK(2).NE.0.OR.MASK(3).NE.0) GO TO 420
      WRITE (6,410)
  410 FORMAT ('0ALL REQUESTED LEVELS ARE UNAUTHORIZED.'/1X,
     1'CONTACT BRANCH MANAGER FOR QUESTIONS REGARDING REJECTED LEVELS.')
      GO TO 220
  420 WRITE (6,430)
  430 FORMAT ('0THE FOLLOWING REQUESTED LEVELS ARE UNAUTHORIZED.  OTHERS
     1 ACCEPTED.' / 1X,'CONTACT BRANCH MANAGER FOR QUESTIONS REGARDING R
     2EJECTED LEVELS.'/)
      N = 0
      DO 450 I = 1,99
      WORD = I/36 + 1
      BIT = MOD(I,36)
      IF (FLD(BIT,1,REJECT(WORD)).EQ.0) GO TO 450
      N = N + 1
```

```
      REJ(N) = I
      IF (N.LT.10) GO TO 450
      WRITE (6,440) (REJ(J),J=1,N)
440   FORMAT (9X,10(I2,','))
      N = 0
450   CONTINUE
      IF (N.GT.0) WRITE (6,440) (REJ(J),J=1,N)
C
C        CHECK FOR CLASSIFIED
C
460   IF (AND(MASK(1),CLASCK).EQ.0.AND.AND(MASK(2),ALL).EQ.0.AND.
     1 AND(MASK(3),ALL).EQ.0) GO TO 500
C
C
470   IF (PASSCK) GO TO 500
      WRITE (6,480)
480   FORMAT ('0YOU HAVE REQUESTED CLASSIFIED DATA')
      COUNT = 0
490   COUNT = COUNT + 1
      IF (COUNT.LE.3) GO TO 510
      WRITE (6,500)
500   FORMAT ('0*** PASSWORD WAS ENTERED INCORRECTLY IN THREE ATTEMPTS.'
     1/ 5X, 'PROGRAM TERMINATES ***')
      STOP ERROR
510   WRITE (6,520)
520   FORMAT (' ENTER PASSWORD')
      READ (5,530,END=220,ERR=550) PASSWD
530   FORMAT (A6)
C
C
      IF (PASS(PASSWD)) GO TO 570
      CALL OVERLY
C
C        ERROR CONDITION
C
      WRITE (6,540) PASSWD
540   FORMAT (' PASSWORD ',A6,' IS INCORRECT'/)
      GO TO 490
550   WRITE (6,560)
560   FORMAT (' *** ERROR ENTERING PASSWORD ***'/)
      GO TO 510
C
570   CALL OVERLY
      PASSCK = .TRUE.
C
C
580   IF (FLD(0,1,MASK(1)).NE.1) GO TO 590
      NOSEX = 1
      SEX(1) = 0
      RETURN
590   NOSEX = 0
      DO 600 I = 1,99
      WORD = I/36 + 1
      BIT = MOD(I,36)
      IF (FLD(BIT,1,MASK(WORD)).EQ.0) GO TO 600
      NOSEX = NOSEX + 1
      SEX(NOSEX) = I
```

```
283.     600  CONTINUE
284.          RETURN
285.          END
```

```
@FOR,SA     N.CONVRT,.CONVRT

      SUBROUTINE CONVRT(VALUE,INDXS,UNICODE,UNITS)
      INTEGER TABLE,UNICODE
C
C     SET UP TABLE TO MAP VARIABLE CODE TO PROPER TABLE NUMBER.
C
      DIMENSION TABLE(199),UNITS(4)
      DATA ANULL /0576377777777/
      DATA (TABLE(I),I=1,72 )/00401,00401,07712,07712,00018,02204,00401,
     .                        00401,00018,00401,00401,00018,00501,00018,
     .                        03808,07712,07712,00018,00401,02204,00018,
     .                        00401,00401,00018,00018,02204,00000,03808,
     .                        07712,03808,03204,02906,03808,03808,
     .                        3800/
C
      DATA(TABLE(I),I=73,199)/00000,00401,02204,00000,03808,03204,00000,
     .                        00401,02204,00000,00000,00000,00401,00401,
     .                        00401,03607,04309,02906,00600,00301,00301,
     .                        04517,04209,04209,04209,04209,0860),00501,
     .                        9900/
C
C
      UNITS(1)='UNITLE'
      UNITS(2)='SS    '
      UNITS(3)='UNITLE'
      UNITS(4)='SS    '
C
C
      IF (FLD(0,36,VALUE).EQ.FLD(0,36,ANULL)) RETURN
C
C
C
C     MAP CODES 1-72 AND 101-172 TO CONTINUOUS TABLE POSITIONS 1-199.
C
      INDEX=INDXS
      IF(INDEX.GT.100) INDEX=INDEX-28
C
C     IF INDEX IS OUT OF RANGE, RETURN VALUE.
C
      IF(INDEX.GE.1.AND.INDEX.LE.199) GO TO 101
      PRINT 102,INDXS
      UNITS(1) = 'UNITS '
      UNITS(2) = 'ERROR '
      RETURN
C
C     CALCULATE LABEL FOR APPROPRIATE UNIT-CODE TABLE SUBROUTINE.
C
  101 IF(TABLE(INDEX).EQ.0) RETURN
      ITMPCD = TABLE(INDEX) / 100
      INDEX = TABLE(INDEX) - ITMPCD* 100
      IF (UNICODE .EQ. 0) UNICODE = - ITMPCD
C
C     BRANCH TO CORRECT UNIT-CODE TABLE SUBROUTINE.
C
      GO TO (1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18),INDEX
```

```
      C
      C     'LINEAR' TABLE.
      C
            1 CALL LINEAR(VALUE,UNICDE,UNITS)
              RETURN
      C
      C     'PLANAR' TABLE.
      C
            2 CALL PLANAR(VALUE,UNICDE,UNITS)
              RETURN
      C
      C     'VOLUME' TABLE.
      C
            3 CALL VOLUME(VALUE,UNICDE,UNITS)
              RETURN
      C
      C     'SHORT TIME' TABLE.
      C
            4 CALL SHRTME(VALUE,UNICDE,UNITS)
              RETURN
      C
      C     'LONG TIME' TABLE.
      C
            5 CALL LNGTME(VALUE,UNICDE,UNITS)
              RETURN
      C
      C     'VELOCITY' TABLE.
      C
            6 CALL SPEEDS(VALUE,UNICDE,UNITS)
              RETURN
      C
      C     'TEMPERATURE' TABLE.
      C
            7 CALL TMPTUR(VALUE,UNICDE,UNITS)
              RETURN
      C
      C     'ANGLE' TABLE.
      C
            8 CALL ANGLES(VALUE,UNICDE,UNITS)
              RETURN
      C
      C     'PER CENT' TABLE.
      C
            9 CALL PERCNT(VALUE,UNICDE,UNITS)
              RETURN
      C
      C     'PRESSURE' TABLE.
      C
           10 CALL PRESSR(VALUE,UNICDE,UNITS)
              RETURN
      C
      C     'ACOUSTIC PRESSURE' TABLE.
      C
           11 CALL ACPRES(VALUE,UNICDE,UNITS)
              RETURN
      C
      C     'FREQUENCY' TABLE.
```

```
      C
   12 CALL CYCLES(VALUE,UNICDE,UNITS)
      RETURN
      C
      C     'ENERGY' TABLE.
      C
   13 CALL ENERGY(VALUE,UNICDE,UNITS)
      RETURN
      C
      C     'POWER' TABLE.
      C
   14 CALL POWER(VALUE,UNICDE,UNITS)
      RETURN
      C
      C     'INTENSITY' TABLE.
      C
   15 CALL INTENS(VALUE,UNICDE,UNITS)
      RETURN
      C
      C     'PRESSURE LEVEL' TABLE.
      C
   16 CALL PRSLVL(VALUE,UNICDE,UNITS)
      RETURN
      C
      C     'DENSITY' TABLE.
      C
   17 CALL DENSTY(VALUE,UNICDE,UNITS)
      RETURN
      C
      C     'DEEBEE' TABLE.
      C
   18 CALL DEEBEE(VALUE,UNICDE,UNITS)
      RETURN
  102 FORMAT(' ERROR--VARIABLE NAME CODE',I3,' OUT OF RANGE.')
      END
```

```
@FOR,S#     N.CVTPRO,.CVTPRO

      SUBROUTINE CVTPRO
      REAL VAL,VALUE
      IMPLICIT INTEGER(A-Z)
C
C     PROGRAM TO CONVERT FROM A VALUE IN SPECIFIED UNITS
C     TO THE APPROPRIATE VALUE IN REQUESTED UNITS.
C
C     'TABLE' GIVES DIRECTION TO APPROPRIATE TABLES.
C
      DIMENSION TABLE(99),UNITS(9),UNIT(2)
      DATA (TABLE(I),I=1,99)/0,11*1,4*2,4*3,5*4,2*5,7*6,
     *                       2*7,2*8,3*19,3*9,3*20,9*10,
     *                       17,19,2*14,3*13,2*18,4*19,2*17,
     *                       2*19,3*15,3*12,3*16,19,0,8,1,19,
     *                       2*13,16,18,4*16,19,2*7,6/
      DATA ANSWER/'YES   '/
C
C     ASK FOR INPUT OF NUMBER OF CONVERSIONS TO BE MADE.
C
  201 PRINT 101
      READ(5,102) NOCON
      IF(NOCON.LE.0) RETURN
      NCON=NOCON
      DO 209 I=1,NOCON
C
C     ASK FOR INPUT OF VALUE, UNITS FROM, UNITS TO.
C
      DO 203 J=1,3
      PRINT 103
      READ(5,102) VAL,UFRM,UTO
C
C     ARE UNITS FROM AND UNITS TO LEGITIMATE?
C
      IF(UFRM.GE.1.AND.UFRM.LE.99) GO TO 202
C
C     ELSE, PRINT ERROR.
C
      PRINT 104,UFRM
      GO TO 203
  202 IF(UTO.GE.1.AND.UTO.LE.99) GO TO 204
C
C     ELSE, PRINT ERROR.
C
      PRINT 105,UTO
  203 CONTINUE
C
C     PRINT 'FIVE ERRORS IN A ROW' MESSAGE.
C
      NCON=NCON-1
      PRINT 106,NCON
      GO TO 209
C
C     SEE IF BOTH CODES ARE OF SAME TYPE.
C
```

```
      204 IF(TABLE(UFRM).EQ.TABLE(UTO)) GO TO 205
C
C     ELSE, PRINT TYPE ERROR.
C
          NCON=NCON-1
          PRINT 107,UFRM,UTO,NCON
          GO TO 209
C
C     CONVERT TO INTERNAL UNITS.
C
      205 INDEX=TABLE(UFRM)
          UNICDE=UFRM
          VALUE=VAL
          K=0
      206 GO TO (1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,201),INDEX
C
C     BRANCH TO CORRECT UNIT-CODE TABLE SUBROUTINE.
C
C
C     'LINEAR' TABLE.
C
        1 CALL LINEAR(VALUE,UNICDE,UNITS)
          GO TO 207
C
C     'PLANAR' TABLE.
C
        2 CALL PLANAR(VALUE,UNICDE,UNITS)
          GO TO 207
C
C     'VOLUME' TABLE.
C
        3 CALL VOLUME(VALUE,UNICDE,UNITS)
          GO TO 207
C
C     'SHORT TIME' TABLE.
C
        4 CALL SHRTME(VALUE,UNICDE,UNITS)
          GO TO 207
C
C     'LONG TIME' TABLE.
C
        5 CALL LNGTME(VALUE,UNICDE,UNITS)
          GO TO 207
C
C     'VELOCITY' TABLE.
C
        6 CALL SPEEDS(VALUE,UNICDE,UNITS)
          GO TO 207
C
C     'TEMPERATURE' TABLE.
C
        7 CALL TMPTUR(VALUE,UNICDE,UNITS)
          GO TO 207
C
C     'ANGLE' TABLE.
C
        8 CALL ANGLES(VALUE,UNICDE,UNITS)
```

```
      GO TO 207
C
C     'PER CENT' TABLE.
C
    9 CALL PERCNT(VALUE,UNICDE,UNITS)
      GO TO 207
C
C     'PRESSURE' TABLE.
C
   10 CALL PRESSR(VALUE,UNICDE,UNITS)
      GO TO 207
C
C     'ACOUSTIC PRESSURE' TABLE.
C
   11 CALL ACPRES(VALUE,UNICDE,UNITS)
      GO TO 207
C
C     'FREQUENCY' TABLE.
C
   12 CALL CYCLES(VALUE,UNICDE,UNITS)
      GO TO 207
C
C     'ENERGY' TABLE.
C
   13 CALL ENERGY(VALUE,UNICDE,UNITS)
      GO TO 207
C
C     'POWER' TABLE.
C
   14 CALL POWER(VALUE,UNICDE,UNITS)
      GO TO 207
C
C     'INTENSITY' TABLE.
C
   15 CALL INTENS(VALUE,UNICDE,UNITS)
      GO TO 207
C
C     'DB LOSS/ATTENUATION' TABLE.
C
   16 CALL DBLOSS(VALUE,UNICDE,UNITS)
      GO TO 207
C
C     'PRESSURE LEVEL' TABLE.
C
   17 CALL PRSLVL(VALUE,UNICDE,UNITS)
      GO TO 207
C
C     'SOURCE LEVEL/TARGET STRENGTH' TABLE.
C
   18 CALL SORGET(VALUE,UNICDE,UNITS)
      GO TO 207
C
C     'MISCELLANEOUS' TABLE.
C
   19 PRINT 111,UFRM,UTO
      GO TO 209
C
```

```
      C     'DENSITY' TABLE.
      C
   23 CALL DENSTY(VALUE,UNICDE,UNITST)
      GO TO 207
  207 IF(K.EQ.1) GO TO 208
      C
      C     SAVE 'FROM' UNITS.
      C
      UNIT(1)=UNITS(1)
      UNIT(2)=UNITS(2)
      C
      C     CONVERT FROM INTERNAL UNITS.
      C
      UNICDE=-UTO
      K=1
      GO TO 204
      C
      C     WRITE CONVERSION.
      C
  208 PRINT 108,VAL,UNIT(1),UNIT(2),VALUE,UNITS(1),UNITS(2)
      NCON=NCON-1
  209 CONTINUE
      C
      C     DO AGAIN?
      C
      PRINT 109
      READ(5,110) RESPNS
      IF(RESPNS.EQ.ANSWER) GO TO 201
      RETURN
  101 FORMAT(' ENTER NUMBER OF CONVERSIONS TO BE MADE.')
  102 FORMAT( )
  103 FORMAT(' ENTER VALUE, "UNITS FROM", AND "UNITS TO".')
  104 FORMAT(' ERROR--"UNITS FROM" CODE',I3,' NOT LEGIT...TRY AGAIN.')
  105 FORMAT(' ERROR--"UNITS TO" CODE',I3,' NOT LEGIT...TRY AGAIN.')
  106 FORMAT(' HOLD IT! THAT IS THREE ERRORS IN A ROW! PLEASE CHECK CODE
     #S CAREFULLY. YOU HAVE',I3,' CONVERSIONS LEFT.')
  107 FORMAT(' ERROR--"UNITS FROM" CODE',I3,' NOT OF SAME TYPE AS "UNIT
     +S TO" CODE',I3,' .',/,' YOU HAVE ',I3,' CONVERSIONS LEFT.')
  108 FORMAT(1H ,G12.5,1H ,2A6,' = ',G12.5,1H ,2A6)
  109 FORMAT(' WOULD YOU LIKE TO DO MORE?')
  110 FORMAT(A6)
  111 FORMAT(' NOTA BENE--"UNITS TO" CODE ',I3,' NOT,/
     + CONVERTIBLE TO "UNITS FROM" CODE ',I3,/)
      END
```

```
@FOR,S# N.CYCLES,,CYCLES

      SUBROUTINE CYCLES(VALUE,UNICDE,UNITS)
C
C     TABLE OF CYCLES MEASURE CONVERSION FACTORS.
C
      INTEGER ENTRY(3,3),UNITS(4),UNICDE
C
C     FACTORS ARE DOUBLE PRECISION.
C
      DOUBLE PRECISION FACTOR(3)
C
C     PUT UNIT CODES INTO ENTRY.
C
      DATA ((ENTRY(I,J),I=1,3)/77,78,79/
C
C     PUT IN ALPHA UNITS.
C
      DATA ((ENTRY(I,J),I=2,3),J=1,3)/'HERTZ ','      ',
     *              'KILOHE','RTZ   ','C.P.H.','      '/
C
C     ENTER INTERNAL UNITS.
C
      UNITS(3)='HERTZ '
      UNITS(4)='      '
C
C     IF UNICDE=0, THEN SET CODE TO STANDARD UNITS.
C
      IF(UNICDE.EQ.0) UNICDE=-77
C
C     SET CONVERSION FACTORS.
C
      DATA (FACTOR(I),I=1,3)/1.0D0,1.0D3,6.0D1/
C
C     IF CODE IS IN TABLE, PERFORM CONVERSION. ALSO, ENTER UNITS ALPHA
C          CODE INTO 'UNITS'.
C
      ICODE=IABS(UNICDE)
      DO 2 I=1,3
      IF(ICODE.NE.ENTRY(1,I)) GO TO 2
      IF(UNICDE.GE.0) VALUE=VALUE*FACTOR(I)
      IF(UNICDE.LT.0) VALUE=VALUE/FACTOR(I)
      DO 1 J=1,2
      UNITS(J)=ENTRY(J+1,I)
    1 CONTINUE
C
C     AT THIS POINT, CONVERSION IS COMPLETE.
C
      RETURN
C
C     ELSE, CHECK REST OF TABLE.
C
    2 CONTINUE
C
C     IF CODE NOT FOUND, RETURN WITH 'UNITS ERROR' IN 'UNITS'.
C
```

```
      UNITS(1)='UNITS '
      UNITS(2)='ERROR '
      PRINT 101,ICODE
101   FORMAT(' ERROR--UNIT CODE',I3,' NOT IN FREQUENCY TABLE.')
      RETURN
      END
```

```
@ELT,L     N.DATECK,.DATLCK

      SUBROUTINE DATECK (IDATE,FDATE,ITIM,FTIM,*)
      IMPLICIT INTEGER (A-Z)
C
      INCLUDE RETREV.LIST
C
C
      IYR = IDATE / 10000
      FYR = FDATE / 10000
      IMO =(IDATE - IYR * 10000) / 100
      FMO =(FDATE - FYR * 10000) / 100
      IDAY = (IDATE - IYR*10000 - IMO*100)
      FDAY = (FDATE - FYR*10000 - FMO*100)
C
      TMP = FYR*12 - (12-FMO)
      TMP1= IYR*12 - (12-IMO)
      IF (TMP-TMP1 .GE. 12) GO TO 50
C
      IF (NMON .EQ. 0 .OR. NMON .EQ. 12) GO TO 50
      DO 30 I=1,NMON
      IF (FMO .GT. IMO) GO TO 20
      IF (MONTHS(I) .GE. IMO .OR.  MONTHS(I) .LE. FMO) GO TO 50
      GO TO 30
   20 IF (MONTHS(I) .GE. IMO .AND. MONTHS(I) .LE. FMO) GO TO 50
   30 CONTINUE
C
      RETURN 5
C
   50 IF (FMYR .EQ. 0) GO TO 60
      IF (FYR .LT. FMYR) RETURN 5
   60 IF (TOYR .EQ. 0) GO TO 70
      IF (IYR .GT. TOYR) RETURN 5
C
   70 IF (FMDAY .EQ. 0) GO TO 80
      IF (FDAY .LT. FMDAY) RETURN 5
   80 IF (TODAY .EQ. 0) GO TO 100
      IF (IDAY .GT. TODAY) RETURN 5
C
  100 IF (FMTIM .EQ. 0 .AND. TOTIM .EQ. 0) RETURN
      IF (TOTIM .LT. FMTIM) GO TO 110
      IF (FTIM .LT. FMTIM .OR. ITIM .GT. TOTIM) RETURN 5
      RETURN
  110 IF (FTIM .GE. FMTIM .AND. ITIM .LE. TOTIM) RETURN 5
      RETURN
      END
```

```
@FOR,S4     N.DATLST,.DATLST

      1.            SUBROUTINE DATLST (OPTION,CHAR,RUNTYP)
      2.      C
      3.      C
      4.            IMPLICIT INTEGER (A-Z)
      5.      C
      6.      C
      7.            COMMON /RUN    /PORA,RILAT,RINS,RFLAT,RFNS,RILON,RIEW,RFLON,RFEW,
      8.           *                RIMO,RIDAY,RIYR,RITIM,RIZON,RFMO,RFDAY,RFYR,RFTIM,
      9.           *                RFZON,NAVCOD,DTASRC,SYSSRC,SYSRCR,SRCTYP,RCRTYP,
     10.           *                CLAS,DESCR,WAVDIR,WAVHT,HTUNIT,WAVPRD,PRDUNT,
     11.           *                SEASTA,WNDDIR,WNDVEL,VELUNT,SWLDIR,SWLHT,
     12.           *                SHTUNT,SWLPRD,SPDUNT,WEATHR,BOTDPH,DPHUNT,BTMSLP,
     13.           *                BATHY,INFO(9),RSTAT(2),RUNNO
     14.            REAL WAVDIR,WAVHT,WAVPRD,SEASTA,WNDDIR,WNDVEL,SWLDIR,
     15.           *     SWLHT,SWLPRD,WEATHR,BOTDPH,BTMSLP
     16.      C
     17.      C
     18.            COMMON /CPV    /NCON,CON(4,30),NPAR,PAR(30,30),NVAR,IVAR(3,30),
     19.           *                NDATS,NKOBS(50),NDATA,VALRNG(2,30,50),DATA(30000)
     20.            DIMENSION ICON(4,30),IPAR(30,30),IVLRNG(2,30,50)
     21.            EQUIVALENCE (CON(1,1),ICON(1,1)),(PAR(1,1),IPAR(1,1)),
     22.           *           (VALRNG(1,1,1),IVLRNG(1,1,1))
     23.            REAL CON,PAR,DATA,VALRNG
     24.      C
     25.            INCLUDE DATCOM,LIST
     26.      C
     27.            REAL SAVE(30)
     28.            DIMENSION UNITS(4),NAME(10),PSUB(30)
     29.            DIMENSION VBUF(10),DUM(2)
     30.            REAL VBUF,DUM
     31.            DIMENSION WORD(4),CHAR(81),ERRCK(4)/999,99,99,999/,ECSR(4)
     32.            INCLUDE GETLST,LIST
     33.            EQUIVALENCE (ECSR(1),EXP1)
     34.      C
     35.            AETYPE = RUNTYP
     36.            IF (AETYPE.NE.3) GO TO 8
     37.            WRITE (6,2)
     38.      2     FORMAT ('0ENTER DESIRED DATA TYPE!'/4X,'ACOUSTIC ONLY(''A''), ENVI
     39.           1RONMENTAL ONLY(''E''), OR BOTH(''B'')')
     40.            READ (5,13,END=1,ERR=6) MODE
     41.            IF (MODE.EQ.'B'.OR.MODE.EQ.' ') GO TO 5010
     42.            IF (MODE.EQ.'E') GO TO 4
     43.            IF (MODE.NE.'A') GO TO 6
     44.            AETYPE = 1
     45.            WRITE (6,3)
     46.      3     FORMAT (5X,'ACOUSTIC ONLY')
     47.            GO TO 8
     48.      4     AETYPE = 2
     49.            WRITE (6,5)
     50.      5     FORMAT (5X,'ENVIRONMENTAL ONLY')
     51.            GO TO 8
     52.      5010  AETYPE = 3
     53.            WRITE (6,5020)
     54.      5020  FORMAT (5X,'ACOUSTIC AND ENVIRONMENTAL')
```

```
      GO TO 8
C
C        ERROR
C
6     WRITE (6,7)
7     FORMAT (' *** FORMAT ERROR OR INVALID DATA TYPE ***')
      GO TO 1
C
C        FIND WHICH RUNS ARE TO BE PRINTED
C
8     WRITE (6,9)
9     FORMAT ('0ENTER EXPER. NO.,CRUISE NO.,STATION NO.,RUN NO./ENTER '
     1'ALL'/ENTER 'END'')
      READ (5,13,END=1,ERR=17) (CHAR(I),I=1,80)
13    FORMAT (80A1)
      WRITE (6,14) (CHAR(I),I=1,80)
14    FORMAT (3X,80A1)
      CRUI = -99999999
      STAI = -99999999
      RUNI = -99999999
      CALL FREEFD (CHAR,WORD,EXPI,NWORDS,4,S17)
      W = ' '
      FLD (0,6,W) = FLD (0,6,EXPI)
      IF (W.EQ.'E') GO TO 900
      IF (W.NE.'A'.AND.NWORDS.GE.1) GO TO 15
      CALL GETINT
      USERAP = .FALSE.
      ALL = .TRUE.
      GO TO 10
C
C        CHECK FOR ERROR
C
15    DO 16 I = 1,NWORDS
16    IF (ECSR(I).LT.1.OR.ECSR(I).GT.ERRCK(I)) GO TO 17
      GO TO 19
17    WRITE (6,18)
18    FORMAT ('0*** FORMAT ERROR OR NUMBER OUT OF RANGE ***')
      GO TO 8
C
19    CALL GETINT
      USERAP = .FALSE.
      ALL = .FALSE.
2000  CALL GETRUN (DATYPE,EXPNO,CRUNO,STANO,S22)
      GO TO 24
22    IF (NRUNSF.GT.0) GO TO 8
      WRITE (6,23)
23    FORMAT ('0*** SPECIFIED RECORD(S) NOT FOUND ***')
      GO TO 8
C
   10 CALL GETRUN (DATYPE,EXPNO,CRUNO,STANO,S900)
C
C          PRINT HEADINGS
C
24    PRINT 11
11    FORMAT ('1')
      IF (DATYPE .EQ. 4) GO TO 20
C
```

```
      WRITE (6,12)
   12 FORMAT (////,' ACOUSTIC DATA')
      GO TO 25
   20 WRITE (6,21)
   21 FORMAT (////,' ENVIRONMENTAL DATA')
C     GET EXPERIMENT NAME AND PRINT
   25 CALL NAMEIT(11,EXPNO,NAME,NR)
      WRITE (6,26) EXPNO,(NAME(J),J=1,NR)
   26 FORMAT (T5,'EXPERIMENT: ',T16,I9,3X,6A6)
      WRITE(6,27) CRUNO,STANO,RUNNO
   27 FORMAT(T5,'CRUISE:',T16,I9/T5,'STATION:',T16,I9/
     * T5,'RUN:',T16,I9)
C
C          SET UP AREA & DATE FOR PRINT
C
      RINS = 'N'
      IF (RILAT .GE. 0) GO TO 30
      RINS = 'S'
      RILAT = -RILAT
C
   30 RFNS = 'N'
      IF (RFLAT .GE. 0) GO TO 35
      RFNS = 'S'
      RFLAT = -RFLAT
C
   35 RIEW = 'E'
      IF (RILON .GE. 0) GO TO 40
      RIEW = 'W'
      RILON = -RILON
C
   40 RFEW = 'E'
      IF (RFLON .GE. 0) GO TO 45
      RFEW = 'W'
      RFLON = -RFLON
C
   45 RIYR = RIDAY / 10000
      RFYR = RFDAY / 10000
      RIMO = (RIDAY - RIYR*10000) / 100
      RFMO = (RFDAY - RFYR*10000) / 100
      RIDAY = RIDAY - RIYR*10000 - RIMO*100
      RFDAY = RFDAY - RFYR*10000 - RFMO*100
C
      IF (RITIM .LE. 2400) GO TO 50
      RITIM = 99999
      RIZON = '+'
   50 IF (RFTIM .LE. 2400) GO TO 55
      RFTIM = 99999
      RFZON = '+'
C
C          PRINT AREA, DATE & TIME
C
   55 R11 = RILAT / 1000
      R12 = RILON / 1000
      R13 = RFLAT / 1000
      R14 = RFLON / 1000
C
      R21 = (RILAT - R11*1000) / 10
```

