NOAA TECHNICAL MEMORANDUM NWS CR-67

AN EXPLANATION OF THE STANDARD HYDROLOGIC EXCHANGE FORMAT (SHEF) AND ITS IMPLEMENTATION IN THE CENTRAL REGION

Geoffrey M. Bonnin
Robert S. Cox, Jr.
Missouri Basin River Forecast Center
National Weather Service
Kansas City, Missouri

April 1983

U.S. DEPARTMENT OF COMMERCE / National Oceanic and Atmospheric Administration / National Weather Service

NOAA TECHNICAL MEMORANDA
National Weather Service, Central Region Subseries

The National Weather Service Central Region (CR) subseries provides an informal medium for the documentation and quick dissemination of results not appropriate, or not yet ready, for formal publication. The series is used to report on work in progress, to describe technical procedures and practices, or to relate progress to a limited audience. Those Technical Memoranda will report on investigations devoted primarily to regional and local problems of interest mainly to regional personnel, and hence will not be widely distributed.

Papers 1 to 15 are in the former series, ESSA Technical Memoranda, Central Region Technical Memoranda (CRTM); papers 16 to 36 are in the former series, ESSA Technical Memoranda, Weather Bureau Technical Memornada (WBTM). Beginning with #37, the papers are now part of the series, NOAA Technical Memoranda NWS.

Papers that have a PB or COM number are available from the National Technical Information Service, U. S. Department of Commerce, 5285 Port Royal Road, Springfield, VA. 22151. Order by accession number shown in parenthesis at the end of each entry. All other papers are available from the National Weather Service Central Region, Scientific Services, Room 1836, 601 E. 12th St. Kansas City, MO 64106.

ESSA Technical Memoranda

CRTM 1 Precipitation Probability Forecast Verification Summary Nov. 1965-Mar. 1966. SSD Staff, WBCRH - May 1966
CRTM 2 A Study of Summer Showers Over the Colorado Mountains. W. G. Sullivan and James O. Severson - June 1966
CRTM 3 Areal Shower Distribution - Mountain Versus Valley Coverage. Wm C. Sullivan and James O. Severson - June 1966
CRTM 4 Heavy Rains in Colorado June 16 and 17, 1965. SSD Staff, WBCRH - July 1966
CRTM 5 The Plum Fire. Wm. G. Sullivan - August 1966
CRTM 6 Precipitation Probability Forecast Verification Summary Nov. 1965-July 1966. SSD Staff, WBCRH - September 1966
CRTM 7 Effect of Diurnal Weather Variations on Soybean Harvest Efficiency. Leonard F. Hand - October 1966
CRTM 8 Climatic Frequency of Precipitation at Central Region Stations. SSD Staff, WBCRH - November 1966
CRTM 9 Heavy Snow or Glazing. Harry W. Waldheuser - December 1966
CRTM 10 Detection of a Weak Frong by WSR-57 Radar. Harry W. Waldheuser - December 1966
CRTM 11 Public Probability Forecasts. SSD Staff, WBCRH - January 1967
CRTM 12 Heavy Snow Forecasting in the Central United States (An Interim Report) SSD Staff - January 1967
CRTM 13 Diurnal Surface Geostrophic Wind Variations Over the Great Plains. Wayne E. Sangster - March 1967
CRTM 14 Forecasting Probability of Summertime Precipitation at Denver. Wm. G. Sullivan and James O. Severson - March 1967
CRTM 15 Improving Precipitation Probability Forecasts Using the Central Region Verification Printout. Lawrence A. Hughes - May 1967
WBTM CR 16 Small-Scale Circulations Associated with Radiational Cooling. Jack R. Cooley - June 1967
WBTM CR 17 Probability Verification Results (6-month and 18-month) Lawrence A. Hughes - June 1967
WBTM CR 18 On the Use and Misuse of the Brier Verification Score. Lawrence A. Hughes - August 1967 (PB 175 771)
WBTM CR 19 Probability Verification Results (24 Months). Lawrence A. Hughes - February 1968
WBTM CR 20 Radar Prediction of the Topeka Tornado. Norman E. Prosser - April 1968
WBTM CR 21 Wind Waves on the Great Lakes. Lawrence A. Hughes - May 1968
WBTM CR 22 Seasonal Aspects of Probability Forecasts: 1. Summer Lawrence A. Hughes - June 1968 (PB185 733)
WBTM CR 23 Seasonal Aspects of Probability Forecasts: 2. Fall Lawrence A. Hughes - September 1968 (PB 185 734)
WBTM CR 24 The Importance of Areal Coverage in Precipitation Probability Forecasting. John T. Curran and Lawrence A. Hughes - Sept. 1968
WBTM CR 25 Meteorological Conditions as Related to Air Pollution Chicago, Illinois, April 12-13, 1963. Charles H. Swan - October 1968
WBTM CR 26 Seasonal Aspects of Probability Forecasts: 3. Winter Lawrence A. Hughes - December 1968 (PB 185 735)
WBTM CR 27 Seasonal Aspects of Probability Forecasts: 4. Spring Lawrence A. Hughes - February 1969 (PB 185 736)
WBTM CR 28 Minimum Temperature Forecasting During Possible Frost Periods at Agricultural Weather Stations in Western Michigan. Marshall A. Soderberg March 1969
WBTM CR 29 An Aid for Tornado Warnings. Harry W. Waldheuser and Lawrence A. Hughes - April 1969
WBTM CR 30 An Aid in Forecasting Significant Lake Snows. H.J. Rothrock - November 1969
WBTM CR 31 A Forecast Aid for Boulder Winds. Wayne E. Sangster - February 1970
WBTM CR 32 An Objective Method for Estimating the Probability of Severe Thunderstorms. Clarence L. David - February 1970
WBTM CR 33 Kentucky Air-Soil Temperature Climatology. Clyde B. Lee - February 1970
WBTM CR 34 Effective Use of Non-Structural Methods in Water Management. Verne Alexander - March 1970
WBTM CR 35 A Note on the Categorical Verification of Probability Forecasts. Lawrence A. Hughes and Wayne E. Sangster - August 1970
WBTM CR 36 A Comparison of Observed and Calculated Urban Mixing Depths. Donald E. Wuerch - August 1970

NOAA TECHNICAL MEMORANDA NWS

NWS CR 37 Forecasting Maximum and Minimum Surface Temperatures at Topeka, Kansas, Using Guidance from the PE Numerical Prediction Model. (FOUS) Morris S. Webb - November 1970 (COM-71-00118)
NWS CR 38 Snow Forecasting for Southeastern Wisconsin. Rheinhart W. Harms - November 1970 (COM-71-00019)
NWS CR 39 A Synoptic Climatology of Blizzards on the North-Central Plains of the United States. Robert E. Black - Feb. 1971 (COM-71-00369)
NWS CR 40 Forecasting the Spring 1969 Midwest Snowmelt Floods. Herman F. Monschein - February 1971 (COM-71-00489)
NWS CR 41 The Temperature Cycle of Lake Michigan 1. (Spring and Summer). Lawrence A. Hughes - April 1971 (COM-71-00545)
NWS CR 42 Dust Devil Meteorology. Jack R. Cooley - May 1971 (COM-71-00628)
NWS CR 43 Summer Shower Probability in Colorado as Related to Altitude. Alois G. Topil. May 1971 (COM-71-00712)
NWS CR 44 An Investigation of the Resultant Transport Wind Within the Urban Complex. Donald E. Wuerch - June 1971 (COM-71-00766)
NWS CR 45 The Relationship of Some Cirrus Formations to Severe Local Storms. William E. Williams - July 1971 (COM-71-00844)
NWS CR 46 The Temperature Cycle of Lake Michigan 2. (Fall and Winter). Lawrence A. Hughes - September 1971 (COM-71-01039)
NWS CR 47 Practical Application of a Graphical Method of Geostrophic Wind Determination. C.B. Johnson - November 1971 (COM-71-01084)

