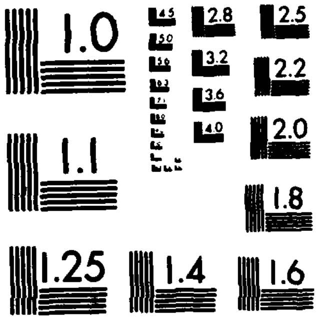
AD-A128 146 A COMPUTER PROGRAM FOR PROCESSING CONDUCTIVITY-TEMPERATURE-DEPTH (CTD) DATA(U) NAVAL OCEAN RESEARCH AND DEVELOPMENT ACTIVITY NSTL STATION MS Z R HALLOCK JAN 83 NORDA-TN-196 1/1

UNCLASSIFIED F/G 9/2 NL

END
DATE FILMED
6-83
DTIC



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

NORDA Technical Note 196

Naval Ocean Research and
Development Activity
NSTL Station, Mississippi 39529

A Computer Program for Processing Conductivity-Temperature-Depth (CTD) Data

Z.R. Hallock
Ocean Science and Technology Laboratory
Oceanography Division

January 1983

ABSTRACT

A software package for processing conductivity-temperature-depth (CTD) data is described. The package includes features for editing, correcting, filtering and pressure sorting to produce working data files for graphic and analysis work.

ACKNOWLEDGMENTS

This work is a description of software developed while the author was employed by the Naval Oceanographic Office (Code 7210). The vast majority of the programming, testing, debugging and use of the described software was performed by Mr. William Teague. Many of the algorithms used were adapted from programs written by Mr. Frank Mulher, who also wrote the preparatory programs. The careful work of Mr. Carlos Mayoral and his colleagues in Code 6320 in determining sensor time-responses and the expert help of Dr. Tom Davis of Code 022 in designing deconvolution filters for temperature, allowed the inclusion of data quality improvement features.

This work was supported by the Ocean Measurements Program through the Ocean Programs Management Office at NORDA, funded by Program Element 63704N, Dr. Rudy Hollman, program manager.

A COMPUTER PROGRAM FOR PROCESSING CONDUCTIVITY-TEMPERATURE-DEPTH (CTD) DATA

I. INTRODUCTION

CTD (conductivity-temperature-depth) data acquired with a Neil Brown Instrument Systems CTD system (or equivalent data) require certain processing steps before they are generally usable for oceanographic analysis. This note describes a processing software package (CTDPRG), which was developed at the Naval Oceanographic Office and has been used in a production mode since 1978. Some minor modifications have been made since that time, but the basic structure has not changed.

Input data for CTDPRG must be in the form of a Fast Easy Binary (FEB) file (Hallock, 1980) and must consist of the physical variables: pressure (dbar), temperature (°C) and conductivity (mmho/cm). Hence, a pre-processing step is required which will be specific to the original data recording format. Output is quite flexible ranging from the input data itself through corrected, pressure-sorted and subsampled data with calculated quantities such as salinity and density.

Section II is a general, functional description of algorithms used. Section III provides details for running the programs with several examples of run setups. Section IV contains complete program listings.

II. GENERAL DESCRIPTION

CTDPRG is designed to handle CTD data acquired as vertical profiles, horizontal tows, or a combination of these. The program was developed around vertically profiled data; however, and much of its complexity goes toward converting the time-base CTD data to functions of pressure, when this conversion is required.

The following list is a summary of the processing steps performed by CTDPRG.

- Read input controls from cards or keyboard.
- Read data from mass storage file.
- Correct isolated spikes (wild data points).
- Apply time-domain filters to pressure, temperature and conductivity (this includes T-C matching).
- Correct conductivity values based on water sample comparisons.
- Apply temperature-pressure dependent cell correction.
- Separate up-going and down-going profiles.
- Pressure-sort data by averaging into 0.1 dbar intervals.

- Low-pass filter, in pressure, and subsample to multiple of 0.1 dbar.
- Compute selected quantities (e.g., salinity, sigma-t) from P, T, C.
- Write results to mass-storage files.

All steps (except the first three and last) are optional and can be selected in several combinations.

Input Data Sets

The basic unit of processing is the input data set. This is, typically, a single CTD cast (down and up, if both were recorded). It might be, however, a series of down and up profiles acquired during one instrument deployment. Usually an input data set corresponds to a single input file, but this is not required. There may be more than one data set per file or a data set may span several files. All this is handled by the input controls. The input data set is in a FEB-file containing time, pressure (dbar), temperature (°C) and conductivity (mmho/cm).* Additional variables could be carried with minor modifications. The last FEB-segment in the data set must have IDOCR(1)=1; preceding segments (for multi-segment data sets) must have IDOCR(1)=0. Input files are thus prepared by a pre-processing program that is tailored for the particular instrument and deployment mode.

Time-Domain Filters

One of the most significant problems with CTD data has been the effects of the response mismatch between temperature and conductivity sensors. A number of attempts have been made to correct this through the design and application of filters. The most widely used method is that described by Fofonoff et al. (1974), which assumes an exponential response for the temperature sensor. More recently, as a result of work done at the Naval Oceanographic Office (NOO) involving direct measurement of instrument response, specific filters have been designed for temperature and conductivity, based on experimentally determined sensor response functions (see Mayoral, 1982). In CTDPRG there are three options: no T-C filtering, filtering, the three-point Fofonoff method (the user specifies only the time-constant), or explicitly supplied filters of the user's choice. These are applied to the raw data, in the time domain, prior to other processing.

A smoothing filter is automatically applied to pressure data to remove high-frequency sensor noise.

Conductivity Calibration

With the newer conductivity cells on the MK III CTD systems, coupled with proper cleaning procedures and frequent laboratory calibration adjustments, conductivity data seldom

*See Hallock (1980) for details on FEB-file structure.

has to be adjusted from the recorded values. However, if salinity checks derived from water samples indicate an offset from proper calibration, a multiplicative correction can be supplied by the user. There is also a manufacturer-supplied correction for pressure and temperature distortion which is applied following the filtering step.

Upcasts and Downcasts

If no editing based on pressure is to be done (i.e., no identification of upcasts, downcasts or constant level data), following conductivity correction, selected additional variables are computed, and an output data set is written that contains the same number of scans (cycles) as the input.

A second option is to retain only upcasts and/or downcasts, based on user defined pressure limits, which are then written as separate output data sets, following computation of additional variables.

A third option, which necessitates the second, is to convert each upcast/downcast to a pressure series as opposed to a time series. This case assumes that the instrument descent/ascent rate is 2 m/sec or less. (If this value is exceeded in the mean, some software modifications would be necessary.) At a nominal descent/ascent rate of 1 m/sec, about three scans of data are acquired every 0.1 dB. If pressure sorting is selected, time, temperature and conductivity values in 0.1 dbar intervals are averaged, creating a pressure series at that resolution. Retrograde (in pressure) data are rejected. The pressure series is then subsampled at some user-defined, integral multiple of 0.1 dbar following the application of a low-pass filter to prevent aliasing. The filter is automatically generated, and its cutoff wavenumber is the new Nyquist wavenumber of the subsampled series. At this point additional variables are computed and data are written to output files. The third option allows the user to select a degree of smoothing/subsampling commensurate with his requirements.

Computed Variables

Including the four input variables, up to twelve are available as output. The user may select any subset of these. The eight computed variables are as follows:

- Salinity (parts per thousand [ppt]) or S: computed with pressure, temperature and conductivity ratio, according to Bennett (1976), Dauphine and Klein (1977) or Lewis (1980).

- Sigma or σ: the in situ density of a water parcel. Computed with pressure, temperature, and salinity according to Fofonoff (1962). (Sigma = (density (g/cc)-1.)*1000.)

- Sigma-t or σ_t: the density of a water parcel with in situ temperature and salinity but at atmospheric pressure.

- Brunt-Väisälä Frequency (cycles per hour) or N: computed with pressure, temperature, and salinity according to Fofonoff

(1962). Essentially a local potential density derivative. The convention is adopted: $N = \sqrt{|N^2|}\ \mathrm{sgn}\ (N^2)$ to allow for instabilities.

- Sound Speed (m/sec): computed with pressure, temperature, and salinity according to Wilson (1960).

- True Depth: for most purposes, pressure in decibars can be interpreted as depth in meters; however, this assumes an average water density of 1.02 g/cc. If depth accurate to more than a few percent are required, this variable can be selected. It is an approximation to:

$$z = \int_0^p \frac{\alpha}{g}\, dp$$

where α = specific volume,
g = acceleration of gravity,
p = pressure,
z = depth below surface,

- Potential Temperature (°C) or θ: The temperature that a parcel of water would have if brought adiabatically to the surface. Computed with pressure, temperature and salinity according to Fofonoff (1962).

- Potential Density or σ_θ: σ_t using θ rather than in situ temperature; the density of a water parcel brought adiabatically to the surface. (σ_θ is calculated using the σ_t routine with θ from the above routine.)

A typical standard processing product might be: retain only downcasts at 1 meter resolution, and store pressure, temperature, salinity and σ_t. This provides a compact version of data for hydrographic analysis work. The best "standard" file is likely to be different for each user group; thus, it is recommended that the input files (edited, raw data) be considered the primary, archived data base.

III. DETAILS OF OPERATION

This section consists of two parts: a tour of CTDPRG and its subroutines; a tutorial on setting up input and running the program.

Functional Program Description

Figure 1 is a block diagram of all the routines in the package. CTDPRG is the driver, whose functions are to read all input controls, to set up filters to be used on input data, and to initiate the processing of each input data set by calling ACCESS.

GENWHF generates a three-point deconvolution filter for temperature based on a user-specified, sensor time-constant. This is described in detail by Fofonoff et al. (1974).

GENFLT is a general filter generating routine (see Brooks, 1976). It is called in CTDPRG to produce a low-pass filter for pressure data to remove sampling noise.

ACCESS controls the majority of the processing. Input data cycles (scans) are accessed, one at a time, via calls to GETREC. If pressure sorting is not selected, output segments are prepared using a user-specified combination of variables from the input as well as quantities computed via DERIVE.

When upcast/downcast separation is specified, each upcast/downcast becomes a separate data set and is written to unit 3/2. Otherwise, all output goes to unit 2 as one data set.

With pressure sorting, which necessitates upcast/downcast separation, input data cycles in 0.1 dB intervals are averaged, producing a pressure series at that resolution. Retrograde data, which result from ship motion or winch reversals, are rejected. For a nominal descent rate of 1 meter per second there should be about 3 cycles per interval. If an interval should turn up empty, it is filled with data from the previous interval. If descent rates systematically exceed 1.5-2 m/s this interval should be increased accordingly.

The pressure sorted data can be subsampled at integral multiples, say n, of 0.1 dB. When this is done, ACCESS calls GENFLT to produce a low-pass filter which is applied to the 0.1 dB data, centered at every nth cycle. The cutoff of the filter is at the new Nyquist of the subsampled series and prevents aliasing. For example: n = 10 implies subsampling every 1 dB; cutoff (half-amplitude point) is at 0.5 cycles/dB; the filter is applied only where samples are to be extracted (e.g., every 10th point), thus avoiding unnecessary computation.

Following pressure sorting and subsampling, DERIVE is called for selected computed variables. Segments are then assembled and written to the output file(s).

Subroutine GETREC is the input program. It reads segments from the input file (unit 4), applies conductivity cell corrections, pressure smoothing filters, conductivity-temperature matching filters and pressure limits. Data are then passed to ACCESS, one cycle at a time. Upper and lower pressure limits, specified by the user, are employed by GETREC to distinguish upcast and downcast data.

NBCCC applies a manufacturer supplied cell correction to conductivity which compensates for pressure and temperature distortions of cell geometry. It also applies a user-specified multiplicative conductivity adjustment (usually = 1.) which corrects for salinity offsets determined by field comparisons.

Time-domain filters are described in Section II.

When an end-of-data set (IDOCR(1) = 1) is detected, control returns to the main program and new input cards are sought. If an end-of-file is encountered, however, GETREC calls IUSET which attempts to dynamically attach another user-specified file name to unit 4. If no file name is supplied, the program terminates. (On some machines dynamic control commands may not be permitted, thus requiring one execution per input file.)

Subroutine DERIVE accepts a cycle of pressure, temperature, conductivity and time from access and, after calling the appropriate variable computation routines, restructures the cycle according to user specifications. The variable computation routines are described in Section II.

Subroutine ZREAD and ZWRIT (described by Hallock, 1980) are input and output routines for FEB-files. A call to ZREAD/ZWRIT transfers one segment of data between main memory and a mass storage file. Typically, a data set (e.g., CTD profile) consists of several FEB segments; each segment is made up of several header blocks followed by a data block. Segments are written sequentially to a mass-storage (disk) file by nonformatted, FORTRAN I/O statements.

Input Control Cards

Essential information for setting up and running CTDPRG is provided through internal documentation (comment cards), which are presented below.

Input Card 1: Printout identification--up to 48 Alphanumeric characters. This can be anything.

Input Card 2: N, LN, NWMAX, KCHOP, NINFIL (Free Format).

N: Subsampling interval for pressure sorted data. Integer multiples of 0.1 dB, e.g., N=20 results in data every 2 dB. N=0 results in no pressure sorting.

LN: Sharpness factor for subsampling filter. This is an integer equal to 1 or greater. Larger values require more computer time. (m = LN*N = half-width of filter. See, GENER1 in Brooks, 1976, for a more detailed explanation.)