```
      R22 = (RILON - R12*1000) / 10
      R23 = (RFLAT - R13*1000) / 10
      R24 = (RFLON - R14*1000) / 10
C
      R31 = RILAT - R11*1000 - R21*10
      R32 = RILON - R12*1000 - R22*10
      R33 = RFLAT - R13*1000 - R23*10
      R34 = RFLON - R14*1000 - R24*10
C
      WRITE (6,57) R11,R21,R31,RINS,R12,R22,R32,RIEW,
     *             R13,R23,R33,RFNS,R14,R24,R34,RFEW,
     *          RIMO,RIDAY,RIYR,RITIM,RIZON,
     *          RFMO,RFDAY,RFYR,RFTIM,RFZON
57    FORMAT (//T11,'INITIAL',T34,'FINAL',/T5,                    'LATITU
     *DE  LONGITUDE  LATITUDE  LONGITUDE',//,T5,    I2,1X,J2,'°',I1,
     *A1,2X,I3,1X,J2,'°',I1,A1,3X,I2,1X,J2,'°',I1,A1,2X,I3,1X,J2,'°',I1,
     *A1,//,T17,'DATE',T26,'TIME',/,T5,'INITIAL',T15,2(I2,'/'),I2,3X,J4,
     *A1,/,T5,'FINAL',T15,2(I2,'/'),I2,3X,J4,A1)
C
C          IF ACOUSTIC RUN, WRITE SYSTEM SOURCE AND TYPE
C
      WRITE (6,59)
59    FORMAT (///T20,' CODE   NAME')
      IF(DATYPE.NE.3) GO TO 64
      WRITE(6,60)
60    FORMAT(T5,'** SONAR SYSTEM **')
      CALL NAMEIT(4,SYSSRC,NAME,NR)
      WRITE(6,61)SYSSRC,(NAME(J),J=1,NR)
61    FORMAT(T5,'SOURCE:',T20,I4,3X,5A6)
      IF(SYSSRC.NE.SYSRCR) CALL NAMEIT(4,SYSRCR,NAME,NR)
      WRITE(6,62)SYSRCR,(NAME(J),J=1,NR)
62    FORMAT(T5,'RECEIVER:',T20,I4,3X,5A6)
      WRITE(6,63)
63    FORMAT(/T5,'** TRANSDUCER TYPE **')
      CALL NAMEIT(5,SRCTYP,NAME,NR)
      WRITE(6,61) SRCTYP,(NAME(J),J=1,NR)
      IF(SRCTYP.NE.RCRTYP) CALL NAMEIT(5,RCRTYP,NAME,NR)
      WRITE(6,62) RCRTYP,(NAME(J),J=1,NR)
C     NAV CODE ETC...
C
64    CALL NAMEIT(3,NAVCOD,NAME,NR)
      WRITE(6,65) NAVCOD,(NAME(J),J=1,NR)
65    FORMAT(/T5,'NAVIGATION:',T20,I4,3X,5A6)
      CALL NAMEIT(6,DTASRC,NAME,NR)
      WRITE(6,66) DTASRC,(NAME(J),J=1,NR)
66    FORMAT(T5,'DATA SOURCE:',T20,I4,3X,5A6)
      CALL NAMEIT(2,CLAS,NAME,NR)
      WRITE(6,67) CLAS,(NAME(J),J=1,NR)
67    FORMAT (T5,'SECURITY CLASS:',I3,3X,5A6)
      ISET=7
      IF(DATYPE.EQ.3) ISET=8
      CALL NAMEIT(ISET,DESCR,NAME,NR)
      WRITE(6,68) DESCR,(NAME(J),J=1,NR)
68    FORMAT(T5,'RUN DESCR.:',T20,I4,3X,6A6)
C
C          IF TYPE IS ENV, WRITE SHIP NAME AND BATHEMETRY AVAILABLE
C
```

```
      IF (DATYPE .EQ. 3) GO TO 80
C
      WRITE (6,70) (INFO(I),I=1,9)
   70 FORMAT (T5,'SHIP''S NAME:  ',9A6)
C
      IF(FLD(33,1,RSTAT(2)).EQ.0) GO TO 80
      WRITE(6,75)
   75 FORMAT(T5,'** BATHYMETRY AVAILABLE - SEE NOTES')
C
   80 IF (NCON .EQ. 0) GO TO 100
      WRITE (6,85)
   85 FORMAT (///,T5,'** CONSTANTS **')
      IF (DATYPE.NE.3) GO TO 82
      WRITE (6,81)
   81 FORMAT (T60,'ACOUSTIC')
      GO TO 84
   82 WRITE (6,83)
   83 FORMAT (T58,'ENVIRONMENTAL')
   84 WRITE (6,86)
   86 FORMAT (T15,'NAME',T33,'VALUE',T48,'UNITS',T61,'METHOD',/)
C
      DO 95 I=1,NCON
      DUM(1) = CON(3,I)
      CALL CHECK (DUM(1),ICON(1,I),UNITS)
      CALL NAMGET (ICON(1,I),NAME)
      IF (ICON(9,I).EQ.0) ICON(9,I) = 1
      WRITE (6,87) (NAME(J),J=1,2),CON(3,I),UNITS(1),UNITS(2),ICON(9,I)
   87 FORMAT (T1,2A6,2X,G16.5,4X,2A6,2X,I9)
   95 CONTINUE
C
  100 IF (NDATS .EQ. 0) GO TO 600
C
C          LOOP TO SET PARAMETER POINTERS TO 1
C
      DO 110 I=1,30
      PSUB(I) = 1
  110 CONTINUE
C
      DSN = 1            @CURRENT DATA SET # FOR PARAMETERS
      RKT = 0            @INIT FOR RO@ COUNTING
      WRITE (6,111)
  111 FORMAT ('1')
      DO 500 I=1,NDUSED
C
C          PRINT DATA SET N
C
      WRITE (6,112) NRS(I)
  112 FORMAT (/T5,'DATA SET ',I2)
C
      IF (NPAR .EQ. 0) GO TO 200   @ GO DO VARIABLES
      IF (NRS(I) .EQ. 1) GO TO 120
C
  114 J = NPAR
  115 PSUB(J) = PSUB(J) +1
      IF (PSUB(J) .LE. IPAR(1,J)) GO TO 117
      PSUB(J) = 1
      J = J -1
```

```
      GO TO 115
  117 DSN = DSN + 1
      IF (DSN .NE. NHS(I)) GO TO 114
C
C          PRINT OUT PARAMETERS
C
  120 WRITE (6,123)
  123 FORMAT (//,T5,'** PARAMETERS **')
      IF (DATYPE.NE.3) GO TO 125
      WRITE (6,124)
  124 FORMAT (T60,'ACOUSTIC')
      GO TO 127
  125 WRITE (6,126)
  126 FORMAT (T58,'ENVIRONMENTAL')
  127 WRITE (6,86)
C
          DO 190 K=1,NPAR
           PSB = PSUB(K)+4
           DUM(1) = PAR(PSB,K)
           CALL CHECK(DUM(1),IPAR(2,K),UNITS)
          CALL NAMGET (IPAR(2,K),NAME)
          IF (IPAR(4,K).EQ.0) IPAR(4,K) = 1
          WRITE (6,87) (NAME(L),L=1,2),PAR(PSB,K),UNITS(1),
     *                 UNITS(2),IPAR(4,K)
  190     CONTINUE
C
C              PRINT VARIABLES
C
  200 WRITE (6,201)
  201 FORMAT (//,T5,'** VARIABLES **')
C
      IF (OPTION.EQ.4) CALL SHORT (NHS(I),DATYPE,IVLRNG,UNITS,NAME,*500)
      L1 = 1
  210 L2 = L1 + 4
      IF (L2 .GT. NVAR) L2 = NVAR
C
      L3 = 0
      DO 250 L4 =L1,L2
      L3 = L3 +1
      IF (1.EQ.1) SAVE(L3) = DATA(L4)
      CALL NAMGET (IVAR(I,L4),VBUF(2*L3-1))
  250 CONTINUE
      T1 = 2*L3
      WRITE (6,255) (VBUF(LP),LP=1,T1)
  255 FORMAT (//,T8,5(1X,2A6))
C
      L3 = 0
      DO 270 L4=L1,L2
      L3 = L3 +1
      DUM(1) = SAVE(L3)
      CALL CHECK (DUM(1),IVAR(I,L4),UNITS)
      FLD(0,36,VBUF(2*L3-1)) = UNITS(1)
      FLD(0,36,VBUF(2*L3  )) = UNITS(2)
  270 CONTINUE
      T1 = 2 * L3
      WRITE (6,275) (VBUF(LP),LP=1,T1)
  275 FORMAT (T8,5(1X,2A6))
```

```
C
      WRITE (6,290) (IVAR(3,LP),LP=L1,L2)
  290 FORMAT (T9,'METHOD',T4,9(9X,I4))
      WRITE (6,291)
  291 FORMAT ()
C
C          COMPUTE BEGINNING SUBSCRIPT FOR DATA
C          IN DATA SET NRS(I)
C
      RKT = 0
      IF (NRS(I) .EQ. 1) GO TO 300
      IK = NRS(I-1)
      DO 295 IXX=1,IK
      RKT = RKT + NROWS(IXX)
  295 CONTINUE
  300 CONTINUE          @ NOT END OF A DO LOOP
C
C          LOOPS TO PRINT DATA VALUES
C
      II = NRS(I)
      L7 = NROWS(II)
      DO 450 L5 = 1,L7
      RKT = RKT +1
C
           L3 = 0
           DO 400 L4 =L1,L2
           L3 = L3 +1
           DSUB = RKT*NVAR - NVAR
           DUM(1) = DATA(DSUB+L4)
           CALL CHECK (DUM(1),IVAR(1,L4),UNITS)
           VBUF(L3) = DATA(DSUB+L4)
  400      CONTINUE
C
      WRITE (6,425) L5,(VBUF(LP),LP=1,L3)
  425 FORMAT (T2,I5,5(G12.5,1X))
C
  450 CONTINUE
C
      IF (L2 .EQ. NVAR) GO TO 490
      L1 = L2 +1
      GO TO 210
  490 WRITE (6,111)
C
  500 CONTINUE
C
  600 IF (ALL) GO TO 10
      GO TO 2000
C
C
  900 WRITE (6,901)
  901 FORMAT (///,T7,'LISTING OF RUN DATA COMPLETED')
      RETURN
      END
```

```
@FOR,S4     N.DBLOSS,.DBLOSS

      SUBROUTINE DBLOSS(VALUE,UNICDE,UNITS)
C
C     TABLE FOR DB LOSS AND ATTENUATION CONVERSION FACTORS.
C
      DIMENSION FACTOR(8)
      INTEGER  ENTRY(3,8),UNITS(4),UNICDE
C
C     PUT UNIT CODES INTO ENTRY.
C
      DATA (ENTRY(1,I),I=1,8)/87,81,82,90,92,93,94,95/
C
C     ENTER CONVERSION FACTORS.
C
      DATA (FACTOR(I),I=1,8)/.8,-59.2,0.,-60.,0.,-.8,59.2,60./
C
C     PUT IN ALPHA UNITS.
C
      DATA ((ENTRY(I,J),I=2,3),J=1,8)/'DB//I ','YARD  ',
     *            'DB//I ','KYARD ','DB//I ','METER ',
     *            'DB//I ','KMETER','DB/MET','ER    ',
     *            'DB/YAR','D     ','DB/KYA','RD    ',
     *            'DB/KME','TER   '/
C
C     INITIALIZE 'UNITS' TO ERROR MESSAGE IN CASE CODE NOT FOUND.
C
      UNITS(1)='UNITS '
      UNITS(2)='ERROR '
C
C     RETRIEVAL IN STANDARD DB LOSS UNITS.
C
      IF(UNICDE.NE.0) GO TO 1
      UNICDE=-82
C
C     ENTER INTERNAL UNITS.
C
      UNITS(3)='DB//I'
      UNITS(4)='METER '
      GO TO 2
C
C     RETRIEVAL IN STANDARD ATTENUATION UNITS.
C
    1 IF(UNICDE.NE.-1) GO TO 2
      UNICDE=-92
C
C     ENTER INTERNAL UNITS.
C
      UNITS(3)='DB/MET'
      UNITS(4)='ER    '
C
C     IF CODE IS IN TABLE, PERFORM CONVERSION. ALSO, ENTER UNITS ALPHA
C          CODE INTO 'UNITS'.
C
    2 ICODE=IABS(UNICDE)
      DO 3 I=1,8
```

```
      IF(ICODE.NE.ENTRY(1,I)) GO TO 3
      IF(UNICDE.GT.0) VALUE=VALUE*FACTOR(I)
      IF(UNICDE.LT.0) VALUE=VALUE-FACTOR(I)
      UNITS(1)=ENTRY(2,I)
      UNITS(2)=ENTRY(3,I)
C
C     AT THIS POINT, CONVERSION IS COMPLETE.
C
      RETURN
    3 CONTINUE
C
C     ELSE, UNIT CODE IS NOT CONVERTIBLE.
C
      PRINT 101,ICODE
      PRINT 112
  112 FORMAT(' ERROR--UNIT CODE NOT CONVERTIBLE!')
      UNITS(1)='UNITS '
      RETURN
  101 FORMAT(' ERROR--UNIT CODE',I3,' NOT IN DB LOSS TABLE.')
      END
```

```
@FOR,SA     N.DEEBEE,.DEEBEE

      SUBROUTINE DEEBEE(VALUE,UNICDE,UNITS)
C
C     TABLE OF DB UNITS.
C
      INTEGER ENTRY(3,27),UNITS(4),UNICDE
C
C     PUT CODES INTO 'ENTRY'.
C
      DATA (ENTRY(1,I),I=1,27)/39,40,41,58,66,67,68,69,72,73,83,80,96,
     * 87,81,82,90,92,93,94,95,64,65,91,101,102,103/
C
C     ENTER ALPHA UNITS.
C
      DATA ((ENTRY(I,J),I=2,3),J=1,27)/'DB/YD**','*2    ',
     *           '//IUBR','2/S/HZ','CM3/6R','AM    ',
     *           'PRESSR',' RATIO','DB//IW','ATT   ',
     *           'DB//IW','/M**2 ','DB//IW','/CM**2',
     *           '//IERG','SC/CM2','DB//IE','RG/CM2',
     *           'IERG/S','EC/CM ','//JLE/','CM2-HZ',
     *           'DB    ','      ','//ERG/','CM2SIY',
     *           'DB    ','      ','DB//I ','KYARD ',
     *           'DB//I ','METER ','DB//I ','KMETER',
     *           'DB/MET','ER    ','DB/YAR','D     ',
     *           'DB/KYA','RD    ','DB/KME','TER   ',
     *           'DB//IU','BARGIM','DB//IU','BARGIY',
     *           'DB//IU','PAMIM ','DB//IU','PABIYD',
     *           '//IUBA','R2SQIY','//IUPA','2 SQIM'/
C
C     IF UNICDE.EQ.0, THEN SET UNITS TO "DB AS MEAS.".
C
      UNITS(1)='DB AS '
      UNITS(2)='MEAS. '
C
C     SET INDICATOR FOR CASE UNPACK IS BYPASSED.
C
      ITST=0
C
C     IF RETRIEVING, UNPACK VALUE AND UNIT CODE.
C
      IF(UNICDE.GE.1) GO TO 1
      ICODE=VALUE
      IF(UNICDE.EQ.0.AND.ICODE.EQ.-99999999) RETURN
      IF(UNICDE.NE.0.AND.ICODE.EQ.-99999999) GOTO 5
      UNICDE=0
      UNICDE=IABS(FLD(29,7,VALUE))
C
C     IF WE AREN'T RETRIEVING, THEN RETURN.
C
      IF(ICODE.EQ.0.AND.UNICDE.EQ.0) RETURN
C
C     SET INDICATOR FOR CASE PACK IS BYPASSED.
C
    5 ITST=1
C
```

A-40

```
C     TEST TO SEE IF CODE IS IN TABLE.
C
   1 ICODE=IABS(UNICDE)
     DO 2 I=1,27
     IF(ICODE.EQ.ENTRY(1,I)) GO TO 3
   2 CONTINUE
C
C     IF CODE NOT FOUND, WRITE ERROR.
C
     PRINT 101,ICODE
     PRINT 102,UNICDE,VALUE,ITST
 102 FORMAT(' UNICDE=',I5,'  VALUE(OCTAL)=',O12,'  ITST=',I5)
     UNITS(1) = 'UNITS '
     UNITS(2) = 'ERROR '
     RETURN
C
C     PACK CODE AND VALUE, IF NECESSARY.
C
   3 IF(ITST.EQ.1) GO TO 4
     FLD(29,7,VALUE)=FLD(29,7,UNICDE)
C
C     SET UNITS, RETURN.
C
   4 UNITS(1)=ENTRY(2,I)
     UNITS(2)=ENTRY(3,I)
     UNITS(3)=ENTRY(2,I)
     UNITS(4)=ENTRY(3,I)
     RETURN
 101 FORMAT(' ERROR--UNIT CODE',I3,' NOT IN DEEBEE TABLE.')
     END
```

```
@FOR,S4     N,DENSTY,.DENSTY

      1.          SUBROUTINE DENSTY(VALUE,UNICDE,UNITS)
      2.    C
      3.    C     TABLE OF DENSITY MEASURE CONVERSION FACTORS.
      4.    C
      5.          INTEGER ENTRY(3,3),UNITS(9),UNICDE
      6.    C
      7.    C     FACTORS ARE DOUBLE PRECISION.
      8.    C
      9.          DOUBLE PRECISION FACTOR(1)
     10.    C
     11.    C     PUT UNIT CODES INTO ENTRY.
     12.    C
     13.          DATA (ENTRY(1,I),I=1,3)/96,96,97/
     14.    C
     15.    C     PUT IN ALPHA UNITS.
     16.    C
     17.          DATA ((ENTRY(I,J),I=2,3),J=1,3)/'GMS/CM','**3   ',
     18.         *     'KG/MET','ER**3 ','LB/FT**','3    '/
     19.    C
     20.    C     ENTER INTERNAL UNITS.
     21.    C
     22.          UNITS(3)='GMS/CM'
     23.          UNITS(9)='**3    '
     24.    C
     25.    C     IF UNICDE=0, THEN SET CODE TO STANDARD UNITS.
     26.    C
     27.          IF(UNICDE.EQ.0) UNICDE=-90
     28.    C
     29.    C     SET CONVERSION FACTORS.
     30.    C
     31.          DATA (FACTOR(I),I=1,3)/1.000,1.003,7.76922304/
     32.    C
     33.    C     INITIALIZE 'UNITS' TO ERROR MESSAGE IN CASE CODE NOT FOUND.
     34.          UNITS(1)='UNITS '
     35.          UNITS(2)='ERROR '
     36.    C
     37.    C     IF CODE IS IN TABLE, PERFORM CONVERSION. ALSO, ENTER UNITS ALPHA
     38.    C          CODE INTO 'UNITS'.
     39.    C
     40.          ICODE=IABS(UNICDE)
     41.          DO 2 I=1,3
     42.          IF(ICODE.NE.ENTRY(1,I)) GO TO 2
     43.          IF(UNICDE.GE.0) VALUE=VALUE*FACTOR(I)
     44.          IF(UNICDE.LT.0) VALUE=VALUE/FACTOR(I)
     45.          UNITS(1)=ENTRY(2,I)
     46.          UNITS(2)=ENTRY(3,I)
     47.    C
     48.    C     AT THIS POINT, CONVERSION IS COMPLETE.
     49.    C
     50.          RETURN
     51.    C
     52.    C     ELSE, CHECK REST OF TABLE.
     53.    C
     54.        2 CONTINUE
```

```
C
C     IF CODE NOT FOUND, RETURN WITH 'UNITS ERROR' IN 'UNITS'.
C
      PRINT 101, ICODE
  101 FORMAT(' ERROR--UNIT CODE',I3,' NOT IN DENSITY TABLE.')
      RETURN
      END
```

```
@FTN,SA      N.DESCTB,.DESCTB

      SUBROUTINE DESCTB (ITEMS,NITEMS,ISET)

C  PROGRAM TO PRINT THE NUMBER OF RUNS (IF NONZERO) OF THE ACOUSTIC
C  OR ENVIRONMENTAL DESCRIPTORS (ISET=0 FOR ACOUSTIC, ISET=7 FOR
C  ENVIRONMENTAL).

      DIMENSION ITEMS(1),NAME(10)
      WRITE (6,10)
10    FORMAT (T43,'NUMBER'/'   CODE',T12,'DESCRIPTOR NAME',T43,'OF RUNS'
     1)
      DO 30 I = 1,NITEMS
      IF (ITEMS(I).EQ.0) GO TO 30
      CALL NAMEIT (ISET,I,NAME,NR)
      IF (NR.GT.5) NR = 5
      WRITE (6,20) ITEMS(I),I,(NAME(J),J=1,NR)
20    FORMAT (T42,I6,T49,I3,'  -  ',5A6)
30    CONTINUE
      RETURN
      END
```

```
@ELT,L      N.DSKRED,.DSKRED

      1.            SUBROUTINE DSKRED(IN,IS,ADR)
      2.      C
      3.      C
      4.            CALL SETADR(10,IN)
      5.            IW = IS * 28
      6.            CALL NTRAN(10,2,IW,ADR,L)
      7.            CALL NTRAN(10,22)
      8.      C
      9.            IF (L .GT. 0) RETURN
     10.      C
     11.            WRITE (6,10) IN,IS,L
     12.         10 FORMAT(//,' ERROR IN DSKRED',3I10)
     13.            RETURN 0
     14.            END
```

```
@ELT,L      N.EAST,.EAST

      INTEGER FUNCTION EAST (A,B)
C
C
      INTEGER A,B
C
      IF ((ABS(A-B) .GT. 180000) GO TO 10
      EAST = MAX(A,B)
      RETURN
C
   10 EAST = B
      IF (A .LE. 0) EAST = A
      RETURN
      END
```

```
@FOR,SB     N.ENERGY,.ENERGY

      1.          SUBROUTINE ENERGY(VALUE,UNICDE,UNITS)
      2.    C
      3.    C     TABLE OF ENERGY MEASURE CONVERSION FACTORS.
      4.    C
      5.          INTEGER ENTRY(3,5),UNITS(4),UNICDE
      6.    C
      7.    C     FACTORS ARE DOUBLE PRECISION.
      8.    C
      9.          DOUBLE PRECISION FACTOR(5)
     10.    C
     11.    C     PUT UNIT CODES INTO ENTRY.
     12.    C
     13.          DATA ((ENTRY(I,J),I=1,5),J=1,3)/61,62,63,88,89/
     14.    C
     15.    C     PUT IN ALPHA UNITS.
     16.    C
     17.          DATA ((ENTRY(I,J),I=2,3),J=1,5)/'ERGS  ','      ',
     18.         *              'JOULES','      ','FT.-LB','S.    ',
     19.         *              'KGM ME','TERS  ','6MBTU','S ,'      '/
     20.    C
     21.    C     ENTER INTERNAL UNITS.
     22.    C
     23.          UNITS(3)='KGM ME'
     24.          UNITS(4)='TERS  '
     25.    C
     26.    C     IF UNICDE=0, THEN SET CODE TO STANDARD UNITS.
     27.    C
     28.          IF(UNICDE.EQ.0) UNICDE=-88
     29.    C
     30.    C     SET CONVERSION FACTORS.
     31.    C
     32.          DATA (FACTOR(I),I=1,5)/1.01970-8,.10197D0,.38255500,100,10^7,5D0/
     33.    C
     34.    C     INITIALIZE 'UNITS' TO ERROR MESSAGE IN CASE CODE NOT FOUND.
     35.    C
     36.          UNITS(1)='UNITS '
     37.          UNITS(2)='ERROR '
     38.    C
     39.    C     IF CODE IS IN TABLE, PERFORM CONVERSION. ALSO, ENTER UNITS ALPHA
     40.    C          CODE INTO 'UNITS'.
     41.    C
     42.          ICODE=IABS(UNICDE)
     43.          DO 2 I=1,5
     44.          IF(ICODE.NE.ENTRY(1,I)) GO TO 2
     45.          IF(UNICDE.GE.0) VALUE=VALUE*FACTOR(I)
     46.          IF(UNICDE.LT.0) VALUE=VALUE/FACTOR(I)
     47.          UNITS(1)=ENTRY(2,I)
     48.          UNITS(2)=ENTRY(3,I)
     49.    C
     50.    C     AT THIS POINT, CONVERSION IS COMPLETE.
     51.    C
     52.          RETURN
     53.    C
     54.    C     ELSE, CHECK REST OF TABLE.
```

```
      C
    2 CONTINUE
      C
      C     IF CODE NOT FOUND, RETURN WITH 'UNITS ERROR' IN 'UNITS'.
      C
          PRINT 101,ICODE
  101 FORMAT(' ERROR--UNIT CODE',I3,' NOT IN ENERGY TABLE.')
          RETURN
          END
```

```
@ELT,L      N,FREEFD,,FREEFD

      1.            SUBROUTINE FREEFD (CHAR,WORD,IWORD,NWORDS,MAX,$)
      2.
      3.      C  THIS PROGRAM ACCEPTS A STRING OF 80 CHARACTERS CONTAINING A VARIABLE NUMBER
      4.      C  OF DATA VALUES INPUTTED IN FREE-FIELD FORMAT.  THE VALUES CAN BE REAL,
      5.      C  INTEGER, OR ALPHANUMERIC.  AN ALPHANUMERIC VALUE IS A VARIABLE LENGTH
      6.      C  STRING OF ALPHANUMERIC CHARACTERS BEGINNING WITH AN ALPHABETIC CHARACTER.
      7.      C  MAX
      8.            IMPLICIT INTEGER (A-Z)
      9.            REAL WORD
     10.            DIMENSION CHAR(1),WORD(1),IWORD(1),VALUE(3)
     11.            DATA COMMA /',',/, Z/037050505050505/
     12.            NWORDS = 0
     13.            CURCHR = 1
     14.            BIT = 0
     15.            W = 1
     16.            B = 0
     17.            CHAR(81) = ' '
     18.
     19.            I = 0
     20.      4     I = I + 1
     21.            IF (I.GT.81) RETURN
     22.            IF (CHAR(I).EQ.',') GO TO 1
     23.            IF (CHAR(I).NE.' ') GO TO 7
     24.            IF (CURCHR.EQ.I) GO TO 3
     25.      1     NWORDS = NWORDS + 1
     26.      C  FLAG VALUE OF -99999999 STORED IN CASE A BLANK VALUE IS FOUND.
     27.            WORD(NWORDS) = -99999999.
     28.      C  SKIP DECODE IF BLANK VALUE.
     29.            IF (I.GT.CURCHR) DECODE (I-CURCHR,2,VALUE,ERR=6) WORD(NWORDS)
     30.      2     FORMAT ()
     31.            IWORD(NWORDS) = WORD(NWORDS)
     32.      C  CHECK IF MAX VALUES HAVE BEEN DECODED
     33.      12    IF (NWORDS.GE.MAX) GO TO 15
     34.            BIT = 0
     35.            W = 1
     36.            B = 0
     37.      3     CURCHR = I + 1
     38.            GO TO 4
     39.
     40.
     41.      C  CHECK IF ALPHABETIC AND IF FIRST CHARACTER OF STRING.
     42.      7     IF (CHAR(I).GT.Z.OR.BIT.GT.0) GO TO 8
     43.      C  LOOP TO STORE ALPHANUMERIC STRING
     44.            NWORDS = NWORDS + 1
     45.            WORD(NWORDS) = ' '
     46.      9     IF (BIT.GE.36) GO TO 11
     47.            FLD(BIT,6,WORD(NWORDS)) = FLD(0,6,CHAR(I))
     48.            BIT = BIT + 6
     49.      11    I = I + 1
     50.            IF (CHAR(I).NE.' '.AND.CHAR(I).NE.',') GO TO 9
     51.            FLD(0,36,IWORD(NWORDS)) = FLD(0,36,WORD(NWORDS))
     52.            GO TO 12
     53.      C  PACK A DATA CHARACTER INTO VALUE.
     54.      8     IF (W.GT.3) GO TO 5
```

```
      FLD(B,6,VALUE(W)) = FLD(0,6,CHAR(I))
      BIT = BIT + 6
      W = (BIT/36) + 1
      B = MOD(BIT,36)
      FLD (B,6,VALUE(W)) = FLD(0,6,COMMA)
      GO TO 4