(continued on back inside cover)

NOAA TECHNICAL MEMORANDUM NWS CR-67

AN EXPLANATION OF THE STANDARD HYDROLOGIC EXCHANGE FORMAT (SHEF) AND ITS IMPLEMENTATION IN THE CENTRAL REGION

Geoffrey M. Bonnin
Robert S. Cox, Jr.
Missouri Basin River Forecast Center
National Weather Service
Kansas City, Missouri

April 1983

UNITED STATES
DEPARTMENT OF COMMERCE
Malcolm Baldrige, Secretary

National Oceanic and
Atmospheric Administration
John V. Byrne, Administrator

National Weather
Service
Richard E. Hallgren, Director

AN EXPLANATION OF THE STANDARD HYDROLOGIC EXCHANGE FORMAT (SHEF) AND ITS IMPLEMENTATION IN THE CENTRAL REGION

Geoffrey M. Bonnin[1]
Robert S. Cox, Jr.[2]
Missouri Basin River Forecast Center
National Weather Service
Kansas City, Missouri

ABSTRACT

Of the numerous federal, state, and local agencies associated with water resources in this country, the efficient communication of hydrologic data is more vital within the National Weather Service than any other agency. Previous attempts to standardize the format in which hydrologic data are transmitted such as the 1961 River Data Code illustrate the fact that we have long recognized the need for such standardization. For various reasons, however, these previous attempts to standardize the transmission format have never maintained widespread support. Present as well as future computer capabilities for the manipulation of hydrologic data demonstrate the fact that now, more than ever before, a standard format for the transfer of hydrologic data is needed. We in the Missouri Basin River Forecast Center (MBRFC) believe that the Standard Hydrologic Exchange Format (SHEF) will fullfill that need.

This paper documents those aspects of SHEF which are most applicable to the Central Region as well as the various systems being developed utilizing SHEF. The paper concludes with a description of the software developed for decoding SHEF.

[1]Geoffrey M. Bonnin, B.E. (Univ. of Qld.), M.I.E., Aust., Hydrologist (Computer Systems)
[2]Robert S. Cox, Jr., B.S.C.E. (Univ. of Mo.), Hydrologist

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF APPENDICES

PART I. SHEF AND ITS IMPLEMENTATION

1. PURPOSE OF STANDARD HYDROLOGIC EXCHANGE FORMAT (SHEF)

a. History

Plans for the current SHEF actually began in 1976 as a cooperative effort between the Northwest River Forecast Center (NWRFC) and the North Pacific Division (NPD) of the U.S. Army Corps of Engineers. This effort began as an enhancement of the Columbia Basin Teletype (CBTT) code and was intended for use in the Columbia River Operational Hydromet Management System (CROHMS). On November 16-18, 1981, a conference of hydrologists from throughout the weather service was held in Kansas City to examine the recurring problems in the transmission of hydrologic data and the possible solutions to these problems. During the conference the data transmission format proposed for CROHMS was presented by NWRFC and discussed. It was agreed that a national standard format for the transmission of hydrologic data should be adapted from the proposed CROHMS format. NWRFC agreed to provide an initial format design prior to a follow-up conference in Kansas City on January 11-15, 1982. This initial format was designed through collaboration with NPD and MBRFC at a meeting on December 16-18, 1981.

The conference in January 1982 was a "meeting of the minds" of hydrologists from all regions of the weather service, the Office of Hydrology, numerous river forecast centers, and WSFOs Topeka and Chicago. An exhaustive examination of NWRFC's proposed format design took place. Its shortcomings were noted, discussed, and the necessary modifications made. The resulting product was christened "SHEF" Standard Hydrologic Exchange Format.

Because the implementation of SHEF on a national scale would be a monumental task, the responsibilities for different phases of the implementation process were assigned as follows:

(1) NWRFC would remain the primary authors of the SHEF design document.
(2) MBRFC would provide the appropriate software to parse and post data transmitted in SHEF.
(3) CNRFC would provide the standard RFC data base design (DATACOL).
(4) OHRFC would provide an applications program (PUPPY), with possible contribution from WSFO Chicago, to accept SHEF data input and generate both a weather wire product and plot files at the WSFO level.
(5) OH, HSD would provide an appropriate OML Chapter and a SHEF user's manual.
(6) MBRFC and OHRFC would coordinate the prototype testing with WSFOs Topeka and Chicago.
(7) MBRFC would provide the standard for RFC hardware configuration.

Since the January conference, numerous modifications and/or enhancements to SHEF have taken place. NWRFC, the authors of the design document, and MBRFC, the authors of the parsing and posting software, have collaborated on these modifications. Hopefully, the result is a format which is simple to use and at the same time is broad ranging enough to meet all user's needs.

b. Significance

The adoption of a standard for data transmission makes possible the most efficient evolution of the software necessary for data encoding, decoding, storage, and manipulation. Likewise, the employment of SHEF on a national scale among all agencies (federal, state, or local) who transmit and/or receive hydrologic data would be a major step in eliminating the necessary duplication of effort in software development which presently exists among "non-standard" systems. In this era of decreasing government spending when agencies are expected to provide improving services with decreasing resources, the establishment of common objectives and the delegation of duties to achieve those objectives is essential. SHEF is a step in the right direction.

Although SHEF was designed to offer a convention for data transmission, it is nevertheless extremely flexible. In order to yield itself to both manual and automated data processing, the code was designed to be both machine and visually readable. Some weather service offices, for example, may feel that a SHEF message before decoding would be perfectly acceptable for transmission over the NOAA Weather Wire for use by non-government interests. Additionally, SHEF has been expanded to accept data which are not strictly hydrologic in nature. The current list of approximately 150 "physical elements" allows for data dealing with aspects of agriculture, meteorology, and fish and wildlife as well as power generation, water quality, and other environmental factors. Expansion of the list of acceptable physical elements to a maximum of 650 data types would be possible in future versions of SHEF.

c. Communications

The primary means of communicating hydrologic data within the Central Region is AFOS. With SHEF implemented, field offices will continue to transmit hydrologic data using the message composition function of AFOS and the proper product identifiers for river and rainfall reports. The body of the messages will, however, be coded in SHEF.

River forecast centers, at the present time, are able to access GOES platform data files on the IBM 360/195 computer in Suitland, MD. This is accomplished via remote job entry in the form of card input. Plans have recently been made for GOES data in the near future to be composed into SHEF and transmitted at predetermined time intervals through AFOS Gateway and on to the national communication circuit.

When these messages are intercepted at each river forecast center they will be passed from the S-230 AFOS computer to the S-140 RFC Gateway computer. The SHEF parsing and posting software will reside in the S-140 and will decode the incoming SHEF data and post it to the RFC hydro-database (DATACOL), also residing in the S-140.

At present, the stored data would then be accessed manually from remote terminals by both the river forecasters and external users of the data. Future intentions are to provide automated transfer of data from the hydro-database to the river forecasting models. A certain amount of manual intervention will always be required, however, to allow the forecasters to monitor the operation of the forecasting models and to override computed data values if necessary. Once the forecasting models have been run to the satisfaction of the river forecasters, the possibility exists that, in the future, the forecasts of river stages would be automatically compiled in SHEF and disseminated to the WSFOs. It is also possible that the public versions of river forecasts, the river statements and flood statements, could be generated automatically at the WSFO level. Even though it would be idealistic to believe that these procedures would never require manual intervention, the automation should help to free field personnel for other duties during critical times.