NWMAX: Maximum no. of cycles in output segments. Typically, 1000.

KCHOP=1: Process all data. No pressure sorting or subsampling. Pressure limits are not applied. Output goes to unit 2.

KCHOP=2: Pressure sort and subsample by N (unless N = 0). Output downcasts to unit 2, upcasts to unit 3.

KCHOP=3: Same as 2 but output only downcasts.

KCHOP=4: Same as 2 but output only upcasts.

NINFIL: No. of input file names to be supplied. If =0, one input file is assumed on unit 4. Otherwise, supplied names are dynamically attached to unit 4 by IUSET (presently configured for UNIVAC).

Input Card 3: (IND(I), I=1, 12) (Free format) output variable selection. A string of 12 integers (1 for yes, 0 for no) indicating variables to be output from the following list: Time, pressure, temperature, salinity, sigma-t, sigma, Brunt-Väisälä frequency, sound speed, depth, potential temperature, potential density.

Input Filename Cards: up to 24 alphanumeric characters, left-justified. If NINFIL=0, no cards are expected.

The following group of cards, the data set card and pressure limit cards, is repeated for each input data set to be processed. .

Input Data Set Card: NMSER, NPLIMS, CFTR, TIMCON, ISAL (Free Format)

NMSER: Sequence no. of input data set to be processed. (Usually this is equal to 1, but if there were, say, n data sets to be skipped beginning in the first input file, NMSER would be set to n+1.)

NPLIMS: No. of pressure limit cards for this data set.

CFTR: Conductivity cell correction factor (default=1).

TIMCON>0: Temperature sensor time constant divided by sampling time interval. WHOI (Fofonoff et al. temperature deconvolution.)

=0: No. temperature or conductivity filtering.

=-1: Use same filters that were used for previous data set. (Not permissible for first data set.)

=-5: Read special filters from cards (unit 5) following pressure limit cards.

All Else: Read special filters from file on unit 8.

ISAL = 0: Salinity computed according to Lewis and Perkin (1980)

= 1: Salinity computed according to Bennet-Dauphinee.

Pressure/Cycle Limit Cards: PTOP, PBOT, DELTAP, I1, I2 (Free format)

PTOP, PBOT are upper lower pressure limits to be applied to input data cycles between cycles I1 and I2. Each sequence of cycles *spanning this pressure interval* is made into an output data set. Pressure limits are irrelevant for KCHOP=1.

DELTAP is a slop factor (usually about 2 dB) that is employed to prevent small pressure fluctuations from tripping up the pressure limit logic. It ensures, for example, that a downcast has ended before the subsequent upcast is recognized. For single-cast data, DELTAP can be set to zero.

I1, I2--There is a global cycle index, IE, which is [illegible] to 1 at the beginning of an input data set. It is incremented for each cycle encountered within the data set. (An input data set typically spans more than one FEB segment and occasionally more than one file.) The first profile start, defined by PTOP, is not sought until IE reaches I1. After each profile is completed, if IE has reached I2, the next pressure limit card is invoked. When the cards are exhausted or the input data set ends, the next data set card is read. This feature allows leading and trailing data within a data set to be skipped, which can save computer time or reject defective data.

Usually, only one pressure limit card is required for each input data set, particularly for a single profile. However, for a series of repeated profiles which are included in a single data set (i.e., a "yo-yo" cast), it may be necessary to change pressure limits several times to maximize data recovery and to reject sections of bad data.

In the simplest case, for a single cast data set, I1 can be set to 1 and I2 to a large number which exceeds the number of cycles in the data set, and everything will be processed. (Default values of 1 and 1,000,000 result for I1 = I2 = 0.)

Special Filter Input: NWTT, NWTC, LAGT, (WT(I), I=1, NWTT), WC(I), I=NWTC) (Free Format)

These are read *from* cards if TIMCON = -5. If TIMECON = -2, -3, -4, -6, -7,... they are read from a file on unit 8. If TIMCON > -2, this input is not expected. (See TIMCON above.)

NWTT, NWTC: No. of temperature, conductivity, filter weights.

LAGT: No. of cycles to lag temperature relative to pressure and conductivity. (The lag might be built into filter, in which case LAGT = 0.)

WT, WC: Temperature, conductivity filter weights.

Examples of Input Controls

Example 1: Three input files, each containing a downcast and an upcast between about 8 m and 2000 m depth. Only downcasts are to be processed. WHOI (Fofonof, et al.,1974) temperature deconvolution is to be used--time constant is 100 ms, sampling interval is 32 ms. Output pressure resolution is to be 1 dbar. Output variables are to be: P, T, S, σ_t, N. Salinity to be computed by practical salinity scale '78.

Card 1: THIS IS EXAMPLE 1.

Card 2: 10,1,1000,3,3

Card 3: 0,1,1,0,1,1,0,1,0,0,0,0

Card 4: FILE1

Card 5: FILE2

Card 6: FILE3

Card 7: 1,1,1,3.125,0

Card 8: 10,2000,1,1,0

Cards 7 and 8 are repeated for Files 2 and 3.

Example 2: One input file containing a continuous record of two downcasts and two upcasts (i.e., one instrument deployment). The instrument was in the water for one hour before the first downcast began and again following the final upcast. Thus, the first usable data scan is about no. 112,000. The four profiles took about one hour so the last usable scan was around 250,000. Temperature deconvolution filters are to be read from unit 8; a 1.01 conduc- is to be made; salinity is to be computed by Bennet-Dauphinee algorithms. Data are to be subsampled at 50 cm. Both upcasts and downcasts are to be saved, between 100 and 400 dbar.

Card 1: THIS IS EXAMPLE 2.

Card 2: 5,1,1000,2,0 (input assumed on unit 4)

Card 3: 0,1,1,0,1,1,0,1,0,0,0,0

Card 4: 1,1,1.01,-3,1

Card 5: 100,400,2,112000,250000

Example 3: One input file containing 500 downcasts and upcasts between about 90 and 210 dbar ("yo-yo" cast). Data are to be processed as in example 1, saving downcasts at 1 dbar resolution. Prior examination of the data reveals that downcasts 201-208 (scans 1.2×10^6 - 1.248×10^6)

extend only to 190 dbar: three sets of pressure limits are thus required, to recover the maximum amount of data.

Card 1: THIS IS EXAMPLE 3.

Card 2: 10,1,1000,3,0

Card 3: 0,1,1,0,1,1,0,1,0,0,0,0

Card 4: 1,3,1,3.125,0

Card 5: 95,205,2,1,1200000

Card 6: 95,185,2,1200000,1248000

Card 7: 95,205,2,1248000,4000000

Example 4: Input file consists of three downcast-upcast pairs. For diagnostic purposes, it is necessary to process downcast data at original resolution (i.e., no pressure sorting). Output variables are to be: time, pressure, temperature, conductivity and salinity. Pressure limits are 10 to 500 dbar. Special filters for temperature and conductivity are to be supplied on cards.

Card 1: THIS IS EXAMPLE 4.

Card 2: 0,0,1000,3,0

Card 3: 1,1,1,1,1,0,0,0,0,0,0,0

Card 4: 1,1,1,-5,0

Card 5: 10,500,0,1,0

Card 6: 5,5,0,.1,.2,.4,.2,.1,.0625,.125,.625

Card 7: .125,.0625

These are representative examples of how CTDPRG may be used. Many other combinations are possible. Some experimentation may be necessary for unusual requirements.

REFERENCES

Bennett, A.S. (1976). Conversion of In Situ Measurements of Conductivity to Salinity. Deep Sea Res. 23(2), p. 157-165.

Brooks, D.A. (1976). Fast and Easy Time Series Analysis at NCSU. Center for Marine and Coastal Studies, North Carolina State University Report.

Dauphinee, T.M. and H.P. Klein (1977). The Effect of Temperature on the Electrical Conductivity of Seawater. Deep Sea Res. 24, p. 891-902.

Fofonoff, N.P. (1962). Physical Properties of S.W. In: The Sea, v. 1, p. 3-30, M.N. Hill (ed.), Wiley.

Fofonoff, N.P., S.P. Hayes and R.D. Millard (1974). WHOI/Brown CTD Microprofiler: Methods of Calibrating and Data Handling. WHOI TR 74-89.

Hallock, Z.R. (1980). The Fast and Easy Binary File. Naval Oceanographic Office Technical Note (TN)7210-12-80, 29 pp.

Lewis, E.L. (1980). The Practical Salinity Scale 1978 and Its Antecedents. IEEE, J. Oceanic Eng., v. OE-5, p. 3-8.

Mayoral, C.R. (1982). Determination of CTD Sensor Responses for Use in Deconvolution Filtering. Proc., International STD Conference and Workshop, 8-11 February, 1982, Marine Technology Society, San Diego Section.

Wilson W.D. (1980). Speed of Sound in Water as a Function of T.P.S. J. Acoust. Soc. Am., v. 32, p. 641-644.