C  RETURN TO ERROR STATEMENT IN CALLING PROGRAM.
5     RETURN 6

C  CHECK IF MORE THAN MAX VALUES HAVE BEEN INPUTTED

15    I = I + 1
      IF (I.GT.80) RETURN
      DO 20 J = I,80
20    IF (CHAR(J).NE.' ') GO TO 25
      RETURN

C  TOO MANY VALUES INPUTTED

25    WRITE (6,30) MAX
30    FORMAT (' *** MORE THAN MAXIMUM OF',I3,' VALUES INPUTTED.  EXCESS
     1VALUES IGNORED ***')
      RETURN

      END
```

```
@FOR,S4     N.GETINT,.GETINT

      SUBROUTINE GETINT
      IMPLICIT INTEGER (A-Z)
      DIMENSION IBFR(3000),DESC(56)
C
C
      INCLUDE RUN,LIST
C
      REAL WAVDIR,WAVHT,WAVPRD,SEASTA,WNDDIR,WNDVEL,SWLDIR,
     .     SWLHT,SWLPRD,WEATHR,BOTDPH,BTMSLP
C
C
      INCLUDE CPV,LIST
      DIMENSION ICON(9,30),IPAR(30,30),IVLRNG(2,30,50),IEQU(3000)
      EQUIVALENCE (CON(1,1),ICON(1,1)),(PAR(1,1),IPAR(1,1)),
     .           (VALRNG(1,1,1),IVLRNG(1,1,1)),(EQU(1))
      REAL CON,PAR,DATA,VALRNG
      INCLUDE GETLST,LIST
C
C
      INCLUDE RETREV,LIST
C
      INCLUDE DATCOM,LIST
C
      USERAP = .TRUE.
      ALL = .TRUE.
C
C
      CALL NTRAN(10,22)
      CALL NTRAN(11,10)
C
C          READ OR BYPASS RETRIEVE COMMON.
C
      IF (USERAP) GO TO 5
      CALL SETADR (11,16)
      GO TO 10
C
5     CALL NTRAN(11,2,428,NLAT,LSTAT)
C
C          NO PARALLEL PROCESSING.
C
      CALL NTRAN(11,22)
C
C          WRITE 'NTRAN ERROR' IF ANY.
C
      IF(LSTAT.GT.0) GO TO 10
      PRINT 101,LSTAT
101   FORMAT(' ERROR--GETINT NTRAN STATUS ',I2)
      STOP
C
10    ICTR=0
      LASTDA=0
      NRUNSF = 0
C
C          INITIALIZATION COMPLETE.
```

```
      C
            RETURN
      C
      C
      C
      C
      C
      C
      C           ROUTINE FOR PROCESSING EACH RUN.
      C
            ENTRY GETRUN(DATYPE,EXPNO,CRUNO,STANO,*)
      C
      C           WHEN RUN COUNTER IS .GE. NUMBER OF POINTERS, WE ARE DONE.
      C
      1     IF(ICTR.GE.NPTRS) RETURN $
      C
            ICTR=ICTR+1
      C
            CALL NTRAN(11,2,56,DESC,LSTAT)
      C
      C           NO PARALLEL PROCESSING.
      C
            CALL NTRAN(11,22)
      C
      C           WRITE NTRAN ERROR, IF ANY.
      C
            IF(LSTAT.GT.0) GO TO 2
            PRINT 102,LSTAT,ICTR
        102 FORMAT(' ERROR--GETRUN NTRAN STATUS ',I2,' ICTR = ',I4,',')
            STOP
      C
      C        CHECK EXP, CRU, AND STA NUMBER
      C
      2     EXPNO = DESC(1)
            CRUNO = DESC(2)
            STANO = DESC(3)
            IF (ALL) GO TO 3
            IF (EXPNO.NE.EXPI) GO TO 1
            IF (CRUI.EQ.-99999999) GO TO 3
            IF (CRUNO.NE.CRUI) GO TO 1
            IF (STAI.EQ.-99999999) GO TO 3
            IF (STANO.NE.STAI) GO TO 1
      C
          3 NEXTDA = FLD( 0,18,DESC(4))
            LENGTH = FLD(18,18,DESC(4))
      C
      C           SET ISUB = SUBSCRIPT OF RUN
      C
            ISUB = DESC(5)
      C
            IF (LASTDA .EQ. NEXTDA) GO TO 15
      C
            CALL DSKRED (NEXTDA,LENGTH,IBFR)
      C
      C           UPDATE LAST DISK ADDRESS PTR
      C
            LASTDA = NEXTDA
```

```
C
C          UNPACK DATA TYPE
C
      DATYPE = FLD(0,18,IBFR(1))
C
C          IF NOT ACOUSTIC OR ENV. DATA, WRITE ERROR
C
      IF (DATYPE .EQ. 3 .OR. DATYPE .EQ. 4) GO TO 15
      PRINT 103, DATYPE
  103 FORMAT (' ERROR -- GETRUN DATA TYPE',I2,' ILLEGAL.')
      STOP
C
C          CHECK RUN NUMBER
C
   15 RUNNO = FLD(0,18,IBFR(ISUB))
      IF (USERAP) GO TO 20
      IF (ALL) GO TO 18
      IF (RUN1.EQ.-99999999) GO TO 18
      IF (RUNNO.NE.RUN1) GO TO 1
   18 IF (AETYPE.EQ.3) GO TO 20
      IF (AETYPE.EQ.2) GO TO 19
      IF (DATYPE.EQ.4) GO TO 1
      GO TO 20
   19 IF (DATYPE.EQ.3) GO TO 1
C
C          FIND PTR TO DATA
C
   20 DPTR = IBFR(ISUB +9)
C
      SECAD = FLD( 0,18,DPTR)
      SLNGTH= FLD(18,18,DPTR)
C
      CALL AUNPCK (IBFR,ISUB)
C
      IF (DATYPE .EQ. 4) GO TO 30
C
      CALL DSKRED (SECAD,SLNGTH,DATA(0))     @ WATCH FOR ZERO SUBSCRIPT!
      GO TO 40
C
C
   30 CALL DSKRED (SECAD,SLNGTH,IEQU(299))
      DO 35 I=1,9
      INFO(I) = IEQU(299+I)
   35 CONTINUE
C
   40 NDUSED = DESC(6)
      DO 50 I=1,50
      NRS(I) = DESC(6+I)
   50 CONTINUE
C
C          CALL TO PERFORM UNIT CONVERSIONS
C
      CALL UNTCVT
C
C          INCREMENT NUMBER OF RUNS FOUND
C
      NRUNSF = NRUNSF + 1
```

```
C
C                    DONE
C
      RETURN
      END
```

```
@ELT,L      N,IDNAME,,IDNAME

      SUBROUTINE IDNAME (CHAR,ID,NAME,FLAG)

C        THIS ROUTINE DECODES THE USER NAME AND ID

      IMPLICIT INTEGER (A-Z)
      DIMENSION CHAR(1),NAME(1)
      DATA A/0060505050505/, Z/0370505050505/
C
C        DECODE NAME
C
      FLAG = 0
      WORD = 1
      BIT = 0
      DO 15 I = 1,5
15    NAME(I) = ' '
      I = 0
40    I = I + 1
      IF (I.GT.80) GO TO 25
      IF (CHAR(I).EQ.' ') GO TO 40
      IF (CHAR(I).LT.A.OR.CHAR(I).GT.Z) GO TO 25
      DO 50 J = 1,30
      FLD (BIT,6,NAME(WORD)) = FLD (0,6,CHAR(I))
      BIT = BIT + 6
      I = I + 1
      IF (CHAR(I).EQ.',') GO TO 5
      IF (BIT.LT.36) GO TO 50
      BIT = 0
      WORD = WORD + 1
50    CONTINUE
      GO TO 5
C
C     ERROR
C
25    WRITE (6,30)
30    FORMAT (' *** ERROR ENTERING USER NAME OR ID ***')
      FLAG = 1
      RETURN
C
C        DECODE ID
C
5     BIT = 0
10    I = I + 1
      IF (I.GT.80) GO TO 25
      IF (CHAR(I).EQ.' ') GO TO 10
      ID = ' '
      DO 20 J = 1,6
      FLD (BIT,6,ID) = FLD (0,6,CHAR(I))
      BIT = BIT + 6
      I = I + 1
      IF (CHAR(I).EQ.' ') RETURN
20    CONTINUE
      RETURN
      END
```

```
@ELT,L      N.INPUT,.INPUT

      SUBROUTINE INPUT (FIRST)
C
C         INPUT IS THE PROGRAM FOR READING IN THE RETRIEVAL REQUEST
C         SELECTION CRITERIA FOR THE NAVDAB RETRIEVAL SYSTEM. THE
C         DATA IS READ INTO COMMON /RETREV/. THIS INPUT PROGNAM IS
C         FOR VERSION 1 MOD 1 OF NAVDAB.
C                                                    8/1/74  GFA
C
C
      IMPLICIT INTEGER (A-Z)
C
      LOGICAL FIRST,BACKUP,CLSREQ
C
      REAL AEQUIV(4,10),EEQUIV(4,10),WORD(20)
C
      INCLUDE RETREV,LIST
C
      DIMENSION TEMP(15),UNITS(4),CHAR(81),MSQRS(2),NAME(10),
     1WORD(20),YEARS(2),DAYS(2),HOURS(2),ECSR(4),ERRCK(4)/999,99,
     299,999/
C
      EQUIVALENCE (AEQUIV,ACPARS),(EEQUIV,ENPARS),(MSQRS(1),FMSQR),
     1(MSQRS(2),TMSQR),(YEARS(1),FMYR),(YEARS(2),TOYR),(DAYS(1),FMDAY),
     2(DAYS(2),TODAY),(HOURS(1),FMTIM),(HOURS(2),TOTIM),(ECSR,EXP)
C
      DATA R /'R'/
C
      DATA W /'W'/
C
      BACKUP = .FALSE.
110   CALL CLASS (NOSEX,SEX,MASK,REQUEST,TNAME,TNBR,FIRST,BACKUP,CLSREQ)
      BACKUP = .TRUE.
C
C           ENTER RETRIEVAL MODE
C
9000  WRITE (6,9010)
9010  FORMAT ('0ENTER DESIRED DATA TYPE:'/4X,'ACOUSTIC ONLY(''A''), ENVI
     1RONMENTAL ONLY(''E''), OR BOTH(''B'')')
      READ (5,52,END=110,ERR=9030) MODE
      IF (MODE.EQ.'B'.OR.MODE.EQ.' ') GO TO 9040
      IF (MODE.EQ.'E') GO TO 9020
      IF (MODE.NE.'A') GO TO 9035
      RUNTYP = 1
      WRITE (6,9015)
9015  FORMAT (5X,'ACOUSTIC ONLY')
      NENDCR = 0
      NENPAR = 0
      NEMTHD = 0
      GO TO 6
9020  RUNTYP = 2
      WRITE (6,9025)
9025  FORMAT (5X,'ENVIRONMENTAL ONLY')
      NACDCR = 0
      NACPAR = 0
```

```
      NAMTHD = 0
      NSTYPE = 2
      NRTYPE = 2
      NSSYST = 0
      NRSYST = 0
      GO TO 6
 9040 RUNTYP = 3
      WRITE (6,9045)
 9045 FORMAT (5X,'ACOUSTIC AND ENVIRONMENTAL')
      GO TO 6
C
C          ERROR
C
 9030 WRITE (6,56)
      GO TO 9000
 9035 WRITE (6,9036)
 9036 FORMAT (' *** INVALID DATA TYPE SPECIFIED ***')
      GO TO 9000
C
C          RETRIEVAL BY EXPERIMENT
C
 6    PRINT 7
 7    FORMAT (/' ENTER EXP. NO.,CRUISE NO.,STATION NO.,RUN NO. RESTRICT
     1ION / ENTER ALL')
      READ (5,70,END=9000,ERR=8) (CHAR(I),I=1,80)
      WRITE (6,12) (CHAR(I),I=1,80)
      EXP = -99999999
      CRU = -99999999
      STA = -99999999
      RUN = -99999999
      CALL FREEFD (CHAR,WORD,EXP,NWORDS,4,88)
      IF (NWORDS.GT.0.AND.CHAR(1).NE.'A') GO TO 3
      EXP = -99999999
      GO TO 4930
C
C               CHECK FOR ERROR
C
 3    DO 9 I = 1,NWORDS
      IF (ECSR(I).LT.1.OR.ECSR(I).GT.ERRCK(I)) GO TO 11
 9    CONTINUE
      GO TO 4910
 8    PRINT 56
      GO TO 6
 11   WRITE (6,78) ECSR(I)
      GO TO 6
C
C                 ANYTHING ELSE TO BE ENTERED?
C
 4910 WRITE (6,4920)
 4920 FORMAT (' HAVE ALL SPECIFICATIONS BEEN ENTERED?')
      READ (5,70,END=6,ERR=4921) (CHAR(I),I=1,80)
      WRITE (6,12) (CHAR(I),I=1,80)
      IF (CHAR(1).EQ.'Y') GO TO 4925
      IF (CHAR(1).EQ.' '.OR.CHAR(1).EQ.'N') GO TO 4930
C
C          ERROR
C
```

```
      WRITE (6,4)
    4 FORMAT (' *** INVALID RESPONSE TO YES OR NO QUESTION ***')
      GO TO 4911
 4921 WRITE (6,56)
      GO TO 4910

 4925 ANY = 0
      NLAT = 0
      ELON = 0
      SLAT = 0
      WLON = 0
      FMYR = 0
      TOYR = 0
      NMON = 0
      FMDAY = 0
      TODAY = 0
      FMTIM = 0
      TOTIM = 0
      NACDCR = 0
      NACPAR = 0
      NAMTHD = 0
      NENDCR = 0
      NENPAR = 0
      NEMTHD = 0
      NSTYPE = 0
      NRTYPE = 0
      NSSYST = 0
      NRSYST = 0
      GO TO 900
C
C               ENTER AREA
C
 4930 ANY = 1
    5 PRINT 10
   10 FORMAT (/,' ENTER WHOLE WORLD(''W''), RECTANGULAR AREA(''R''), OR
     1MSQR AREA(XXX:XXX)')
      READ (5,70,END=6,ERR=43) (CHAR(I),I=1,80)
      WRITE (6,12) (CHAR(I),I=1,80)
   12 FORMAT (3X,80A1)
      CALL FREEFD (CHAR,WORD,MSQRS,NMSQRS,2,$43)
      IF (NMSQRS.LT.1) GO TO 43
      IF (R.EQ.MSQRS(1)) GO TO 40      @ RECTANGULAR AREA
      IF (W.EQ.MSQRS(1)) GO TO 30      @ WHOLE WORLD
C
C               MSQR REQUEST
C
      IF (NMSQRS.EQ.1) TMSQR = FMSQR
C
      IF (FMSQR .GT.   0 .AND. FMSQR .LT. 289) GO TO 16
      IF (FMSQR .GT. 299 .AND. FMSQR .LT. 624) GO TO 16
      IF (FMSQR .GT. 900 .AND. FMSQR .LT. 937) GO TO 16
      GO TO 17
C
   16 IF (TMSQR .GT.   0 .AND. TMSQR .LT. 289) GO TO 25
      IF (TMSQR .GT. 299 .AND. TMSQR .LT. 624) GO TO 25
      IF (TMSQR .GT. 900 .AND. TMSQR .LT. 937) GO TO 25
C
```

```
C                ERROR
C
17    PRINT 19,FMSQR,TMSQR
   19 FORMAT (' *** ERROR-- MSQR ',J3,' OR ',J3,' IS ILLEGAL ***')
      GO TO 5
C
C                CONVERT TO LONG & LAT
C
   25 CALL MSQCVT (FMSQR,TMSQR,NLAT,ELON,SLAT,WLON)
C
C                GO TO NEXT INPUT ITEM
C
      GO TO 50
C
C                WHOLE WORLD
C
30    NLAT = 0
      ELON = 0
      SLAT = 0
      WLON = 0
      GO TO 50
C
C                RECTANGULAR AREA
C
C                   NORTHERN BOUNDARY
C
40    WRITE (6,41)
41    FORMAT (' UPPER BOUNDARY:')
      NLATNS = ' '
      READ (5,70,END=5,ERR=46) (CHAR(I),I=1,80)
      CALL FREEFD (CHAR,WORD,IWORD,NWURDS,3,$46)
      IF (NWORDS.EQ.0) GO TO 46
      IF (IWORD(1).EQ.-99999999) IWORD(1) = 0
      NLATD = IWORD(1)
      IF (NWORDS.EQ.3) GO TO 42
      NLATM = 0
      FLD(0,6,NLATNS) = FLD(0,6,IWORD(2))
      GO TO 5000
42    IF (WORD(2).EQ.-99999999.) WORD(2) = 0.
      NLATM = WORD(2) * 10.
      FLD(0,6,NLATNS) = FLD(0,6,IWORD(3))
C
C                CHECK FOR ERROR
C
5000  IF (NLATD .LT. 0 .OR. NLATD .GT. 90) GO TO 46
      IF (NLATM .LT. 0 .OR. NLATM .GT.599) GO TO 46
      IF (NLATNS .NE. 'N' .AND. NLATNS .NE. 'S') GO TO 46
      NLAT = NLATD * 1000 + NLATM
      IF (NLATNS .EQ. 'S') NLAT = -NLAT
C
C                   SOUTHERN BOUNDARY
C
59    WRITE (6,60)
60    FORMAT (' LOWER BOUNDARY:')
      SLATNS = ' '
      READ (5,70,END=40,ERR=47) (CHAR(I),I=1,80)
      CALL FREEFD (CHAR,WORD,IWORD,NWURDS,3,$47)
```

```
      IF (N#ORDS.EQ.0) GO TO 47
      IF (I#ORD(1).EQ.-99999999) I#ORD(1) = 0
      SLATD = I#ORD(1)
      IF (N#ORDS.EQ.3) GO TO 61
      SLATM = 0
      FLD(0,6,SLATNS) = FLD(0,6,I#ORD(2))
      GO TO 5100
61    IF (W#ORD(2).EQ.-99999999.) W#ORD(2) = 0.
      SLATM = W#ORD(2) * 10.
      FLD(0,6,SLATNS) = FLD(0,6,I#ORD(3))
C
C                 CHECK FOR ERROR
C
5100  IF (SLATD .LT. 0 .OR. SLATD .GT. 90) GO TO 47
      IF (SLATM .LT. 0 .OR. SLATM .GT.599) GO TO 47
      IF (SLATNS .NE. 'N' .AND. SLATNS .NE. 'S') GO TO 47
      SLAT = SLATD * 1000 + SLATM
      IF (SLATNS .EQ. 'S') SLAT = -SLAT
      IF (SLAT.GT.NLAT) GO TO 46
C
C                     WESTERN BOUNDARY
C
65    WRITE (6,66)
66    FORMAT (' LEFT  BOUNDARY:')
      W1ONE# = ' '
      READ (5,70,END=59,ERR=49) (CHAR(I),I=1,80)
      CALL FREEFD (CHAR,W#ORD,I#ORD,NW#ORDS,3,$49)
      IF (N#ORDS.EQ.0) GO TO 49
      IF (I#ORD(1).EQ.-99999999) I#ORD(1) = 0
      W1OND = I#ORD(1)
      IF (N#ORDS.EQ.3) GO TO 67
      W1ONM = 0
      FLD(0,6,W1ONE#) = FLD(0,6,I#ORD(2))
      GO TO 5300
67    IF (W#ORD(2).EQ.-99999999.) W#ORD(2) = 0.
      W1ONM = W#ORD(2) * 10.
      FLD(0,6,W1ONE#) = FLD(0,6,I#ORD(3))
C
C                 CHECK FOR ERROR
C
5300  IF (W1OND .LT. 0 .OR. W1OND .GT.180) GO TO 49
      IF (W1ONM .LT. 0 .OR. W1ONM .GT.599) GO TO 49
      IF (W1ONE# .NE. 'E' .AND. W1ONE# .NE. 'W') GO TO 49
      W1ON = W1OND * 1000 + W1ONM
      IF (W1ONE# .EQ. 'W') W1ON = -W1ON
C
C                     EASTERN BOUNDARY
C
62    WRITE (6,63)
63    FORMAT (' RIGHT BOUNDARY:')
      E1ONE# = ' '
      READ (5,70,END=65,ERR=48) (CHAR(I),I=1,80)
      CALL FREEFD (CHAR,W#ORD,I#ORD,NW#ORDS,3,$48)
      IF (N#ORDS.EQ.0) GO TO 48
      IF (I#ORD(1).EQ.-99999999) I#ORD(1) = 0
      E1OND = I#ORD(1)
      IF (N#ORDS.EQ.3) GO TO 64
```

```
      ELONM = 0
      FLD(3,6,ELONE#) = FLD(0,6,I#ORD(2))
      GO TO 5200
44    IF (I#ORD(2).EQ.-99999999.) #ORD(2) = 0.
      ELONM = #ORD(2) * 10.
      FLD(0,6,ELONE#) = FLD(0,6,I#ORD(3))
C
C
C               CHECK FOR ERROR
C
5200  IF (ELOND .LT. 0 .OR. ELOND .GT.180) GO TO 48
      IF (ELONM .LT. 0 .OR. ELONM .GT.599) GO TO 48
      IF (ELONE# .NE. 'E' .AND. ELONE# .NE. 'W') GO TO 48
      ELON = ELOND * 1000 + ELONM
      IF (ELONE# .EQ. 'W') ELON = -ELON
C
C               PRINT OUT
C
      T1 = NLATM /10
      T2 = NLATM - T1 *10
      T3 = ELONM /10
      T4 = ELONM - T3 *10
      T5 = SLATM /10
      T6 = SLATM - T5 *10
      T7 = #LONM /10
      T8 = #LONM - T7 *10
C
      PRINT 45,NLATD,T1,T2,NLATNS,SLATD,T5,T6,SLATNS,
     .          #LOND,T7,T8,#LONE#,ELOND,T3,T4,ELONE#
   45 FORMAT (1X,I3,1X,2I2,'°',I1,A1,'',I4,1X,2I2,'°',I1,A1,'',
     .          I4,1X,2I2,'°',I1,A1,'',I4,1X,2I2,'°',I1,A1)
C
C               GO TO NEXT INPUT ITEM
C
      GO TO 50
C
C               #RITE ERROR
C
43    #RITE (6,56)
      GO TO 5
46    PRINT 68
68    FORMAT (' *** ERROR IN RECTANGULAR BOUNDARY SPECIFICATION ***')
      GO TO 40
47    PRINT 68
      GO TO 59
48    PRINT 68
      GO TO 62
49    PRINT 68
      GO TO 65
C
C
C          CHECK FOR ZERO BOUNDARIES
C
   50 PRINT 51
   51 FORMAT (/' ARE YEAR, MONTH, DAY, AND TIME BOUNDARIES',
     .' TO BE CONSIDERED?')
      READ (5,52,END=5,ERR=57) BOUNDS
```

```
      52 FORMAT (A1)
         IF (BOUNDS.EQ.'Y') GO TO 53
         IF (BOUNDS.EQ.' '.OR.BOUNDS.EQ.'N') GO TO 99
C
C                ERROR
C
         WRITE (6,4)
         GO TO 50
   57    WRITE (6,56)
         GO TO 50
C
C             READ IN YEAR BOUNDARIES
C
      53 PRINT 54
      54 FORMAT (' ENTER YEAR BOUNDARIES')
         READ (5,70,END=50,ERR=55) (CHAR(I),I=1,80)
         FMYR = 0
         TOYR = 0
         CALL FREEFD (CHAR,WORD,YEARS,NWORDS,2,$55)
         IF (NWORDS.EQ.0) GO TO 72
         IF (NWORDS.EQ.1) TOYR = FMYR
   15    FORMAT ()
C
C                CHECK FOR ERROR
C
         IF (FMYR .LT. 0 .OR.  FMYR .GT. 99) GO TO 39
         IF (TOYR .LT. 0 .OR.  TOYR .GT. 99) GO TO 58
         IF (TOYR.LT.FMYR) GO TO 38
         PRINT 15,FMYR,TOYR
         GO TO 72
C
C                ERROR
C
      55 PRINT 56
      56 FORMAT (' *** READ FORMAT ERROR ***')
         GO TO 53
   39    WRITE (6,78) FMYR
         GO TO 53
   58    WRITE (6,78) TOYR
         GO TO 53
   38    WRITE (6,89)
         GO TO 53
C
C
C
C                READ IN MONTHS
C
      72 PRINT 71
      71 FORMAT (' ENTER MONTH NUMBERS')
         READ (5,70,ERR=75,END=53) (CHAR(I),I=1,80)
   70    FORMAT (80A1)
         CALL FREEFD (CHAR,WORD,MONTHS,NMON,12,$75)
         IF (NMON.EQ.0) GO TO 80
C
C                CHECK FOR ERROR
C
   76    DO 73 I=1,NMON
```

```
      IF (MONTHS(I) .LT. 1 .OR. MONTHS(I) .GT. 12) GO TO 77
   73 CONTINUE
C
C                PRINT
C
      PRINT 74,(MONTHS(I),I=1,NMON)
   74 FORMAT (1X,14,10(',',I4))
      GO TO 80
C
C                ERROR
C
   75 PRINT 56
      GO TO 72
   77 WRITE (6,78) MONTHS(I)
   78 FORMAT (' *** ENTERED NUMBER',I5,' OUT OF RANGE ***')
      GO TO 72
C
C
C          READ IN DAY BOUNDARIES
C
   80 PRINT 81
   81 FORMAT (' ENTER DAY BOUNDARIES')
      READ (5,70,END=72,ERR=85) (CHAR(I),I=1,80)
      FMDAY = 0
      TODAY = 0
      CALL FREEFD (CHAR,WORD,DAYS,NWORDS,2,$85)
      IF (NWORDS.EQ.0) GO TO 90
      IF (NWORDS.EQ.1) TODAY = FMDAY
C
C                CHECK FOR ERROR
C
      IF (FMDAY .EQ. 0 .AND. TODAY .EQ. 0) GO TO 83
      IF (FMDAY .LT. 1 .OR. FMDAY .GT. 31) GO TO 86
      IF (TODAY .LT. 1 .OR. TODAY .GT. 31) GO TO 87
      IF (TODAY .LT. FMDAY) GO TO 88
C
C                PRINT
C
   83 PRINT 15,FMDAY,TODAY
      GO TO 90
C
C                ERROR
C
   85 PRINT 56
      GO TO 80
   86 WRITE (6,78) FMDAY
      GO TO 80
   87 WRITE (6,78) TODAY
      GO TO 80
   88 WRITE (6,89)
   89 FORMAT (' *** INVALID RANGE SPECIFIED ***')
      GO TO 80
C
C
C          READ IN TIME BOUNDARIES
C
   90 PRINT 91
```

```
      91 FORMAT (' ENTER TIME BOUNDARIES')
         READ (5,70,END=80,ERR=95) (CHAR(I),I=1,80)
         FMTIM = 0
         TOTIM = 0
         CALL FREEFD (CHAR,WORD,HOURS,NWORDS,2,595)
         IF (NWORDS.EQ.0) GO TO 120
         IF (NWORDS.EQ.1) TOTIM = FMTIM
C
C              CHECK FOR ERROR
C
         IF (FMTIM .LT. 0 .OR. FMTIM .GT. 2359) GO TO 96
         IF (TOTIM .LT. 0 .OR. TOTIM .GT. 2359) GO TO 97
C
C              PRINT
C
         PRINT 15,FMTIM,TOTIM
         GO TO 120
C
C              ERROR
C
      95 PRINT 56
         GO TO 90
   96    WRITE (6,78) FMTIM
         GO TO 90
   97    WRITE (6,78) TOTIM
         GO TO 90
C
C          BOUNDARIES EQUAL ZERO
C
   99    FMYR = 0
         TOYR = 0
         NMON = 0
         FMDAY = 0
         TODAY = 0
         FMTIM = 0
         TOTIM = 0
         PRINT 103
  103 FORMAT (' YEAR,MONTH,DAY,AND TIME BOUNDARIES WILL BE IGNORED',)
C
C
C          PROCESS ACOUSTIC REQUEST UNLESS ENV. ONLY
C
  120    IF (RUNTYP.EQ.2) GO TO 212
         PRINT 121
  121    FORMAT ('0      ACOUSTIC REQUEST(S)')
C
C          BYPASS SOURCE/RECEIVER QUESTIONS
C
 1110    WRITE (6,1120)
 1120    FORMAT ('0DO YOU WANT TO ENTER TRANSDUCER TYPE(S) OR SONAR SYSTEM(
        1S)?')
         READ (5,1130,END=1150,ERR=1150) ANSWER
 1130    FORMAT (A3)
         WRITE (6,1140)
 1140    FORMAT (3A,A3)
         W = ' '
         FLD (0,6,W) = FLD (0,6,ANSWER)
```