2. DESCRIPTION OF SHEF

This paper is not intended to provide a complete description of the capabilities of the Standard Hydrologic Exchange Format. For further information, the reader should refer to Standard Hydrologic Exchange Format-Version 1, Northwest River Forecast Center, November 18, 1982; or to the upcoming SHEF User's Manual authored by OH, HSD.

SHEF consists of three basic message formats, each designed for a specific purpose. These formats are:

- ".A" format for single station, single or multiple parameter transmissions.
- ".B" format for multiple station, single or multiple parameter transmissions.
- ".E" format for single station, single parameter, evenly-spaced time series.

Any data which can be transmitted using one of the SHEF formats can also be transmitted using the other two. For data to be transmitted in the most efficient manner, however, the purpose of each format should be understood. As long as each format is used correctly, a single message may contain as many different formats as is desirable.

a. The .A Format

The .A format was designed for transmitting one or more parameters observed at various times for a single station. The .A format represents SHEF in its most fundamental form. It is used most efficiently when the message to be transmitted contains data from fewer than three stations or data from three or more stations which are of various types and/or observation times.

As can be seen from Figure 1, the .A format consists of two parts: the Positional Fields and the Data String. The Positional Fields must, as the name implies, be positioned precisely as diagrammed and consist of:

- the format specifier (mandatory, must begin in column 1).
- the station identifier (mandatory, three to eight alphanumeric characters).
- the date (mandatory, four or six numeric characters).
- the time zone code (optional, will default to GMT if not specified).

Each of the Positional Fields must be separated by one or more mandatory blanks.

The "D" Date/Data Type Elements of the Data String are the:

- observation time (optional, will default to 1200 GMT or 0000 local time)
- creation date (mandatory for forecast data only).
- data string qualifier (optional).
- duration code variable (optional).
- units code (optional).

The "D" Date/Data Type Elements may occur in any order and apply to all subsequent data values unless overridden by another "D" Date/Data Type Element encountered in the Data String.

The type of data being reported is indicated by using an appropriate two to seven character "parameter code", such as: HG for observed instantaneous river stage, PPP for precipitation observed since the previous 7 A.M., or TAIRAW for observed weekly maximum air temperature. Each parameter code must have a corresponding data value; the two separated by one or more blanks. The Data String must also be separated from the Positional Fields by one or more blanks. Each of the elements of the Data String must be separated by a mandatory slash (/).

In its simplest form a .A formatted message would be transmitted as:

.A ΔKCDM7 Δ0217/HG Δ4.5

where:

- .A is the format specifier (the "dot" must occur in column 1).
- Δ denotes a blank character.
- KCDM7 is the station identifier (NWS Communications Handbook No. 5).
- 0217 is the four-digit code for month, day (year defaults to year nearest the current date).
- Time zone is not specified, therefore it defaults to GMT.
- Observation time is not specified, therefore it defaults to 1200.
- Units are not specified, therefore they default to English.
- HG is the parameter code.
- 4.5 is the data value.

The message therefore decodes as a river stage observed to be 4.5 feet at 1200Z on February 17th at the Missouri River gage at Kansas City, MO. Further examples of .A formatted messages can be found in Figure 4.

b. The .B Format

When a message to be coded will contain reports of similar observation times and data types from three or more stations, it would be transmitted most efficiently in the .B format. The .B format is a header-driven, tabular format which yields itself very well to the "morning round-up" type of messages. It consists of three basic parts: the Header, the Body, and the Terminator. See Figure 2 for the basic format structure.

(1) The Header -

(a) The station identifier in a .A format line will be replaced by the Message Source in the .B format header line. The Message Source is intended to be a 3 to 8 character alphanumeric designation for the office/agency from which the message originates. For messages originating from NWS offices, the three-character station IDs will be used.

(b) The Parameter Codes in the Data String of the .B header line will not be followed directly by the corresponding data values. Rather, the Parameter Codes, separated by slashes, represent a list of all data types being reported from the stations contained in the body of the .B message.

(2) The Body -

The Body of the .B format message contains the station identifiers and the data values corresponding, in order, to the Parameter Codes listed in the .B header. One or more blanks are required between the station id and the first data value; the data values are separated by a slash (/). The station ids and their corresponding data values may be encoded one station per line or may be "packed" with several stations per line using commas as delimiters between the last data value for one station and the identifier for the next station. In no instance should a stations identifier and data value occur on separate lines.

(3) The Terminator -

The body of the .B format message is followed by the Terminator, a ".END" in columns 1-4 of the last line of the message.

A simple .B format message containing a morning round-up of observed river stages would appear as:

```
.B MCI 820217 DH11/HG
KCDM7 4.5, SJSM7 8.9, KCKK1 6.0
KCCM7 4.2, LKCM7 3.9, SMNM7 12.7
GLLM7 9.6
.END
```

Further examples of .B formatted messages can be found in Figures 5 and 6.

c. The .E Format

The .E format has been designed for the efficient transmission of evenly-spaced time series. The .E format is also similar in structure to the .A format with the following exceptions:

(1) Each .E format message can contain only one Parameter Code.
(2) The Parameter Code must only be specified once, not before every data value as in the .A format.
(3) The .E format must contain a special Date/Data Type Element "DIXXX" for specifying the time interval between reported data values.
(4) The observation time specified in the .E format line applies only to the first data value. The observation time of each subsequent data value is one time interval later than the previous data value.

The basic structure of the .E format is shown in Figure 3. The .E format used to transmit river stages observed at 6-hour intervals would appear as:

.E KCDM7 0216 DH18/HG/DIH6/5.6/5.9/6.2/5.8

The observation times would be decoded as 1800Z on February 16th and 0000Z, 0600Z, and 1200Z on February 17th, respectively. Further examples of .E formatted messages can be found in Figure 7.

d. Parameter Codes

In order for SHEF to handle the wide variety of hydrologic data which is communicated by the weather service and other agencies on a routine basis, the seven-character Parameter Codes were devised. When fully specified, the seven-character code PEDTSEP contains six keys which break down as follows:

PE = Physical Element (gage height, precipitation, lake elevation, water equivalent, etc.)
D = Duration Code (instantaneous, hourly, daily, etc.)
T = Type Code (observed, forecast, etc.)
S = Source Code (may indicate how data was created or transmitted)
E = Extremum Code (daily maximum, weekly minimum, seasonal maximum, etc.)
P = Probability Code (90%, 10%, etc.)

To reduce manual and communications requirements, each of the six keys (except PE) have been provided standard defaults so that most hydrologic data can be transmitted with a minimum key of two characters. Further explanation of the elements of the Parameter Code and a list of all acceptable codes are provided in the SHEF design document and User's Manual.

e. Other Rules for Coding Messages in SHEF

The following is a brief outline of some of the other features and/or enhancements of SHEF. The reader is encouraged to refer to the SHEF design document and the SHEF User's Manual for further details.

(1) Continuation lines - The .A and .E format lines and the .B header line can be continued, if required. The standard record length for SHEF messages is 80 characters. A line can be specified as a continuation of a previous line by entering a numeric value (1-9) in column 3 after the format specifier, e.g. ".A1", ".E1", ".B9", etc. The body of a .B format message may not be continued.