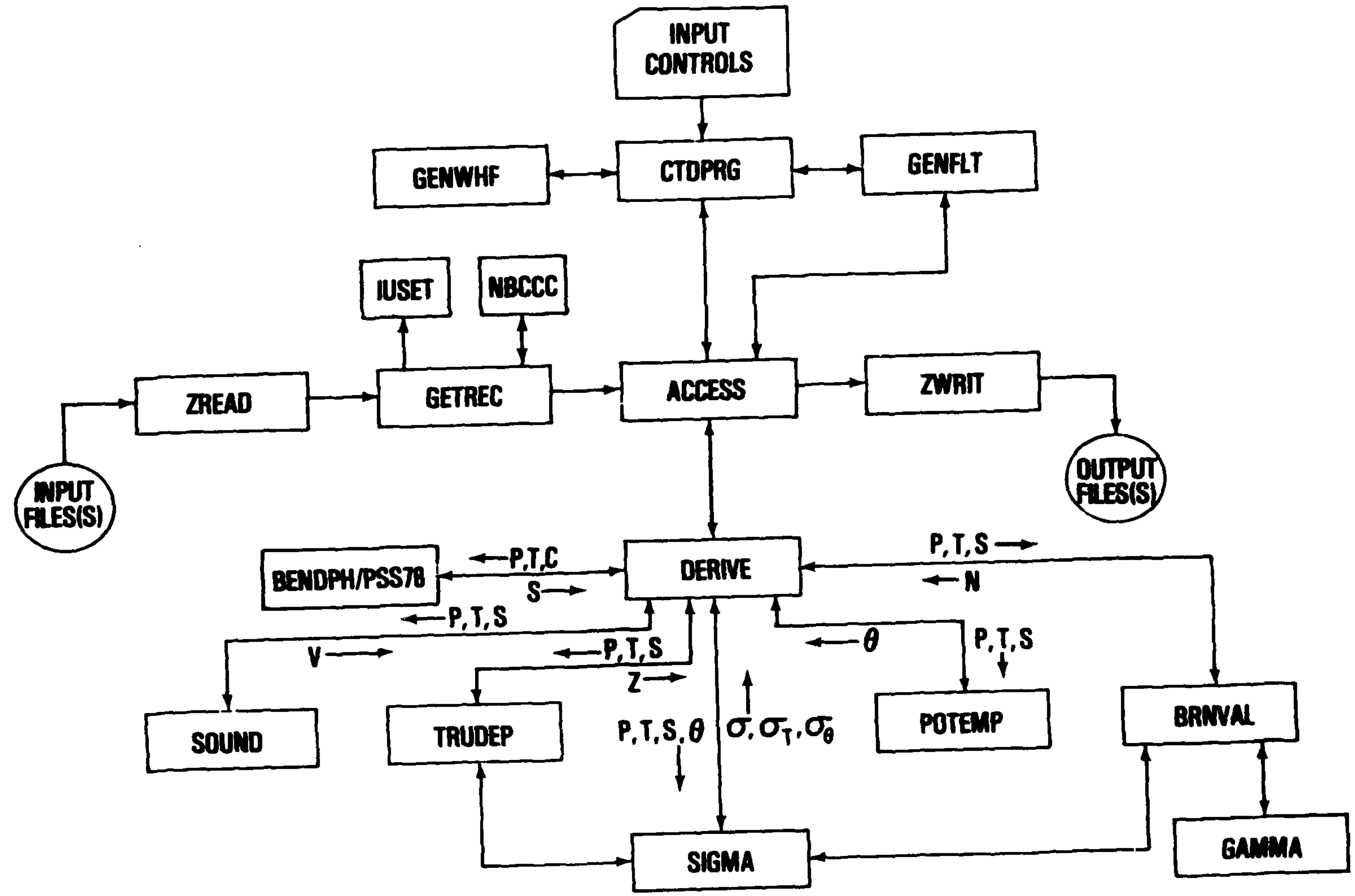


Figure 1. Functional Block Diagram of CTD Processing Programs

APPENDIX

```
PF*UD7*PRGD(1).CTDPRG
            COMPILER(DIAG=3)
      C
      C                     BASIC CTD PROCESSING PROGRAM
      C
      C       INPUT FILES (UNIT 4) ARE PROPERLY PREPARED FEB FILES OF
      C       RAW CTD DATA. VARIABLE ARE TIME, PRESSURE, TEMPERATURE,
      C       CONDUCTIVITY. NMFR, NMBR SHOULD BE CRUISE AND STATION
      C       NUMBER, RESPECTIVELY. IDCR(1)=1 MARKS THE END OF AN
      C       INPUT DATA SET (CAST). THERE CAN BE MORE THAN ONE INPUT
      C       FILE PER DATA SET (SEE BELOW). OUTPUT GOES TO UNITS
      C       2 AND 3. UNSEPARATED DATA (KCHOP=1) AND DOWNCASTS GO TO UNIT
      C       2 WHILE UPCASTS GO TO UNIT 3. FOR FURTHER DETAILS SEE
      C       EXTERNAL DOCUMENTATION. INPUT CONTROL CARDS ARE DES-
      C       CRIBED BELOW.
      C
            COMMON/WHAT/NIND,IND(20),KCHOP,NPLINS,PL(3,100),IPLIN(2,100),
           *   NUMAX,IPWSET(20),IEOLU,NWSER
            COMMON/SALINE/KSAL
            COMMON/IOFILE/NINFIL,IFEB(120)
            COMMON/CALIB/MP,MT,MC,LAGT,WP(60),WT(60),WC(60),PDIF,
           *   TDIF,CDIF,CFTR
            DIMENSION LABEL(20),IRUN(8)
            DATA (LABEL(I),I=1,12) /'TIME','PRESS','TEMP','COND','SAL','SIGT',
           *   'SIGSTP','BVFCPH','SNDVEL','TDEPTH','TEMPOT','SIGPOT'/
            DATA MP,JA,JB,JC,F/10,2,1,1,.1/
            DATA ISW/1/
            DATA PDIF,TDIF,CDIF/.5,.2,.2/
      C
      C ****** INPUT CARD 1: PRINTOUT IDENTIFICATION. UP TO 48 CHARS.
      C                      CAN BE ANYTHING - EVEN BLANK CARD.
      C
            READ(5,1020,END=999) IRUN
       1020 FORMAT(8A6)
            WRITE(6,1021)IRUN
       1021 FORMAT(1H1,' CTD PROCESSING RUN: ',8A6//)
      C
      C
      C
      C ****** INPUT CARD 2: N,LN,NUMAX,KCHOP,NINFIL
      C
      C               N:      SUBSAMPLING INTERVAL FOR PRESSURE SORTING.
      C                       INTEGER MULTIPLES OF .1 DB. I.E. N=20 RESULTS
      C                       IN DATA EVERY 2 DB. N=0 RESULTS IN NO SORTING.
      C
      C               LN:     SHARPNES FACTOR FOR DECIMATION FILTER.
      C                       THIS IS AN INTEGER EQUAL TO 1 OR GREATER.
      C                       IT DETERMINES THE NO. OF WEIGHTS AND THUS
      C                       THE SHARPNESS OF THE DECIMATION FILTER.
      C
      C               NUMAX:  MAX NO. OF CYCLES IN OUTPUT SEGMENTS.
      C                       (AS A RULE, THIS SHOULD BE ABOUT 1000.)
      C
      C               KCHOP=1: PROCESS ALL DATA; NO PRESSURE SORTING
      C                       OR SUBSAMPLING. PRESSURE LIMITS ARE
      C                       NOT APPLIED.
      C
      C               KCHOP=2: OUTPUT UPCASTS AND DOWNCASTS;
      C                       PRESSURE SORT AND SUBSAMPLE BY N.
      C
      C               KCHOP=3: SAME AS 2 BUT OUTPUT ONLY DOWNCASTS.
      C
      C               KCHOP=4: SAME AS 2 BUT PROCESS ONLY UPCASTS.
      C
      C               NINFIL: NO. OF INPUT FILENAMES TO BE SUPPLIED.
      C                       IF NINFIL=0, ONE INPUT FILE IS ASSUMED ON
      C                       UNIT 4. OTHERWISE, SUPPLIED NAMES ARE ASSIGNED
      C                       DYNAMICALLY TO UNIT 4 BY SR IUSET. THIS MAY
      C                       NOT WORK ON ALL MACHINES.
      C
            READ(5,1000,END=999)N,LN,NUMAX,KCHOP,NINFIL
       1000 FORMAT()
            WRITE(6,6000)N,LN,NUMAX,KCHOP
            KCHOPI=KCHOP
       6000 FORMAT(1X,'N=',I5,1X,'LN=',I5,1X,'NUMAX=',
           &I5,1X,'KCHOP=',I5)
      C
      C ****** INPUT CARD 3: OUTPUT VARIABLE SELECTION. A STRING OF
      C                      12 INTEGERS (1 FOR YES, 0 FOR NO)
      C                      INDICATING VARIABLES TO BE OUTPUT, FROM
      C                      THE FOLLOWING LIST.
      C
```

```
C                          TIME,PRESSURE,TEMPERATURE,CONDUCTIVITY,
C                          SALINITY,SIGMA-T, SIGMA, B-V FREQUENCY,
C                          SOUND SPEED, TRUE DEPTH, POTENTIAL
C                          TEMPERATURE,POTENTIAL DENSITY.
C
      READ(5,1001,END=999)(IND(I),I=1,12)
C
      J=0
      DO 11 I=1,20
      IF(IND(I).EQ.0)GO TO 11
      J=J+1
      IPWSET(J)=LABEL(I)
 11   CONTINUE
      NIND=J
C
      WRITE(6,6005)
 6005 FORMAT(//1X,'VARIABLES REQUESTED')
      WRITE(6,6006)(IPWSET(K),K=1,J)
 6006 FORMAT((1X,10(A6,1X)))
C
C ****** INPUT FILENAME CARDS (FORMAT 4A6)
C
C                      ONE CARD FOR EACH INPUT FILENAME OF UP TO
C                      24 CHARACTERS (INCLUDING *). IF NINFIL=0,
C                      NO CARDS ARE EXPECTED.
C
      DO 993 I=1,4
  993 IFEB(I)='      '
      IF(NINFIL.EQ.0) GO TO 990
      DO 991 I=1,NINFIL
      I1=(I-1)*4+1
      I2=I1+3
  991 READ(5,1001,END=999) (IFEB(II),II=I1,I2)
 1001 FORMAT(4A6)
  990 CONTINUE
C
C
C
C                      THE FOLLOWING GROUP OF CARDS, THE DATA SET
C                      CARD AND PRESSURE LIMIT CARDS, IS REPEATED
C                      FOR EACH INPUT DATA SET TO BE PROCESSED.
C
C ****** INPUT DATA SET CARD:  NNSER,NPLIMS,CFTR,TIMCON,ISAL
C
C     NNSER            SEQUENCE NO. OF INPUT DATA SET TO BE PROCESSED.
C
C     NPLIMS           NO. OF PRESSURE LIMIT CARDS FOR THIS DATA SET.
C
C     CFTR             CONDUCTIVITY CELL FACTOR (DEFAULT=1.)
C
C     TIMCON    GT 0:  TEMPERATURE TIME CONSTANT/SAMPLING TIME INTERVAL.
C                      WHOI TEMPERATURE DECONVOLUTION.
C
C                = 0:  NO FILTERS ON TEMP OR COND.
C
C               = -1:  USE FILTERS FOR PREVIOUS DATA SET. (NOT
C                      PERMISSIBLE ON FIRST DATA SET.)
C
C               = -5:  READ SPECIAL FILTERS FROM CARDS FOLLOWING
C                      PRESSURE LIMIT CARD(S).
C
C              ELSE:   READ SPECIAL FILTER CARD IMAGES FROM UNIT 8.
C
C     ISAL       = 0:  SALINITY COMPUTED IAW PSS78.
C
C                = 1:  SALINITY COMPUTED IAW BENNET-DAUPHINEE.
 15   READ(5,1000,END=999)NNSER,NPLIMS,CFTR,TIMCON,ISAL
      NSAL=2
      IF(ISAL.NE.0)NSAL=1
C
      NT=1
      NC=1
      NWTT=1
      NWTC=1
      WT(1)=1.
      WC(1)=1.
      IFILT=2
      IF(TIMCON.LT.0.) IFILT=4
      IF(TIMCON.LT.-1.9)IFILT=3
      IF(ABS(TIMCON).LT..001)IFILT=1
C
      IUFILT=0
      IF(ABS(TIMCON+5.).LT..1)IUFILT=5
```

```
      C
            IF(CFTR.LT..0)CFTR=1.
      C
            WRITE(6,6050)NNSER,NPLIMS,CFTR
       6050 FORMAT(//' SERIES SEQUENCE NO. ',I4,
           *   ';  NO. OF PRESSURE LIMITS =',I4//
           *   ' CONDUCTIVITY CELL FACTOR = ',G12.6//)
      C
      C ****** PRESSURE/CYCLE LIMIT CARD(S)
      C
      C                      PTOP,PBOT,DELTAP,I1,I2
      C
      C     PTOP/PBOT        ARE UPPER/LOWER PRESSURE LIMITS TO BE
      C                      APPLIED TO INPUT DATA CYCLES BETWEEN
      C                      CYCLES I1 AND I2. EACH SEQUENCE OF
      C                      CYCLES SPANNING THIS PRESSURE
      C                      INTERVAL IS MADE INTO AN OUTPUT DATA
      C                      SET OR PROFILE. (PRESSURE LIMITS
      C                      ARE IRRELEVANT FOR KCHOP=1.)
      C
      C     DELTAP           IS A SLOP FACTOR (USUALLY ABOUT 2 DB)
      C                      WHICH IS EMPLOYED TO PREVENT SMALL PRESSURE FLUC-
      C                      TUATIONS FROM TRIPPING UP THE PRESSURE LIMIT
      C                      LOGIC. IT ENSURES THAT A DOWNCAST HAS ENDED
      C                      BEFORE THE SUBSEQUENT UPCAST IS RECOGNIZED, ETC.
      C
      C     I1,I2            THERE IS A GLOBAL CYCLE INDEX, SAY IE, WHICH
      C                      IS SET TO 1 AT THE BEGINNING OF AN INPUT DATA
      C                      SET. IT IS INCREMENTED FOR EACH CYCLE ENCOUNTERED
      C                      WITHIN THE DATA SET. (AN INPUT DATA SET TYPICALLY
      C                      SPANS MORE THAN ONE SEGMENT AND OCCASIONALLY
      C                      MORE THAN ONE FILE.) THE FIRST PROFILE
      C                      START, DEFINED BY PTOP, IS NOT SOUGHT
      C                      UNTIL IE IS GE I1. AFTER EACH PROFILE
      C                      IS COMPLETED, IF IE IS GE I2, THE NEXT PRESSURE
      C                      LIMIT CARD IS INVOKED. WHEN THE CARDS ARE EX-
      C                      HAUSTED OR INPUT DATA SET ENDS, THE NEXT
      C                      DATA SET CARD IS READ. IF THE ENTIRE INPUT
      C                      DATA SET IS TO BE SCANNED, SET I1=I2=0.
      C
      C
            II=1
            DO 30 L=1,NPLIMS
            READ(5,1000)(PL(I,II),I=1,3),(IPLIM(I,II),I=1,2)
            IF(IPLIM(1,II).LE.0)IPLIM(1,II)=1
            IF(IPLIM(2,II).LE.0)IPLIM(2,II)=1000000
      C
            WRITE(6,6010)NNSER,(PL(I,II),I=1,3),(IPLIM(I,II),I=1,2)
       6010 FORMAT(1X,'SERIES',1X,I2,';',3X,'PTOP=',F5.0,1X,';',3X,'PBOT=',
           *   F5.0,';',3X,'DELTAP=',
           &F5.0,';',3X,'CYCLE1=',I7,';',3X,'CYCLE2=',I7)
      C
            II=II+1
         30 CONTINUE
      C
            IF(IFILT.EQ.4.AND.ISW.EQ.1) GO TO 95
            GO TO (21,22,23,100),IFILT
      C
      C            WHOI FILTER GENERATION SECTION
      C
         22 IF(NWTT.LT.3)NWTT=3
            CALL GENWHF(NWTT,TIMCON,WT)
      C
            WRITE(6,6075)NWTT,TIMCON
       6075 FORMAT(/' WOODS HOLE DECONVOLUTION CHOSEN'//
           *   ' NO. OF WEIGHTS = ',I4,
           *   '   TIME CONSTANT = ',G11.5//)
      C
            NT=(NWTT+1)/2
            NC=NT
            NWTC=NWTT
            DO 26 I=1,NWTC
         26 WC(I)=1./NWTC
      C
            GO TO 21
      C
      C   SPECIAL FILTER INPUT SECTION
      C
         23 WRITE(6,2310)IUFILT
       2310 FORMAT(//' SUPPLIED TEMPERATURE, CONDUCTIVITY FILTERS'
           *   /' TO BE READ ON UNIT',I4//)
      C
      C ****** SPECIAL FILTERS: NO. TEMP WTS, NO. COND WTS,
```

```
C                              TEMP LAG, TEMP WTS, COND WTS.
C
C                         THESE VALUES ARE READ FROM CARDS IF TIMCON=-6.
C                         OTHERWISE (I.E. TIMCON=-2,-3,-4,-6,...,)
C                         THEY ARE READ FROM UNIT 8. IF TIMCON BE -1,
C                         THIS READ STATEMENT IS SKIPPED.
C
      READ(IUFILT,1000,END=999) NWTT,NWTC,LAGT,(WT(I),I=1,NWTT),
     *    (WC(I),I=1,NWTC)
C
      WRITE(6,7601)NWTT,NWTC,LAGT
 7601 FORMAT(/' NO. OF TEMPERATURE WEIGHTS: ',I3//
     *    ' NO. OF CONDUCTIVITY WEIGHTS: ',I3//
     *    ' TEMPERATURE LAG = ',I2,' DATA CYCLES'/)
C
C
      NT=(NWTT+1)/2
      NC=(NWTC+1)/2
      NTCHK=NT*2-1
      NCCHK=NC*2-1
      IF(NWTT.EQ.NTCHK.AND.NWTC.EQ.NCCHK) GO TO 21
C
      WRITE(6,2800)
 2800 FORMAT(//' NO. OF WEIGHTS MUST BE ODD. TRY AGAIN.'//)
      STOP
C
C  CREATE PRESSURE SMOOTHING FILTER.
C
   21 WRITE(6,2100)
 2100 FORMAT(/' PRESSURE SMOOTHING FILTER'/)
C
      CALL GENFLT(NP,JA,JB,JC,F,WP)
C
      NWTP=2*NP-1
      SUMP=0.
      SUMC=0.
      SUMT=0.
      DO 25 I=1,NWTP
   25 SUMP=SUMP+WP(I)
      DO 27 I=1,NWTT
   27 SUMT=SUMT+WT(I)
      DO 28 I=1,NWTC
   28 SUMC=SUMC+WC(I)
C
      WRITE(6,2700)NWTP,SUMP,NWTT,SUMT,NWTC,SUMC
 2700 FORMAT(//' SUM OF ',I3,'   PRESSURE WEIGHTS = ',G11.5//
     *    ' SUM OF ',I3,'   TEMPERATURE WEIGHTS = ',G11.5//
     *    ' SUM OF ',I3,'   CONDUCTIVITY WEIGHTS = ',G11.5//)
C
C                 ****** PROCES ONE DATA SET. ******
C
  100 CALL ACCESS(N,LN)
      ISW=2
C
C
      GO TO 15
C
  999 WRITE(6,6040)
 6040 FORMAT(1X,'***END OF JOB***')
      STOP
C
   95 WRITE(6,9500)
 9500 FORMAT(//' PREVIOUS FILTER SPECIFIED ON INITIAL PASS.'
     *    ' INCONSISTENT.'//)
      STOP
C
      END
```

```
@PRT,S PF*UO74*RGD.ACCESS
```

```
PF*UD7*PRGD(1).ACCESS
              COMPILER(DIAG=3)
              SUBROUTINE ACCESS(N,LN)
        C
              COMMON/WMAT/NIND,IND(20),KCHOP,NPLINS,PL(3,100),IPLIN(2,100),
             &NWMAX,IPWSET(20),IEOLU
              COMMON/RDOCF/FDOCR(40)
              COMMON/RDOCI/IDOCR(40)
              COMMON/RDOCA/ADOCR(100)
              COMMON / WHDR / LW,NW,NBW,NNBW,NNFW,NFW,NIW,NAW,IPW(20)
              COMMON / RHDR / LR,NR,NBR,NNBR,NNFR
              COMMON /WDOCF/FDOCW(40) /WDOCI/IDOCW(40) /WDOCA/ADOCW(100)