```
      IF (A.EQ.'Y') GO TO 1151
      NSTYPE = 0
      NRTYPE = 0
      NSSYST = 0
      NRSYST = 0
      IF (W.EQ.' '.OR.W.EQ.'N') GO TO 135
C
C        ERROR
C
1150  WRITE (6,4)
      GO TO 1110
C
C              READ IN SOURCE TYPES
C
1151  PRINT 1152
1152  FORMAT ('0ENTER SOURCE TYPE(S)')
      READ (5,70,END=1110,ERR=1155) (CHAR(I),I=1,80)
      CALL FREEFD (CHAR,WORD,STYPE,NSTYPE,10,$1155)
      IF (NSTYPE.EQ.0) GO TO 1161
C
C              CHECK FOR ERROR
C
1154  DO 1153 I = 1,NSTYPE
      IF (STYPE(I).LT.1.OR.STYPE(I).GT.99) GO TO 1157
1153  CONTINUE
C
C              PRINT
C
      DO 1156 I = 1,NSTYPE
      CALL NAMEJT (5,STYPE(I),NAME,NR)
1156  WRITE (6,139) STYPE(I),(NAME(J),J=1,NR)
      GO TO 1161
C
C              ERROR
C
1155  PRINT 56
      GO TO 1151
1157  WRITE (6,78) STYPE(I)
      GO TO 1151
C
C        BACKUP
C
1158  IF (BOUNDS.EQ.'Y') GO TO 90
      GO TO 50
C
C              READ IN RECEIVER TYPES
C
1161  PRINT 1162
1162  FORMAT('0ENTER RECEIVER TYPE(S)')
      READ (5,70,END=1151,ERR=1165) (CHAR(I),I=1,80)
      CALL FREEFD (CHAR,WORD,RTYPE,NRTYPE,10,$1165)
      IF (NRTYPE.EQ.0) GO TO 1171
C
C              CHECK FOR ERROR
C
1164  DO 1163 I = 1,NRTYPE
      IF (RTYPE(I).LT.1.OR.RTYPE(I).GT.99) GO TO 1167
```

```
1163  CONTINUE
C
C                PRINT
C
      DO 1166 I = 1,NRTYPE
      CALL NAMEIT (5,RTYPE(I),NAME,NR)
1166  WRITE (6,139) RTYPE(I),(NAME(J),J=1,NR)
      GO TO 1171
C
C                ERROR
C
1165  PRINT 56
      GO TO 1161
1167  WRITE (6,78) RTYPE(I)
      GO TO 1161
C
C                READ IN SOURCE SYSTEMS
C
1171  PRINT 1172
1172  FORMAT('0ENTER SOURCE SYSTEM(S)')
      READ (5,70,END=1161,ERR=1175) (CHAR(I),I=1,80)
      CALL FREEFD (CHAR,WORD,SSYST,NSSYST,10,$1175)
      IF (NSSYST.EQ.0) GO TO 1181
C
C                CHECK FOR ERROR
C
      DO 1173 I = 1,NSSYST
      IF (SSYST(I).LT.1.OR.SSYST(I).GT.99) GO TO 1177
1173  CONTINUE
C
C                PRINT
C
      DO 1176 I = 1,NSSYST
      CALL NAMEIT (4,SSYST(I),NAME,NR)
1176  WRITE (6,139) SSYST(I),(NAME(J),J=1,NR)
      GO TO 1181
C
C                ERROR
C
1175  PRINT 56
      GO TO 1171
1177  WRITE (6,78) SSYST(I)
      GO TO 1171
C
C                READ IN RECEIVER SYSTEMS
C
1181  PRINT 1182
1182  FORMAT ('0ENTER RECEIVER SYSTEM(S)')
      READ (5,70,END=1171,ERR=1185) (CHAR(I),I=1,80)
      CALL FREEFD (CHAR,WORD,RSYST,NRSYST,10,$1185)
      IF (NRSYST.EQ.0) GO TO 135
C
C                CHECK FOR ERROR
C
1184  DO 1183 I = 1,NRSYST
      IF (RSYST(I).LT.1.OR.RSYST(I).GT.99) GO TO 1187
1183  CONTINUE
```

```
      C
      C                PRINT
      C
            DO 1186 I = 1,NRSYST
            CALL NAMEIT (9,RSYST(I),NAME,NN)
      1186  #RITE (6,139) RSYST(I),(NAME(J),J=1,NR)
            GO TO 135
      C
      C                ERROR
      C
      1185  PRINT 56
            GO TO 1181
      1187  #RITE (6,78) RSYST(I)
            GO TO 1181
      C
      C                READ IN ACOUSTIC DESCRIPTORS
      C
      C
      135   PRINT 136
      136   FORMAT ('CENTER ACOUSTIC DESCRIPTOR(S)')
            READ (5,70,ERR=140,END=145) (CHAR(I),I=1,80)
            CALL FREEFD (CHAR,#ORD,ACDCRS,NACDCR,10,$140)
            IF (NACDCR.EQ.0) GO TO 140
      C
      C                    CHECK FOR ERROR
      C
      138   DO 137 I=1,NACDCR
            IF (ACDCRS(I) .LT. 1 .OR. ACDCRS(I) .GT. 99) GO TO 142 @ TABLE LIMIT 99
        137 CONTINUE
      C
      C                    PRINT
      C
            DO 141 I = 1,NACDCR
            CALL NAMEIT (8,ACDCRS(I),NAME,NN)
            #RITE (6,139) ACDCRS(I),(NAME(J),J=1,NR)
      139   FORMAT (3X,I4,' - ',10A6)
      141   CONTINUE
            GO TO 160
      C
      C                    ERROR
      C
        140 PRINT 56
            GO TO 135
      142   #RITE (6,78) ACDCRS(I)
            GO TO 135
      C
      C        BACKUP
      C
      145   IF (#.EQ.'Y') GO TO 1181
            GO TO 1110
      C
      C
      C                READ ACOUSTIC PARAMETERS
      C
      C
      160   PRINT 161
      161   FORMAT ('CENTER ACOUSTIC PARAMETERS (NAME,MIN,MAX,UNIT,METHODS) /
```

```
682.            (ENTER ''END''')
683.      C
684.            NACPAR = 0
685.      C                       LOOP TO READ PARAMETERS
686.      C
687.      165   NACPAR = NACPAR + 1
688.      167   PRINT 170,NACPAR
689.      170   FORMAT ('0PARAMETER ',I2)
690.            READ (5,70,END=177,ERR=175) (CHAR(I),I=1,80)
691.            IF (CHAR(1).EQ.'E') GO TO 180
692.      C
693.      C                 CHECK FOR PARAMETER ARRAY LIMIT
694.      C
695.            IF (NACPAR.LE.10) GO TO 179
696.            WRITE (6,173)
697.      173   FORMAT (' *** INPUT RESTRICTED TO 10 ACOUSTIC PARAMETERS ***')
698.            GO TO 181
699.      C
700.      C                 STORE MIN, MAX, AND UNIT VALUE IN CASE NOT SPECIFIED
701.      C
702.      179   WORD(2) = -99999999.
703.            WORD(3) = -99999999.
704.            IWORD(4) = 0
705.            CALL FREEFD (CHAR,WORD,IWORD,NWORDS,14,$175)
706.            IF (NWORDS.EQ.0) GO TO 175
707.            ACPARS(1,NACPAR) = IWORD(1)
708.            AEQUIV(2,NACPAR) = WORD(2)
709.            AEQUIV(3,NACPAR) = WORD(3)
710.            IF (IWORD(4).EQ.-99999999) IWORD(4) = 0
711.            ACPARS(4,NACPAR) = IWORD(4)
712.            NAMTHD(NACPAR) = NWORDS - 4
713.            IF (NAMTHD(NACPAR).LT.0) NAMTHD(NACPAR) = 0
714.      C
715.      C
716.      C                       CHECK FOR ERRORS
717.      C
718.            IF (ACPARS(1,NACPAR).LT.1.OR.ACPARS(1,NACPAR).GT.72) GO TO 179
719.            DUM = 0
720.            CALL CONVRT (DUM,ACPARS(1,NACPAR),-ACPARS(4,NACPAR),UNITS)
721.            IF (UNITS(1) .EQ. 'UNITS ') GO TO 182
722.      C
723.      C                 PRINT PARAMETER INFORMATION
724.      C
725.            CALL NAMGET (ACPARS(1,NACPAR),TEMP)
726.            IF (AEQUIV(2,NACPAR).EQ.-99999999.) GO TO 198
727.            IF (AEQUIV(3,NACPAR).EQ.-99999999.) GO TO 196
728.            PRINT 195,TEMP(1),TEMP(2),ACPARS(2,NACPAR),ACPARS(3,NACPAR),
729.           *        UNITS(1),UNITS(2)
730.      195   FORMAT (1X,2A6,' BETWEEN',E15.5,' AND',E15.5,' IN ',2A6)
731.            GO TO 202
732.      196   PRINT 197, TEMP(1),TEMP(2),ACPARS(2,NACPAR),UNITS(1),UNITS(2)
733.      197   FORMAT (1X,2A6,' GREATER THAN ',E15.5,' IN ',2A6)
734.            GO TO 202
735.      198   IF (AEQUIV(3,NACPAR).EQ.-99999999.) GO TO 200
736.            PRINT 199, TEMP(1),TEMP(2),ACPARS(3,NACPAR),UNITS(1),UNITS(2)
737.      199   FORMAT (1X,2A6,' LESS THAN    ',E15.5,' IN ',2A6)
738.            GO TO 202
```

```
      200   PRINT 201, TEMP(1),TEMP(2),UNITS(1),UNITS(2)
      201   FORMAT (1X,2A6,' IN ',2A6)
C
C                 CHECK FOR METHODS ERRORS
C
      202   N = NAMTHD(NACPAR)
            IF (N.EQ.0) GO TO 165
            DO 171 I = 1,N
            AMTHDS(I,NACPAR) = I@ORD(I+4)
      171   IF (AMTHDS(I,NACPAR).LT.1.OR.AMTHDS(I,NACPAR).GT.999) GO TO 176
            PRINT 172, (AMTHDS(I,NACPAR),I=1,N)
      172   FORMAT (' METHODS: ',14,10(',',I4))
            GO TO 165
C
C                       ERROR
C
      175 PRINT 56
            GO TO 167
      176   PRINT 178
      178   FORMAT (' *** ERROR IN METHODS SPECIFICATION ***')
            GO TO 167
      179   WRITE (6,78) ACPARS(1,NACPAR)
            GO TO 167
      182   WRITE (6,183) I@ORD(4)
      183   FORMAT (' *** UNITS CODE',I5,' IS INVALID ***')
            GO TO 167
C
      177   IF (NACPAR.EQ.1) GO TO 135
            NACPAR = NACPAR - 1
            GO TO 167
C
      180   PRINT 12, (CHAR(I),I=1,3)
      181   NACPAR = NACPAR - 1
C
C
C
C                 END OF ACOUSTICS
C
      210   IF (RUNTYP.EQ.1) GO TO 900
        212 PRINT 211
        211 FORMAT ('0       ENVIRONMENTAL REQUEST(S)')
C
C                 READ IN ENV DESCRIPTORS
C
C
      235   PRINT 236
      236   FORMAT ('0ENTER ENVIRONMENTAL DESCRIPTOR(S)')
            READ (5,70,ERR=240,END=243) (CHAR(I),I=1,80)
            CALL FREEFD (CHAR,@ORD,ENDCRS,NENDCR,10,$240)
            IF (NENDCR.EQ.0) GO TO 240
C
C                       CHECK FOR ERROR
C
      238   DO 237 I=1,NENDCR
            IF (ENDCRS(I) .LT. 1 .OR. ENDCRS(I) .GT. 99) GO TO 242 @ TABLE LIMIT 99
        237 CONTINUE
C
```

```
C                    PRINT
C
      DO 239 I = 1,NENDCR
      CALL NAMEIT (7,ENDCRS(I),NAME,NR)
      WRITE (6,139) ENDCRS(I),(NAME(J),J=1,NR)
239   CONTINUE
      GO TO 260
C
C                    ERROR
C
240 PRINT 56
      GO TO 235
242   WRITE (6,78) ENDCRS(I)
      GO TO 235
C
C        BACKUP
C
243   IF (RUNTYP.EQ.3) GO TO 160
      IF (BOUNDS.EQ.'Y') GO TO 90
      GO TO 50
C
C
C              READ ENVIRONMENTAL PARAMETERS
C
C
C
260   PRINT 261
261   FORMAT ('0ENTER ENVIRONMENTAL PARAMETERS (NAME,MIN,MAX,UNIT,METHOD
     1S) / ENTER ''END''')
      NENPAR = 0
C
C                    LOOP TO READ THEM
C
265   NENPAR = NENPAR + 1
267   PRINT 170,NENPAR
      READ (5,70,END=277,ERR=275) (CHAR(I),I=1,80)
      IF (CHAR(1).EQ.'E') GO TO 280
C
C              CHECK FOR PARAMETER ARRAY LIMIT
C
      IF (NENPAR.LE.10) GO TO 274
      WRITE (6,273)
273   FORMAT (' *** INPUT RESTRICTED TO 10 ENVIRONMENTAL PARAMETERS ***'
     1)
      GO TO 281
C
274   WORD(2) = -99999999.
      WORD(3) = -99999999.
      IWORD(4) = 0
      CALL FREEFD (CHAR,WORD,IWORD,NWORDS,14,&275)
      IF (NWORDS.EQ.0) GO TO 275
      ENPARS(1,NENPAR) = IWORD(1)
      EEQUIV(2,NENPAR) = WORD(2)
      EEQUIV(3,NENPAR) = WORD(3)
      IF (IWORD(4).EQ.-99999999) IWORD(4) = 0
      ENPARS(4,NENPAR) = IWORD(4)
      NEMTHD(NENPAR) = NWORDS - 4
      IF (NEMTHD(NENPAR).LT.0) NEMTHD(NENPAR) = 0
```

```
C
C
C                    CHECK FOR ERRORS
C
      IF (ENPARS(1,NENPAR).LT.101.OR.ENPARS(1,NENPAR).GT.172) GO TO 279
      DUM =0
      CALL CONVRT (DUM,ENPARS(1,NENPAR),-ENPARS(4,NENPAR),UNITS)
      IF (UNITS(1) .EQ. 'UNITS ') GO TO 282
C
C              PRINT PARAMETER INFORMATION
C
      CALL NAMGET (ENPARS(1,NENPAR),TEMP)
      IF (EEQUIV(2,NENPAR).EQ.-99999999.) GO TO 269
      IF (EEQUIV(3,NENPAR).EQ.-99999999.) GO TO 268
      PRINT 195,TEMP(1),TEMP(2),ENPARS(2,NENPAR),ENPARS(3,NENPAR),
     .          UNITS(1),UNITS(2)
      GO TO 272
  268 PRINT 197, TEMP(1),TEMP(2),ENPARS(2,NENPAR),UNITS(1),UNITS(2)
      GO TO 272
  269 IF (EEQUIV(3,NENPAR).EQ.-99999999.) GO TO 270
      PRINT 199, TEMP(1),TEMP(2),ENPARS(3,NENPAR),UNITS(1),UNITS(2)
      GO TO 272
  270 PRINT 201, TEMP(1),TEMP(2),UNITS(1),UNITS(2)
C
C              CHECK FOR METHODS ERRORS
C
  272 N = NEMTHD(NENPAR)
      IF (N.EQ.0) GO TO 265
      DO 271 I = 1,N
      EMTHDS(I,NENPAR) = IWORD(I+4)
  271 IF (EMTHDS(I,NENPAR).LT.1.OR.EMTHDS(I,NENPAR).GT.999) GO TO 276
      PRINT 172, (EMTHDS(I,NENPAR),I=1,N)
      GO TO 265
C
C                    ERROR
C
  275 PRINT 56
      GO TO 267
  276 PRINT 178
      GO TO 267
C
  277 IF (NENPAR.EQ.1) GO TO 235
      NENPAR = NENPAR - 1
      GO TO 267
  279 WRITE (6,78) ENPARS(1,NENPAR)
      GO TO 267
  282 WRITE (6,183) IWORD(4)
      GO TO 267
C
C
  280 PRINT 12, (CHAR(I),I=1,3)
  281 NENPAR = NENPAR - 1
C
C
C              END OF ENVIRONMENTAL
C
C
```

```
      900   WRITE (6,910)
      910   FORMAT ('0*** SEARCH IN PROGRESS ***')
            RETURN
            END
```

```
@FOR,S*     N.INTENS,,INTENS

      SUBROUTINE INTENS(VALUE,UNICDE,UNITS)
C
C     TABLE OF INTENSITY LEVEL CONVERSION FACTORS.
C
      INTEGER ENTRY(3,3),UNITS(4),UNICDE
C
C     FACTORS ARE DOUBLE PRECISION.
C
      DOUBLE PRECISION FACTOR(3)
C
C     PUT UNIT CODES INTO ENTRY.
C
      DATA ((ENTRY(I,J),I=1,3),J=1,3)/74,75,76/
C
C     PUT IN ALPHA UNITS.
C
      DATA ((ENTRY(I,J),I=2,3),J=1,3)/'@ATTS/','M**2  ',
     *          '@ATTS/','CM**2 ','@TS/CM','**2/HZ'/
C
C     ENTER INTERNAL UNITS.
C
      UNITS(3)='@ATTS/'
      UNITS(4)='M**2  '
C
C     IF UNICDE=0, THEN SET CODE TO STANDARD UNITS.
C
      IF(UNICDE.EQ.0) UNICDE=-74
C
C     SET CONVERSION FACTORS.
C
      DATA (FACTOR(I),I=1,3)/100.,1.004,1.004/
C
C     INITIALIZE 'UNITS' TO ERROR MESSAGE IN CASE CODE NOT FOUND.
C
      UNITS(1)='UNITS '
      UNITS(2)='ERROR '
C
C     IF CODE IS IN TABLE, PERFORM CONVERSION. ALSO, ENTER UNITS ALPHA
C          CODE INTO 'UNITS'.
C
      ICODE=IABS(UNICDE)
      DO 2 I=1,3
      IF(ICODE.NE.ENTRY(1,I)) GO TO 2
      IF(UNICDE.GE.0) VALUE=VALUE*FACTOR(I)
      IF(UNICDE.LT.0) VALUE=VALUE/FACTOR(I)
      UNITS(1)=ENTRY(2,I)
      UNITS(2)=ENTRY(3,I)
C
C     AT THIS POINT, CONVERSION IS COMPLETE.
C
      RETURN
C
C     ELSE, CHECK REST OF TABLE.
C
```

```
      2 CONTINUE
C
C     IF CODE NOT FOUND, RETURN WITH 'UNITS ERROR' IN 'UNITS'.
C
      PRINT 101,ICODE
  101 FORMAT(' ERROR--UNIT CODE',I3,' NOT IN INTENSITY TABLE.')
      RETURN
      END
```

```
@FOR,$N     N.INVNTY,,INVNTY

      1.            SUBROUTINE INVNTY
      2.
      3.      C     ROUTINE TO PREPARE AN INVENTORY OF THE RUNS MEETING THE SELECTION
      4.      C     CRITERIA.  THE EXPERIMENT, CRUISE, STATION, AND RUN NUMBER FOR EACH
      5.      C     RUN ARE PRINTED.
      6.
      7.            IMPLICIT INTEGER (A-Z)
      8.            DIMENSION NAME(10)
      9.
     10.            COMMON /RUN   /PDRA,RILAT,RINS,RFLAT,RFNS,RILON,RIEW,RFLON,RFEW,
     11.           *               RIMO,RIDAY,RIYR,RITIM,RIZON,RFMO,RFDAY,RFYR,RFTIM,
     12.           *               RFZON,NAVCOD,DTASRC,SYSSRC,SYSRCR,SRCTYP,RCRTYP,
     13.           *               CLAS,DESCR,#AVDIR,#AVHT,HTUNIT,#AVPRD,PRDUNT,
     14.           *               SEASTA,#NDDIR,#NOVEL,VELUNT,S#LDIR,S#LHT,
     15.           *               SHTUNT,S#LPRD,SPDUNT,#EATHR,BOTDPH,DPHUNT,BTMSLP,
     16.           *               BATHY,INFO(9),RSTAT(2),RUNNO
     17.
     18.            WRITE (6,5)
     19.      5     FORMAT ('1INVENTORY OF RUNS SATISFYING SELECTION CRITERIA:' //T15,
     20.           1'EXPER.',T40,'EXPER.',T50,'CRUISE',T58,'STATION',
     21.           2T68,'RUN'/T3,'TYPE',T15,'NUMBER',T23,'EXPER. NAME',T39,
     22.           3'REFERENCE',T50,'NUMBER',T58,'NUMBER',T67,'NUMBER'/)
     23.            CALL GETINT
     24.      10    CALL GETRUN (DATYPE,EXPNO,CRUNO,STANO,$30)
     25.            CALL NAMEIT (1,EXPNO,NAME,NR)
     26.            NR1 = NR - 1
     27.            NR2 = NR - 3
     28.            IF (DATYPE.EQ.4) GO TO 20
     29.            WRITE (6,15) (NAME(I),I=NR1,NR),CRUNO,STANO,RUNNO,EXPNO,
     30.           1(NAME(I),I=1,NR2)
     31.      15    FORMAT (T39,'(',A6,A2,T50,I4,T58,I4,T67,I4,T2,'ACOUSTIC',T15,I4,
     32.           12X,3A6)
     33.            GO TO 10
     34.      20    WRITE (6,21) (NAME(I),I=NR1,NR),CRUNO,STANO,RUNNO,EXPNO,
     35.           1(NAME(I),I=1,NR2)
     36.      21    FORMAT (T39,'(',A6,A2,T50,I4,T58,I4,T67,I4,T2,'ENVIRONMENTAL',T15,
     37.           1I4,2X,3A6)
     38.            GO TO 10
     39.      30    RETURN
     40.            END
```

```
@FOR.S*     N,LINEAR,.LINEAR

      SUBROUTINE LINEAR(VALUE,UNICDE,UNITS)
C
C     TABLE OF LINEAR MEASURE CONVERSION FACTORS.
C
      INTEGER ENTRY(3,12),UNITS(4),UNICDE
C
C     FACTORS ARE DOUBLE PRECISION.
C
      DOUBLE PRECISION FACTOR(12)
C
C     PUT UNIT CODES INTO ENTRY.
C
      DATA (ENTRY(1,I),I=1,12)/2,3,4,5,6,7,8,9,10,11,12,86/
C
C     PUT IN ALPHA UNITS.
C
      DATA ((ENTRY(I,J),I=2,3),J=1,12)/'MILLIM','ETERS ',
     *                 'CENTIM','ETERS ','METERS','      ',
     *                 'KILOME','TERS  ','INCHES','      ',
     *                 'FEET  ','      ','YARDS ','      ',
     *                 'KILOYA','RDS   ','NAUT. ','MILES ',
     *                 'STAT. ','MILES ','FATHOM','S     ',
     *                 'MICRON','S     '/
C
C     ENTER INTERNAL UNITS.
C
      UNITS(3)='METERS'
      UNITS(4)='      '
C
C     IF UNICDE = 0, THEN SET CODE TO STANDARD UNITS.
C
      IF(UNICDE.EQ.0) UNICDE=-4
C
C     SET CONVERSION FACTORS.
C
      DATA (FACTOR(I),I=1,12)/.00100,.0100,.100,1.00,2.5400050-2,
     *                        .3048000D0,.9144019217D0,9.1440192170-2,
     *                        1.8520D3,1.4093503,1.8288036D0,1.00-9/
C
C     IF CODE IS IN TABLE, PERFORM CONVERSION. ALSO, ENTER UNITS ALPHA
C          CODE INTO 'UNITS'.
C
      ICODE=IABS(UNICDE)
      DO 2 I=1,12
      IF(ICODE.NE.ENTRY(1,I)) GO TO 2
      IF(UNICDE.GE.0) VALUE=VALUE*FACTOR(I)
      IF(UNICDE.LT.0) VALUE=VALUE/FACTOR(I)
      DO 1 J=1,2
      UNITS(J)=ENTRY(J+1,I)
    1 CONTINUE
C
C     AT THIS POINT, CONVERSION IS COMPLETE.
C
      RETURN
```

```
      C
      C     ELSE, CHECK REST OF TABLE.
      C
    2 CONTINUE
      C
      C     IF CODE NOT FOUND, RETURN WITH 'UNITS ERROR' IN 'UNITS'.
      C
      UNITS(1)='UNITS '
      UNITS(2)='ERROR '
      PRINT 101,ICODE
  101 FORMAT(' ERROR--UNIT CODE',I3,' NOT IN LINEAR TABLE.')
      RETURN
      END
```

```
@FOR,S*     N.LNGTME,.LNGTME

      1.          SUBROUTINE LNGTME(VALUE,UNICOE,UNITS)
      2.    C
      3.    C     TABLE OF LNGTME MEASURE CONVERSION FACTORS.
      4.    C
      5.          INTEGER ENTRY(3,2),UNITS(4),UNICDE
      6.    C
      7.    C     FACTORS ARE DOUBLE PRECISION.
      8.    C
      9.          DOUBLE PRECISION FACTOR(2)
     10.    C
     11.    C     PUT UNIT CODES INTO ENTRY.
     12.    C
     13.          DATA ((ENTRY(I,J),I=1,2)/26,27/
     14.    C
     15.    C     PUT IN ALPHA UNITS.
     16.    C
     17.          DATA ((ENTRY(I,J),I=2,3),J=1,2)/'MONTHS',' ',
     18.         *               'YEARS ',' '/
     19.    C
     20.    C     ENTER INTERNAL UNITS.
     21.    C
     22.          UNITS(3)='MO.   '
     23.          UNITS(4)='      '
     24.    C
     25.    C     IF UNICDE=0, THEN SET CODE TO STANDARD UNITS.
     26.    C
     27.          IF(UNICDE.EQ.0) UNICDE=-26
     28.    C
     29.    C     SET CONVERSION FACTORS.
     30.    C
     31.          DATA (FACTOR(I),I=1,2)/100,1200/
     32.    C
     33.    C     IF CODE IS IN TABLE, PERFORM CONVERSION. ALSO, ENTER UNITS ALPHA
     34.    C          CODE INTO 'UNITS';
     35.    C
     36.          ICODE=IABS(UNICDE)
     37.          DO 2 I=1,2
     38.          IF(ICODE.NE.ENTRY(1,I)) GO TO 2
     39.          IF(UNICDE.GE.0) VALUE=VALUE*FACTOR(I)
     40.          IF(UNICDE.LT.0) VALUE=VALUE/FACTOR(I)
     41.          DO 1 J=1,2
     42.          UNITS(J)=ENTRY(J+1,I)
     43.        1 CONTINUE
     44.    C
     45.    C     AT THIS POINT, CONVERSION IS COMPLETE.
     46.    C
     47.          RETURN
     48.    C
     49.    C     ELSE, CHECK REST OF TABLE.
     50.    C
     51.        2 CONTINUE
     52.    C
     53.    C     IF CODE NOT FOUND, RETURN WITH 'UNITS ERROR' IN 'UNITS'.
     54.    C
```

```
      UNITS(1)='UNITS '
      UNITS(2)='ERROR '
      PRINT 101,ICODE
  101 FORMAT(' ERROR--UNIT CODE',I3,' NOT IN LONG TIME TABLE.')
      RETURN
      END
```