(2) Revision of previous data - If two similar messages containing data of the same type and observation time for a given station are transmitted at different times, the first message received at the RFC will be posted to the data base. The later transmission will not over write the previous transmission unless it has been specified to be a revision. The user may do this by encoding an "R" in column 3 after the format specifier (.AR, .BR, .ER). Lines in the revision mode may be continued by adding a numeric value (1-9) after the "R" or by replacing the "R" with a numeric value (.AR1, .E3, etc.)

(3) Missing Data vs. Non-Reports - A data value which would routinely be reported but is missing may be reported by encoding a "+", "M", or -9999 as the data value. A non-report, or data which would not routinely be expected may be reported by entering no data value between successive slashes (/) in the Data String. (See Example #10)

(4) Trace of Precipitation - The character "T" encoded as a data value denotes a trace of precipitation and is applicable only to physical elements PC, PP, PY, SD, SF, and SW. Spelling out the word "TRACE" will create a decoding error.

(5) Change of Observation Times - The observation time may be respecified within any of the three formats, either explicitly or implicitly.

(a) Explicit respecification- Date/time changes may be accomplished through the use of the "D" Date/Data Type Elements DJ,DY,DM,DD,DH, and DN.

(b) Implicit respecification - Date time changes may be accomplished through the use of the Date Relative Code "DRX XX". The Date Relative Code increments or decrements the latest explicit specification of date and time by a specified amount. For instance, DRH+6 would increment the last explicit observation time by 6 hours.

(6) Stranger Stations - If the user wishes to transmit data to the RFC from a station for which a station identifier has not been previously assigned, he may do so by designating it as a "stranger station". In the Central Region, stranger stations will be designated by encoding an "X" followed by seven numeric characters as the station identifier. The seven digits represent a 3-digit latitude followed by a 4-digit longitude to the nearest tenth of a degree. Data for stranger stations will not be posted to the hydro-database. Therefore, if reports from a particular stranger station become routine, a coordinated station identifier should be established.

(7) Comments - A colon (:) at any point on a line will terminate the decoding process on that line. Any plain language information which the user wishes to relay to the RFC may be included after the colon.

(8) Blanks - If 15 consecutive blanks are encountered on any data line, the decoding process is terminated for that line. Care must be taken, therefore, especially in .B tabular type messages to assure that no more than 14 consecutive blanks are encoded.

(9) .END Terminator - The .END is required to terminate all .B format messages, however, it is <u>not</u> required to terminate .A or .E format messages.

(10) Number of Data Values - The number of data values in the body of a .B message must be equal to the number of parameter codes in the .B header line. This implies the appropriate number of slashes for each station identifier in the .B body.

(11) Observation Time - The Observation Time in a SHEF message is the time at which the data were <u>observed</u> and <u>not</u> the time at which the message is transmitted. For data with which a duration is associated (24-hr. precipitation, mean daily discharge), the observation time is, by convention, the end of the time period.

(12) Negative Data Values - Negative data is encoded using a minus sign (-). Any other code, such as a "B" will be unacceptable.

3. IMPLEMENTATION OF SHEF

The implementation of SHEF in the Central Region will consist of many facets. Just to institute the new data transmission format and otherwise maintain the status quo would require the composition of new AFOS preformats, the assignment of station identifiers, the training of personnel on the format, and the other housekeeping duties necessary to make the conversion a smooth one. Coincidentally with the implemention of SHEF, however, the implementation of the PUPPY applications program and the Remote Observation System Automation (ROSA) project will occur. The long-term result of this full scale implementation will surely be more efficient communication of hydrologic data and improved data applications with less manual effort at the WSFO/WSO level. The short-term effort required to accomplish the implementation, however, bears mentioning.

a. Schedule

Before SHEF can be implemented on a national scale, the SHEF format, the parsing and posting software, the Datacol hydro-database, and the PUPPY program should be thoroughly tested as a complete package. This will be accomplished through a prototype test involving MBRFC, WSFO STL, and WSOs SGF, MCI, and COU. Although the PUPPY program will not be ready for testing until July 1983, the prototype test has been underway since mid-February and will continue until the modified PUPPY program is completed and tested. The final version of SHEF, the decoding software, and the PUPPY program will then be ready for delivery to field offices this summer.

b. WSFO Responsibilities

The Service Hydrologists should be aware that within the next six months they will be asked to implement SHEF, insuring that:

(1) New AFOS message preformats are composed in SHEF. MBRFC will be available to advise the SHs on the correct use of the SHEF code.

(2) All data points should have established station identifiers, preferrably from NWS Communications Handbook No. 5. The assignment of station ids should be coordinated with DATAC, the Regional Hydrologist, and the appropriate RFC.

(3) The WSFO/WSO personnel should be trained in the correct use of the SHEF code and proper encoding of data into the new preformats. Making full use of SHEF's capability to allow the encoding of comments in messages would be advisable when composing the new preformats. This can significantly decrease the amount of memorization required in instituting the new code. See Examples 12,13 & 14 for sample preformats displaying the usefulness of comments.

(4) Future messages should continue to be encoded in SHEF using established station identifiers or the correct code for stranger stations. As far as the RFC's will be concerned, if a message has not been received in correct SHEF, it has not been received.

(5) The WSFO/WSOs should continue to transmit messages over the AFOS communications lines using the correct product identifiers for river and rainfall reports as outlined by the regional office.

SHEF was designed to be flexible, enabling the conversion to the new code to be as painless as possible for the field offices. The WSFO/WSOs should, in most cases, be able to maintain their AFOS preformats in much the same format as they presently exist. The implementation of SHEF should have very little negative impact on the field offices standard operating procedure.

c. PUPPY

(1) Schedule

PUPPY is an applications program written by OHRFC to automate the transmission of hydrologic data at the WSFO level. At the time of writing this paper, the PUPPY program is being modified by OHRFC to, among other things, accept data input in SHEF. It is scheduled for completion in July 1983.

(2) Description of PUPPY program

For a full description of PUPPY the reader is encouraged to refer to "PUPPY Program Software Documentation" by Daniel P. Provost, OHRFC. PUPPY should be run on a hard disk in the S-230 for reliable interface with AFOS and was designed to generate five different types of files:

(a) A file of all data reported from stranger stations.
(b) A file in PLT format for plotting data at a GDM on a national map background.
(c) A file in PLT format for plotting data on a local map background. Plotting is accomplished by running the hydro version of the PMOD program.
(d) Files containing data to be transmitted over the NOAA weather wire.
(e) A file of data to be transmitted to the RFCs. With the advent of SHEF, this file will not be necessary.

(3) Impact on WSFO

The standard procedure for a WSFO/WSO will be to:

(a) Transmit accumulated data to RFC coded in SHEF.
(b) As required and data accumulates, run PUPPY to generate plot files and/or weather wire products, using SHEF data as input.

This would mean that the data would get to the RFC without delay and that the WSFO/WSO would not have to encode the data twice for the RFC and the weather wire. In addition, the user is capable of plotting any data he desires on an established map background.

d. Remote Observation System Automation (ROSA)

The ROSA project is being developed by Central Region Headquarters and MBRFC to automate the current manual process of receiving telephone reports from hydrologic/agricultural observers and entering the data into AFOS preformatted messages. The ROSA project is a complete system involving encoders, Data General MPT/100 micro computers, and the AFOS computers.

Weather service hydro/ag contract observers in the Central Region will be provided encoding devices to attach to their telephones. A transmission code has been developed specifically for the ROSA observers, with the capabilities of the encoders in mind. Using this code the observers will enter their data into the encoders and, when ready, transmit their observations over the telephone lines to one of the four ROSA MPT/100 micro computers.