              COMMON / WDATA / VW(10010)
        C
        C
              COMMON / DIAGS / NSBR,NSBW,NNNR,NNNW,NNIP,NNF,NNI,NNA,IRST,IWST
              DATA NNNR,NNNW,NNIP,NNF,NNI,NNA/1000,1000,13,40,40,74/
              DATA NFW,NIW,NAW/40,40,100/
        C
              DIMENSION W(200),ICT(200),RECOUT(4,200),REMOUT(20)
              DIMENSION Q(20),SAVREC(4)
              DATA JA,JB,JC/2,1,3/
              LOGICAL IEOLU
              KCHOP1=KCHOP
              IDOCW(5)=0
              IF(N.EQ.1)LN=1
              IF(KCHOP.NE.1)GO TO 300
              N=0
              IUD=2
          300 CONTINUE
              IDOCW1=0
              KNTZ=0
              IDOCW(4)=0
              IF(NNBW.NE.NNBWP)IDOCW(5)=0
              IDOCW(5)=IDOCW(5)+1
              JPX=2
              JN2X=0
              IF(IND(8).EQ.0)GO TO 325
              DO 324 IX=1,8
          324 IF(IND(IX).NE.0)JN2X=JN2X+1
          325 CONTINUE
              NW=0
              LW=NIND
              IF(NWMAX.GT.NNNW)NWMAX=NNNW
              DO 301 J=1,LW
              IPW(J)=IPWSET(J)
          301 CONTINUE
              INIT=1
              ISW=2
              IF(N.LT.1)ISW=1
              IEOLU=.FALSE.
          101 GO TO (1,2),ISW
        C  STRAIGHT-THROUGH PROCESSING SECTION
            1 CALL GETREC(590,Q,IUD)
              KUD=IUD*2-3
        C  DERIVE OPTION
           70 IF(IEOLU)RETURN
              CALL DERIVE(Q,INIT)
              INIT=0
           80 NW=NW+1
              IJ0=(NW-1)*LW
              DO 71 J=1,LW
              IJ=IJ0+J
           71 VW(IJ)=Q(J)
              IF(NW.EQ.NWMAX+1) GO TO 95
              GO TO 1
        C  DECIMATION SECTION
        C  GET FIRST RECORD
        C      IUD=1:  UP PROFILE      IUD=2:  DOWN PROFILE
            2 CALL GETREC(590,Q,IUD)
              KUD=IUD*2-3
        C  SET FIRST INTERVAL
              P=Q(JPX)
              IP=P*10.+.5
              IP1=IP
              IP0=IP1+LN*N*KUD
              IP2=IP0+LN*N*KUD
              IPP=IP
              N1=1
              M=LN*N
              NWT=2*M-1
              IF(N.EQ.1) GO TO 72
              IF(N.EQ.NPREV.AND.LN.EQ.LNPREV) GO TO 72
```

```
      FO=1./N
      WRITE(6,9311)
 9311 FORMAT(/' DECIMATION FILTER'/)
      CALL GENFLT(N,JA,JB,JC,FO,W)
      WRITE(6,7801)
 7801 FORMAT(//)
      NPREV=N
      LNPREV=LN
      GO TO 72
C   GET NEXT RECORD
   75 CALL GETREC(S9D,Q,IUD)
      KUD=IUD*2-3
   73 P=Q(JPX)
      IP=P*10.+.5
C   CHECK FOR P-REVERSAL
   72 IF((IP-IPP)*KUD.LT.0) GO TO 75
      IPP=IP
C   CHECK FOR P OUT OF INTERVAL
      IF((IP-IP1)*KUD.LE.0) GO TO 75
      IF((IP-IP2)*KUD.GE.0) GO TO 60
C   LOAD RECORD INTO FILTER ARRAY
      JP=(IP-IP1)*KUD
      ICT(JP)=ICT(JP)+1
      DO 76 J=3,4
   76 RECOUT(J,JP)=RECOUT(J,JP)+Q(J)
      RECOUT(1,JP)=RECOUT(1,JP)+Q(1)
      GO TO 75
C   COMPLETE AVERAGING
   60 DO 65 K=K1,NWT
      KCT=ICT(K)
      IF(KCT.NE.0) GO TO 161
      KNTHOL=KNTHOL+1
      KNT2=KNT2+1
      DO 163 J=3,4
  163 RECOUT(J,K)=SAVREC(J)
      RECOUT(1,K)=SAVREC(1)
      GO TO 65
  161 CONTINUE
      IF(KNT2.GT.1)WRITE(6,3981)KNT2,PO
 3981 FORMAT(1X,'GAP OF',I5,1X,'AT P =',G12.6)
      KNT2=0
      DO 63 J=3,4
      RECOUT(J,K)=RECOUT(J,K)/KCT
   63 SAVREC(J)=RECOUT(J,K)
      RECOUT(1,K)=RECOUT(1,K)/KCT
      SAVREC(1)=RECOUT(1,K)
   65 CONTINUE
C   IF NO FILTER, SKIP
      IF(N.EQ.1) GO TO 61
C   CONVOLVE FILTER
      WSUM=0.
      JCT=0
      DO 64 J=3,4
   64 REKOUT(J)=0.
      REKOUT(1)=0.
      DO 66 K=1,NWT
      IF(ICT(K).EQ.0) GO TO 66
      WSUM=WSUM+W(K)
      JCT=JCT+1
      DO 67 J=3,4
   67 REKOUT(J)=REKOUT(J)+RECOUT(J,K)*W(K)
      REKOUT(1)=REKOUT(1)+RECOUT(1,K)*W(K)
   66 CONTINUE
      IF(JCT.EQ.0.OR.JCT.EQ.NWT) GO TO 69
      DO 68 J=3,4
   68 REKOUT(J)=REKOUT(J)/WSUM
      REKOUT(1)=REKOUT(1)/WSUM
      GO TO 69
   61 DO 62 J=3,4
   62 REKOUT(J)=RECOUT(J,1)
      REKOUT(1)=RECOUT(1,1)
   69 CONTINUE
      PO=FLOAT(IPO)/10.
      REKOUT(2)=PO
C RECORD NOW COMPLETE
C   DERIVE OPTION
      CALL DERIVE(REKOUT,INIT)
      INIT=0
C   LOAD INTO OUTPUT ARRAY
   81 NW=NW+1
      IJO=(NW-1)*LW
      DO 51 J=1,LW
      IJ=IJO+J
```

```
   51 VW(IJ)=RECOUT(J)
      IF(NW.EQ.NWMAX+1)GO TO 95
C   RESET ALL SUMS AND COUNTERS
   48 IP0=IP0+N*KUD
      IP1=IP0-LN*N*KUD
      IP2=IP0+LN*N*KUD
      K1=NWT-N+1
      NWTMN=NWT-N
      DO 45 K=1,NWTMN
      KK=K+N
      ICT(K)=ICT(KK)
      RECOUT(1,K)=RECOUT(1,KK)
      DO 45 J=3,4
   45 RECOUT(J,K)=RECOUT(J,KK)
      DO 46 K=1,N
      KK=NWT-K+1
      ICT(KK)=0
      RECOUT(1,KK)=0.
      DO 46 J=3,4
   46 RECOUT(J,KK)=0.
 47   IF(IEQ.U) RETURN
      IF(IDOCW1.EQ.1)GO TO 300
      GO TO 73
   55 CALL GETREC(590,0,IUD)
      KUD=IUD*2-3
   96 IDOCW(1)=0
C** MORE STUFF HERE **
   97 CONTINUE
C     SHIFT N**2
      IF(JN2X.EQ.0)GO TO 122
      NWM1=NW-1
      JBVF1=JN2X
      DO 121 J=1,NWM1
      VW(JBVF1)=VW(JBVF1+LW)
      JBVF1=JBVF1+LW
 121  CONTINUE
 122  CONTINUE
      IF(NW.GT.NWMAX)NW=NWMAX
      NMBW=NMBR
      NMFW=NMFR
      GO TO (93,92),IUD
   92 IDOCW(21)=0
      IDOCW(4)=IDOCW(4)+1
      DO 981 IQ=1,40
 981  FDOCW(IQ)=FDOCR(IQ)
      IDOCW(2)=IDOCR(2)
      IDOCW(3)=IDOCR(3)
      IDOCW(10)=IDOCR(10)
      IDOCW(11)=IDOCR(11)
      IDOCW(12)=IDOCR(12)
      DO 982 IQ=1,NAW
 982  ADOCW(IQ)=ADOCR(IQ)
      CALL ZWRIT(2,IF,0)
      NMBWP=NMBW
      GO TO 145
   93 IDOCW(21)=1
      IDOCW(4)=IDOCW(4)+1
      DO 983 IQ=1,40
 983  FDOCW(IQ)=FDOCR(IQ)
      IDOCW(2)=IDOCR(2)
      IDOCW(3)=IDOCR(3)
      IDOCW(10)=IDOCR(10)
      IDOCW(11)=IDOCR(11)
      IDOCW(12)=IDOCR(12)
      DO 984 IQ=1,NAW
 984  ADOCW(IQ)=ADOCR(IQ)
      CALL ZWRIT(3,IF,0)
      NMBWP=NMBW
 145  IF(NNTHOL.GT.0)WRITE(6,6910)NNTHOL,NBW
 6910 FORMAT(1X,I5,1X,*HOLES IN DATA REPLACED WITH PRECEDING*,
     &1X,*RECORD IN SEGMENT*,1X,I5)
      NNTHOL=0
C
 94   NW=1
      IDOCW(1)=0
C     SHIFT RECORD NW+1 TO 1
      NWMAXL=NWMAX*LW
      DO 120 J=1,LW
      VW(J)=VW(NWMAXL+J)
 120  CONTINUE
      IF(IF.GT.1) GO TO 98
      GO TO (70,48),ISW
   90 CONTINUE
```

```
      IDOCW(1)=1
      IDOCWI=1
      INII=1
      DO 405 IK99=1,NNT
      ICT(IK99)=0
      DO 405 IK4=1,4
405   RECOUT(IK4,IK99)=0
      IF(NW.EQ.0)GO TO 97
      GO TO 97
   58 WRITE(6,9800) IF
 9800 FORMAT(//' PROBLEM IN ZWRIT'//)
      STOP
      END
```

```
@PRT,S PF*U074PRGD.GETREC
```

```
PF*U074PRGD(1).GETREC
      COMPILER(DIAG=3)
      SUBROUTINE GETREC(*,Q,IUD)
C
      DIMENSION Q(20)
      COMMON/WHAT/NIND,IND(20),KCHOP,NPLIMS,PL(3,100),IPLIM(2,100),
     *   NNMAX,IPWSET(20),IEOLU,NMSER
      COMMON/CALIB/MP,MT,MC,LAGT,WP(60),WT(60),WC(60),PDIF,
     *   TDIF,CDIF,CFTR
      COMMON/RHDR/LR,NR,NBR,NNBR,NMFR,NFR,NIR,NAR,IPR(20)
      COMMON /RDATA/ VR(4,1100)
      COMMON/RDOCF/FDOCR(40)
      COMMON/RDOCI/IDOCR(40)
      COMMON/RDOCA/ADOCR(100)
      COMMON/DIAGS/MSGR,MSGW,NNNR,NNNW,NNIP,NNF,NNI,NNA,IRST,IWST
      DATA NNNR,NNIP,NNF,NNI,NNA/1000,20,40,40,100/
      DATA ISW/1/,JM,JP,JT,JC/1,2,3,4/
      DATA IIU/4/
      DATA ISET,INEXT/1,1/
      LOGICAL IEOLU
      GO TO (6,7),ISET
    6 CONTINUE
      IDOCR(1)=0
      IIT=MT+LAGT
      I1=MC
      IF(IIT.GT.MC)I1=IIT
      IF(MP.GT.I1)I1=MP
      MMT=2*MT-1
      MMC=2*MC-1
      MMP=2*MP-1
      ISET=2
      IRST=2*I1-1
    7 CONTINUE
C
C INITIALIZE
C
      GO TO (1,2),ISW
    1 NSW=4
      CALL IUSET
  780 NMSER1=NID1+1
      IF(NMSER1.EQ.NMSER)GO TO 781
      CALL ZREAD(IIU,IF,Q)
      IF(IF.EQ.1)GO TO 782
      IF(IDOCR(1).EQ.1)NID1=NID1+1
      GO TO 780
  782 CALL IUSET
      GO TO 780
  781 CONTINUE
      IDOCR(1)=0
      ISW=2
      IE=I1-1
      ICPF=0
      IBL1=-1
      GO TO 5
    4 NSW=1
  783 IE=IE-1
```

```
      CALL IUSET
    5 CONTINUE
      NNNR1=NNNR
C
    2 IE=IE+1
C
      IBL=((IE+I1-2+ICPF)/1000)+1
      IF(IBL.EQ.IBL1)GO TO 160
      IF(IDOCR(1).EQ.1)GO TO 20
      IF(IBL1.EQ.-1)GO TO 133
      IRST=2*I1-1
      IF(I1.EQ.1)GO TO 133
      IBEG=NR+1
      NRST=NR+IRST-1
      J=0
      DO 131 I=IBEG,NRST
      J=J+1
      DO 132 II=1,LR
  132 VR(II,J)=VR(II,I)
  131 CONTINUE
  133 CONTINUE
      CALL ZREAD(IIU,IF,0)
      IF(IF.EQ.1)GO TO 783
      NRST=NR+IRST-1
      JIN=INEXT-1
      IF(JIN.EQ.0)JIN=1
      IF(IE.LT.IPLIM(1,JIN))GO TO 190
      IRSTP=IRST+IPLIM(1,JIN)-1
      JJ=0
      DO 191 J=IRST,NRST
      JJ=JJ+1
      JM1=J-1
      JP1=J+1
      JJM1=JJ-1
      IF(IBL1.EQ.-1.AND.JM1.LE.IRSTP)GO TO 191
      DIFP=ABS(VR(2,J)-VR(2,JM1))
      DIFT=ABS(VR(3,J)-VR(3,JM1))
      DIFC=ABS(VR(4,J)-VR(4,JM1))
      DIFPP=ABS(VR(2,JP1)-VR(2,JM1))
      DIFTT=ABS(VR(3,JP1)-VR(3,JM1))
      DIFCC=ABS(VR(4,JP1)-VR(4,JM1))
      IF(DIFP.LT.PDIF.OR.PDIF.LE.0)GO TO 192
      IF(DIFPP.GE.(PDIF))GO TO 192
      WRITE(6,6192)NBR,JJ,IPR(2),VR(2,J),VR(2,JM1)
 6192 FORMAT(1X,'SEG',1X,I5,1X,'CYCLE',I5,1X,A6,1X,F10.3,1X,
     &'REPLACED BY',1X,F10.3,T132,'$')
      VR(2,J)=VR(2,JM1)
  192 IF(DIFT.LT.TDIF.OR.TDIF.LE.0)GO TO 193
      IF(DIFTT.GE.(TDIF))GO TO 193
      WRITE(6,6192)NBR,JJ,IPR(3),VR(3,J),VR(3,JM1)
      VR(3,J)=VR(3,JM1)
  193 IF(DIFC.LT.CDIF.OR.CDIF.LE.0)GO TO 191
      IF(DIFCC.GE.(CDIF))GO TO 191
      WRITE(6,6192)NBR,JJ,IPR(4),VR(4,J),VR(4,JM1)
      VR(4,J)=VR(4,JM1)
  191 CONTINUE
  190 CONTINUE
      IF(IDOCR(1).EQ.1)NID1=NID1+1
      IF(NR.LT.1000)ICPF=1000-NR+ICPF
      IEE=I1
      IF(IBL1.EQ.-1)IEE=I1+IRST-1
      NNNR1=NR
      IBL1=IBL
      IF(IF.EQ.0)GO TO 160
      IF(IF.GT.1)GO TO 22
      GO TO 4
   22 CONTINUE
      WRITE(6,9000)IF
 9000 FORMAT(1X,'***ERROR IN ZREAD, IF=',I1,'***')
      STOP
C
  160 CONTINUE
C
C CONVOLVE FILTERS WITH DATA
C
      DO 11 J=1,4
   11 Q(J)=0.
      Q(1)=VR(JN,IEE)
      DO 12 I=1,NMT
      IT=IEE-MT+I+LAGT
   12 Q(3)=Q(3)+VR(JT,IT)*WT(I)
      DO 13 I=1,NMC
      IC=IEE-MC+I
```

```
      Q(4)=Q(4)+VR(JC,IC)*WC(I)
   13 CONTINUE
      DO 14 I=1,NMP
      IP=IEE+I-MP
      Q(2)=Q(2)+VR(JP,IP)*WP(I)
   14 CONTINUE
C
C APPLY MANUFACTURERS CELL CORRECTION AND CFTR
C
      CALL NBCCC(Q(2),Q(3),Q(4),CFTR)
C
      IEE=IEE+1
C
C KCHOP=1:ALL DATA,    KCHOP=2:UP AND DOWN CAST
C KCHOP=3 DOWN CAST ONLY    KCHOP =4:UP CAST ONLY
      GO TO (19,100,100,100),KCHOP
C
   19 IF(INEXT.GT.NPLIMS)GO TO 200
      IE1=IPLIM(1,INEXT)
      IENEXT=IPLIM(2,INEXT)
      IF(IE.LT.IE1)GO TO 2
      IF(IE.LT.IENEXT)GO TO 25
      INEXT=INEXT+1
      GO TO 19
C END OF LU OR FILE
C
  200 INEXT=1
   20 ISW=1
      ISET=1
      EOLU=.TRUE.
      NID1P1=NID1+1
      IF(IDOCR(1).EQ.1)NID1P1=NID1
      IF(NMSER.NE.NID1P1)WRITE(6,9432)NID1P1,NMSER
 9432 FORMAT(///1X,'CONGRATULATIONS TURKEY!!! YOU HAVE TRIED TO PROCESS
     & SERIES',I3,'INSTEAD OF SERIES',I3,1X/' CHECK PRESSURE LIMITS'///)
      RETURN1
   25 RETURN
C
  100 P=Q(JP)
C
      GO TO (110,110,110,122),KSW
C
C SET UP P-LIMITS
C
  122 P1T=PL(1,1)
      IF(P1T.LE.0.)P1T=AINT(P+1.)
      P2T=P1T+PL(3,1)
      P1B=PL(2,1)
      IF(P1B.LE.0.)P1B=99000.
      P2B=P1B-PL(3,1)
      IE1=IPLIM(1,1)
      IENEXT=IPLIM(2,1)
      INEXT=2
      IF(IE.LT.IE1)GO TO 2
C
C SEARCH FOR FIRST PROFILE START
  114 GO TO (140,140,140,141),KCHOP
  140 IF(P.LT.P1T)GO TO 117
      GO TO (141,141,142,141),KCHOP
  141 IF(P.GT.P1B)GO TO 118
  142 GO TO 2
C
C SKIP FOLLWING RUBBISH IF PLIM EXHAUSTED
C
C
C
  110 IF(IE.LT.IE1)GO TO 2
C
      GO TO (111,117,118,114),KSW
C
C
C CHECK FOR P INSIDE LIMITS
C
  111 IF(P.GT.P1B) GO TO 113
      IF(P.LE.P1T) GO TO 112
      GO TO (116,115),IUD
C
C END OF PROFILE
C
  112 KSW=2
      GO TO (151,151,2,151),KCHOP
  113 KSW=3
      GO TO (151,151,151,2),KCHOP
```

```
  151 CONTINUE
      ICOLU=.FALSE.
      IF(IE.GE.IPLIN(2,NPLINS))GO TO 20
      IF(IDOCR(1).EQ.1)GO TO 20
      IF(IE.LE.IENEXT)GO TO 124
      IF(INEXT.GT.NPLINS)GO TO 124
C INCREMENT PLINS
      P1T=PL(1,INEXT)
      P1B=PL(2,INEXT)
      P2T=P1T+PL(3,INEXT)
      P2B=P1B-PL(3,INEXT)
      IE1=IPLIN(1,INEXT)
      IENEXT=IPLIN(2,INEXT)
      INEXT=INEXT+1
  124 CONTINUE
      RETURN1
C
C AT TOP
C
  117 KSW=2
      IF(P.LT.P2T) GO TO 2
      IUD=2
      KSW=1
  115 GO TO (25,25,25,2),KCHOP
C
C AT BOTTOM
C
  118 KSW=3
      IF(P.GT.P2B) GO TO 2
      IUD=1
      KSW=1
  116 GO TO (25,25,2,25),KCHOP
C
      END
```