```
@ELT,L     N.MSQCVT,.MSQCVT

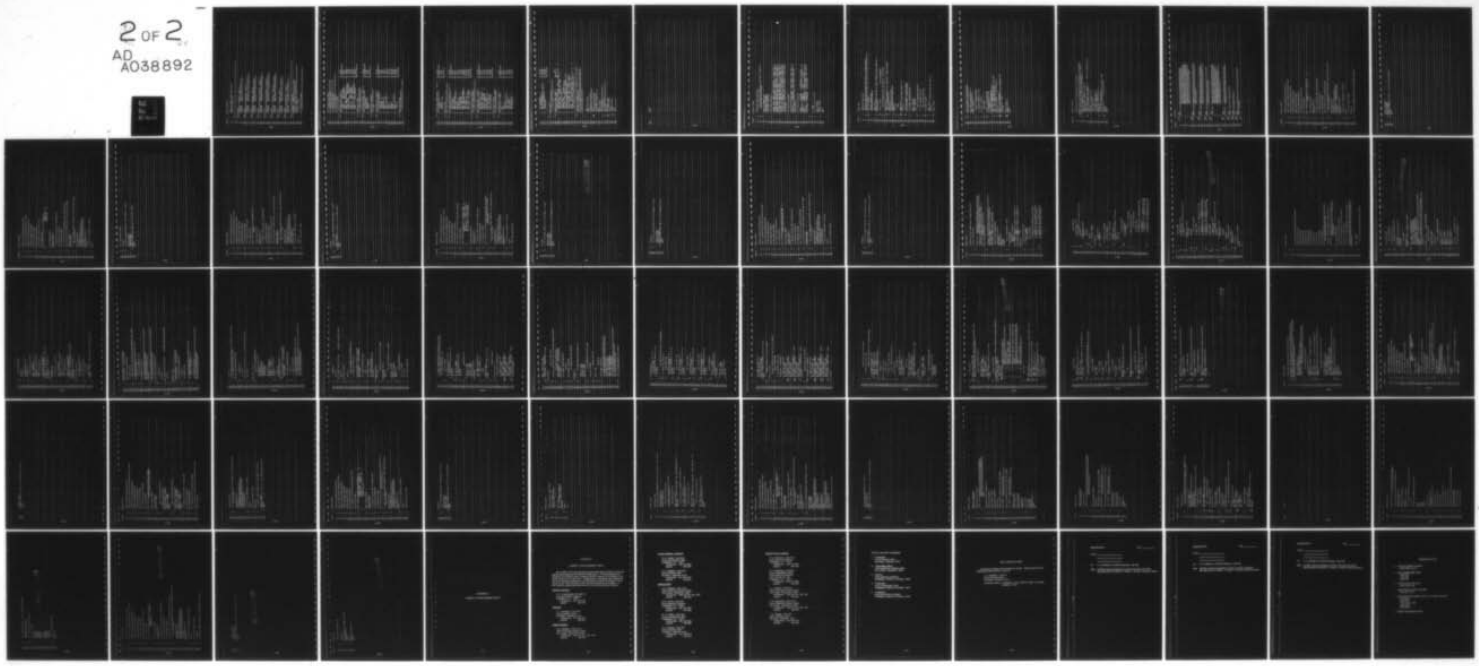
      SUBROUTINE MSQCVT(MSQNO1,MSQNO2,LAT1,LON1,LAT2,LON2)
C
C         PROGRAM TO CONVERT INPUT OF TWO MARSDEN SQUARES INTO
C         LATITUDES AND LONGITUDES OF TWO POINTS DEFINING THE
C         MINIMUM RECTANGLE INCLUDING THE MARSDEN SQUARES.
C
C
C         INITIALIZE THE FIRST SQUARE.
C
      DIMENSION LON(2),LAT(2)
    1 LNADD=10      @TO GET LONGITUDE.
      LTSUB=0       @TO GET LATITUDE.
      ISGN=1        @SIGN FOR NORTHERN HEMISPHERE.
      I=1       @INDICATES FIRST SQUARE IS IN PROCESS.
      MSQ=MSQNO1
C
C         IF SQUARE IS IN SOUTHERN HEMISPHERE, BRANCH TO
C         CODE.
C
    3 IF(MSQ.LT.300.OR.MSQ.GT.623) GO TO 4
      MSQ=MSQ-264       @ADJUSTMENT FOR COMMON CODE BELOW.
      ISGN=-1       @SIGN FOR SOUTHERN HEMISPHERE.
C
C         BRANCH TO CALCULATIONS.
C
      GO TO 5
C
    4 IF(MSQ.GT.623) MSQ=MSQ-613
      MSQ=MSQ-1
C
C         CALCULATE LONGITUDE AND LATITUDE.
C
    5 LON(I)=MOD(MSQ,36)*10+LNADD
      LAT(I)=(MSQ/36+ISGN+LTSUB)*10
C
C         ADJUST LONGITUDE TO ABS(LON(I)) .LE. 180.
C
      IF(LON(I).LT.180) LON(I)=-LON(I)
      IF(LON(I).GE.180) LON(I)=360-LON(I)
C
C         IF DONE WITH THIS CASE, RETURN.
C
      IF(I.EQ.2) GO TO 99
C
C         INITIALIZE FOR SECOND SQUARE.
C
      LNADD=0
      LTSUB=1
      MSQ=MSQNO2
      I=2       @INDICATES SECOND SQUARE IN PROCESS.
C
C         START SECOND PASS.
C
      GO TO 3
```

```
      99 LON1=LON(2)*1000
         LON2=LON(1)*1000
         LAT1=LAT(2)*1000
         LAT2=LAT(1)*1000
         RETURN
         END
```

AD-A038 892 NAVAL SEA SYSTEMS COMMAND WASHINGTON D C F/G 17/1
NAVSEA OCEAN ENVIRONMENTAL ACOUSTIC DATA BANK - NAVDAB - IN SUP--ETC(U)
DEC 75

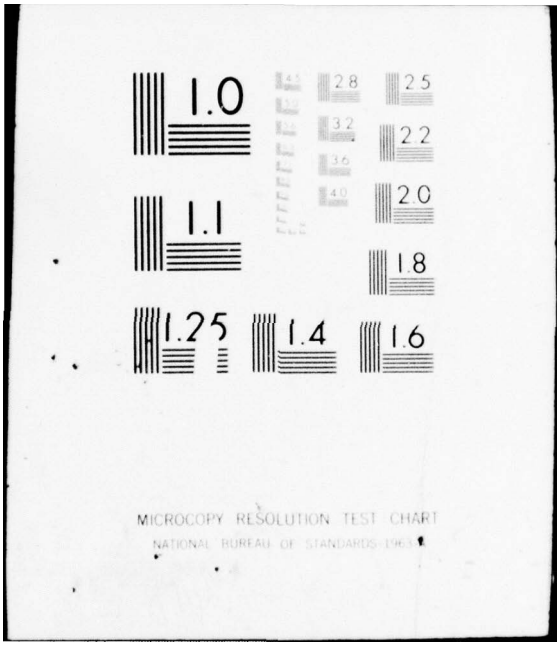
UNCLASSIFIED NAVSEA-06H1/036-EVA/MOST- NL

2 OF 2
AD A038892



END

DATE FILMED
5-77


1.0
1.1
1.25
1.4
1.6
2.8
2.5
3.2
2.2
3.6
2.0
4.0
1.8
MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

```
@FOR,SH     N.NAMEIT,.NAMEIT

      SUBROUTINE NAMEIT (ISET,ICODE,NAME,NR)
      IMPLICIT INTEGER (A-Z)
C
C.....
C
C**** ISET=1
C**** EXPERIMENT PARAMETERS  (TABLE 16)
      PARAMETER     T16S=1,T16E=T16S+51,NRT16=10
      PARAMETER     I#D16S=1,I#D16E=I#D16S + NRT16 - 1
C
C**** ISET=2
C**** CLASSIFICATION PARAMETERS  (TABLE 17)
      PARAMETER     T17S=T16E+1,T17E=T17S+4,NRT17=3
      PARAMETER     I#D17S=I#D16E+1,I#D17E=I#D17S+NRT17-1
C
C**** ISET=3
C**** NAVIGATION PARAMETERS  (TABLE 9)
      PARAMETER     T9S=T17E+1,T9E=T9S+26,NRT9=14
      PARAMETER     I#D9S=I#D17E+1,I#D9E=I#D9S+NRT9-1
C
C**** ISET=4
C**** SYSTEM (SOURCE/RECVR) PARAMETERS   (TABLE 11)
      PARAMETER     T11S=T9E+1,T11E=T11S+6,NRT11=3
      PARAMETER     I#D11S=I#D9E+1,I#D11E=I#D11S+NRT11-1
C
C**** ISET=5
C**** TYPE (SOURCE/RECVR) PARAMETERS   (TABLE 12)
      PARAMETER     T12S=T11E+1,T12E=T12S+42,NRT12=14
      PARAMETER     I#D12S=I#D11E+1,I#D12E=I#D12S+NRT12-1
C
C**** ISET=6
C**** DATA SOURCE PARAMETERS   (TABLE 10)
      PARAMETER     T10S=T12E+1,T10E=T10S+29,NRT10=9
      PARAMETER     I#D10S=I#D12E+1,I#D10E=I#D10S+NRT10-1
C
C**** ISET=7
C**** ENVIR RUN PARAMETERS  (TABLE 7)
      PARAMETER     T7S=T10E+1,T7E=T7S+36,NRT7=14
      PARAMETER     I#D7S=I#D10E+1,I#D7E=I#D7S+NRT7-1
C
C**** ISET=8
C**** ACOUSTIC RUN PARAMETERS   (TABLE 4)
      PARAMETER T4S=T7E+1,T4E=T4S+73,NRT4=40
      PARAMETER     I#D4S=I#D7E+1,I#D4E=I#D4S+NRT4-1
C
C.....
C**** DIMENSIONING PARAMETERS
      PARAMETER     NMAX1=T4E,NMAX2=I#D4E,NGRP=8
      DIMENSION I#ORD(NMAX1),I#D(NMAX2),NAME(1)
      DIMENSION INDXST(NGRP),INDXND(NGRP),NSETS(NGRP),IOFSTR(NGRP)
      DIMENSION IC(NGRP)
C
      DATA (IOFSTR(I),I=1,NGRP)/
     1    I#D16S,I#D17S,I#D9S,I#D11S,I#D12S,I#D10S,I#D7S,I#D4S/
```

```
      DATA (INDXST(I),I=1,NGRP)/
     1    T16S,T17S,T9S,T11S,T12S,T10S,T7S,T4S/
      DATA (INDXND(I),I=1,NGRP)/
     1    T16E,T17E,T9E,T11E,T12E,T10E,T7E,T4E/
      DATA (NSETS(I),I=1,NGRP)/
     1    NRT16,NRT17,NRT9,NRT11,NRT12,NRT10,NRT7,NRT4/
      DATA (IC(I),I=1,NGRP)/
     1    0,0,1,0,1,1,0,0/
C
C**** EXPERIMENT NAMES  (TABLE 16.                    @   ISET=1
      DATA (IWORD(I),I=T16S,T16E)/                    @ ICDE,NWDS,OFSET
     1    'FASOR I      (REF: 024-005)',              @   1,5,0
     2    'FASOR II     (REF: 026-002)',              @   2,5,5
     3    'AMOS  (REF: 042-001)',                     @   3,4,10
     4    'LORAD, HAWAII      (REF:088-001)',         @   4,6,14
     5    'LORAD, ALASKA      (REF:198-003)',         @   5,6,20
     6    'JOAST III    (REF: 142-001)',              @   6,5,26
     7    'JOAST IV     (REF: 142-002)',              @   7,5,31
     8    'FASOR III    (REF: 027-001)',              @   8,5,36
     9    'GULF OF ALASKA     (REF: 195-001)',        @   9,6,41
     1    'SUDS I       (REF: 086-002)'               @  10,5,47
     *    /                                           @END OF EXPR NMS
C**** EXPERIMENT OFFSETS
      DATA (IWD(I),I=I6D16S,IWD16E)/
     1    0,5,10,14,20,26,31,36,41,47/
C
C**** CLASSIFICATION NAMES   (TABLE 17)               @   ISET=2
      DATA (IWORD(I),I=T17S,T17E)/                    @ ICDE,NWDS,OFSET
     1    'UNCLASSIFIED',                             @   1,2,0
     2    'CONFIDENTIAL',                             @   2,2,2
     3    'SECRET'                                    @   3,1,4
     *    /                                           @END SECUR NMS
C**** CLASSIFICATION OFFSETS
      DATA (IWD(I),I=IWD17S,IWD17E)/
     1    0,2,4/
C
C**** NAVIGATION NAMES  (TABLE 9)                     @   ISET=3
      DATA (IWORD(I),I=T9S,T9E)/                      @ ICDE,NWDS,OFSET
     1    'NOT AVAIL /OR APPL',                       @   1,3,0
     2    'SATELLITE    ',                            @   2,2,3
     3    'CELESTIAL    ',                            @   3,2,5
     4    'DEAD RECKON,',                             @   4,2,7
     5    'LORAN C      ',                            @   5,2,9
     6    'EST. - SEE NOTES  ',                       @   6,3,11
     7    'LORAN A      ',                            @   7,2,14
     8    'DECCA ',                                   @   8,1,16
     9    'OMEGA ',                                   @   9,1,17
     1    'SHORAN',                                   @  10,1,18
     1    'RAYDIST      ',                            @  11,2,19
     2    'COMB. - SEE NOTES ',                       @  12,3,21
     3    'SINS  ',                                   @  13,1,24
     4    'ACOUSTIC     '                             @  14,2,25
     *    /                                           @END NAV NMS
C**** NAVIGATION OFFSETS
      DATA (IWD(I),I=IWD9S,IWD9E)/
     1    0,3,5,7,9,11,14,16,17,18,19,21,24,25/
C
```

```
C**** SYSTEM NAMES (SOURCE/RECVR)   (TABLE 11)      @   ISET=4
      DATA (IWORD(I),I=T11S,T11E)/                 @ ICDE,NWDS,OFSET
     1   'NOT APPLIC. ',                           @   1,2,0
     2   'AN/BQS-5    ',                           @   2,2,2
     3   'AN/SQS-26 (AXR)  '                       @   3,3,4
     *   /                                         @END SYS NMS
C**** SYSTEM OFFSETS
      DATA (IWD(I),I=IWD11S,IWD11E)/ 0,2,4/
C
C**** TYPE NAMES (SOURCE/RECVR)  (TABLE 12)        @   ISET=5
      DATA (IWORD(I),I=T12S,T12E)/                 @ ICDE,NWDS,OFSET
     1   'NOT APPLIC. ',                           @   1,2,0
     2   'HORIZ. ARRAY',                           @   2,2,2
     3   'OMNI. HYDROPHONE  ',                     @   3,3,4
     4   'DIR. PROJ. - CW PULSED  ',               @   4,4,7
     5   'EXPLOSIVE   ',                           @   5,2,11
     6   'AMBIENT NOISE     ',                     @   6,3,13
     7   'SUS CHARGE, MK61  ',                     @   7,3,16
     8   'DIR. HYDROPHONE   ',                     @   8,3,19
     9   'OMNI. PROJ. - CW PULSED ',               @   9,4,22
     1   'VERT. HYDROPHONE STRING ',               @  10,4,26
     1   'VERT. ARRAY ',                           @  11,2,30
     2   'SUS CHARGE, MK82  ',                     @  12,3,32
     3   'OMNI. PROJ. - FM PULSED ',               @  13,4,35
     4   'DIR. PROJ. - FM PULSED  '                @  14,4,39
     *   /                                         @END TYP NMS
C**** TYPE (SOURCE/RECVR) OFFSETS
      DATA (IWD(I),I=IWD12S,IWD12E)/
     1   0,2,4,7,11,13,16,19,22,26,30,32,35,39/
C
C**** DATA SOURCE NAMES   (TABLE 10)               @   ISET=6
      DATA (IWORD(I),I=T10S,T10E)/                 @ ICDE,NWDS,OFSET
     1   'NOT APPLIC. ',                           @   1,2,0
     2   'GRAPHICAL, PRELIMINARY  ',               @   2,4,2
     3   'GRAPHICAL, PUBLISHED    ',               @   3,4,6
     4   'TABULAR, PRELIMINARY    ',               @   4,4,10
     5   'TABULAR, PUBLISHED      ',               @   5,4,14
     6   'PUNCHED CARDS     ',                     @   6,3,18
     7   'MAG. TAPE, DIGITAL',                     @   7,3,21
     8   'MAG. TAPE, ANALOG ',                     @   8,3,24
     9   'PUNCHED PAPER TAPE'                      @   9,3,27
     *   /                                         @END DSR NMS
C**** DATA SOURCE OFFSETS
      DATA (IWD(I),I=IWD10S,IWD10E)/
     1   0,2,6,10,14,18,21,24,27/
C
C**** ENVIR RUN NAMES   (TABLE 7)                  @   ISET=7
      DATA(IWORD(I),I=T7S,T7E)/                    @ ICDE,NWD,OFSET
     1   'BT    ',                                 @   1,1,0
     2   'XBT   ',                                 @   2,1,1
     3   'AXBT  ',                                 @   3,1,2
     4   'HYDROGRAPHIC CAST ',                     @   4,3,3
     5   'CORE, PISTON (GRAVITY)  ',               @   5,4,6
     6   'CORE, SHIPEK',                           @   6,2,10
     7   'GRAB SAMPLE ',                           @   7,2,12
     8   'THERMISTOR CHAIN  ',                     @   8,3,14
     9   'DEPTH/SOUND VEL.  ',                     @   9,3,17
```

```
     1     'SAL/TEMP/DEPTH     ',                    @  10,3,20
     1     'SAL/TEMP/DEPTH/3V  ',                    @  11,3,23
     2     'SEA SFC/WIND SPEED TIME SERIES',         @  12,6,26
     3     'ENVIR. CONSTANTS ONLY    ',              @  13,9,31
     4     'BATHYMETRY  '                            @  14,2,35
     .     /                                         @END ENV NMS
C
C**** ENVIR OFFSETS
      DATA (IWD(I),I=IWD7S,IWD7E)/
     1   0,1,2,3,4,10,12,14,17,20,23,26,31,35/
C
C**** ACOUSTIC RUN NAMES  (TABLE 4)                  @  ISET=8
      DATA(IWORD(I),I=T4S,T4E)/                      @ICDE
     1   'DIRECT PATH ','SINGLE BR   ','N/U   ','SD    ','CZ    ',
     2   'SHALLOW WATER     ','BOT. REFL. LOSS - 1ST B.',
     3   'BOT. REFL. LOSS - NORMAL INC. ','N/U   ',
     4   'VOLUME SCAT.',2*'N/U   ',
     5   'AMBIENT NOISE     ',2*'N/U   ',
     6   'BOT. REFL. LOSS - 2ND B.','2ND BR',
     7   'SD, BR, CZ  ','DEEP REFR.  ','SURFACE SCAT',
     8   'SD,RSR','SD,DEEP REFR','3RD BR',
     1   'BOT. REFL. LOSS - 3RD B.','SD,DP ','BR CZ ',
     2   'SD,BR ','SOFAR ','N/U   ','BEAM PATTERN',
     3   2*'N/U   ','R.A.P.','SD,BC ','N/U   ',
     4   'SHADOW ZONE ','MULTI - MODE','VERT. BEAM PAT. - CZ    ',
     .   'VERT. BEAM PAT. - BB/OOT','SOFAR TO SD '/
C
C**** ACOUSTIC RUN OFFSETS (LT 0 NOT PRECEDED WITH PROP. LOSS )
      DATA(IWD(I),I=IWD4S,IWD4E)/
     1 0,2,-4,5,6,7,-10,-14,-19,-20,-22,-23,-24,-27,-28,-29,
     2 33,34,36,-38,40,41,43,-44,48,49,50,51,-52,-53,-55,-56,
     3 57,58,-59,60,62,-64,-68,72/
C
C
      NR=0
      ICDE=ICODE+IC(ISET)
      JMP=1
      IF(ICDE.LE.0) GO TO 999
      IF(INSETS(ISET)-ICDE)999,50,60
50    JMP=2
60    IOFFI=IOFSTR(ISET)+ICDE-1
      IF(ISET.NE.8) GO TO 65
      IF(IWD(IOFFI).LT.0) GO TO 65
      NR=2
      NAME(1)='PROP. '
      NAME(2)='LOSS  '
65    IST=INDXST(ISET)+IABS(IWD(IOFFI))
      GO TO (70,80),JMP
70    IEND=INDXST(ISET)+IABS(IWD(IOFFI+1))-1
      GO TO 90
80    IEND=INDXND(ISET)
90    DO 100 I=IST,IEND
      NR=NR+1
100   NAME(NR)=IWORD(I)
      RETURN
999   NR=1
      NAME(1)='******'
```

```
224.          RETURN
227.          END
```

```
$FOR,S=     N.NAMGET,.NAMGET

      1.            SUBROUTINE NAMGET (CODE,NAME)
      2.      C
      3.      C           NAMGET (NAME GET) RETURNS THE ALPHA NAME IN 'NAME' FOR
      4.      C           THE NAME CODE 'CODE' ENTERED.
      5.      C
      6.            IMPLICIT INTEGER (A-Z)
      7.            REAL*8 DATA(199),TEMP
      8.            DIMENSION N(2),NAME(2)
      9.            EQUIVALENCE (TEMP,N(1))
     10.      C
     11.            DATA (DATA(I),I=1,72) /
     12.      C
     13.           *'SOURCE DEPTH','RECVR  DEPTH','FREQUENCY   ','FREQ. BAND  ',
     14.           *'SOURCE LEVEL','PULSE LENGTH','WATER DEPTH ','TRANS. DEPTH',
     15.           *'COLUMN STRTH','SHALLOW D.L.','GREATER D.L.','PROP. LOSS  ',
     16.           *'HORIZ. RANGE','STH REF LOSS','GRAZING ANGLE','LOW FREQ LIM',
     17.           *'HI FREQ LIM ','VOL SCAT STR','DEPTH       ','TRAVEL TIME ',
     18.           *'SUR SCAT STR','RPTR SCE DTH','RPTR RCR DTH','TARGET STR. ',
     19.           *'REVERB. LVL.','REPT. RATE  ','REL RESPONSE','ANGULAR DEV.',
     20.           *'GEO MEAN FRQ','DEPRESS. ANG','NOR REPEATER','RNGE/DPTH SS',
     21.           *'S HOR BEAMW ','S VERT BEAMW','R HOR BEAMW ','R VERT BEAMW',
     22.           *36*'NOT USED    '/
     23.      C
     24.            DATA (DATA(I),I=73,85)/
     25.      C
     26.           *'WAVE DIR COD','WAVE HEIGHT ','WAVE PERIOD ','SEA STATE   ',
     27.           *'WIND DIR    ','WIND SPEED  ','SWELL DIR CD','SWELL HEIGHT',
     28.           *'SWELL PERIOD','WEATHER CODE','BOTTOM SLOPE','NOT USED    ',
     29.           *'BOTTOM DEPTH'/
     30.      C
     31.            DATA (DATA(I),I=86,199)/
     32.      C
     33.           *'DEPTH       ','NOT USED    ','TEMPERATURE ','SALINITY    ',
     34.           *'SOUND SPEED ','SIGMA-T     ','HI DEPTH LIM','LO DEPTH LIM',
     35.           *'DENSITY     ','POROSITY    ','SAND        ','SILT        ',
     36.           *'CLAY        ','GRAIN SIZE  ','TRK DISTANCE','TIME OF DAY ',
     37.           *'WAVE DIR.   ','SWELL DIR.  ','HORIZ. RANGE','ATTEN FACTOR',
     38.           *'SS GRADIENT ','THICKNESS   ',37*'NOT USED    '/
     39.      C
     40.      C
     41.      C
     42.            J = CODE
     43.            IF (J .GT. 72) J = J -28
     44.      C
     45.            TEMP = DATA(J)
     46.            NAME(1) = N(1)
     47.            NAME(2) = N(2)
     48.      C
     49.            RETURN
     50.            END
```

```
@ELT,L     N.NOTE,.NOTE

      1.          SUBROUTINE NOTE (DATA)
      2.    C
      3.    C     ROUTINE FOR THE SELECTIVE PRINTING OF NOTES.  THE USER RESPONDS TO
      4.    C     QUESTIONS ASKING THE EXPERIMENT NUMBER AND CRUISE NUMBER FOR WHICH NOTES
      5.    C     ARE TO BE PRINTED.
      6.    C
      7.          IMPLICIT INTEGER (A-Z)
      8.          DIMENSION DATA(1),NAME(10)
      9.    C
     10.          WRITE (6,5)
     11.    5     FORMAT (' SPECIFY THE EXP. AND CRUISE NUMBER FOR WHICH NOTES ARE T
     12.         1O BE PRINTED')
     13.    C
     14.    10    PNXTE = 262145                    @ OCTAL 000001000001
     15.    C
     16.    C          INPUT EXPERIMENT NUMBER AND CRUISE NUMBER FOR NOTES
     17.    C
     18.    15    WRITE (6,20)
     19.    20    FORMAT (/' ENTER EXPERIMENT NO., CRUISE NO. / ENTER ',6H'END')
     20.          READ (5,30,ERR=90) FLAG
     21.    30    FORMAT (A1)
     22.          IF (FLAG.EQ.'E') GO TO 900
     23.          READ (0,35,ERR=90) EXPNO,CRUSNO
     24.    35    FORMAT ()
     25.          GO TO 50
     26.    C
     27.    C          ERROR
     28.    C
     29.    90    WRITE (6,95)
     30.    95    FORMAT (' *** READ FORMAT ERROR ***')
     31.          GO TO 15
     32.    C
     33.    50    FLD(18,18,SECADR) = FLD(0,18,PNXTE)
     34.          FLD (18,18,NSEC) = FLD(18,18,PNXTE)
     35.    C
     36.          CALL DSKRED (SECADR,NSEC,DATA)
     37.    C
     38.          EXNO = DATA(2)
     39.          PNXTE = DATA(15)
     40.          PCRU = DATA(16)
     41.    C
     42.    C          CHECK IF REQUESTED EXPERIMENT
     43.    C
     44.          IF (EXNO.EQ.EXPNO) GO TO 60
     45.    C
     46.    C          CHECK FOR NEXT EXPERIMENT
     47.    C
     48.    55    IF (PNXTE) 50,,50
     49.    C
     50.          WRITE (6,57)
     51.    57    FORMAT (' *** SPECIFIED RECORD NOT FOUND ***')
     52.          GO TO 10
     53.    C
     54.    60    PNXTC = PCRU
```

```
      C
   70 FLD(18,18,SECADR) = FLD(0,18,PNXTC)
      FLD(18,18,NSEC) = FLD(18,18,PNXTC)
      C
      CALL DSKRED (SECADR,NSEC,DATA)
      CRUNO = FLD(0,18,DATA(2))
      PNXTC = DATA(3)
      C
      C          CHECK IF REQUESTED CRUISE
      C
      IF (CRUNO.EQ.CRUSNO) GO TO 80
      C
      IF (PNXTC) 70,55,70
      C
      C          PRINT NOTES
      C
   80 NCARDS = FLD(18,18,DATA(2))
      IF (NCARDS.EQ.0) GO TO 100
      N = 19*NCARDS + 17
      CALL NAMEIT (1,EXNO,NAME,NR)
      WRITE (6,90) EXNO,(NAME(I),I=1,NR)
   90 FORMAT (1H1,' NOTES FOR EXPERIMENT',I5,2X,10A6)
      WRITE (6,91) CRUNO,(DATA(I),I=18,N)
   91 FORMAT (T3,'CRUISE    ',I5//(1X,13A6,A2))
      GO TO 10
  100 WRITE (6,110) EXNO,CRUNO
  110 FORMAT (' THERE ARE NO NOTES FOR EXPERIMENT',I5/
     1T25,'CRUISE    ',I5)
      GO TO 10

  900 WRITE (6,910)
  910 FORMAT (3X,'END')
      RETURN
      END
```

```
@FOR,SW     U.PARATB,.PARATB

      SUBROUTINE PARATB (ITEMS,DSITMS,NITEMS,ITABLE)
C
C  PROGRAM TO PRINT THE NUMBER OF RUNS FOR CONSTANTS, AND THE NUMBER
C  OF RUNS AND DATA SETS FOR PARAMETERS AND VARIABLES (IF NONZERO).
C  (ITABLE=2 FOR ACOUSTIC, ITABLE=5 FOR ENVIRONMENTAL).
      IMPLICIT INTEGER (A-Z)
      DIMENSION ITEMS(1),DSITMS(1),NAME(2)
      WRITE (6,10)
   10 FORMAT (T30,'NUMBER',T40,'NUMBER OF'/'   CODE',T12,'PARAMETER NAME
     1',T30,'OF RUNS',T40,'DATA SETS'/)
      CODE = 0
      IF (ITABLE.EQ.5) CODE = 100
      DO 30 I = 1,NITEMS
      CODE = CODE + 1
      IF (ITEMS(I).EQ.0) GO TO 30
      CALL NAMGET (CODE,NAME)
      ENCODE (6,15,VALUE) DSITMS(I)
   15 FORMAT (I6)
      IF (DSITMS(I).EQ.0) VALUE = ' '
      WRITE (6,20) CODE,NAME(1),NAME(2),ITEMS(I),VALUE
   20 FORMAT (T4,I3,'  -  ',2A6,T29,I6,T40,A6)
   30 CONTINUE
      RETURN
      END
```