The MPT/100's will reside in WSFOs Bismarck, Des Moines, Minneapolis, and Sioux Falls and shall impose as little burden as possible on these offices. When an SAO, SSM, SSI, or OMR report is transmitted to a ROSA computer, the computer will check it for errors, assign the proper AFOS identifier, attach the appropriate WMO header, and transfer the message immediately to AFOS without translation to SHEF. Other Hydro/Ag reports transmitted to the ROSA computers will be transferred at independently selectable thresholds in:

(1) time - i.e. certain times each hour.
(2) number of reports - whenever a certain number of reports have been received from a certain HSA, send out the collection.
(3) data magnitude - when a preset data value is equalled or exceeded transfer the message directly to AFOS.

The messages will be coded in the SHEF .A format, grouped according to HSA, and transmitted using product identifiers CCCRR3CCC (where CCC is the WSFO ID of the HSA from which the report originates, not the ROSA computer site).

For a Flash Flood criteria rainfall report, the ROSA observer will have the ability to transmit his report using a special data type code When a ROSA computer receives a report containing this data type, it will check the report for errors, assign the proper AFOS identifier (CCCRR4XXX, where XXX is the ID of the office with warning responsibilities), and transfers the message immediately to AFOS in the SHEF .A format. It is intended that weather service offices designate the appropriate CCCRR4XXX to set off the alarm on at least one of their AFOS consoles. In this way the responsible office will be assured of receiving rainfall reports of a critical nature when flash flooding may be imminent. WSFO/WSOs should be alerted to the fact that the CCCRR4XXX product IDs should not be used for any other purpose.

The ROSA computers shall also have the capability to:

- sense when AFOS is down.
- transfer data files to another ROSA computer if necessary.
- receive files from another ROSA computer, transfer same files to AFOS, verify proper file transfer, and mark the files sent.
- provide accounting information to aid in WSFO preparation of the Quarterly Observer Payroll.
- maintain an error log.

PART II. THE SHEF DECODER

1. PROGRAM OUTLINE

The process of decoding a message in SHEF and getting the data filed in a database can be conceptualised as a two step process. The two steps being parsing and posting. The parsing step reduces the SHEF format to its essential data. The posting step takes this data and files it in the database.

PARSHEF is a piece of software which accomplishes the parsing step. By parsing the entire SHEF message in one pass and creating as output a file of data, the parsing and posting steps become independent. As a result, PARSHEF is independent of the target database and the only interface required is a means to initiate the software and the reading of the output file SHEFOUT. SHEFOUT contains the data which has been derived from the SHEF message in a wholly machine readable form.

The process of parsing and posting a SHEF message involves initiating PARSHEF and then upon it's completion, reading data item by item from SHEFOUT and posting it.

PARSHEF decodes the SHEF message by examining it character by character. There is no extensive back or forward checking of characters, merely a sequential interpretation. This eliminates the need to maintain a significant character buffer and an extensive series of character pointers. For convenience, the message is read line by line to an eighty character buffer by the routine NEXTCH.

NEXTCH is used to provide the next character in the message for interpretation. It' also takes care of reading lines to the buffer, and terminating the parsing process on a line (fifteen consecutive blanks or a colon).

The software architecture recognizes that while different formats exist in SHEF, the majority of the elements which make up each format are common to each format. This has been addressed by designing software modules which deal with the common elements and then writing driving modules for each format.

There is a master driver routine (SHDRIVE) which searches the message for SHEF code. Upon finding some, it determines the format type (using SHFOR) and then calls the appropriate format driver (SHDECA, SHDECB or SHDECE) to decode that format. After the format is decoded, SHDRIVE starts looking for more SHEF code until the end of the message is completed.

During the decoding process, it is possible for PARSHEF to detect coding errors in the message. A file ($LPT.DU) is maintained to display these errors. The file is a display of the original message with error messages interspersed. These error messages consist of an up arrow and a statement of the error number. Some implementations of PARSHEF have replaced the error number with a description of the error.

The up arrow points to the last character that the software looked at. In most cases, this provides a good indication of where the error is.

As an example of the decoding process, we will examine the general logic flow of the .A format decode driver; SHDECA.

As a first step, a determination is made as to whether this is a continuation or not and if this continuation is in fact legitimate.

The positional data is then decoded using SHPOS, and then the data and data type elements are decoded using SHDTYPE. Following this should be sequences of parameter codes and data values. The parameter code is extracted with SHPCODE and the value with SHREAL. The data item and all it's attributes are then written to the output file, whereupon we try to find another parameter code.

During the process of extracting the parameter code and the data value, various checks are made for legitimacy as well as performing units and time conversions. The majority of the validation is done in the decoding modules as the data is extracted. For example, the parameter codes are validated in SHPCODE as they are decoded, and the date and data type elements are validated in SHDTYPE. This simplifies the validation procedures and allows the errors to be flagged in the correct position in the error file without the use of complex pointers as referred to earlier.

The philosophy for SHDECE and SHDECB, the .E and .B decoder drivers is similar to that for SHDECA and varies only in the areas of uniqueness of each format.

2. FILES

There are five files used by the prototype:

TESTFILE
SHEFOUT
SHEFPARM
DOTBTEMP
$LPT.DU

a. TESTFILE

TESTFILE is the message to be decoded. It is read sequentially in NEXTCH. The file contains character data on lines no more than 80 characters long.

b. SHEFOUT

SHEFOUT is the file containing the results of the decoding process. It is sequential and unformatted.

Each record of the file is a self sufficient and complete description of an item of data and all its attributes.

There is provision in SHDRIVE to create an activity flag in the first record to indicate whether or not the file is in use.

The file has records written to it from SHDECA, SHDECE and SHDOTB.

All times on the file are in Greenwich Mean Time and all units are "English" as specified in SHEF.

Each record of the file is written with the following statement:

```
WRITE (JCHN) IDSTN, NYEAR, NMON, NDAY, NHOUR, NMIN,
1                   KYEAR, KMON, KDAY, KHOUR, KMIN,
2            KODP, KODE, IDUR, KODT, KODS, KODEX, CODP
3            VALUE, LWAL, IREV, MSOURCE, IDOTE
```

The list variables are defined as follows:

IDSTN: Eight element Integer array, containing the eight characters of the station identifier. If there are fewer than eight characters, they are stored starting in the first array element with the remaining elements containing a blank.

NYEAR: Integer, containing the last two digits of the year of the observation date.

NMON: Integer, containing the month number of the observation date. January is month number one.

NDAY: Integer, containing the day of the month of the observation date.

NHOUR: Integer, containing the hour of the day of the observation date on a 24 hour clock.

NMIN: Integer, containing the minute of the observation date.

KYEAR:
KMON:
KDAY:
KHOUR:
KMIN: The creation date of the data coded in the same manner as the observation date. If there was no creation date specified, all elements are set to zero.

KODP: Integer, containing the first character of the Physical Element code.

KODE: Integer, containing the second element of the Physical Element code.

IDUR: Integer, containing a coded number which represents the duration. This duration code specifies the units of time and the number of units as follows:

0XXX	minutes
1XXX	hours
2XXX	days
3XXX	months
4XXX	years
5000	unspecified
5001	seasonal
5002	entire period of record
5004	time period beginning at 7AM local time prior to the observation time and ending at the observation time.
5005	unknown
6XXX	months - end of month

where XXX is the number of units. i.e., eight days would be coded as 2008, instantaneous as 0.

KODT: Integer, containing the Type code character.

KODS: Integer, containing the Source code character.

KODEX: Integer, containing the Extremum code character.

CODP: Real, containing the value of the Probability code as a decimal i.e., fifty percent probability is coded as 0.5. The mean (P code of M) is coded as -0.5. If the Probability code is unspecified, it is coded as -1.0.