@PRT,S PF*U074PRGD.IUSET

```
PF*U074PRGD(1).IUSET
      COMPILER(DIAG=3)
      SUBROUTINE IUSET
      COMMON /IOFILE/NINFIL,IFEB(120)
      DIMENSION LABL(6)
      DATA IFILE/0/
      IFILE=IFILE+1
      I=(IFILE-1)*4+1
      II3=I+3
      IF(IFILE.GT.NINFIL)GO TO 900
      ENCODE(36,12,LABL)(IFEB(J),J=I,II3)
   12 FORMAT('@USE 4.,',4A6,' . ')
      CALL ERTRAN(6,'@FREE,A 4. . ')
      CALL ERTRAN(6,LABL)
      WRITE(6,6000)(IFEB(J),J=I,II3)
 6000 FORMAT(1X,'FILE ASSIGNED',1X,4A6)
      RETURN
  900 IF(IFILE.EQ.1) GO TO 901
      I=I-4
      II3=II3-4
      WRITE(6,9000)(IFEB(K),K=I,II3)
 9000 FORMAT(//' PROGRAM TERMINATED BEFORE REACHING LIMIT,'
     *   ' AT END OF INPUT FILE ',4A6)
      STOP
  901 WRITE(6,9001)
 9001 FORMAT(//' NO INPUT FILE NAMES SUPPLIED.'//
     *   ' INPUT FILE ASSUMED ON UNIT 4'//)
      RETURN
      END
```