```
@PDP.LF      N.PDP,.PDP

      RETREV PROC
            COMMON /RETREV/NLAT,ELON,SLAT,WLON,FMYR,TOYR,NMON,MONTHS(12),
           .                FMDAY,TODAY,FMTIM,TOTIM,NOSEX,SEX(10),NACDCR,
           .                ACDCRS(10),NACPAR,ACPARS(9,10),NENDCR,ENDCRS(10),
           .                NENPAR,ENPARS(9,10),RUNTYP,NPTRS,NSTYPE,STYPE(10),
           .                NRTYPE,RTYPE(10),NSSYST,SSYST(10),NRSYST,RSYST(10),
           .                NAMTHD(10),AMTHDS(10,10),NEMTHD(10),EMTHDS(10,10),
           .                ANY,EXP,CRU,STA,RUN,CLSREQ,TNBR,TNAME(6),MASK(3),
           .                RQUEST(3),DATE,DAY,YEAR,TIME,MINUTE,SECOND
      END
      EXPMNT PROC
            COMMON /EXPMNT/EXNO,PNXTE,PCRU,EAILAT,EAILON,EAIDAT,EAFLAT,EAFLON,
           .                EAFDAT,EEILAT,EEILON,EEIDAT,EEFLAT,EEFLON,EEFDAT,
           .                EASTAT(2),EESTAT(2)
      END
      CRUISE PROC
            COMMON /CRUISE/CRUNO,PNXTC,CAILAT,CAILON,CAIDAT,CAFLAT,CAFLON,
           .                CAFDAT,CEILAT,CEILON,CEIDAT,CEFLAT,CEFLON,CEFDAT,
           .                CASTAT(2),CESTAT(2),PACS,PENS,NCARDS,NOTES(14,132)
      END
      ASTATN PROC
            COMMON /ASTATN/SANO,SANRUN,SAILAT,SAILON,SAIDAT,SAFLAT,SAFLON,
           .                SAFDAT,SASTAT(2),PNXTAS,PENVS
      END
      RUN    PROC
            COMMON /RUN   /PORA,RILAT,RINS,RFLAT,RFNS,RILON,RIEW,RFLON,RFEW,
           .                RIMO,RIDAY,RIYR,RITIM,RIZON,RFMO,RFDAY,RFYR,RFTIM,
           .                RFZON,NAVCOD,DTASRC,SYSSRC,SYSRCR,SRCTYP,RCRTYP,
           .                CLAS,DESCR,BAVDIN,BAVHT,HTUNIT,BAVPRD,PRDUNT,
           .                SEASTA,BNDDIR,BNDVEL,VELUNT,SWLDIR,SWLPRD,
           .                SHTUNT,SWLPRO,SPDUNT,WEATHR,BOTDPH,DPHUNT,BTMSLP,
           .                BATHY,INFO(9),RSTAT(2),RUNNO
      END
      CPV    PROC
            COMMON /CPV   /NCON,CON(9,30),NPAR,PAR(30,30),NVAR,IVAR(3,30),
           .                NDATS,NROWS(50),NDATA,VALRNG(2,30,50),DATA(30000)
      END
      SECTO  PROC
            COMMON /SECTO /ID(7),IDATE,DUMMY(20)
      END
      DATCOM PROC
            COMMON /DATCOM/NDUSED,NRS(50)
      END
      GETLST PROC
            COMMON /GETLST/EXPI,CRUI,STAI,RUNI,ALL,NRUNSF,USERAP,AETYPE
            LOGICAL ALL,USERAP
      END
```

```
@FOR,SN     N.PERCNT,.PERCNT

      SUBROUTINE PERCNT(VALUE,UNICDE,UNITS)
C
C     TABLE OF PERCNT MEASURE CONVERSION FACTORS.
C
      INTEGER ENTRY(3,3),UNITS(4),UNICDE
C
C     FACTORS ARE DOUBLE PRECISION.
C
      DOUBLE PRECISION FACTOR(3)
C
C     PUT UNIT CODES INTO ENTRY.
C
      DATA (ENTRY(1,I),I=1,3)/92,93,94/
C
C     PUT IN ALPHA UNITS.
C
      DATA ((ENTRY(I,J),I=2,3),J=1,3)/'PER CE','NT    ',
     .              '(0/00)','      ','PPM   ','      '/
C
C     ENTER INTERNAL UNITS.
C
      UNITS(3)='PERCEN'
      UNITS(4)='T     '
C
C     IF UNICDE=0, THEN SET CODE TO STANDARD UNITS.
C
      IF(UNICDE.EQ.0) UNICDE=-92
C
C     SET CONVERSION FACTORS.
C
      DATA (FACTOR(I),I=1,3)/1.0D0,0.10D0,1.0D-4/
C
C     IF CODE IS IN TABLE, PERFORM CONVERSION. ALSO, ENTER UNITS ALPHA
C          CODE INTO 'UNITS'.
C
      ICODE=IABS(UNICDE)
      DO 2 I=1,3
      IF(ICODE.NE.ENTRY(1,I)) GO TO 2
      IF(UNICDE.GE.0) VALUE=VALUE*FACTOR(I)
      IF(UNICDE.LT.0) VALUE=VALUE/FACTOR(I)
      DO 1 J=1,2
      UNITS(J)=ENTRY(J+1,I)
    1 CONTINUE
C
C     AT THIS POINT, CONVERSION IS COMPLETE.
C
      RETURN
C
C     ELSE, CHECK REST OF TABLE.
C
    2 CONTINUE
C
C     IF CODE NOT FOUND, RETURN WITH 'UNITS ERROR' IN 'UNITS'.
C
```

```
      UNITS(1)='UNITS '
      UNITS(2)='ERROR '
      PRINT 101,ICODE
101   FORMAT(' ERROR--UNIT CODE',I3,' NOT IN PER CENT TABLE.')
      RETURN
      END
```

```
@FOR,SW     N.PLANAR,.PLANAR

      SUBROUTINE PLANAR(VALUE,UNICDE,UNITS)
C
C     TABLE OF AREA MEASURE CONVERSION FACTORS.
C
      INTEGER ENTRY(3,9),UNITS(9),UNICDE
C
C     FACTORS ARE DOUBLE PRECISION.
C
      DOUBLE PRECISION FACTOR(9)
C
C     PUT UNIT CODES INTO ENTRY.
C
      DATA (ENTRY(1,I),I=1,9)/13,14,15,16/
C
C     PUT IN ALPHA UNITS.
      DATA ((ENTRY(I,J),I=1,2),J=1,9)/'CM**2 ','      ',
     *                                'METERS','**2   ',
     *                                'FEET**','2     ',
     *                                'YARDS**','2    '/
C
C     ENTER INTERNAL UNITS.
C
      UNITS(3)='METERS'
      UNITS(9)=' SQ.  '
C
C     IF UNICDE=0, THEN SET CODE TO STANDARD UNITS.
C
      IF(UNICDE.EQ.0) UNICDE=-14
C
C     SET CONVERSION FACTORS.
C
      DATA (FACTOR(I),I=1,9)/10000.0D0,1.0D0,7.6249D0,.836100/
C
C     INITIALIZE 'UNITS' TO ERROR MESSAGE IN CASE CODE NOT FOUND.
C
      UNITS(1)='UNITS '
      UNITS(2)='ERROR '
C
C     IF CODE IS IN TABLE, PERFORM CONVERSION. ALSO, ENTER UNITS ALPHA
C          CODE INTO 'UNITS'.
C
      ICODE=IABS(UNICDE)
      DO 2 I=1,9
      IF(ICODE.NE.ENTRY(1,I)) GO TO 2
      IF(UNICDE.GE.0) VALUE=VALUE*FACTOR(I)
      IF(UNICDE.LT.0) VALUE=VALUE/FACTOR(I)
      DO 1 J=1,2
      UNITS(J)=ENTRY(J+1,I)
    1 CONTINUE
C
C     AT THIS POINT, CONVERSION IS COMPLETE.
C
      RETURN
C
```

```
      C     ELSE, CHECK REST OF TABLE.
      C
    2 CONTINUE
      C
      C     IF CODE NOT FOUND, RETURN WITH 'UNITS ERROR' IN 'UNITS'.
      C
          UNITS(1)='UNITS '
          UNITS(2)='ERROR '
          PRINT 101,ICODE
  101 FORMAT(' ERROR--UNIT CODE',I3,' NOT IN AREA TABLE.')
          RETURN
          END
```

```
@FOR,Sa     N.POWORK,,POWORK

      SUBROUTINE POWORK(VALUE,UNICODE,UNITS)
C
C     TABLE OF POWER MEASURE CONVERSION FACTORS.
C
      INTEGER ENTRY(3,2),UNITS(4),UNICODE
C
C     FACTORS ARE DOUBLE PRECISION.
C
      DOUBLE PRECISION FACTOR(2)
C
C     PUT UNIT CODES INTO ENTRY.
C
      DATA (ENTRY(1,I),I=1,2)/59,60/
C
C     PUT IN ALPHA UNITS.
C
      DATA ((ENTRY(I,J),I=2,3),J=1,2)/'ERGS/S','EC.   ',
     .                '&ATTS ','      '/
C
C     ENTER INTERNAL UNITS.
C
      UNITS(3)='WATTS/'
      UNITS(4)='      '
C
C     IF UNICODE=0, THEN SET CODE TO STANDARD UNITS.
C
      IF(UNICODE.EQ.0) UNICODE=-60
C
C     SET CONVERSION FACTORS.
C
      DATA (FACTOR(I),I=1,2)/1.0D-7,100/
C
C     INITIALIZE 'UNITS' TO ERROR MESSAGE IN CASE CODE NOT FOUND.
C
      UNITS(1)='UNITS '
      UNITS(2)='ERROR '
C
C     IF CODE IS IN TABLE, PERFORM CONVERSION. ALSO, ENTER UNITS ALPHA
C          CODE INTO 'UNITS':
C
      ICODE=IABS(UNICODE)
      DO 2 I=1,2
      IF(ICODE.NE.ENTRY(1,I)) GO TO 2
      IF(UNICODE.GE.0) VALUE=VALUE*FACTOR(I)
      IF(UNICODE.LT.0) VALUE=VALUE/FACTOR(I)
      UNITS(1)=ENTRY(2,I)
      UNITS(2)=ENTRY(3,I)
C
C     AT THIS POINT, CONVERSION IS COMPLETE.
C
      RETURN
C
C     ELSE, CHECK REST OF TABLE.
C
```

```
      2 CONTINUE
C
C     IF CODE NOT FOUND, RETURN WITH 'UNITS ERROR' IN 'UNITS'.
C
      PRINT 101,ICODE
  101 FORMAT(' ERROR--UNIT CODE',I3,' NOT IN POWER TABLE.')
      RETURN
      END
```

```
@FOR,SN     N.PRESSR,,PRESSR

      1.          SUBROUTINE PRESSR(VALUE,UNICDE,UNITS)
      2.    C
      3.    C     TABLE OF PRESSR MEASURE CONVERSION FACTORS.
      4.    C
      5.          INTEGER ENTRY(2,9),UNITS(4),UNICDE
      6.    C
      7.    C     FACTORS ARE DOUBLE PRECISION.
      8.    C
      9.          DOUBLE PRECISION FACTOR(9)
     10.    C
     11.    C     PUT UNIT CODES INTO ENTRY.
     12.    C
     13.          DATA (ENTRY(1,I),I=1,9)/48,49,50,51,52,53,54,55,56/
     14.    C
     15.    C     PUT IN ALPHA UNITS.
     16.    C
     17.          DATA ((ENTRY(I,J),I=2,3),J=1,9)/'MICRO ','BARS  ',
     18.         *                'PASCALS','      ','MICRO ','PAS   ',
     19.         *                'KG./CM','**2   ','LBS./I','N.**2 ',
     20.         *                'LBS./F','I.**2 ','IN. ME','RCURY ',
     21.         *                'MM. ME','RCURY ','ATMOSP','HERES '/
     22.    C
     23.    C     ENTER INTERNAL UNITS.
     24.    C
     25.          UNITS(3)='KG/CM**'
     26.          UNITS(4)='2    '
     27.    C
     28.    C     IF UNICDE=0, THEN SET CODE TO STANDARD UNITS.
     29.    C
     30.          IF(UNICDE.EQ.0) UNICDE=-51
     31.    C
     32.    C     SET CONVERSION FACTORS.
     33.    C
     34.          DATA (FACTOR(I),I=1,9)/101.971D0,1.02D-3,1.0200,1D0,.703D0,
     35.         *                       9.882D-3,.345362D0,0.772212D0,
     36.         *                     1.033222887D0/
     37.    C
     38.    C     IF CODE IS IN TABLE, PERFORM CONVERSION. ALSO, ENTER UNITS ALPHA
     39.    C          CODE INTO 'UNITS'.
     40.    C
     41.          ICODE=IABS(UNICDE)
     42.          DO 2 I=1,9
     43.          IF(ICODE.NE.ENTRY(1,I)) GO TO 2
     44.          IF(UNICDE.GE.0) VALUE=VALUE*FACTOR(I)
     45.          IF(UNICDE.LT.0) VALUE=VALUE/FACTOR(I)
     46.          DO 1 J=1,2
     47.          UNITS(J)=ENTRY(J+1,I)
     48.        1 CONTINUE
     49.    C
     50.    C     AT THIS POINT, CONVERSION IS COMPLETE.
     51.    C
     52.          RETURN
     53.    C
     54.    C     ELSE, CHECK REST OF TABLE.
```

```
      C
    2 CONTINUE
      C
      C     IF CODE NOT FOUND, RETURN WITH 'UNITS ERROR' IN 'UNITS'.
      C
          UNITS(1)='UNITS '
          UNITS(2)='ERROR '
          PRINT 101, ICODE
  101 FORMAT(' ERROR--UNIT CODE',I3,' NOT IN PRESSURE TABLE.')
          RETURN
          END
```


BEST AVAILABLE COPY

```
      C
    2 CONTINUE
      C
      C     IF CODE NOT FOUND, RETURN WITH 'UNITS ERROR' IN 'UNITS'.
      C
          UNITS(1)='UNITS '
          UNITS(2)='ERROR '
          PRINT 101, ICODE
  101 FORMAT(' ERROR--UNIT CODE',I3,' NOT IN PRESSURE TABLE.')
          RETURN
          END
```

```
BFOR,SA     N.PRSLVL,.PRSLVL

      SUBROUTINE PRSLVL(VALUE,UNICDE,UNITS)
C
C     TABLE OF PRESSURE LEVEL MEASURE CONVERSION FACTORS.
C
      INTEGER ENTRY(3,3),UNITS(9),UNICDE
C
C     FACTORS ARE DOUBLE PRECISION.
C
      DOUBLE PRECISION FACTOR(3)
C
C     PUT UNIT CODES INTO ENTRY.
C
      DATA (ENTRY(1,I),I=1,3)/57,70,71/
C
C     PUT IN ALPHA UNITS.
C
      DATA ((ENTRY(I,J),I=2,3),J=1,3)/'DB//1U','BAR   ',
     *            '//.000','2D/CM2','DB//1U','PAS   '/
C
C     ENTER INTERNAL UNITS.
C
      UNITS(3)='DB//1U'
      UNITS(9)='PAS   '
C
C     IF UNICDE=0, THEN SET CODE TO STANDARD UNITS.
C
      IF(UNICDE.EQ.0) UNICDE=-71
C
C     SET CONVERSION FACTORS.
C
      DATA (FACTOR(I),I=1,3)/-100.0D0,-26.0D0,0D0/
C
C     INITIALIZE 'UNITS' TO ERROR MESSAGE IN CASE CODE NOT FOUND.
C
      UNITS(1)='UNITS '
      UNITS(2)='ERROR '
C
C     IF CODE IS IN TABLE, PERFORM CONVERSION. ALSO, ENTER UNITS ALPHA
C          CODE INTO 'UNITS'.
C
      ICODE=IABS(UNICDE)
      DO 2 I=1,3
      IF(ICODE.NE.ENTRY(1,I)) GO TO 2
      IF(UNICDE.GE.0) VALUE=VALUE+FACTOR(I)
      IF(UNICDE.LT.0) VALUE=VALUE-FACTOR(I)
      UNITS(1)=ENTRY(2,I)
      UNITS(2)=ENTRY(3,I)
C
C     AT THIS POINT, CONVERSION IS COMPLETE.
C
      RETURN
C
C     ELSE, CHECK REST OF TABLE.
C
```

```
      2 CONTINUE
C
C     IF CODE NOT FOUND, RETURN WITH 'UNITS ERROR' IN 'UNITS'.
C
      PRINT 101,ICODE
  101 FORMAT(' ERROR--UNIT CODE',I3,' NOT IN PRESSURE LEVEL TABLE.')
      RETURN
      END
```

```
GELT,L      N.RANGE,.RANGE

      SUBROUTINE RANGE (NTHNGS,THNGS,*,NMTHDS,MTHDS,MTHNGS)
C
      IMPLICIT INTEGER (A-Z)
C
C
      COMMON /CPV   /NCON,CON(9,30),NPAR,PAR(30,30),NVAR,IVAR(3,33),
     *               NDATS,NHOBS(50),NDATA,VALRNG(2,30,50),DATA(30000)
      DIMENSION ICON(9,30),IPAR(30,30),IVLRNG(2,30,50)
      EQUIVALENCE (CON(1,1),ICON(1,1)),(PAR(1,1),IPAR(1,1)),
     *            (VALRNG(1,1,1),IVLRNG(1,1,1))
      REAL CON,PAR,DATA,VALRNG
C
C
      DIMENSION THNGS(9,10),UNITS(9),PSUB(30),NMTHDS(10),MTHDS(10,10)
      DIMENSION ATHNGS(9,10),ITHNGS(9,10),RTHNGS(9,10)
      REAL ATHNGS,RTHNGS
      EQUIVALENCE (ATHNGS,ITHNGS),(THNGS,RTHNGS)
C
      COMMON /DATCOM/ NDUSED,NRS(50)
C
      IF(NTHNGS.NE.0) GO TO 3
      NDUSED=NDATS
      DO 2 I=1,NDATS
      NRS(I)=1
    2 CONTINUE
      RETURN
C
C          LOOP TO MOVE THNGS TO ATHNGS
C
    3 DO 10 I=1,NTHNGS
          DO 5 J=1,9
          ITHNGS(J,I)= THNGS(J,I)
    5     CONTINUE
      CALL CONVRT (ATHNGS(2,I),ITHNGS(1,I),ITHNGS(9,I),
     *             UNITS)
      CALL CONVRT (ATHNGS(3,I),ITHNGS(1,I),ITHNGS(9,I),
     *             UNITS)
   10 CONTINUE
C
      IF (NCON .EQ. 0) GO TO 125
C
          DO 100 J=1,NTHNGS
             NM = NMTHDS(J)
C
             DO 90 I=1,NCON
C
             IF (ICON(1,I) .NE. THNGS(1,J)) GO TO 90
C
              IF (RTHNGS(2,J) .EQ. -99999999.) GO TO 50
             IF (CON(3,I) .LT. ATHNGS(2,J)) RETURN 3
C
   50         IF (RTHNGS(3,J) .EQ. -99999999.) GO TO 60
             IF (CON(3,I) .GT. ATHNGS(3,J)) RETURN 3
C
```

```
      C                    CHECK METHODS
      C
   60       IF (NM.EQ.0) GO TO 90
            DO 70 L = 1,NM
               IF (MTHDS(L,J).EQ.ICON(4,I)) GO TO 90
   70       CONTINUE
      C
      C        METHODS DID NOT MATCH
            RETURN 3
      C
   90       CONTINUE
      C
  100    CONTINUE
      C
      C         INITIALIZE NDUSED
      C
  125 NDUSED = 0
      C
            DO 130 I=1,30
            PSUB(I) = 1
  130    CONTINUE
      C
      C
        IF (NDATS .EQ. 0) RETURN
      C
            DO 500 I=1,NDATS
      C
          IF (I .EQ. 1) GO TO 150
      C
            J = NPAR
  140    PSUB(J) = PSUB(J) + 1
            IF (PSUB(J) .LE. IPAR(1,J)) GO TO 150
            PSUB(J) = 1
            J = J -1
            GO TO 140
      C
      C                    LOOP FOR REQUESTS
      C
  150       DO 450 K=1,NTHNGS
            NM = NMTHDS(K)
      C
      C                        CHECK FOR PARAMETERS
      C
            IF (NPAR .EQ. 0) GO TO 300
      C
      C                        LOOP FOR PARAMETERS
      C
                DO 290 J=1,NPAR
      C
                IF (IPAR(2,J) .NE. ITHNGS(1,K)) GO TO 290
      C
                JJ = PSUB(J)+4
                IF (RTHNGS(2,K) .EQ. -99999999.) GO TO 200
                 IF (PAR(JJ,J) .LT. ATHNGS(2,K)) GO TO 500
      C
  200           IF (RTHNGS(3,K) .EQ. -99999999.) GO TO 210
                IF (PAR(JJ,J) .GT. ATHNGS(3,K)) GO TO 500
```

```
C
C                    CHECK METHODS
C
210                  IF (NM.EQ.0) GO TO 290
                     DO 220 L = 1,NM
                        IF (METHDS(L,K).EQ.IPAR(4,J)) GO TO 290
220                  CONTINUE
C
C                 METHODS DID NOT MATCH
                     GO TO 500
C
  290                CONTINUE
C
C                       LOOP FOR VARIABLES
C
  300                DO 900 J=1,NVAR
C
                     IF (IVAR(1,J) .NE. THNGS(1,K)) GO TO 900
C
                     IF (RTHNGS(2,K) .EQ. -99999999.) GO TO 350
                     IF (VALRNG(2,J,1) .LT. ATHNGS(2,K)) GO TO 500
C
  350                IF (RTHNGS(3,K) .EQ. -99999999.) GO TO 360
                     IF (VALRNG(1,J,1) .GT. ATHNGS(3,K)) GO TO 500
C
C                    CHECK METHODS
C
360                  IF (NM.EQ.0) GO TO 400
                     DO 370 L = 1,NM
                        IF (METHDS(L,K).EQ.IVAR(3,J)) GO TO 900
370                  CONTINUE
C
C                 METHODS DID NOT MATCH
                     GO TO 500
C
  900                CONTINUE
C
  950          CONTINUE
C
C                 INCREMENT COUNTER
C
  475      NDUSED = NDUSED + 1
           NRS(NDUSED) = I
C
  500      CONTINUE
C
       IF (NDUSED .EQ. 0) RETURN 3
       RETURN
C
       END
```

BEST AVAILABLE COPY

```
SELT.L      N.SELECT,.SELECT

      C
      C
      C
      C
            PARAMETER NACPT=36,NENPT=35,NACDT=90,NENDT=19
      C
            IMPLICIT INTEGER (A-Z)
      C
            LOGICAL LISTF,FIRST
      C
            INCLUDE RETREV.LIST
      C
            INCLUDE EXPHNT.LIST
      C
            INCLUDE CRUISE.LIST
      C
            INCLUDE ASTATN.LIST
      C
            INCLUDE RUN.LIST
      C
            INCLUDE CPV.LIST
      C
            INCLUDE SECTO.LIST
      C
            INCLUDE DATCOM.LIST
      C
            REAL *AVDIR,*AVMT,*AVPRD,SEASTA,*NDDIR,*NDVEL,SWLDIR,
           *     SWLMT,SWLPRD,*EATHR,DOTDPH,DTMSLP
      C
            DIMENSION ICON(9,30),IPAR(30,30),IVLRNG(2,30,50)
      C
            EQUIVALENCE (CON(1,1),ICON(1,1)),(PAR(1,1),IPAR(1,1)),
           *            (VALRNG(1,1,1),IVLRNG(1,1,1)),(AEQUIV,ACPANS),
           *            (EEQUIV,ENPANS)
      C
            REAL CON,PAR,DATA,VALRNG
      C
            REAL AEQUIV(9,10),EEQUIV(9,10),*ORD
      C
            DIMENSION IDTI(56),DISK(7),ASTATS(2),ESTATS(2)
      C
            DIMENSION IBFR(3000),CHAR(80)
      C
            DATA DISK /*   NAVSEA UNDER*ATER ACOUSTIC DATA BANK    */
      C
            DIMENSION ACPT(NACPT),ENPT(NENPT),ACDT(NACDT),ENDT(NENDT),
           1ACPOST(NACPT),ENPOST(NENPT),TMPDT(NENDT),TMPPT(NENPT),
           1TMPOST(NENPT)
      C
      C
            LISTF = .FALSE.
            FIRST = .TRUE.
      C
      C         INIT. UNIT 19 SO WE CAN USE SETADR WITH NTRAN
```

```
      C
            CALL NTRAN(10,22)
      C
      C        READ IN SECTOR ZERO
      C
            CALL DSKRED (0,1,ID)
      C
      C        MAKE SURE IT'S CORRECT DISK
      C
            DO 10 I=1,7
            IF (DISK(I) .NE. ID(I)) GO TO 15
         10 CONTINUE
            GO TO 18
      C
      C        WRITE ERROR
      C
         15 WRITE(6,16)
         16 FORMAT (' ERROR -- WRONG DISK MOUNTED')
            STOP
      C
         18 CALL ERTRAN ('DATE',DATE,TIME)
            FLD(C,12,DAY) = FLD(12,12,DATE)
            FLD(C,12,YEAR) = FLD(24,12,DATE)
            FLD(C,12,MINUTE) = FLD(12,12,TIME)
            FLD(C,12,SECOND) = FLD(24,12,TIME)
            IF (FIRST) WRITE (6,25) (ID(I),I=1,7),DATE,DAY,YEAR,TIME,MINUTE,
           1SECOND
         25 FORMAT ('1',T30,'NAVDAB'/T12,6A6,A3,')',T14,'('/T23,2(A2,
           1'/'),A2,4X,2(A2,':'),A2///1X,'TO RETURN TO PREVIOUS QUESTION, ENTE
           1R WLOF'/)
      C
         20 PNXTE = 262145      @ OCTAL 000001000001
      C
      C        READ RETRIEVAL DATA
      C
         26 CALL INPUT (FIRST)
            FIRST = .FALSE.
      C
            CALL STORE
      C
      C        INITIALIZE INVENTORY COUNTERS
      C
            DO 31 I = 1,NACPT
            ACPDST(I) = 0
         31 ACPT(I) = 0
            DO 32 I = 1,NENPT
            ENPDST(I) = 0
         32 ENPT(I) = 0
            DO 33 I = 1,NACDT
         33 ACDT(I) = 0
            DO 34 I = 1,NENDT
         34 ENDT(I) = 0
      C
            DATYPE=0       @ SET ACOUSTIC FLAG
      C
            IF (RUNTYP.EQ.2) DATYPE = 1
      C
```

BEST AVAILABLE COPY