VALUE: Double Precision Real, containing the value of the piece of data.

LWAL: Integer, containing the Data Qualifier character.

IREV: Integer, set to one if this is data is intended as a revision of previous data. It is set to zero otherwise.

MSOURCE: Eight element Integer array, containing the eight characters of the data source (.B format). The characters are coded in the same way as those in IDSTN. The source is blank for .A and .E format.

IDOTE: Integer, set to one if the data item is the first item in a string of time series data, set to two if it is a subsequent time series item, set to zero otherwise.

c. SHEFPARM

SHEFPARM is the SHEF parameter file. It contains virtually all the information necessary to validate and interpret SHEF. The file is never written to. It is read in SHPCODE.

The file organized randomly with eight byte (Data General) unformatted records. It contains information in various sections as follows:

records 1-676: Each record corresponds to a theoretical Physical element code (PE), yielding twenty-six squared records (AA is record 1, BA is record 27 and ZZ is record 676). Each record indicates whether or not the corresponding PE code exists, and if it does, it contains the conversion from SI to English units. (Double Precision Real variable).

The conversion term is -9D9 if the PE exists. It is 1.0 if there is no conversion factor. It is -1.0 if the conversion is for a temperature (°C to °F). Otherwise, the value of the conversion term is multiplied with the Metric data to give English units data.

records 677-702: Each record corresponds to a theoretical Duration code (A-Z) yielding twenty six records with Duration code A being record 677.

Each record contains an Integer (IVAL). If IVAL is -9, the corresponding Duration code doesn't exist. Otherwise, IVAL has a coded value for the duration. This code is the same as for IDUR on the SHEFOUT file.

records 702-1378: Each record corresponds to a theoretical Type and Source code combination (TS) yielding twenty-six squared records arranged in the same order as the Physical Element records. Each record contains an Integer (IVAL) which is set to 1 if the corresponding TS code exists and -9 if it doesn't.

records 1379-1404: Each record corresponds to a theoretical Extremum code (A-Z) yielding twenty six records with Extremum code A in record 1379. Each record contains an Integer which is set to 1 if the Extremum code exists and -9 otherwise.

records 1405-1439: Each record corresponds to a theoretical probability code (A-Z, 1-9) yielding 35 records with Probability code A at record 1405 and Probability code 1 following code Z. Each record contains a Double Precision Real value which is the probability (as a decimal) of the corresponding code. P code M (mean) has a value of -0.5. If the P code doesn't exist, the value is -9D9.

records 1440-2115: Each record corresponds to a theoretical Physical Element code (PE) yielding twenty-six squared records arranged in the same order as records 1-676. This section of the file is used to indicate Send Codes and Duration defaults other than I (Instantaneous).

Each record contains an Integer. If neither a Send Code nor a Duration default other than I exists, then the value of the Integer is -9. If either of these do exist, the integer points to a position in the file as follows:

records 2116-....: This section of the file is split into two record groups. Thus records 2116 and 2117 are group one etc. Each group contains either the actual Parameter Code (PEDTSEP) or the PED section of the Parameter Code to indicate a Duration default. The first record of the group contains four integers with the P,E,D and T characters. The second record contains three integers with S,E and P. If the code is for a default Duration, T,S,E and P are blank. Should the default Duration be Instantaneous, then there is no entry here, because that is the standard default Duration.

Certain Send Codes are designed such that they denote an observation time which is at 7AM local time prior to the observation time in the message. These send codes are signified by a negative P character on the file. In summary, the Send Code translation or the default Duration is in the two record group starting at record 2116 which is indicated by the integer in the record corresponding to PE in records 1440-2115.

d. DOTBTEMP

DOTBTEMP is a temporary scratch file written by SHDECB and read by SHDOTB. It is sequential and unformatted. SHDECB reads the .B header line and for every Parameter Code writes a record to DOTBTEMP which contains all of the header information for that Parameter Code. SHDOTB decodes the body of the .B format applying the sequential header information in DOTBTEMP to the corresponding data items.

The data in each record of DOTBTEMP is essentially the same as that in SHEFOUT.

e. $LPT.DU

$LPT.DU is a sequential formatted file containing a log of errors from the decoding process. The file is not read. It is written by NEXTCH, SHERR and SHDRIVE.

As NEXTCH reads lines from the SHEF message, it writes that line to $LPT.DU. Should an error occur in the decoding process, the error handler, SHERR writes two lines to $LPT.DU. The first is a line of blanks with an up-arrow pointing to the last character processed. The second line is a statement of the error number. A list of errors and their error numbers appears in Appendix C .

At the completion of the decoding process, SHDRIVE writes a statement of the total number of errors which have occurred in the message.

3. SOFTWARE MODULES

a. PARSHEF

This is the mainline module. It allocates the logical unit numbers, performs all file opens and closes, sets the SHEFOUT file activity flag and calls the driver routine SHDRIVE.

All software interfacing with the decoder can be achieved through this module with the rest of the software being transparent to the database through SHDRIVE.

b. SHDRIVE

This module handles the SHEFOUT file activity flag, waiting until the file is free and then grabbing control of the file.

The current date and time to be used throughout the decoding process is read from the system. The module then searches for SHEF within the message and calls the appropriate routine to decode it.

At the end of the process, the number of errors which have occurred during decoding are written to the error log file $LPT.DU.

c. SHFOR

Searches for SHEF, or more specifically, a dot in column one followed by A,B or E. It will then determine the format type (.A, .B or .E) and check to see if the revision flag is set.

d. SHDECA

Controls the decoding process for the .A format.

Checks are made to see whether this .A is a continuation. If it is, the previous format must have been a .A with no errors. All of the parameters are initialized at this point in a manner which depends on whether it is a continuation or not.

The positional data is extracted with SHPOS and then date and data type elements with SHTYPE. Pairs of Parameter code and data values should follow, and these are extracted with SHPCODE and SHREAL. After furhter checks of the data, it is written to the file SHEFOUT and the process of calls to SHDTYPE, SHPCODE and SHREAL is repeated until the end of the line is reached.

If the line is a continuation, the extraction of positional data is by-passed.

After data is extracted, and before it is written to file, a number of checks must be made: The possibility and validity of the Trace code must be checked. The SI to English units conversion is performed, and the Qualifier (if any) is extracted and validated.

All that remains before the data can be written to file is a series of time checks and conversions however because of the flexibility of time specification in SHEF these are quite complex, and they vary significantly from format to format.

If the observation time is specified as 7AM previous day and if the time is specified as GMT and if the time is not explicit (i.e. a Date Relative specification) the conversion can be made.

Adjustments are then made for Data Relative, if necessary, for year, month and day followed by the conversion to Zulu. This conversion involves a test for whether or not a local time zone default has been used. The Date Relative adjustments for hour and minute can now be made and the data can be written to file.

e. SHDECE

Controls the decoding process for the .E format. This module is similar to SHDECA except for the following items.

After reading the first data value, no more Parameter Codes or date and data type elements are expected. Should they occur, an error is generated.

The extra complication of a time increment is added to the time calculations and they become exceedingly complex.

f. SHDECB

This module controls the decoding of the .B format header. It operates in a manner similar to SHDECA except for the following items.

No data values are expected in the header. The positional data returned by SHPOS contains the Source rather than the Station ID. Rather than write each data item to SHEFOUT, all of the header information for each Parameter Code is written to the working file DOTBTEMP.

At the conclusion of the header, SHDOTB is called to decode the body of the .B format.

g. SHDOTB

This module decodes the body of the .B format. It uses the header information in file DOTBTEMP to prepare the final output to SHEFOUT.