@PRT,S PF*U074PRGD.GENWHF

```
PF*UD74PRGD(1).GENWHF
        SUBROUTINE GENWHF(NWT,TIMCON,W)
C
C**     NWT = WOODS HOLE FILTER SIZE ,
C**     TIMCON = TIME CONSTANT/SAMPLING TIME INTERVAL OF INSTRUMENT .
C**     W = WOODS HOLE FILTER ARRAY NAME
C
        DIMENSION W(1)
        FNWT=FLOAT(NWT)
        IF(NWT.EQ.0) RETURN
        DO 10 I=1,NWT
        F=FLOAT(I)
        W(I)=(1./FNWT)+((TIMCON)*(12.*F-6.*(FNWT+1.))/
       1  (FNWT*(FNWT**2.-1.)))
     10 CONTINUE
        RETURN
        END

@PRT,S PF*UD74PRGD.GENFLT
```

```
PF*UD74PRGD(1).GENFLT
C*********************************************************
C
C GENFLT IS ESSENTIALLY GENER1 FROM THE FESTSA PACKAGE.
C (SEE: FAST AND EASY TIME SERIES ANALYSIS AT NCSU -
C EDITED BY D. A. BROOKS - AN NCSU INTERNAL REPORT)
C
C   M - HALF-WIDTH OF FILTER
C   JA - 1,-1 FOR ONE-SIDED FILTER FADING TO THE RIGHT,LEFT
C         2 FOR A SYMMETRICAL FILTER OF (2M-1) TERMS.
C   JB - 0 FOR LAG WINDOW APPLICATION, LARGEST TERM=1.
C         1 FOR LOW PASS FILTER.
          2 FOR HIGH PASS FILTER.
C   JC - 1 BARTLETT TAPER
C         2 PARZEN TAPER
C         3 COSINE(TUKEY) TAPER
C         4 LANCZOS TAPER
C          5 LANCZOS SQUARED TAPER
C
C   FILTER WEIGHTS ARE RETURNED IN SER
C
C*********************************************************
C
         COMPILER(DIAG=3)
         SUBROUTINE GENFLT(M,JA,JB,JC,F,SER)
        WRITE(6,6666)M,JA,JB,JC,F
   6666 FORMAT(' GENFLT: M=',I6,'  JA=',I3,'  JB=',I3,
     *    '  JC=',I3,'  F=',G12.6)
         DIMENSION SER(1)
        PI=3.14159265
        FM=M
        DO 100 I=1,M
        FI=I
        U=(FI-1.)/FM
        GO TO(40,50,60,61,61),JC
     61 IF(U-.0000001)40,40,63
     63 IF(JC-4)64,64,65
     64 SER(I)=  SIN(PI*U) /(PI*U)
        GO TO 70
     65 SER(I)=(SIN(PI*U)/(PI*U))**2
        GO TO 70
     40 SER(I)=1.-U
        GO TO 70
     60 SER(I)=.5*(1.+COS(PI*U))
        GO TO 70
     50 IF(ABS(U)-.5)51,52,52
     51 SER(I)=1.-6.*U**2+6.*ABS(U**3)
        GO TO 70
     52 SER(I)=2.*(1.-ABS(U))**3
     70 IF(F-.00000001)100,100,71
     71 IF(ABS(PI*(FI-1.)*F)-.00001)100,100,73
     73 SER(I)=SER(I)*SIN(PI*(FI-1.)*F)/(PI*(FI-1.)*F)
```

```
1C0 CONTINUE
1C2 IF(JB)130,130,103
1C3 SUM=.5*SER(1)
    DO 104 I=2,M
1C4 SUM=SUM+SER(I)
    DO 129 I=1,M
110 SER(I)=SER(I)/(2.*SUM)
    IF(JB-1)129,129,120
120 SER(I)=-SER(I)
    IF(I-1)129,121,129
121 SER(I)=SER(I)+1.
129 CONTINUE
130 JAB=JA+2
    GO TO(150,150,140,180),JAB
140     CONTINUE
    RETURN
150 MB2=M/2
    DO 155 I=1,MB2
    X=SER(I)
    II=M-I+1
    SER(I)=SER(II)
155 SER(II)=X
    RETURN
180 MM=2*M-1
CO    WRITE(6,6900)(I,SER(I),I=1,MM)
69C0 FORMAT(I5,G12.6)
    DO 185 I=M,1,-1
    II=I+M-1
185 SER(II)=SER(I)
    MM1=M-1
    DO 181 I=1,MM1
    II=MM-I+1
181 SER(I)=SER(II)
    RETURN
    END
```

@PRT,S PF*UD74PRGD.DERIVE

```
PF*UD74PRGD(1).DERIVE
          COMPILER(DIAG=3)
          SUBROUTINE DERIVE(Q,INIT)
C
          DIMENSION Q(20),KND(20)
          COMMON /WHAT/NIND,IND(20)
          COMMON/SALINE/KSAL
          DATA CF,ISKIP/1.,1/
          LOGICAL IOBV,IOPT,IOPD,IOSV,IODT,IOSIG
C
          P=Q(2)
          T=Q(3)
          C=Q(4)
          GO TO (21,22),ISKIP
C
       21 K=0
          ISKIP=2
          DO 1 I=1,20
          IF(IND(I).EQ.0) GO TO 1
          ILAST=I
          K=K+1
          KND(K)=I
          IOBV=(I.EQ.8.OR.IOBV)
          IOSV=(I.EQ.9.OR.IOSV)
          IODT=(I.EQ.10.OR.IODT)
          IOPT=(I.EQ.11.OR.I.EQ.12.OR.IOPT)
          IOPD=(I.EQ.12.OR.IOPD)
          IOSIG=(I.EQ.6.OR.I.EQ.7.OR.IOSIG)
        1 CONTINUE
C
          KSW=4
          IF(ILAST.LT.5)KSW=1
          IF(ILAST.EQ.5) KSW=2
          IF(IOSIG)KSW=3
```

```
      IF(IOSIG.AND.(ILAST.GE.7)) KSW=5
C
   22 GO TO (50,51,51,51,51),KSW
C
   51 GO TO (511,512),KSAL
  511 CALL BENDPH(P,T,C,S)
      GO TO 513
  512 CALL PSS78(P,T,C,S)
  513 Q(5)=S
C
      GO TO (50,50,52,53,52),KSW
C
   52 CALL SIGMA(P,T,S,SIGT,SIG)
      Q(6)=SIGT
      Q(7)=SIG
C
      GO TO (50,50,50,53,53),KSW
C
   53 IF(IOSV)CALL SOUND(T,S,P,SV)
      Q(9)=SV
      IF(IOPT)CALL POTEMP(P,T,S,TPOT)
      Q(11)=TPOT
      IF(IOPD)CALL SIGMA(P,TPOT,S,SIGPOT,XX)
      Q(12)=SIGPOT
      IF(IOBV)CALL BRNVAL(T,P,S,G,INIT,EN,PRIGHT,IXBV)
      Q(8)=EN
      IF(IODT)CALL TRUDEP(P,SIG,INIT,DEP)
      Q(10)=DEP
C
   50 CONTINUE
C
      DO 2 I=1,NIND
      K=KND(I)
    2 Q(I)=Q(K)
C
      RETURN
C
      END
```

@PRT,S PF*UD74PRGD.NBCCC

```
PF*UD74PRGD(1).NBCCC
      COMPILER(DIAG=3)
C
      SUBROUTINE NBCCC (P,T,C,CF)
C
C NBCCC CORRECTS FOR TEMPERATURE AND PRESSURE EFFECTS ON CELL
C AND APPLIES CALIBRATION CORRECTION, CF.
C
      DATA ALPHA/6.5DE-6/,FK/67.5E6/
C
      C=CF*C*(1.-ALPHA*(T-15.)+(P*1.45038)/FK)
C
      RETURN
      END
```

@PRT,S PF*UD74PRGD.BENDPH

```
PF*U074PRGD(1).BENDPH
          COMPILER(DIAG=3)
C
           SUBROUTINE BENDPH(P1,T1,C1,SPPT)
C
C THIS ROUTINE USES THE BENNETT(1976), AND DAUPHINEE(1976),
C EQUATIONS TO COMPUTE SALINITY FROM PRESSURE, TEMPERATURE,
C AND CONDUCTIVITY.
C
C         P1:PRESSURE IN DECIBARS
C         T1:TEMPERATURE IN DEGREES CELSIUS
C         C1:CONDUCTIVITY IN MMHO/CM
C         RPRIM:CONDUCTIVITY RATIO
C
C          COMPUTE CONDUCTIVITY RATIO
          RPRIM=C1/42.906
C
C         CORRECT RPRIM TO ZERO PRESSURE
           F=((6.166E-15*P1-5.4805E-10)*P1+1.60836E-5)*P1/
     1     ((3.169E-4*T1+3.0786E-2)*T1+1.)
          RST=RPRIM/(1.+F)
C
C         CORRECT RST FOR TEMPERATURE EFFECTS AT 35PPT WHERE
C         RTW=C 35,T,0/C 35,15,0
C
          RTW=((((1.35868E-9*T1-7.26682E-7)*T1+1.11099E-4)*T1+2.005294E-2)
     1 *T1+.6765524
          RS=RST/RTW
C
C         CALCULATE SALINITY
         RSP=((((-1.32311)*RS+5.98624)*RS-10.61869)*RS+12.18882)*RS+28.05671)
     1*RS-.08996
           RSRSM1=RS*(RS-1.)*((4.42E-2-.46E-3*T1-4.E-3*RS)*T1
     1          +(1.25E-4-2.9E-6*T1)*P1)
           SPPT=RSP+RSRSM1
C
           RETURN
           END

@PRT,S PF*U074PRGD.PSS78
```

```
PF*U074PRGD(1).PSS78
          COMPILER(DIAG=3)
           SUBROUTINE PSS78(P,T,C,SAL)
C THIS ROUTINE USES THE PRACTICAL SALINITY SCALE (1978)
C EQUATIONS TO COMPUTE SALINITY FROM PRESSURE,
C TEMPERATURE AND CONDUCTIVITY .
C (IEEE JOURNAL (JAN 1980))
C
C         P1:PRESSURE (DB)
C         T1:TEMPERATURE (DEG-C)
C         C1:CONDUCTIVITY (MMHO/CM)
C         RP:CONDUCTIVITY RATIO.
C
C COMPUTE CONDUCTIVITY RATIO
C
        RP=C/42.914
C
          ALPH=(2.070E-5*P-6.370E-10*P**2.+3.989E-15*P**3.)
     &  /(1+3.426E-2*T+4.464E-4*T**2.+
     &  4.215E-1*RP-3.107E-3*RP*T)
C
C  ALPH=F(R,T,P) IS THE FRACTIONNAL INREASE IN COND DUE TO PRESS.
C
          SWTCO=6.766097E-1+2.00564E-2*T+1.104259E-4*T**2.
     &  -6.9698E-7*T**3.+1.0031E-9*T**4.
C
C  THIS IS : C(35,T,0)/C(35,15,0)
C
          RRT=RP/(SWTCO*(1.+ALPH))
C
```

```
      SAL=(0.0080-0.1692*RRT**0.5+25.3851*RRT
     & +14.0941*RRT**1.5-7.0261*RRT**2.+2.7081*RRT**2.5)
     & +(((T-15.)/(1.+(0.0162*(T-15.))))*
     & (0.0005-0.0056*RRT**0.5-0.0066*RRT
     & -0.0375*RRT**1.5+0.0636*RRT**2.
     & -0.0144*RRT**2.5))
C
      RETURN
      END
```

@PRT,S PF*U074PRGD.SIGMA

```
PF*U074PRGD(1).SIGMA
      SUBROUTINE  SIGMA(P,T,S,SIGT,SIGSTP)
      TM398=T-3.98
      SIGSIG=(-TM398*TM398*(283.+T))/(503.57*(T+67.26))
      SIG0=S*(S*(.67678614E-5*S-.48249614E-3)+.81487058)
     1 -.93445863 2E-1
      AT=T*(.47867E-2-T*(.98185E-4-T*.10843E-5))
      BT=T*(.18030E-4-T*(.8164E-6-T*.1667E-7))
      SIGT= SIGSIG+(SIG0+.1324)*(1.-AT+BT*(SIG0-.1324))
C     THIS IS SIGMA-T
      SIG28=SIG0-28.
      T1=4886./(1.+1.83E-5*P)
      T2=227.+T*(28.33-T*(.551-T*.0041))
      T3=1.E-4*P*(105.5+T*(9.50-T*.158-1.5E-4*P))
      T4=.1*SIG28*(147.3-T*(2.72-0.04*T)
     1 -1.E-4*P*(32.4-T*(.87-.02*T)))
      T5=1.E-2*(SIG28*SIG28)*(4.5-.1*T-P*(1.8E-4-.6E-5*T))
      XMU=1.E-9*(T1-T2+T3-T4+T5)
      ALFA=(1.-XMU*P)/(SIGT*.001+1.)
      RHOSTP=1./ALFA
      SIGSTP=(RHOSTP-1.)*1000.
C     THIS IS SIGMA-STP
      RETURN
      END
```