```
C          INIT NCONTA, NCONTE
C
      NCONTE = 0
      NCONTA = 0
C
      IF (ANY.EQ.0) GO TO 55
      ASTATS(1) = 0
      ASTATS(2) = 0
C
      IF (NACPAR .EQ. 0) GO TO 40
C
      DO 35 I=1,NACPAR
      ICOD = IABS(IACPARS(1,I) -1)
      ISUB =1
      IF (ICOD .LE. 35) GO TO 30
      ISUB = 2
      ICOD = IABS(ICOD -36)
C
      IF (ICOD .LT. 36) GO TO 30
      WRITE (6,28) ACPARS(1,I)
   28 FORMAT (' ILLEGAL ACOUSTIC CODE',I10)
      GO TO 26
   30 FLD(ICOD,1,ASTATS(ISUB)) = 1
   35 CONTINUE
C
   40 ESTATS(1) =0
      ESTATS(2) =0
C
      IF (NENPAR .EQ. 0) GO TO 55
C
      DO 50 I=1,NENPAR
      ICOD = IABS(ENPARS(1,I) -101)
      ISUB = 1
      IF (ICOD .LE. 35) GO TO 45
      ISUB = 2
      ICOD = IABS(ICOD -36)
C
      IF (ICOD .LT. 36) GO TO 45
      WRITE (6,43) ENPARS(1,I)
   43 FORMAT (' ILLEGAL ENVIRONMENTAL CODE',I10)
      GO TO 26
   45 FLD(ICOD,1,ESTATS(ISUB)) = 1
   50 CONTINUE
C
   55 CALL NTRAN(1,10)
C
      CALL NTRAN(1,1,422,NLAT,L)
      CALL NTRAN(1,22)
C
      IF (L .GT. 0) GO TO 65
C
C          WRITE ERROR
C
      WRITE (6,60) L
   60 FORMAT (' ERROR -- WRITING /RETREV/ TO TEMP FILE',I10)
      STOP
C
```

```
      65 FLD(18,18,SECADR) = FLD( 0,18,PNXTE)
         FLD(18,18,NSEC)   = FLD(18,18,PNXTE)
C
         CALL DSKRED (SECADR,NSEC,DATA)
C
         CALL UNPKXP (DATA)
C
C           CHECK AREA, DATE AND STATUS BITS FOR EXPERIMENT LEVEL RECORD
C
         IF (EXNO.NE.EXP.AND.EXP.NE.-99999999) GO TO 70
         IF (ANY.EQ.0) GO TO 80
C
C            CHECK FOR ENVIRONMENTAL ONLY
C
         IF (DATYPE.EQ.1) GO TO 67
C
         CALL AREACK(NLAT,ELON,SLAT,WLON,EAILAT,EAILON,EAFLAT,EAFLON,$70)
         CALL DATECK(EAIDAT,EAFDAT,0,0,$70)
         CALL STATCK(ASTATS,EASTAT,$70)
         GO TO 69
C
      67 CALL AREACK (NLAT,ELON,SLAT,WLON,EEILAT,EEILON,EEFLAT,EEFLON,$70)
         CALL DATECK(EEIDAT,EEFDAT,0,0,$70)
C
      69 CALL STATCK(ESTATS,EESTAT,$70)
C
C
C           CONTINUE ON
C
         GO TO 80
C
      70 IF (PNXTE .EQ. 0) GO TO 200       @ FINISHED PASS THRU DATA BANK
C
         GO TO 65
C
C
C
C
      80 PNXTC = PCHU
C
      85 FLD(18,18,SECADR) = FLD( 0,18,PNXTC)
         FLD(18,18,NSEC)   = FLD(18,18,PNXTC)
C
         CALL DSKRED (SECADR,NSEC,DATA)
C
C           UNPACK IT
C
         CALL UNPKCR (DATA)
C
C
         IF (CRUNO.NE.CRU.AND.CHU.NE.-99999999) GO TO 90
         IF (ANY.EQ.0) GO TO 100
C            CHECK FOR ENVIRONMENTAL ONLY
C
         IF (DATYPE.EQ.1) GO TO 87
         CALL AREACK(NLAT,ELON,SLAT,WLON,CAILAT,CAILON,CAFLAT,CAFLON,$90)
         CALL DATECK(CAIDAT,CAFDAT,0,0,$90)
         CALL STATCK(ASTATS,CASTAT,$90)
```

```
226.            GO TO 88
227.      C
228.         87 CALL AREACK(NLAT,ELON,SLAT,WLON,CEILAT,CEILON,CEFLAT,CEFLON,$90)
229.            CALL DATECK(CEIDAT,CEFDAT,0,0,$90)
230.      C
231.         88 CALL STATCK(ESTATS,CESTAT,$90)
232.      C
233.      C
234.      C           CONTINUE ON
235.      C
236.            GO TO 100
237.      C
238.         90 IF (PNXTC .EQ. 0) GO TO 70   @ FINISHED WITH CRUISE LEVEL
239.      C
240.            GO TO 85
241.      C
242.      C
243.      C
244.        100 PNXTAS = PACS
245.      C
246.      C           CHECK FOR ENVIRONMENTAL ONLY
247.      C
248.            IF (DATYPE.EQ.0) GO TO 105
249.            PNXTAS=PENS
250.            ASTATS(1)=ESTATS(1)
251.            ASTATS(2)=ESTATS(2)
252.            NACDCR=NENDCR
253.            NACPAR=NENPAR
254.      C
255.            DO 101 I=1,10
256.            ACDCRS(I) = ENDCRS(I)
257.            NAMTHD(I) = NEMTHD(I)
258.            DO 101 J = 1,10
259.            AMTHDS(I,J) = EMTHDS(I,J)
260.        101 CONTINUE
261.      C
262.            DO 102 I=1,9
263.            DO 102 J=1,10
264.        102 ACPARS(I,J)=ENPARS(I,J)
265.      C
266.        105 FLD(18,18,SECADR) = FLD( 0,18,PNXTAS)
267.            FLD(18,18,NSEC)   = FLD(18,18,PNXTAS)
268.            TEMP = PNXTAS
269.      C
270.            CALL DSKNED (SECADR,NSEC,IBFR)
271.      C
272.      C           UNPACK IT
273.      C
274.            CALL ASUNPK (IBFR,TYPE)
275.      C
276.            IF (SANO.NE.STA.AND.STA.NE.-99999999) GO TO 110
277.            IF (ANY.EQ.0) GO TO 120
278.            CALL AREACK(PLAT,ELON,SLAT,WLON,SAILAT,SAILON,SAFLAT,SAFLON,$110)
279.            CALL DATECK(SAIDAT,SAFDAT,0,0,$110)
280.            CALL STATCK(ASTATS,SASTAT,$110)
281.      C
282.      C
```

```
C          CONTINUE ON
C
      GO TO 120
C
  110 IF (PNXTAS .NE. 0) GO TO 105       @ READ NEXT ACOUSTIC STATION
C
      GO TO 90
C
  120 PNEXTS = PNXTAS
C
C
      EFLAG = 0
C
C          CHECK IF ACOUSTIC OR ENVIRONMENTAL ONLY
C
      IF (RUNTYP.NE.3) GO TO 155
C
C
      FLD(18,18,SECADR) = FLD( 0,18,PENVS)
      FLD(18,18,NSEC)   = FLD(18,18,PENVS)
      TEMP = PENVS
C
      CALL DSKRED (SECADR,NSEC,IBFR)
C
C          UNPACK IT
C
      CALL ASUNPK (IBFR,TYPE)
C
      IF (ANY.EQ.0) GO TO 135
      CALL STATCK(ESTATS,SASTAT,$130)
      GO TO 135
C
  130 PNXTAS = PNEXTS
      GO TO 110
C
C               REWIND 12 & INIT # ENV RUNS COUNTER
C
  135 CALL NTRAN (12,1C)
      EFLAG = 0
      IPTR = 1
C
C               INITIALIZE TEMPORARY ENV. COUNTERS
C
      DO 133 I = 1,NENDT
  133 TMPDT(I) = 0
      DO 134 I = 1,NENPT
      TMPDST(I) = 0
  134 TMPPT(I) = 0
C
C               LOOP TO EXAMINE RUNS
C
      DO 150 I=1,SANRUN
C
      OPTR = IPTR
C
      CALL AUNPCK (IBFR,IPTR)
C
```

```
      IF (RUNNO.NE.RUN.AND.RUN.NE.-99999999) GO TO 150
      CALL RANGE (NENPAR,ENPARS,$150,NEMTHD,EMTHDS,EQUIV)
      IF (ANY.EQ.0) GO TO 190
      CALL STATCK (ESTATS,NSTAT,$150)
C
C           CHECK ENV. MODES
C
      IF (NENDCR .EQ. 0) GO TO 190
      DO 136 J=1,NENDCR
      IF (ENDCRS(J) .EQ. DESCR) GO TO 190
  136 CONTINUE
      GO TO 150
C
  190 DO 191 J = 1,NOSEA
      IF (SEX(J).EQ.0) GO TO 192
      IF (CLAS.EQ.SEA(J)) GO TO 192
  191 CONTINUE
      GO TO 150
C
C                   WRITE DATA TO 12
C
  192 IOT(1) = SANO
      IOT(2) = TEMP
      IOT(3) = OPTR
      IOT(4) = NOUSED
      DO 193 J=1,50
      IOT(J+4) = NRS(J)
  193 CONTINUE
C
      CALL NTRAN (12,1,54,IOT(1),L)
      CALL NTRAN (12,22)
      IF (L .EQ. 54) GO TO 197
C
C           ERROR
C
      WRITE (6,199) L
  199 FORMAT (' ERROR-- WRITE TO 12',I6)
      STOP
C
C
C                       INCREMENT COUNTER
C
  197 EFLAG = EFLAG +1
C
      TMPDT(DESCR) = TMPDT(DESCR) + 1

      IF (NCON.EQ.0) GO TO 1470
      DO 1460 J = 1,NCON
      INDEX = ICON(1,J) - 100
 1460 TMPPT(INDEX) = TMPPT(INDEX) + 1
 1470 IF (NPAR.EQ.0) GO TO 1490
      DO 1480 J = 1,NPAR
      INDEX = IPAR(2,J) - 100
      TMPPT(INDEX) = TMPPT(INDEX) + 1
 1480 TMPOST(INDEX) = TMPOST(INDEX) + NOUSED
 1490 IF (NVAR.EQ.0) GO TO 150
      DO 1495 J = 1,NVAR
```

```
            INDEX = IVAN(I,JI) - ICO
            IMPPT(INDEX) = IMPPT(INDEX) + 1
      1495  IMPDST(INDEX) = IMPDST(INDEX) + NOUSED
C
  150 CONTINUE
C
C                       CHECK FOR NONE FOUND
C
            IF (EFLAG .EQ. 0) GO TO 130  @ GO GET NEXT ACOUSTIC STATION
C
C                   READ BACK IN ACOUSTIC RUN
            FLD(10,18,SECADR) = FLD( 0,18,PENVS)
            FLD(10,18,NSEC)   = FLD(10,18,PENVS)
            TEMP = PENVS
C
            CALL DSKRED (SECADR,NSEC,IBFR)
C
C                   UNPACK IT
C
            CALL ASUNPK (IBFR,TYPE)
C
C                   SET COUNTER AND BUFFER POINTER
C
  155 AFLAG = 0
            IPTR = 11
C
C                   INIT FOR THIS STATION OUTPUT TO 11
C
            IOT1(1) = EANO
            IOT1(2) = CRUNO
            IOT1(3) = SANO
            IOT1(9) = TEMP
C
C                   LOOP FOR PROCESSING ACOUSTIC RUNS
C
            DO 170 I=1,SANRUN
C
C                       SET OLD PTR = NEW
C
            OPTR = IPTR
C
C                       UNPACK RUN
C
            CALL AUNPCK (IBFR,IPTR)
C
            IF (RUNNO.NE.RUN.AND.RUN.NE.-99999999) GO TO 170
            CALL RANGE (NACPAR,ACPARS,S170,NAMTHD,AMTHDS,AEQUIV)
            IF (ANY.EQ.0) GO TO 161
            TILAT = MAX (RILAT,RFLAT)
            TFLAT = MIN (RILAT,RFLAT)
            TILON = EAST (RILON,RFLON)
            TFLON = WEST (RILON,RFLON)
            CALL AREACK(NLAT,ELON,SLAT,WLON,TILAT,TILON,TFLAT,TFLON,S170)
            CALL DATECK(RIDAY,RFDAY,RITIM,RFTIM,S170)
            CALL STATCK(ASTATS,RSTAT,S170)
C
C     PRINT 159,DESCR,(ACOCRST(J),J=1,NACOCR)
```

```
C 159 FORMAT(8I10)
      IF (NACDCN .EQ. 0) GO TO 156
      DO 160 J=1,NACDCN
      IF (ACDCNS(J) .EQ. DESCN) GO TO 156
  163 CONTINUE
      GO TO 170
C
C              SKIP SOURCE/RECEIVER CHECKS IF ENVIRONMENTAL
C
  156 IF (DATYPE.EQ.1) GO TO 161
C
C          CHECK SOURCE TYPE
C
      IF (NSTYPE.EQ.0) GO TO 158
      DO 157 J = 1,NSTYPE
      IF (STYPE(J).EQ.SRCTYP) GO TO 158
  157 CONTINUE
      GO TO 170
C
C          CHECK RECEIVER TYPE
C
  158 IF (NRTYPE.EQ.0) GO TO 164
      DO 163 J = 1,NRTYPE
      IF (RTYPE(J).EQ.RCRTYP) GO TO 164
  163 CONTINUE
      GO TO 170
C
C          CHECK SOURCE SYSTEM
C
  164 IF (NSSYST.EQ.0) GO TO 168
      DO 166 J = 1,NSSYST
      IF (SSYST(J).EQ.SYSSRC) GO TO 166
  166 CONTINUE
      GO TO 170
C
C          CHECK RECEIVER SYSTEM
C
  168 IF (NRSYST.EQ.0) GO TO 161
      DO 171 J = 1,NRSYST
      IF (RSYST(J).EQ.SYSRCR) GO TO 161
  171 CONTINUE
      GO TO 170
C
C          CHECK FOR SECURITY
C
  161 DO 167 J = 1,NUSEX
      IF (SEX(J).EQ.0) GO TO 169
      IF (CLAS.EQ.SEX(J)) GO TO 169
  167 CONTINUE
      GO TO 170
C
  169 IOTI(5) = OPTR
      IOTI(6) = NOUSEO
      DO 1610 J=1,50
      IOTI(J+6) = NRS(J)
 1610 CONTINUE
C
```

```
C
      CALL NTRAN (11,1,56,IOT11,L)
      CALL NTRAN (11,22)
      IF (L .EQ. 56) GO TO 165
      WRITE (6,162) L,I,SANRUN
  162 FORMAT (' ERROR IN ACOUSTIC WRITE TO 11',3110)
      STOP
C
C          INCREMENT ACOUSTIC OR ENV. ONLY COUNTER
C
  165 AFLAG = AFLAG +1
C
C          INCREMENT DESCRIPTON AND PARAMETER TABLE COUNTERS
C
C               BRANCH IF ENV. ONLY
C
      IF (DATYPE.EQ.1) GO TO 1750
C
      ACDT(DESCR) = ACDT(DESCR) + 1

      IF (NCON.EQ.0) GO TO 1670
      DO 1660 J = 1,NCON
      INDEX = ICON(1,J)
 1660 ACPT(INDEX) = ACPT(INDEX) + 1
 1670 IF (NPAR.EQ.0) GO TO 1690
      DO 1680 J = 1,NPAR
      INDEX = IPAR(2,J)
      ACPT(INDEX) = ACPT(INDEX) + 1
 1680 ACPDST(INDEX) = ACPDST(INDEX) + NDUSED
 1690 IF (NVAR.EQ.0) GO TO 170
      DO 1695 J = 1,NVAR
      INDEX = IVAR(1,J)
      ACPT(INDEX) = ACPT(INDEX) + 1
 1695 ACPDST(INDEX) = ACPDST(INDEX) + NDUSED
      GO TO 170
C
C          INCREMENT ENV. ONLY COUNTERS
C
 1750 ENDT(DESCR) = ENDT(DESCR) + 1

      IF (NCON.EQ.0) GO TO 1770
      DO 1760 J = 1,NCON
      INDEX = ICON(1,J) - 100
 1760 ENPT(INDEX) = ENPT(INDEX) + 1
 1770 IF (NPAR.EQ.0) GO TO 1790
      DO 1780 J = 1,NPAR
      INDEX = IPAR(2,J) - 100
      ENPT(INDEX) = ENPT(INDEX) + 1
 1780 ENPDST(INDEX) = ENPDST(INDEX) + NDUSED
 1790 IF (NVAR.EQ.0) GO TO 170
      DO 1795 J = 1,NVAR
      INDEX = IVAR(1,J) - 100
      ENPT(INDEX) = ENPT(INDEX) + 1
 1795 ENPDST(INDEX) = ENPDST(INDEX) + NDUSED
C
  170 CONTINUE
C
```

```
C
C                CHECK FOR NO ACOUSTIC RUNS FOUND
C
      IF (AFLAG .EQ. 0) GO TO 110
C
C                INCREMENT ENV. COUNTERS FROM TEMPORARY COUNTERS
C
      IF (RUNTYP.NE.3) GO TO 1698
      DO 1696 J = 1,NENDT
1696  ENDT(J) = ENDT(J) + TMPDT(J)
      DO 1697 J = 1,NEMPT
      ENPT(J) = ENPT(J) + TMPPT(J)
1697  ENPOST(J) = ENPOST(J) + TMPOST(J)
C
C          REWIND 12
C
1698  CALL NTRAN (12,10)
C
C          CHECK EFLAG
C
      IF (EFLAG .EQ. 0) GO TO 190
C
C          LOOP TO MOVE ENV RUN POINTERS TO 11
C
      DO 180 I=1,EFLAG
C
C                READ FROM 12
C
      CALL NTRAN (12,2,54,IOT(1,(3),L)
      CALL NTRAN (12,22)
      IF (L .EQ. 54) GO TO 179
      STOP ERR 10
C
C                WRITE TO 11
C
179   CALL NTRAN (11,1,56,IOT(1,L)
      CALL NTRAN (11,22)
      IF (L .EQ. 56) GO TO 180
      WRITE (6,175) L,I,EFLAG
175   FORMAT (' ERROR IN ENV WRITE TO 11',3(10)'
      STOP
C
180   CONTINUE
C
C
190   NCONTA = NCONTA + AFLAG
      NCONTE = NCONTE + EFLAG
      GO TO 110
C
C
C          CHECK FOR ENVIRONMENTAL ONLY
C
C
200   IF (DATYPE.EQ.0) GO TO 203
      TEMP=NCONTA
      NCONTA=NCONTE
      NCONTE=TEMP
C
```

```
      C           PRINT RESULTS
      C
        203 WRITE (6,201) NCONTA,NCONTE
      201   FORMAT (//'THERE ARE:',I5,' ACOUSTIC RUN(S)',/,T9,'AND',I5,' ENVI
           *RONMENTAL RUN(S) WHICH MEET SELECTION CRITERIA')
      C
      C           UPDATE COMMON RETNEV ON DISK...
      C
            CALL NTRAN (11,10)
            NPTRS = NCONTA + NCONTE
            CALL NTRAN (11,1,422,NLAT,L)
            CALL NTRAN (11,22)
            IF (L .EQ. 422) GO TO 209
            WRITE (6,331) L
        331 FORMAT (' ERROR WRITING NPTRS TO [1',I10)
            STOP
      C
      C           DISPLAY MENU OPTIONS
      C
      C  IF SUMMARY DATA LISTING APPLIC. NUMBER IS CHANGED, CALL SHORT STATEMENT IN
      C  DATLST MUST ALSO BE CHANGED.
      209   IF (LISTF) GO TO 208
            LISTF = .TRUE.
        213 WRITE (6,205)
        205 FORMAT (//,T10,'NAVDAB APPLICATION OPTIONS',//,
           *       ' 1  - RESPECIFY SELECTION CRITERIA',/,
           *       ' 2  - INVENTORY OF RUNS SATISFYING SELECTION CRITERIA'/
           *       ' 3  - PRINT THE NOTES'/
           *       ' 4  - SUMMARY DATA LISTING BY RUN'/
           *       ' 5  - COMPLETE DATA LISTING BY RUN'/
           *       ' 6  - ACOUSTIC DESCRIPTOR INVENTORY SATISFYING SELECTION'
           *         /
           *       ' 7  - ENVIRONMENTAL DESCRIPTOR INVENTORY SATISFYING SELE'
           *         'CTION'/
           *       ' 8  - ACOUSTIC PARAMETER INVENTORY SATISFYING SELECTION'/
           *       ' 9  - ENVIRONMENTAL PARAMETER INVENTORY SATISFYING SELEC'
           *         'TION'/
           *       ' 10 - SINGLE UNIT CONVERSIONS',/,
           *       ' 11 - EXECUTE USER APPLICATION PROGRAM',/)
      C
      C           READ INPUT
      C
      C
        208 WRITE (6,209)
      209   FORMAT (//' ENTER APPLICATION OPTION NUMBER / ENTER ''LIST'' / ENT
           *ER ''DONE'''')
      202   READ (5,211,END=208,ERR=210) (CHAR(I),I=1,80)
      211   FORMAT (80A1)
            WRITE (6,207) (CHAR(I),I=1,80)
      207   FORMAT (3X,80A1)
            CALL FREEFD (CHAR,WORD,APPL,NWORDS,1,$210)
            IF (NWORDS.EQ.0) GO TO 210
            W = ' '
            FLD(3,6,W) = FLD(3,6,APPL)
            IF (W.EQ.'L') GO TO 213
            IF (W.EQ.'D') GO TO 340
            GO TO 250
```

BEST AVAILABLE COPY

```
C
C              ERROR ON READ
C
210   WRITE (6,215)
215   FORMAT (' *** READ FORMAT ERROR ***')
      GO TO 208
220   WRITE (6,221) APPL
221   FORMAT (' *** ENTERED NUMBER',I5,' OUT OF RANGE ***')
      GO TO 208
C
C
250   IF(APPL.LE.0.OR.APPL.GT.11) GO TO 220
      GO TO (18,280,290,300,310,320,340,370,380,320,330),APPL
C
C
C          INVENTORY PROGRAM
C
280   CALL INVNTT
      GO TO 208
C
C          NOTES PROGRAM
C
290   CALL NOTE (DATA)
      GO TO 208
C
C          LIST PROGRAM
C
300   CALL DATLST (APPL,CHAR,RUNTYP)
      GO TO 208
C
C          UNIT CONVERSION
C
320 CALL CVTPRO
      GO TO 208
C
C          EXECUTE USER PGM
C
330 WRITE (6,333)
333 FORMAT (T20,'ENTER @XQT USERS PROGRAM')
      STOP
C
C          DONE
C
340 WRITE (6,341)
341 FORMAT (T14,'NAVDAB RETRIEVAL SYSTEM -- SESSION COMPLETE')
      STOP
C
C          ACOUSTIC DESCRIPTOR TABLE
C
350   WRITE(6,355)
355   FORMAT ('1','ACOUSTIC DESCRIPTOR INVENTORY SATISFYING SELECTI'
     1'ON'/)
      CALL DESCTB (ACDT,NACDT,8)
      GO TO 208
C
C          ENVIRONMENTAL DESCRIPTOR TABLE
C
```

```
      WRITE (6,365)
365   FORMAT ('1','ENVIRONMENTAL DESCRIPTON INVENTORY SATISFYING SEL'
     1'ECTION'/)
      CALL DESCTB (ENDT,NENDT,7)
      GO TO 208
C
C          ACOUSTIC PARAMETER TABLE
C
370   WRITE (6,375)
375   FORMAT ('1','ACOUSTIC PARAMETER INVENTORY SATISFYING SELECTIO'
     1'N'/)
      CALL PARATB (ACPT,ACPDST,NACPT,2)
      GO TO 208
C
C          ENVIRONMENTAL PARAMETER TABLE
C
380   WRITE (6,385)
385   FORMAT ('1','ENVIRONMENTAL PARAMETER INVENTORY SATISFYING SELE'
     1'CTION'/)
      CALL PARATB (ENPT,ENPDST,NENPT,6)
      GO TO 208
C
      END
```

BEST AVAILABLE COPY

```
@FOR,S* N.SHORT,.SHORT

      SUBROUTINE SHORT (NRS,DATYPE,IVLRNG,UNITS,NAME,$)
C
C  SHORT PROVIDES A SUMMARY DATA LISTING OF THE RUNS MEETING THE USER'S
C  REQUIREMENTS.  THIS IS IDENTICAL TO THE LONG LISTING EXCEPT THE RANGE OF
C  THE VARIABLES ARE PRINTED INSTEAD OF THE ACTUAL VALUES.  LONG RUN LISTING
C  STATEMENTS ARE BYPASSED BY USING RETURN 6.
C
      IMPLICIT INTEGER (A-Z)
      REAL VALRNG,VALMIN,VALMAX
      DIMENSION IVLRNG(2,30,50),UNITS(1),NAME(1)
      COMMON /CPV   /NCON,CON(4,30),NPAR,PAR(30,30),NVAR,IVAR(3,30),
     .              NDATS,NRUNS(50),NDATA,VALRNG(2,30,50),DATA(30000)
C
C           PRINT HEADING FOR VARIABLES
C
      IF (DATYPE.NE.3) GO TO 10
      WRITE (6,5)
    5 FORMAT (/T25,'MINIMUM     MAXIMUM',T60,'ACOUSTIC')
      GO TO 20
   10 WRITE (6,15)
   15 FORMAT (/T25,'MINIMUM     MAXIMUM',T50,'ENVIRONMENTAL')
   20 WRITE (6,30)
   30 FORMAT (T15,'NAME',T26,'VALUE',T37,'VALUE',T48,'UNITS',T61,
     1'METHOD'/)
C
C           LOOP THROUGH VARIABLES FOR DATA SET NRS
C
      DO 50 I = 1,NVAR
      CALL NAMGET (IVAR(1,I),NAME)
      VALMIN = VALRNG(1,I,NRS)
      VALMAX = VALRNG(2,I,NRS)
      CALL CHECK (VALMIN,IVAR(1,I),UNITS)
      CALL CHECK (VALMAX,IVAR(1,I),UNITS)
      WRITE (6,40) (NAME(J),J=1,2),VALMIN,VALMAX,UNITS(1),UNITS(2),
     1IVAR(3,I)
   40 FORMAT (T11,2A6,G12.5,T35,G11.5,T47,2A6,2X,I4)
   50 CONTINUE
      WRITE (6,60)
   60 FORMAT (///)
      RETURN 6
      END
```

```
@F04,SF     N.SHRTME,.SHRTME

      1.          SUBROUTINE SHRTME(VALUE,UNICDE,UNITS)
      2.    C
      3.    C     TABLE OF SHRTME MEASURE CONVERSION FACTORS.
      4.    C
      5.          INTEGER ENTRY(3,5),UNITS(4),UNICDE
      6.    C
      7.    C     FACTORS ARE DOUBLE PRECISION.
      8.    C
      9.          DOUBLE PRECISION FACTOR(5)
     10.    C
     11.    C     PUT UNIT CODES INTO ENTRY.
     12.    C
     13.          DATA (ENTRY(I,1),I=1,5)/21,22,23,24,25/
     14.    C
     15.    C     PUT IN ALPHA UNITS.
     16.    C
     17.          DATA ((ENTRY(I,J),I=2,3),J=1,6)/'MILLIS','ECONDS',
     18.         .                'SECOND','S    ','MINUTE','S    ',
     19.         .                'HOURS ','     ','DAYS  ','     '/
     20.    C
     21.    C     ENTER INTERNAL UNITS.
     22.    C
     23.          UNITS(3)='SEC.  '
     24.          UNITS(4)='      '
     25.    C
     26.    C     IF UNICDE=0, THEN SET CODE TO STANDARD UNITS.
     27.    C
     28.          IF(UNICDE.EQ.0) UNICDE=-22
     29.    C
     30.    C     SET CONVERSION FACTORS.
     31.    C
     32.          DATA (FACTOR(I),I=1,5)/.001D0,1.0D0,60.D0,3.6D3,8.64D4/
     33.    C
     34.    C     IF CODE IS IN TABLE, PERFORM CONVERSION. ALSO, ENTER UNITS ALPHA
     35.    C          CODE INTO 'UNITS'.
     36.    C
     37.          ICODE=IABS(UNICDE)
     38.          DO 2 I=1,5
     39.          IF(ICODE.NE.ENTRY(1,I)) GO TO 2
     40.          IF(UNICDE.GE.0) VALUE=VALUE*FACTOR(I)
     41.          IF(UNICDE.LT.0) VALUE=VALUE/FACTOR(I)
     42.          DO 1 J=1,2
     43.          UNITS(J)=ENTRY(J+1,I)
     44.        1 CONTINUE
     45.    C
     46.    C     AT THIS POINT, CONVERSION IS COMPLETE.
     47.    C
     48.          RETURN
     49.        2 CONTINUE
     50.    C
     51.    C     IF CODE NOT FOUND, RETURN WITH 'UNITS ERROR' IN 'UNITS'.
     52.    C
     53.          UNITS(1)='UNITS '
     54.          UNITS(2)='ERROR '
```