The process is similar to that described in SHDECA except for the following items.

The station ID is decoded directly rather than by using SHPOS. The header information is read from DOTBTEMP and then either data or date and data type elements are decoded using SHREAL and SHDTYPE. Because of the possibility of date and data type elements in the .B body which can override the values in the header, a series of dummy variables and flags are used to keep track of the applicable parameters.

h. SHPOS

This module decodes the positional fields of each of the three formats.

The station ID (or source, in the case of the .B format) is extracted and validated. It should be noted that this validation is for format only. The responsibility for checking whether a particular station exists lies with the data base specific posting software.

The date group is extracted and validated and the default year computations are made if necessary.

The time zone code is extracted and validated. The Standard Time difference between the time zone and GMT is computed, as is a flag to indicate Daylight or Standard time.

i. SHDTYPE

This module decodes the date and data type elements. All of those elements begin with the letter D which is followed by a one or two letter code. The primary letter designates the nature of the element and the secondary letter is sometimes necessary to discriminate between elements further.

SHDTYPE is split into independant sections on the basis of the primary letter. The module searches for a D, if it finds one, it gets the next character and checks for a valid date or data type element. If this second character is valid, the rest of the element is decoded and validated.

SHDTYPE continues searching for date and data type elements until it doesn't find any more. It will then return to the calling routine.

j. SHPCODE

This module decodes and validates the Parameter Code and assigns the SI to English conversion factor.

The Parameter code is extracted in its entirety and a preliminary check is made for character validity. Each element is then validated and interpreted by reference to the SHEFPARM file.

k. SHTADJ

This module makes adjustments to dates and times by adding and subtracting any date/time unit from a given date and time. It is used to make conversions between local and GMT time zones, to realise Date Relative adjustments and for time series time incrementation.

l. SHCAL

This module calculates a calendar month and day of the month given the ordinal day of the year and whether or not it is a leap year.

It was originally intended that this be done explicitly with integer arithmetic, however there are problems with the algorithm. This algorithm remains part of the code (in comments) in the hope that someone will make it work.

m. SHLEAP

This module determines whether a given year is a leap year using an explicit integer arithmetic method.

n. SHTDAT

This module is used to check if a date is in fact a valid date. i.e. no February 29 unless it is a leap year.

o. SHLOCL

This module determines whether daylight or standard time is in effect if it hasn't been explicitly specified. The determination is based on a table of daylight/standard time change over dates for each year 1976 through 1999.

The module will modify the difference between local time and GMT depending on whether daylight or standard time is in effect.

p. SHINT

This module is used to extract an integer number from a string of digits. The number of digits to scan is passed as an argument. Should an error occur, the routine will return the number extracted so far and the number of digits in that number.

q. SHREAL

This module is used to extract real numbers from the message. The numbers may be signed and the decimal point is optional. Scientific notation is not acceptable.

An 'M' for Missing is decoded as -9999 and a 'T' for Trace is returned as $-9x10^{10}$. This Trace value is a flag for the calling routine which later converts it to 0.001 (inches).

The number of digits read is returned as an argument.

r. NEXTCH

This module is used to provide the next character in the message to be interpreted.

NEXTCH reads the message line by line, maintaining an eighty character buffer, with a pointer to the last character provided. If the buffer is exhausted, the routine will automatically read a new line. The routine will also stop reading the line if fifteen consecutive blanks or a colon is encountered.

As each new line is read, it is written on the error log file $LPT.DU.

s. CODES

This module is a BLOCKDATA module defining the characters in the labeled COMMON block CODES.

t. IRANG

Tests an integer variable to see if it lies within certain boundaries.

u. SHERR

This module is the error handler. Upon detection of an error, the software passes the error number to SHERR. It then writes an up arrow on $LPT.DU in the position of the last character read and then a statement of the error number. Some implementations of PARSHEF will write the actual error message from Appendix C.

4. COMMON BLOCKS

There is a large body of data maintained in labelled COMMON throughout PARSHEF. Following is a brief description of each of these blocks.

a. CODES

Contains all the alpha and numeric characters recognized in SHEF.

b. BUFFER

The buffer containing the current line being decoded, the buffer pointer and the number of sequential blanks encountered on the line.

c. CHAR

The last character read by the system at any time.

d. LUNS

Logical unit numbers and file record counters.

e. FORMAT

Current and previous format types and a flag to indicate whether or not the last format had an error.

f. DATREL

The Date Relative variables.

g. DATA

The SHEF data variables.

h. ERROR

The total number of errors in the message.

i. DATIM

The current system date.

j. DOTBEE

The number of parameters in the .B header and the number of errors in this .B format.

k. DTYPE

A flag for SHDTYPE to indicate a .E format.

l. DOTEE

The value of the observation date and time in the .E time series format. DOTEE maintains the value of the starting date and time for the time series.

5. PORTABILITY

One of the original objectives for the SHEF decoder was to write it in such a way that the task of transporting it across different Fortran compilers was simplified. There are a number of features of the code which have already been identified as likely to cause a problem.

PARSHEF was originally written for the Data General Fortran 5 Compiler Revision 6.13 running under a Mapped RDOS operating system on a Data General S/140 computer.

a. Input Output

The majority of the I/O is standard unformatted sequential. All of the initialization and logical unit number assignment occurs in the mainline module to provide for convenient interfacing. Only one file, DOTBTEMP is subjected to a REWIND (in SHDECB and SHDOTB).

The READ of the message file in NEXTCH uses ERR= and END= branches.

SHEFPARM is the only file that is read in more than one place, however, these all occur in SHPCODE. The file is random and the record to be read is based on a calculation which involves character variables.

b. Character Variables

SHEF makes extensive use of Integer variables which contain character data. The characters are ASCII which means that the character codes are sequential A through Z and 0 through 9. This enables tests to be made to see if a character is alpha (for example) by testing if it is in the numeric range equivalent to A through Z. All of these tests take place through IRANG. It also allows computations to be made with the variables. The record number of the random access to SHEFPARM is made with such a computation and both SHREAL and SHINT make computations based on these character variables.

The character is stored in the right byte or low order byte of the variable with the other byte set to a null, so that the variable has an integer value equivalent to the ASCII code.

Certain compilers may have problems with character and non-character variables mixed in COMMON blocks.

c. Alternate RETURNS

Most of the modules use an alternate RETURN (RETURN I). For this version of the software, I is counted by SUBROUTINE statement argument position with the first argument counted as one.

eg. for the statements;

```
CALL ABC (A, $50, B, $100, C)
SUBROUTINE ABC (A, $, B, $, C)
```

to return to statement 100,
the RETURN statement would be: RETURN 4

d. Variable and Symbolic Names

All names are six characters or fewer except for the module SHDTYPE. Most are unique at five characters, however there are a few exceptions.

e. Comments

Use has been made of the Data General Fortran V facility to allow comments on a statement line after a semi-colon.

6. SUPPORT SOFTWARE

Three support programs are provided; OUTPUT, CPARM and LSTSETUP. OUTPUT will read the SHEFOUT file and create a formatted printer listing of what is on the output file. CPARM is used to create the file SHEFPARM by reading a card image file INPUTPARM.

LSTSETUP will read SHEFPARM and create a listing of what is there. Each of these three modules is quite specific to DG FORTRAN 5.

The file INPUTPARM is prepared as follows.

Command Card:

col 1; *
col 2: a number 1 through 6

This card indicates which parameters are to follow.