@PRT,S PF*U074PRGD.SOUND

```
PF*U074PRGD(1).SOUND
      SUBROUTINE SOUND(T,SAL,PRES,SVEL)
C      REVISED PRESSURE CONVERSION  JMO
C     REFERENCE - WILSON, W.D.,1960, EQUATION FOR THE SPEED OF
C     SOUND IN SEA WATER, JOUR. ACOUST. SOC. AMER., 32(13),1357.
C
C     PRES: PRESSURE IN DECIBARS FROM SEA SURFACE .
C     P : TOTAL PRESSURE IN KG/CM2 ABSOLUTE .
C     SAL : SALINITY IN PARTS PER 1000 .
C     T : TEMPERATURE IN DEGREES CELSIUS .
C     SOUND =  SOUND VELOCITY IN METERS PER SECOND .
C
      P = (PRES + 10.1325) * 0.1019716
      S = SAL -35.
C
      VT = T*(4.5721+T*(-4.4532E-2+T*(-2.6045E-4+7.9851E-6*T)))
      VS = S*(1.39799+1.69202E-3*S)
      VP = P*(0.160272+P*(1.0268E-5+P*(3.5216E-9-3.3603E-12*P)))
C
      VSTP = S*(T*(-1.1244E-2+7.7711E-7*T)+P*(7.7016E-5-1.2943E-7*P+
     1T*(3.1580E-8+1.5790E-9*T)))+
```

```
          2P*(T*(-1.8607E-4+T*(7.4812E-6+4.5283E-8*T)))+
          3P*P*(T*(-2.5294E-7+1.8563E-9*T-1.9646E-10*P))
     C
           SOUND = 1449.14 + VT + VP + VS + VSTP
     C
          SVEL = SOUND
          RETURN
          END
```

```
@PRT,S PF*U074P*GD.TRUDEP
```

```
PF*U074PRGD(1).TRUDEP
            COMPILER(DIAG=3)
           SUBROUTINE TRUDEP(PRESS,SIGSTP,INIT,TRUDP)
            COMMON/MDOCI/IDOCW(1)
            COMMON/RDOCF/FDOCR(1)
     C
            IF(INIT.NE.1) GO TO 2
             XLAT=ABS(FDOCR(3))
          GLAT = XLAT*0.01745329
          TGRAV=980612.3-2586.157*COS(GLAT*2.0)+2.8852446*COS(GLAT*4.0)
            GRAV=1.0E-6*TGRAV
            TRUDP=0.
            PRVPRS=0.
            RHOP=0.
     C
        2  RHO=(SIGSTP/1000.)+1.
             PRS=PRESS
            IF(RHOP.EQ.0.)RHOP=RHO
            GRAV2=GRAV*(1.+2.28E-7*TRUDP)
            TRUDP=TRUDP+2.*(PRS-PRVPRS)/(GRAV2*(RHOP+RHO))
     C
            PRVPRS=PRS
            RHOP=RHO
            RETURN
            END
```

```
@PRT,S PF*U074P*GD.POTEMP
```

```
PF*U074PRGD(1).POTEMP
          COMPILER(DIAG=3)
          SUBROUTINE POTEMP(P,T,S,PTEMP)
     C
     C     THIS SUBROUTINE COMPUTES DELT, THE CHANGE IN TEMPERATURE
     C     DUE TO ADIABATIC CHANGE IN SEA PRESSURE FROM INSITU POSITION
     C     TO THE SURFACE. THE POTENTIAL TEMPERATURE, PTEMP, IS THEN
     C         PTEMP = T-DELT  ( IN DEG CELCIUS )
     C     REFERENCE:  THE SEA, 1962, VOL 1, PG 17
     C
          DEL1=-1.6E-5+T*(1.014E-5+T*(-1.27E-7+2.7E-9*T))
          DEL2=1.322E-6-2.62E-8*T+4.1E-9*S
          DEL3=9.14E-9-1.55E-13*P+T*(-2.77E-10+9.5E-13*T)
     C
          DELT=P*(DEL1+S*DEL2+P*DEL3)
          PTEMP=T-DELT
     C
          RETURN
          END
```

```
@PRT,S PF*U074P*GD.BRNVAL
```

```
PF*U074PRGD(1).BRNVAL
               COMPILER(DIAG=3)
              SUBROUTINE BRNVAL(TJP1,PJP1,SJP1,G,INIT,ENCPH,PJ,IXBV)
      C
              G=9.80665
              IF (INIT.EQ.1) GO TO 100
              SBARJ=(SJP1+SJ)/2.
              TBARJ=(TJP1+TJ)/2.
              PBARJ=(PJP1+PJ)/2.
              DELPJ=(PJP1-PJ)/2.
              GAMBAR=GAMMA(TBARJ,PBARJ,SBARJ)*.001
              GAMDEL=GAMBAR*DELPJ
      C
              CALL SIGMA(PBARJ,TJP1-GAMDEL,SJP1,DUM,SSTP1)
              ALPHA1=1./(SSTP1*.001+1.)
      C
              CALL SIGMA(PBARJ,TJ+GAMDEL,SJ,DUM,SSTP2)
              ALPHA2=1./(SSTP2*.001+1.)
              E=(ALPHA2-ALPHA1)/(2.*DELPJ)
                RHOJPM=((SSTP1*.001+1.)+(SSTP2*.001+1.))/2.
              EN2RPS=.1*((RHOJPM*G)**2)*(E)
                IF(IXBV.GT.0) GO TO 80
              ENRPS=ABS(EN2RPS)**.5
              ENCPH=(ENRPS/6.2832)*3600.
                GO TO 90
          80    ENCPH=EN2RPS
          90    IF(E.LT.0.) ENCPH=-ENCPH
      C
          51  TJ=TJPM
              PJ=PJPM
              SJ=SJPM
              TJPM=TJP1
              PJPM=PJP1
              SJPM=SJP1
              RETURN
      C
         100  TJPM=TJP1
              PJPM=PJP1
              SJPM=SJP1
              GO TO 91
              END
```

```
@PRT,S PF*U074PRGD.GAMMA
```

```
PF*U074PRGD(1).GAMMA
              FUNCTION GAMMA(T,P,S)
               T2=T*T
              P2=P*P
              S2=S*S
              T4=T2*T2
              GAMMA=-4.21E-2+1.022E-2*T-1.478E-4*T2+3.45E-6*T2*T
             1-3.3E-8*T4+2.427E-3*S-5.0E-7*S*T+9.0E-8*S*T2
             2-6.5E-6*S2+2.291E-5*P-7.62E-7*P*T
             3+7.5E-9*P*T2-1.06E-7*P*S+1.978E-9*P*S*T
             4-4.53E-10*P2+1.56E-11*P2*T
              RETURN
               END
```

```
@PRT,S F.ZREAD
```

PF*UO74PR6D(1).ZREAD

```
      COMPILER(DIAG=3)
      SUBROUTINE ZREAD(IU,IF,IBL)
C
C  THIS SUBROUTINE IS THE READ HALF OF AN INPUT-OUTPUT
C  PACKAGE FOR HANDLING NON-FORMATTED, FORTRAN
C  WRITTEN DATA FILES, COMMONLY REFERRED TO AS
C  FEB(FAST & EASY BINARY) FILES.
C
      COMMON / RHDR / LR,NR,NBR,NMBR,NMFR,NFR,NIR,NAR,IPR(1)
      COMMON /RDOCF/FDOCR(1) /RDOCI/IDOCR(1) /RDOCA/ADOCR(1)
      COMMON / RDATA / VR(1)
C
      COMMON / DIAGS / MSGR,MSGW,NNNR,NNNW,NNIP,NNF,NNI,NNA,IRST,INST
      COMMON / JPOS / JUNIT(30)
      COMMON/DRDCON/ JFLG,ISECR(30)
      DIMENSION IUNIT(30)
      LOGICAL B1,B210,B10,B35,B45,B69
      DATA MSGR / 2 /
      DATA LLSW/1/, IRST/1/
C
      B1=MSGR.EQ.1
      B210=MSGR.GE.2.AND.MSGR.LE.10
      B10=MSGR.EQ.10
      B35=MSGR.EQ.3.OR.MSGR.EQ.5.OR.MSGR.EQ.7.OR.MSGR.EQ.9.OR.MSGR.EQ.10
      B45=MSGR.EQ.4.OR.MSGR.EQ.5.OR.MSGR.GE.8.AND.MSGR.LE.10
      B69=MSGR.GE.6.AND.MSGR.LE.9
C
C
C
      IBLK=IBL
      IPOS=JUNIT(IU)
      IF(IPOS.EQ.0) IPOS=1
      ISEC=ISECR(IU)
      IF(IBL.EQ.0) IBLK=IUNIT(IU)
      IF(IBLK.LT.IPOS) GO TO 5
    4 IF(IBLK.EQ.IPOS) GO TO 3
C
      READ(IU,END=99,ERR=98)LR,NR,NBR,NMBR,NMFR,(IPR(I),I=1,LR),
     *   NFR,NIR,NAR
      IC0=0
      IC1=0
      IC2=0
      IC3=0
      IC4=0
      IC0=(LR+8+30)/28
      IF(NFR.NE.0)IC1=(NFR+(((NFR-1)/221)*3)+30)/28
      IF(NAR.NE.0)IC2=(NAR+(((NAR-1)/221)*3)+30)/28
      IF(NIR.NE.0)IC3=(NIR+(((NIR-1)/221)*3)+30)/28
      IF(NR.NE.0)IC4=((NR*LR)+(((LR*NR-1)/221)*3)+30)/28
      ISEC=ISEC+IC1+IC2+IC3+IC4+IC0
      CALL SETADR(IU,ISEC)
C1
      IPOS=IPOS+1
      IUNIT(IU)=IPOS
      JUNIT(IU)=IPOS
      GO TO 4
C
    5 IF=0
      REWIND IU
      IF(IBL.EQ.0) IBLK=1
      IPOS=1
      IUNIT(IU)=IPOS
      JUNIT(IU)=IPOS
      ISECR(IU)=0
      ISEC=0
      GO TO 4
    3 CONTINUE
C
      READ(IU,END=99,ERR=98)LR,NR,NBR,NMBR,NMFR,(IPR(I),I=1,LR),
     *   NFR,NIR,NAR
      IC0=0
      IC1=0
      IC2=0
      IC4=0
      IC3=0
      IC0=(LR+8+30)/28
      IF(NFR.NE.0)IC1=(NFR+(((NFR-1)/221)*3)+30)/28
      IF(NAR.NE.0)IC2=(NAR+(((NAR-1)/221)*3)+30)/28
      IF(NIR.NE.0)IC3=(NIR+(((NIR-1)/221)*3)+30)/28
      IF(NR.NE.0)IC4=((NR*LR)+(((LR*NR-1)/221)*3)+30)/28
      ISEC=ISEC+IC1+IC2+IC3+IC4+IC0
      IF(NR.GT.NNNR.OR.LR.GT.NNIP.OR.NFR.GT.NNF.
```

```
     *    OR.NIR.GT.NNI.OR.NAR.GT.NNA) GO TO 95
      IF((NFR+NIR+NAR).EQ.0) GO TO 11
      IF(NFR.GT.0)READ(IU,END=99,ERR=98)(FDOCR(I),I=1,NFR)
      IF(NIR.GT.0)READ(IU,END=99,ERR=98)(IDOCR(I),I=1,NIR)
      IF(NAR.GT.0)READ(IU,END=99,ERR=98)(ADOCR(I),I=1,NAR)
   11 CONTINUE
C1
C1
      NL=NR*LR
      N1=(IRST-1)*LR+1
      N2=N1+NL-1
      IF(JFLG.NE.1) GO TO 8
      CALL SETADR(IU,ISEC)
      GO TO 9
    8 READ(IU,END=99,ERR=98)(VR(J),J=N1,N2)
    9 CONTINUE
C
C
      IPOS=IPOS+1
      IUNIT(IU)=IPOS
      JUNIT(IU)=IPOS
      ISECR(IU)=ISEC
C
      IF(NSGR.EQ.0) GO TO 108
      IF(B210) WRITE(6,1000)IU,NMFR,NBR,NMBR,NR,LR,NFR,NIR,NAR
 1000 FORMAT(' READ UNIT',I3,';  FILE ',A6,
     *   ';  SEGNUM',I4,';  SEGNAM ',A6,';  N=',I6,
     *   ';  L=',I4,'  NF=',I4,'  NI=',I4,'  NA=',I4)
C
      IF(B1) WRITE(6,1011) IU,NMFR,NBR,NMBR,NR,LR,NFR,NIR,NAR
 1011 FORMAT(' RD ',I4,2X,A6,2X,I4,2X,A6,2X,I6,4I4)
C
      IF(B35) WRITE(6,1012)(IPR(I),I=1,LR)
 1012 FORMAT(' PARAMETERS: '12(2X,A6)/(13X,12(2X,A6)))
C
      IF(.NOT.B45) GO TO 110
      IF((NFR+NIR+NAR).EQ.0) GO TO 110
      WRITE(6,1013)
 1013 FORMAT(' ADDL DATA:')
      IF(NFR.GT.0)WRITE(6,1100)(FDOCR(I),I=1,NFR)
      IF(NIR.GT.0)WRITE(6,1101)(IDOCR(I),I=1,NIR)
      IF(NAR.GT.0)WRITE(6,1102)(ADOCR(I),I=1,NAR)
 1100 FORMAT(10G11.5)
 1101 FORMAT(1X,12I6)
 1102 FORMAT(1X,12A6)
C
  110 IF(.NOT.B69) GO TO 107
      JL=IRST*LR
      J1=JL-LR+1
      WRITE(6,1014)(VR(I),I=J1,JL)
      JL=(NR+IRST-1)*LR
      J1=JL+1-LR
      WRITE(6,1015)(VR(J),J=J1,JL)
 1014 FORMAT(' FIRST CYCLE:',10G11.5/(13X,10G11.5))
 1015 FORMAT(' LAST CYCLE: ',10G11.5/(13X,10G11.5))
C
  107 IF(.NOT.B10) GO TO 108
      WRITE(6,1017)
      IQ1=IRST
      IQ2=IQ1+NR-1
      DO 106 I=IQ1,IQ2
      JL=I*LR
      J1=JL+1-LR
      WRITE(6,1016) I,(VR(J),J=J1,JL)
  106 CONTINUE
 1016 FORMAT(5X,I5,3X,10G12.6)
 1017 FORMAT(//' LISTING OF DATA'///)
C
  108 IF=0
      IUP=IU
      RETURN
C
C
   95 IF=5
      WRITE(6,1005)NNNR,NNIP,NNF,NNI,NNA,
     *   NG,LR,NFR,NIR,NAR
 1005 FORMAT(//' A DIMENSION IS TOO SMALL.'//
     *   ' NNNR=',I6,'   NNIP=',I6,'   NNF=',I6,
     *   '   NNI=',I6,'   NNA=',I6//' NR=',I6,
     *   '   LR=',I6,'   NFR=',I6,'   NIR=',I6,'   NAR=',I6//)
      RETURN
   98 IF=2
      WRITE(6,1002) IU
```

```
 1002 FORMAT(' READ ERROR ON UNIT ',I3)
      GO TO 90
   99 IF=1
      WRITE(6,1001) IU
 1001 FORMAT(' EOF ON UNIT ',I3)
   90 REWIND IU
      IPOS=0
      IUNIT(IU)=IPOS
      JUNIT(IU)=IPOS
      ISECR(IU)=0
      RETURN
      END
```