```
      PRINT 101,ICODE
  101 FORMAT(' ERROR--UNIT CODE',I3,' NOT IN SHORT TIME TABLE.')
      RETURN
      END
```

```
@FOR,S  N.SORGET,,SONGET

      SUBROUTINE SORGET(VALUE,UNICDE,UNITS)
C
C     TABLE OF SOURCE LEVEL/TARGET STRENGTH CONVERSION FACTORS.
C
      INTEGER ENTRY(3,3),UNITS(4),UNICDE
C
C     FACTORS ARE DOUBLE PRECISION.
C
      DOUBLE PRECISION FACTOR(3)
C
C     PUT UNIT CODES INTO ENTRY.
C
      DATA (ENTRY(I,1),I=1,3)/69,65,91/
C
C     PUT IN ALPHA UNITS.
C
      DATA ((ENTRY(I,J),J=2,3),J=1,3)/'DB//1U','BAR@1M',
     *            'DB//1U','BAR@1Y','DB//1U','PAS@1M'/
C
C     ENTER INTERNAL UNITS.
C
      UNITS(3)='DB//1U'
      UNITS(4)='PAS@1M'
C
C
C          CODE TO PACK DB//1ERG/CM2@1 YD.
C
      IF (UNICDE .NE. 96) GO TO 5
      FLD(18,18,VALUE) = VALUE
      FLD( 0,18,VALUE) = -UNICDE
      UNITS(1) = '//ERG/'
      UNITS(2) = 'CM2@YD'
      RETURN
C
C          CODE TO UNPACK DB//1ERG/CM2@1 YD.
C
    5 IF (UNICDE .NE. -96) GO TO 6
      UNICDE = FLD( 3,18,VALUE)
      VALUE = FLD (18,18,VALUE)
      UNITS(1) = '//ERG/'
      UNITS(2) = 'CM2@YD'
      RETURN
C
C     IF UNICDE=0, THEN SET CODE TO STANDARD UNITS.
C
    6 IF(UNICDE.EQ.0) UNICDE=-91
C
C     SET CONVERSION FACTORS.
C
      DATA (FACTOR(I),I=1,3)/-130.000,-99.200,000/
C
C     INITIALIZE 'UNITS' TO ERROR MESSAGE IN CASE CODE NOT FOUND.
C
      UNITS(1)='UNITS '
```

```
      UNITS(2)='ERROR '
C
C     IF CODE IS IN TABLE, PERFORM CONVERSION. ALSO, ENTER UNITS ALPHA
C          CODE INTO 'UNITS'.
C
      ICODE=IABS(UNICDE)
      DO 2 I=1,3
      IF(ICODE.NE.ENTRY(1,I)) GO TO 2
      IF(UNICDE.GE.0) VALUE=VALUE*FACTOR(I)
      IF(UNICDE.LT.0) VALUE=VALUE-FACTOR(I)
      UNITS(1)=ENTRY(2,I)
      UNITS(2)=ENTRY(3,I)
C
C     AT THIS POINT, CONVERSION IS COMPLETE.
C
      RETURN
C
C     ELSE, CHECK REST OF TABLE.
C
    2 CONTINUE
C
C     IF CODE NOT FOUND, RETURN WITH 'UNITS ERROR' IN 'UNITS'.
C
      PRINT 101,ICODE
  101 FORMAT(' ERROR--UNIT CODE',I3,' NOT IN SOURCE/TARGET TABLE.')
      RETURN
      END
```

```
@FOR,S) N.SPEEDS,.SPEEDS

      SUBROUTINE SPEEDS(VALUE,UNICDE,UNITS)
C
C     TABLE OF SPEEDS MEASURE CONVERSION FACTORS.
C
      INTEGER ENTRY(3,8),UNITS(4),UNICDE
C
C     FACTORS ARE DOUBLE PRECISION.
C
      DOUBLE PRECISION FACTOR(8)
C
C     PUT UNIT CODES INTO ENTRY.
C
      DATA (ENTRY(1,I),I=1,8)/28,29,30,31,32,33,34,99/
C
C     PUT IN ALPHA UNITS.
C
      DATA ((ENTRY(I,J),I=2,3),J=1,8)/'CM/SEC','.    ',
     .                'METERS','/SEC. ','FT./SE','C.    ',
     .                'YARDS/','SEC.  ','KNOTS ','      ',
     .                'KM./HR','.    ','MILES/','HR.   ',
     .                'FRLNGS','/FRTNT'/
C
C     ENTER INTERNAL UNITS.
C
      UNITS(3)='METERS'
      UNITS(4)='/SEC. '
C
C     IF UNICDE=0, THEN SET CODE TO STANDARD UNITS.
C
      IF(UNICDE.EQ.0) UNICDE=-29
C
C     SET CONVERSION FACTORS:
C
      DATA (FACTOR(I),I=1,8)/.01D0,1D0,.3048D0,.9144D0,.514500,.2777777D0
     .                      ,.44694444D0,1.662D-4/
C
C     IF CODE IS IN TABLE, PERFORM CONVERSION. ALSO, ENTER UNITS ALPHA
C          CODE INTO 'UNITS':
C
      ICODE=IABS(UNICDE)
      DO 2 I=1,6
      IF(ICODE.NE.ENTRY(1,I)) GO TO 2
      IF(UNICDE.GE.0) VALUE=VALUE*FACTOR(I)
      IF(UNICDE.LT.0) VALUE=VALUE/FACTOR(I)
      DO 1 J=1,2
      UNITS(J)=ENTRY(J+1,I)
    1 CONTINUE
C
C     AT THIS POINT, CONVERSION IS COMPLETE.
C
      RETURN
C
C     ELSE, CHECK REST OF TABLE.
C
```

```
      2 CONTINUE
C
C     IF CODE NOT FOUND, RETURN WITH 'UNITS ERROR' IN 'UNITS'.
C
      UNITS(1)='UNITS '
      UNITS(2)='ERROR '
      PRINT 101,ICODE
  101 FORMAT(' ERROR--UNIT CODE',I3,' NOT IN VELOCITY TABLE.')
      RETURN
      END
```

```
@ELT,L      N.STATCK,.STATCK

      1.            SUBROUTINE STATCK (MASK,STATUS,*)
      2.      C
      3.            IMPLICIT INTEGER (A-Z)
      4.            DIMENSION MASK(2),STATUS(2)
      5.      C
      6.      C
      7.            DO 10 I=1,2
      8.            CK     = AND(MASK(I),STATUS(I))
      9.            IF (CK .NE. MASK(I)) RETURN 1
     10.         10 CONTINUE
     11.      C
     12.            RETURN
     13.      C
     14.            END
```

```
@ELT,L      N.STORE,.STORE

      1.            SUBROUTINE STORE
      2.
      3.      C     THIS PROGRAM STORES COMMON/RETREV/ INTO THE USAGE FILE (UNIT USGUNT).
      4.
      5.            PARAMETER USGUNT = 19
      6.            INCLUDE RETREV,LIST
      7.            IMPLICIT INTEGER (A-Z)
      8.      C
      9.      C        READ SECTOR 1 OF USAGE FILE
     10.      C
     11.            CALL NTRAN (USGUNT,10,2,1,LOC,LSTAT,22)
     12.            IF (LSTAT.GT.0) GO TO 20
     13.      C
     14.      C        ERROR READING USAGE FILE
     15.      C
     16.            WRITE (6,10) LSTAT
     17.     10     FORMAT (' *** ERROR READING USAGE FILE - ERROR CODE',I4,' ***')
     18.            STOP ERROR
     19.      C
     20.      C        WRITE THE RETRIEVAL REQUESTS TO THE USAGE FILE
     21.      C
     22.     20     CALL SETADR (USGUNT,LOC)
     23.            CALL NTRAN (USGUNT,1,426,NLAT,LSTAT,22)
     24.            IF (LSTAT.GT.0) GO TO 50
     25.      C
     26.      C        ERROR WRITING TO USAGE FILE
     27.      C
     28.     30     WRITE (6,40) LSTAT
     29.     40     FORMAT (' *** ERROR WRITING TO USAGE FILE - ERROR CODE',I4,' ***')
     30.            STOP ERROR
     31.      C
     32.      C        WRITE NEW LOCATION SPECIFICATION
     33.      C
     34.     50     LOC = LOC + 16
     35.            CALL NTRAN (USGUNT,10,1,1,LOC,LSTAT,22)
     36.            IF (LSTAT.LE.0) GO TO 30
     37.            RETURN
     38.            END
```

```
@FOR,SN     N.TMPTUR,,TMPTUR

      SUBROUTINE TMPTUR(VALUE,UNICDE,UNITS)
C
C     TABLE OF TMPTUR MEASURE CONVERSION FACTORS.
C
      INTEGER ENTRY(3,9),UNITS(9),UNICDE
C
C     PUT UNIT CODES INTO ENTRY.
C
      DATA (ENTRY(1,I),I=1,9)/35,36,97,98/
C
C     PUT IN ALPHA UNITS.
C
      DATA ((ENTRY(I,J),I=2,3),J=1,9)/'FAHREN','HEIT  ',
     .                 'DEG. C','ELSIUS','KELVIN','      ',
     .                 'RANKIN','ES    '/
C
C     ENTER INTERNAL DEGREE UNITS.
C
      UNITS(3)='DEGREE'
      UNITS(4)='S CENT'
C
C     IF UNICDE=0, THEN SET CODE TO STANDARD UNITS.
C
      IF(UNICDE.EQ.0) UNICDE=-36
C
C     IF CODE IS IN TABLE, PERFORM CONVERSION. ALSO, ENTER UNITS ALPHA
C          CODE INTO 'UNITS'.
C
      ICODE=IABS(UNICDE)
      DO 1 I=1,9
      IF(ICODE.NE.ENTRY(1,I)) GO TO 1
      GO TO (2,3,4,5,1
    1 CONTINUE
C
C     IF CODE NOT FOUND, RETURN WITH 'UNITS ERROR' IN 'UNITS'.
C
      UNITS(1)='UNITS '
      UNITS(2)='ERROR '
      PRINT 101,ICODE
      RETURN
    2 IF(UNICDE.GE.0) VALUE=(VALUE-32.)*5./9.
      IF(UNICDE.LT.0) VALUE=VALUE*9.75.+32.
      UNITS(1)=ENTRY(2,I)
      UNITS(2)=ENTRY(3,I)
      RETURN
    3 UNITS(1)=ENTRY(2,I)
      UNITS(2)=ENTRY(3,I)
      RETURN
    4 IF(UNICDE.GE.0) VALUE=VALUE-273.15
      IF(UNICDE.LT.0) VALUE=VALUE+273.15
      UNITS(1)=ENTRY(2,I)
      UNITS(2)=ENTRY(3,I)
      RETURN
    5 IF(UNICDE.GE.0) VALUE=(VALUE-491.67)*5./9.
```

```
      IF(UNICODE.LT.0) VALUE=VALUE*9./5.+491.67
      UNITS(1)=ENTRY(2,I)
      UNITS(2)=ENTRY(3,I)
      RETURN
101   FORMAT(' ERROR--UNIT CODE ',I3,' NOT IN TEMPERATURE TABLE.')
      END
```

```
QELT,L      N.UNPKCR,.UNPKCR

      SUBROUTINE UNPKCR(DATA)
      IMPLICIT INTEGER(A-Z)
C
      DIMENSION DATA(1),INOTES(1)
C
      COMMON /CRUISE/CRUNO,PRNTC,CAILAT,CAILON,CAIDAT,CAFLAT,CAFLON,
     *               CAFDAT,CEILAT,CEILON,CEIDAT,CEFLAT,CEFLON,CEFDAT,
     *               CASTAT(2),CESTAT(2),PACS,PENS,NCARDS,NOTES(14,132),
     *               CAFLG
C
      EQUIVALENCE(INOTES(1),NOTES(1,1))
C
      CRUNO=FLD(0,18,DATA(2))
C
C     UNPACK NUMBER OF CARDS.
C
      NCARDS=FLD(18,18,DATA(2))
C
      PRNTC=DATA(3)
C
C     UNPACK ACOUSTIC INITIAL AND FINAL POINT BOUNDS.
C
      CALL UPKREA(' ',CAILAT,CAILON,CAIDAT,0,' ',
     *               CAFLAT,CAFLON,CAFDAT,0,' ',DATA(4))
C
C     UNPACK ENVIRONMENT INITIAL AND FINAL POINT BOUNDS.
C
      CALL UPKREA(' ',CEILAT,CEILON,CEIDAT,0,' ',
     *               CEFLAT,CEFLON,CEFDAT,0,' ',DATA(8))
C
      CASTAT(1)=DATA(12)
      CASTAT(2)=DATA(13)
C
      CESTAT(1)=DATA(14)
      CESTAT(2)=DATA(15)
C
      PACS=DATA(16)
      PENS=DATA(17)
C
C     UNPACK NOTES.
C
      N=14*NCARDS
      DO 1 I=1,N
    1 INOTES(I)=DATA(I+17)
      RETURN
      END
```

```
@ELT,L     N,UNPKAP,,UNPKAP

      1.            SUBROUTINE UNPKAP(P(DATA)
      2.            IMPLICIT INTEGER(A-Z)
      3.      C
      4.            DIMENSION DATA(1)
      5.      C
      6.            COMMON /EXPMNT/EXNO,PNXTE,PCRU,EAILAT,EAILON,EAIDAT,EAFLAT,EAFLON,
      7.           *               EAFDAT,EEILAT,EEILON,EEIDAT,EEFLAT,EEFLON,EEFDAT,
      8.           *               EASTAT(2),EESTAT(2),EAFLG
      9.      C
     10.      C
     11.      C     UNPACK EXPERIMENT NUMBER.
     12.      C
     13.            EXNO=DATA(2)
     14.      C
     15.      C     UNPACK ACOUSTIC INITIAL AND FINAL POINT BOUNDS.
     16.      C
     17.            CALL UPKREAL(' ',EAILAT,EAILON,EAIDAT,0,' ',
     18.           *               EAFLAT,EAFLON,EAFDAT,0,' ',DATA(3))
     19.      C
     20.      C     UNPACK ENVIRONMENT INITIAL AND FINAL POINT BOUNDS.
     21.      C
     22.            CALL UPKREAL(' ',EEILAT,EEILON,EEIDAT,0,' ',
     23.           *               EEFLAT,EEFLON,EEFDAT,0,' ',DATA(7))
     24.      C
     25.            EASTAT(1)=DATA(11)
     26.            EASTAT(2)=DATA(12)
     27.      C
     28.            EESTAT(1)=DATA(13)
     29.            EESTAT(2)=DATA(14)
     30.      C
     31.            PNXTE=DATA(15)
     32.      C
     33.            PCRU=DATA(16)
     34.            RETURN
     35.            END
```

```
@FOR,S4     U.UNTCVT,.UNTCVT

      1.            SUBROUTINE UNTCVT
      2.
      3.      C        THIS ROUTINE CONVERTS THE SELECTED CONSTANTS, PARAMETERS, AND
      4.      C        VARIABLES FROM SYSTEM UNITS TO THE REQUESTED UNITS.
      5.
      6.            IMPLICIT INTEGER (A-Z)
      7.            REAL CON,PAR,DATA
      8.            DIMENSION ICON(4,30),IPAR(30,30),UNITS(4)
      9.            INCLUDE CPV.LIST
     10.            INCLUDE DATCOM.LIST
     11.            EQUIVALENCE (CON(1,1),ICON(1,1)),(PAR(1,1),IPAR(1,1))
     12.
     13.            IF (NCON.EQ.0) GO TO 20
     14.      C
     15.      C        CONVERT THE CONSTANTS
     16.      C
     17.            DO 10 I = 1,NCON
     18.      10    CALL CHECK (CON(3,I),ICON(1,I),UNITS)
     19.      C
     20.      C        CHECK IF THERE ARE ANY DATA SETS
     21.      C
     22.      20    IF (NDATS.EQ.0) RETURN
     23.      C
     24.      C     DATA SET LOOP
     25.      C
     26.            DO 150 I = 1,NDUSED
     27.            IF (NPAR.EQ.0) GO TO 90
     28.      C
     29.      C        CONVERT THE PARAMETERS
     30.      C
     31.            DO 80 L = 1,NPAR
     32.            J = IPAR(1,L) + 4
     33.            DO 80 K = 5,J
     34.      80    CALL CHECK (PAR(K,L),IPAR(2,L),UNITS)
     35.      C
     36.      C        COMPUTE BEGINNING SUBSCRIPT FOR DATA IN DATA SET NRS(I)
     37.      C
     38.      90    RKT = 0
     39.            IF (NRS(I).EQ.1) GO TO 120
     40.            IK = NRS(I-1)
     41.            DO 110 IXX = 1,IK
     42.      110   RKT = RKT + NROWS(IXX)
     43.      120   II = NRS(I)
     44.            L7 = NROWS(II)
     45.            DO 140 LS = 1,L7
     46.            RKT = RKT + 1
     47.      C
     48.      C        CONVERT THE VARIABLES
     49.      C
     50.            DO 130 L4 = 1,NVAR
     51.            DSUB = RKT * NVAR - NVAR
     52.      130   CALL CHECK (DATA(DSUB+L4),IVAR(1,L4),UNITS)
     53.      140   CONTINUE
     54.      150   CONTINUE
```

```
55.     RETURN
56.     END
```

```
$ELT,L       V.UPKREA,,UPKREA

      SUBROUTINE UPKREA(PORA,ILAT,ILON,IDAT,ITIM,IZON,
     .                  FLAT,FLON,FDAT,FTIM,FZON,IOT)
C
C     MAKE ALL VARIABLES INTEGER AND DIMENSION IOT.
C
      IMPLICIT INTEGER (A-Z)
      DIMENSION IOT(4)
C
C     UNPACK POINT OR AREA BIT.
C
      PORA='P'
      IF(IOT(1).LT.0) PORA='A'
C
C     LOOP TO UNPACK INITIAL DATA FIRST, THEN FINAL DATA.
C
      DO 2 I=1,3,2
C
C     INITIALIZE INTERMEDIATE VARIABLES TO 0.
C
      IT1=0
      IT2=0
      IT3=0
      IT4=0
      IT5=0
      SIGN=0
      ITMP=0
C
C     UNPACK THE DATE.
C
      ITMP=FLD(1,17,IOT(I))
      IT1=ITMP/100
      IT1=((ITMP-IT1*100)*10000+IT1
C
C     UNPACK THE TIME.
C
      IT2=FLD(18,12,IOT(I))
C
C     UNPACK THE ZONE.
C
      FLD(1,5,IT3)=FLD(30,5,IOT(I))
C
C     UNPACK THE LATITUDE.
C
      IT4=FLD(0,17,IOT(I+1))
C
C     GET LATITUDE SIGN BIT, ADJUST LATITUDE ACCORDINGLY.
C
      SIGN=FLD(35,1,IOT(I))
      IF(SIGN.GT.0) IT4=-IT4
C
C     UNPACK THE LONGITUDE.
C
      IT5=FLD(18,18,IOT(I+1))
C
```

```
      C     GET LONGITUDE SIGN BIT, ADJUST LONGITUDE ACCORDINGLY.
      C
            SIGN=FLD(17,1,IOTC(I+1))
            IF(SIGN.GT.0) ITS=-ITS
      C
      C     IF I IS NOT 1,WE ARE ALMOST DONE.
      C
            IF(I.EQ.1) GOTO 1
      C
            FDAT=IT1
            FTIM=IT2
            FZON=IT3
            FLAT=IT4
            FLON=ITS
            RETURN
      C
          1 IDAT=IT1
            ITIM=IT2
            IZON=IT3
            ILAT=IT4
            ILON=ITS
      C
      C     UNPACK THE FINAL POINT DATA.
      C
          2 CONTINUE
            END
```

BEST AVAILABLE COPY

```
SP34,S.      1.VOLUME,.VOLUME

  1.           SUBROUTINE VOLUME(VALUE,UNICDE,UNITS)
  2.     C
  3.     C     TABLE OF VOLUME MEASURE CONVERSION FACTORS.
  4.     C
  5.           INTEGER ENTRY(3,4),UNITS(4),UNICDE
  6.     C
  7.     C     FACTORS ARE DOUBLE PRECISION.
  8.     C
  9.           DOUBLE PRECISION FACTOR(4)
 10.     C
 11.     C     PUT UNIT CODES INTO ENTRY.
 12.     C
 13.           DATA (ENTRY(I,1),I=1,4)/17,18,19,20/
 14.     C
 15.     C     PUT IN ALPHA UNITS.
 16.     C
 17.           DATA ((ENTRY(I,J),I=2,3),J=1,4)/'CM**3 ','       ',
 18.          *                 'METERS','**3   ','FEET**','3     ',
 19.          *                 'YARDS*','**3   '/
 20.     C
 21.     C     ENTER INTERNAL UNITS.
 22.     C
 23.           UNITS(3)='METERS'
 24.           UNITS(4)=' CU.  '
 25.     C
 26.     C     IF UNICDE=1, THEN SET CODE TO STANDARD UNITS.
 27.     C
 28.           IF(UNICDE.EQ.1) UNICDE=-18
 29.     C
 30.     C     SET CONVERSION FACTORS.
 31.     C
 32.           DATA (FACTOR(I),I=1,4)/1.0D-6,1.0,2.8317D-2,.7645D
 33.     C
 34.     C     IF CODE IS IN TABLE, PERFORM CONVERSION. ALSO, ENTER UNITS ALPHA
 35.     C          CODE INTO 'UNITS'.
 36.     C
 37.           ICODE=IABS(UNICDE)
 38.           DO 2 I=1,4
 39.           IF(ICODE.NE.ENTRY(1,I)) GO TO 2
 40.           IF(UNICDE.GE.0) VALUE=VALUE*FACTOR(I)
 41.           IF(UNICDE.LT.0) VALUE=VALUE/FACTOR(I)
 42.           DO 1 J=1,2
 43.           UNITS(J)=ENTRY(J+1,I)
 44.         1 CONTINUE
 45.     C
 46.     C     AT THIS POINT, CONVERSION IS COMPLETE.
 47.     C
 48.           RETURN
 49.     C
 50.     C     ELSE, CHECK REST OF TABLE.
 51.     C
 52.         2 CONTINUE
 53.     C
 54.     C     IF CODE NOT FOUND, RETURN WITH 'UNITS ERROR' IN 'UNITS'.
```

BEST_AVAILABLE COPY

```
      C
            UNITS(1)='UNITS '
            UNITS(2)='ERROR '
            PRINT 101,ICODE
  101 FORMAT(' ERROR--UNIT CODE=',I3,' NOT IN VOLUME TABLE.')
            RETURN
            END
```

BEST AVAILABLE COPY

```
DELT.L      M.MEST,.MEST

  1.            INTEGER FUNCTION MEST (A,B)
  2.      C
  3.      C
  4.            INTEGER A,B
  5.      C
  6.            IF ((ABS(A-B) .GT. 1000000) GO TO 10
  7.            MEST = MIN(A,B)
  8.            RETURN
  9.      C
 10.        10 MEST = B
 11.            IF (A .GT. 0) MEST = A
 12.            RETURN
 13.            END
```


BEST AVAILABLE COPY

APPENDIX B

CURRENT NAVDAB STEERING GROUP

APPENDIX B

CURRENT NAVDAB STEERING GROUP

The NAVSEA Ocean Environmental Acoustic Data Bank (NAVDAB) is under the general sponsorship and direction of NAVSEA 06H1-4. NAVDAB functions as a subsidiary of the Mobile Sonar Technology (MOST) Acoustics of the Medium Committee (ACME) for NAVSEA 06H1. Dissemination of information and requests for any of the documents pertaining to NAVDAB may be obtained by sending a formal request to any of the following current NAVDAB Steering Group members. A list of official mailing addresses follows the list of Steering Group members.

Executive Secretary

A. P. Franceschetti, SEA 06H1-4
Naval Sea Systems Command
Washington, D.C. 20362
Commercial (202) 692-3166
Autovon 222-3166

Chairman

L. A. Smothers, Code 3073
Naval Undersea Center
San Diego, California 92132
Commercial (714) 225-2316
Autovon 933-2316

Branch Managers

F. R. DiNapoli, Code TA113
Naval Underwater Systems Center
New London, Connecticut 06320
Commercial (203) 442-0771 Ext. 2647
Autovon 636-2647

Branch Managers (Continued)

W. H. Geddes, Code 3440
Naval Oceanographic Office
Washington, D.C. 20373
Commercial (202) 433-3994
Autovon 288-3994

L. A. Smothers, Code 3073
Naval Undersea Center
San Diego, California 92132
Commercial (714) 225-2316
Autovon 933-2316

Steering Group

F. R. DiNapoli, Code TA113
Naval Underwater Systems Center
New London, Connecticut 06320
Commercial (203) 442-0771 Ext. 2647
Autovon 636-2647

R. H. Ferris, Code 8120
Naval Research Laboratory
Washington, D.C. 20375
Commercial (202) 767-3359
Autovon 297-3359

W. H. Geddes, Code 3440
Naval Oceanographic Office
Washington, D.C. 20373
Commercial (202) 433-3994
Autovon 288-3994

R. F. Hosmer, Code 3071
Naval Undersea Center
San Diego, California 92132
Commercial (714) 225-2316
Autovon 933-2316

Steering Group (Continued)

W. A. Kuperman, Code 8120
Naval Research Laboratory
Washington, D.C. 20375
Commercial (202) 767-3210
Autovon 297-3210

K. V. Mackenzie, Code 9420
Naval Oceanographic Office
Naval Research Laboratory
4555 Overlook Ave. SW
Washington, D.C. 20375
Commercial (202) 767-2830
Autovon 297-2830

L. C. Maples, Code TA1A
Naval Underwater Systems Center
New London, Connecticut 06320
Commercial (203) 442-0771 Ext. 2501
Autovon 636-2501

S. R. Santaniello, Code TA111
Naval Underwater Systems Center
New London, Connecticut 06320
Commercial (203) 442-0771 Ext. 2675
Autovon 636-2675

J. A. Whitney, Code 3073
Naval Undersea Center
San Diego, California 92132
Commercial (714) 225-2316
Autovon 933-2316

OFFICIAL MAILING ADDRESSES

1. Commander
   Naval Undersea Center
   San Diego, California 92132

2. Commanding Officer
   Naval Underwater Systems Center
   New London, Connecticut 06320

3. Director
   Naval Research Laboratory
   Washington, District of Columbia 20375

4. Commander
   Naval Oceanographic Office
   Washington, District of Columbia 20373

5. Commander
   Naval Sea Systems Command
   Washington, District of Columbia 20362

USER COMMENTS FORM

Comments concerning this document are invited. Please send one of the following attached forms or a letter to:

L. A. Smothers, Code 307
Naval Undersea Center
San Diego, California 92132

(Envelope address: Commander, Naval Undersea Center, San Diego, California 92132)

MEMORANDUM Date: _____

From: _____

_____

_____

To: L. A. Smothers, NAVDAB Chairman, Code 3073

Subj: NAVSEA Ocean Environmental Acoustic Data Bank (NAVDAB),
SEA 06H1/036-EVA/MOST- 5 (Volume 4: Details of Retrieval Phase)

CUT

MEMORANDUM Date: _____

From: _____

_____

_____

To: L. A. Smothers, NAVDAB Chairman, Code 3073

Subj: NAVSEA Ocean Environmental Acoustic Data Bank (NAVDAB),
SEA 06H1/036-EVA/MOST- 5 (Volume 4: Details of Retrieval Phase)

CUT

MEMORANDUM

Date: _____

From: _____

_____

_____

To: L. A. Smothers, NAVDAB Chairman, Code 3073

Subj: NAVSEA Ocean Environmental Acoustic Data Bank (NAVDAB), SEA 06H1/036-EVA/MOST- 5 (Volume 4: Details of Retrieval Phase)

CUT

DISTRIBUTION LIST

5 Naval Sea Systems Command
NSEA-06H1 (5 cys)

3 Naval Oceanographic Office
Code 3432
Code 3440
Code 6100

2 Naval Research Laboratory
Code 8120 (2 cys)

7 Naval Undersea Center, San Diego
Code 307 (7 cys)

6 Naval Underwater Systems Center, New London Laboratory
Code TA1A
Code TA42
Code TA111 (2 cys)
Code TA112
Code TA113

2 Defense Documentation Center


FILM
5