@PRT,S F.ZWRIT

```
PF*UO74PRGO(1).ZWRIT
      COMPILER(DIAG=3)
      SUBROUTINE ZWRIT(JU,IF,IBL)
C
C  THIS SUBROUTINE IS THE WRITE HALF OF AN INPUT-OUTPUT
C  PACKAGE FOR HANDLING NON-FORMATTED, FORTRAN
C  WRITTEN DATA FILES, COMMONLY REFERRED TO AS
C  FEB(FAST & EASY BINARY) FILES.
C
C
      COMMON / WHDR / LW,NW,NBW,NMBW,NMFW,NFW,NIW,NAW,IPW(1)
      COMMON /WDOCF/FDOCW(1) /WDOCI/IDOCW(1) /WDOCA/ADOCW(1)
      COMMON / WDATA / VW(1)
C
      COMMON / RHDR / LR,NR,NBR,NMBR,NMFR,NFR,NIR,NAR,IPR(1)
      COMMON /RDOCF/FDOCR(1) /RDOCI/IDOCR(1) /RDOCA/ADOCR(1)
      COMMON / RDATA / VR(1)
C
C
      COMMON / DIAGS / MSGR,MSGW,NNNR,NNNW,NNTP,NNF,NNI,NNA,IRST,IWST
      LOGICAL B1,B210,B10,B35,B45,B69
      COMMON / JPOS / JUNIT(30)
      COMMON /DRDCOM/ JFLG,ISECR(30)
      DIMENSION IUNIT(30),KUNIT(30)
      DATA MSGW / 2 /
      DATA LLSW / 1 /, IRST, IWST / 1, 1 /
C
      IW=1
      IF(JU.LT.0) IW=2
      IU=ABS(JU)
C
C
      B1=MSGW.EQ.1
      B210=MSGW.GE.2.AND.MSGW.LE.10
      B10=MSGW.EQ.10
      B35=MSGW.EQ.3.OR.MSGW.EQ.5.OR.MSGW.EQ.7.OR.MSGW.EQ.9.OR.MSGW.EQ.10
      B45=MSGW.EQ.4.OR.MSGW.EQ.5.OR.MSGW.GE.8.AND.MSGW.LE.10
      B69=MSGW.GE.6.AND.MSGW.LE.9
C
      IBLK=IBL
      IPOS=JUNIT(IU)
      IF(IPOS.EQ.0) IPOS=1
      IF(IBL.EQ.0) IBLK=IUNIT(IU)
      IF(IBLK.LT.IPOS) GO TO 5
    4 IF(IBLK.EQ.IPOS) GO TO 3
C
      READ(IU,END=99,ERR=98)LQ,NQ,(XQ,I=1,3),(XQ,J=1,LQ),NFW,NIW,NAW
      IC0=0
      IC1=0
      IC2=0
      IC3=0
      IC4=0
      IC0=(LQ+30)/28
      IF(NFW.NE.0)IC1=(NFW+(((NFW-1)/22)+3)+30)/28
      IF(NIW.NE.0)IC2=(NIW+(((NIW-1)/22)+3)+30)/28
      IF(NAW.NE.0)IC3=(NAW+(((NAW-1)/22)+3)+30)/28
```

```
      IF(NQ.NE.0)IC4=((NQ*LQ)+(((NQ*LQ)-1)/221)*3)+30)/28
      ISEC=ISEC+IC0+IC1+IC2+IC3+IC4
      CALL SETADR(IU,ISEC)
      IPOS=IPOS+1
      IUNIT(IU)=IPOS
      JUNIT(IU)=IPOS
      GO TO 4
C
    5 REWIND IU
      IPOS=1
      ISEC=0
      ISECR(IU)=0
      IUNIT(IU)=IPOS
      JUNIT(IU)=IPOS
      IF (IBL.NE.3) GO TO 4
C
    2 READ(IU,END=6,ERR=8)LQ,NQ,(XQ,I=1,3),(XQ,J=1,LQ),NFQ,NIQ,NAQ
      IC0=0
      IC1=0
      IC2=0
      IC3=0
      IC4=0
      IC0=(LQ+38)/28
      IF(NFQ.NE.0)IC1=(NFQ+(((NFQ-1)/221)*3)+30)/28
      IF(NIQ.NE.0)IC2=(NIQ+(((NIQ-1)/221)*3)+30)/28
      IF(NAQ.NE.0)IC3=(NAQ+(((NAQ-1)/221)*3)+30)/28
      IF(NQ.NE.0)IC4=((NQ*LQ)+(((NQ*LQ)-1)/221)*3)+30)/28
      ISEC=ISEC+IC0+IC1+IC2+IC3+IC4
      CALL SETADR(IU,ISEC)
      IPOS=IPOS+1
      IUNIT(IU)=IPOS
      JUNIT(IU)=IPOS
      IF(IPOS.EQ.KUNIT(IU)) GO TO 3
C2
      GO TO 2
C
    8 CONTINUE
      KUNIT(IU)=IPOS
      IBLK=1
      IPOS=1
      REWIND IU
      ISEC=0
      ISECR(IU)=0
      IF(KUNIT(IU).EQ.1) GO TO 3
      GO TO 2
C
    6 CONTINUE
      BACKSPACE IU
      WRITE(6,1001)IU
C
C
    3 CONTINUE
      NBW=IPOS
      GO TO (81,82),IW
   81 IF(NW.GT.NNNW.OR.LW.GT.NNIP.OR.NFW.GT.NNF.
     *    OR.NIW.GT.NNI.OR.NAW.GT.NNA) GO TO 95
      WRITE(IU,ERR=97)LW,NW,NBW,NNBW,NNFW,(IPW(I),I=1,LW),NFW,NIW,NAW
      IF((NFW+NIW+NAW).EQ.0) GO TO 87
      IF(NFW.GT.0)WRITE(IU,ERR=97)(FDOCW(I),I=1,NFW)
      IF(NIW.GT.0)WRITE(IU,ERR=97)(IDOCW(I),I=1,NIW)
      IF(NAW.GT.0)WRITE(IU,ERR=97)(ADOCW(I),I=1,NAW)
   87 CONTINUE
C1
C1
      NL=NW*LW
      N1=(IWST-1)*LW+1
      N2=N1+NL-1
      WRITE(IU,ERR=98)(VW(J),J=N1,N2)
C
      GO TO 83
   82 WRITE(IU,ERR=97)LR,NR,NBW,NNBR,NNFR,(IPR(I),I=1,LR),NFR,NIR,NAR
      IF((NFR+NIR+NAR).EQ.0) GO TO 88
      IF(NFR.GT.0) WRITE(IU,ERR=97)(FDOCR(I),I=1,NFR)
      IF(NIR.GT.0)WRITE(IU,ERR=97)(IDOCR(I),I=1,NIR)
      IF(NAR.GT.0)WRITE(IU,ERR=97)(ADOCR(I),I=1,NAR)
   88 CONTINUE
C1
C1
      NL=NR*LR
      N1=(IRST-1)*LR+1
      N2=N1+NL-1
      WRITE(IU,ERR=97)(VR(J),J=N1,N2)
C
```

```
   83 CONTINUE
C
C
      IPOS=IPOS+1
      ISECR(IU)=ISEC
      IUNIT(IU)=IPOS
      JUNIT(IU)=IPOS
      KUNIT(IU)=IPOS
      GO TO (84,85 ),IW
   84 IF(B210) WRITE(6,1000)IU,NMFW,NBW,NMBW,NW,LW,NFW,NIW,NAW
 1000 FORMAT(' WRITE UNIT',I3,';  FILE ',A6,
     *    ';  SEGNUM',I4,';  SEGNAM ',A6,';  N=',I6,
     *    ';  L=',I4,'  NF=',I4,'  NI=',I4,'  NA=',I4)
C
C
      IF(B1) WRITE(6,1011) IU,NMFW,NBW,NMBW,NW,LW,NFW,NIW,NAW
 1011 FORMAT(' WRT ',I4,2X,A6,2X,I4,2X,A6,2X,I6,4I4)
C
      IF(B35) WRITE(6,1012)(IPW(I),I=1,LW)
 1012 FORMAT(' PARAMETERS: '12(2X,A6)/(13X,12(2X,A6)))
C
      IF(.NOT.B45) GO TO 110
      IF((NFW+NIW+NAW).EQ.0) GO TO 110
      WRITE(6,1013)
 1013 FORMAT(' ADDL DATA:')
      IF(NFW.GT.0)WRITE(6,1100)(FDOCW(I),I=1,NFW)
      IF(NIW.GT.0)WRITE(6,1101)(IDOCW(I),I=1,NIW)
      IF(NAW.GT.0)WRITE(6,1102)(ADOCW(I),I=1,NAW)
 1100 FORMAT(10G11.5)
 1101 FORMAT(1X,12I6)
 1102 FORMAT(1X,12A6)
C
  110 IF(.NOT.B69) GO TO 107
      JL=IWST*LW
      J1=JL-LW+1
      WRITE(6,1014)(VW(I),I=J1,JL)
      JL=(NW+IWST-1)*LW
      J1=JL+1-LW
      WRITE( 6,1015)(VW(J),J=J1,JL)
 1014 FORMAT(' FIRST CYCLE:',10G11.5/(13X,10G11.5))
 1015 FORMAT(' LAST CYCLE: ',10G11.5/(13X,10G11.5))
C
  107 IF(.NOT.B10) GO TO 108
      WRITE(6,1017)
      IQ1=IWST
      IQ2=IQ1+NW-1
      DO 106 I=IQ1,IQ2
      JL=I*LW
      J1=JL+1-LW
      WRITE( 6,1016) I,(VW(J),J=J1,JL)
  106 CONTINUE
 1016 FORMAT(5X,I5,3X,10G12.6)
 1017 FORMAT(//' LISTING OF DATA'///)
C
      GO TO 86
   85 IF(B210) WRITE(6,1000)IU,NMFR,NBW,NMBR,NR,LR,NFR,NIR,NAR
C
      IF(B1) WRITE(6,1011) IU,NMFR,NBW,NMBR,NR,LR,NFR,NIR,NAR
C
      IF(B35) WRITE(6,1012)(IPR(I),I=1,LR)
C
      IF(.NOT.B45) GO TO 109
      IF((NFR+NIR+NAR).EQ.0) GO TO 109
      WRITE(6,1013)
      IF(NFR.GT.0)WRITE(6,1100)(FDOCR(I),I=1,NFR)
      IF(NIR.GT.0)WRITE(6,1101)(IDOCR(I),I=1,NIR)
      IF(NAR.GT.0)WRITE(6,1102)(ADOCR(I),I=1,NAR)
C
  109 IF(.NOT.B69) GO TO 117
      JL=IRST*LR
      J1=JL+1-LR
      WRITE( 6,1014)(VR(I),I=J1,JL)
      JL=(NR+IRST-1)*LR
      J1=JL+1-LR
      WRITE( 6,1015)(VR(J),J=J1,JL)
C
  117 IF(.NOT.B10) GO TO 108
      WRITE(6,1017)
      IQ1=IRST
      IQ2=IQ1+NR-1
      DO 116 I=IQ1,IQ2
      JL=I*LR
      J1=JL+1-LR
```

```
      WRITE(6,1016) I,(VR(J),J=J1,JL)
  116 CONTINUE
C
C
   86 CONTINUE
  108 IF=0
      IUP=IU
      RETURN
C
C
   95 IF=5
      WRITE(6,1005)NNNW,NNIP,NNF,NNI,NNA,
     *   NW,LW,NFW,NIW,NAW
 1005 FORMAT(//' A DIMENSION IS TOO SMALL.'//
     *    ' NNNW=',I6,'    NNIP=',I6,'    NNF=',I6,
     *    '     NNI=',I6,'    NNA=',I6//' NW=',I6,
     *    '     LW=',I6,'    NFW=',I6,'    NIW=',I6,'    NAW=',I6//)
      RETURN
   97 IF=3
      WRITE(6,1003) IU
 1003 FORMAT(' WRITE ERROR ON UNIT ',I3)
      LU=IU
      GO TO 90
   98 IF=2
      WRITE(6,1002) IU
 1002 FORMAT(' READ ERROR ON UNIT ',I3)
      LU=IU
      GO TO 90
   99 IF=1
      WRITE(6,1001) IU
 1001 FORMAT(' EOF ON UNIT ',I3)
      LU=IU
      GO TO 90
C
      ENTRY RESETW(KU)
      LU=KU
   90 REWIND LU
      IPOS=0
      IUNIT(LU)=0
      JUNIT(LU)=0
      ISECR(LU)=0
      RETURN
      END
```

@BRKPT PRINTS

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER NORDA Technical Note 196	2. GOVT ACCESSION NO. AD-A128 146	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) A Computer Program for Processing Conductivity-Temperature-Depth (CTD) Data		5. TYPE OF REPORT & PERIOD COVERED Final
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Z.R. Hallock		8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS Naval Ocean Research and Development Activity NSTL Station, Mississippi 39529		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS P.E. 63704N
11. CONTROLLING OFFICE NAME AND ADDRESS Naval Ocean Research and Development Activity NSTL Station, Mississippi 39529		12. REPORT DATE January 1983
		13. NUMBER OF PAGES 39
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)		15. SECURITY CLASS. (of this report) UNCLASSIFIED
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Distribution Unlimited

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

18. SUPPLEMENTARY NOTES

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

CTD Data Processing

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

A software package for processing conductivity-temperature-depth (CTD) data is described. The package includes features for editing, correcting, filtering and pressure sorting to produce working data files for graphic and analysis work.

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

S/N 0102-LF-014-6601

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)