1 = Physical Element Codes
2 = Duration Codes
3 = Type/Source Codes
4 = Extremum Codes
5 = Probability Codes
6 = Send Codes or Duration Defaults

Physical Element Codes:

col 1 - 2 . Physical Element Code
col 3 - 22: Conversion factor SI to English

Duration Codes:

Col 1 : Duration Code
col 4-9 : The integer translation of the Duration Code
(see 2.b IDUR)

Type/Source Codes:

col 1 - 2 : Type Source Code
col 4 - 6 : 1.0 to indicate it exists

Extremum Codes:

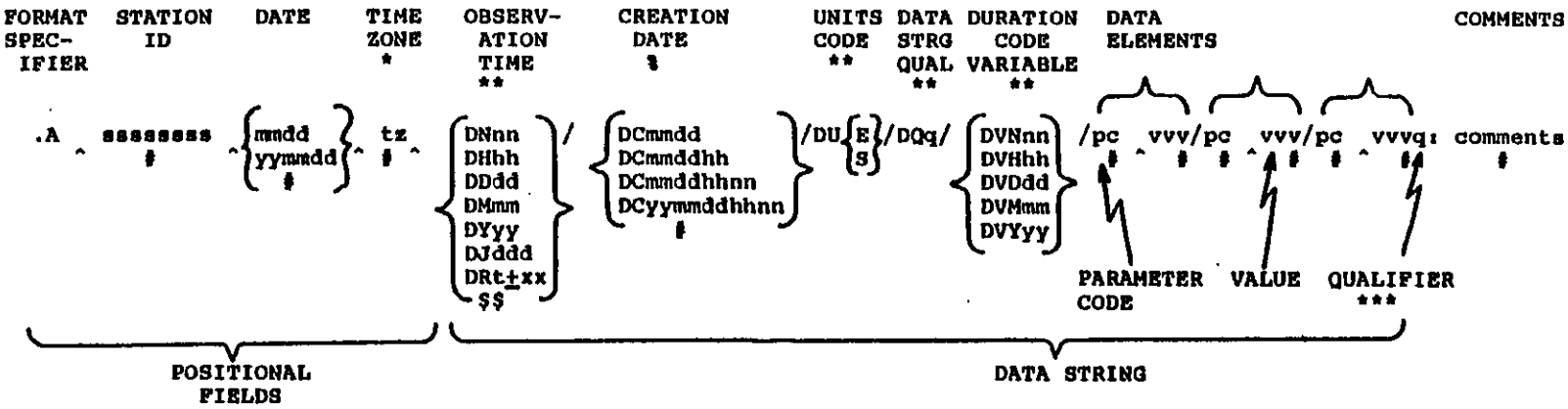
col 1 : Extremum Code
col 4 - 6 : 1.0 to indicate it exists

Probability Codes:

col 1 : Probability Code
col 3 - 22: The equivalent probability

Send Codes or Duration Defaults

col 1 - 2 : Physical Element or Send Code
col 4 - 10: The fully expanded parameter code for Send codes or
the three character PED combination for Duration defaults.
col 12 : for send codes, place a '1' in this column if the observation
time is previous 7AM.



* Denotes optional field which defaults to prespecified value and is fixed positionally in the message

** Denotes optional fields which default to prespecified values and can occur anywhere in the data string

*** Denotes optional field which occurs directly after the data element value 'v'

^ Denotes at least 1 mandatory blank used as a delimiter

# Denotes variable length fields

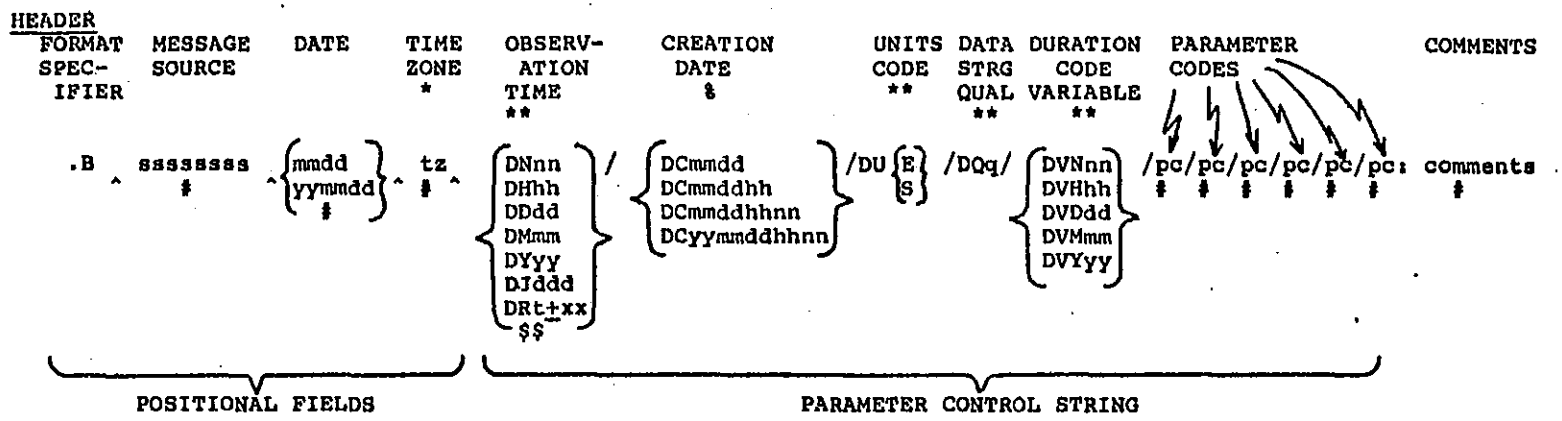
$$ Denotes fields that can be expanded (see table at right)

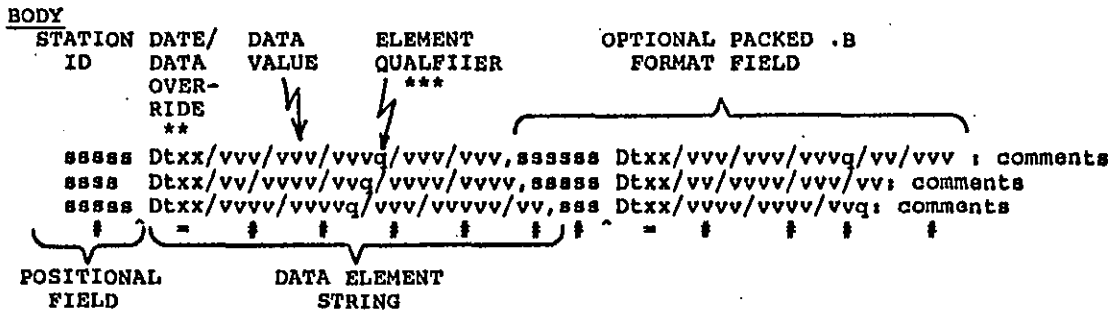
% Denotes mandatory field for forecast data only.

EXPANSION TABLE

OBSERVATION TIME	RELATIVE DATE (DRt)
DNnn	DRN±xx
DHhhnn	DRH±xx
DDddhhnn	DRD±xx
DMmmddhhnn	DRM±xx
DYyymmddhhnn	DRY±xx
DJyyddd	DRE±mm

FIGURE 1. GENERALIZED .A FORMAT MESSAGE





TERMINATOR
.END

* Denotes optional field which defaults to prespecified value and is fixed positionally in the message

** Denotes optional fields which default to prespecified values and can occur anywhere in the data string

*** Denotes optional field which occurs directly after the value

^ Denotes at least 1 mandatory blank used as a delimiter

\# Denotes variable length fields

, Denotes packed .B FORMAT delimiter

$$ Denotes fields that can be expanded (see table at right)

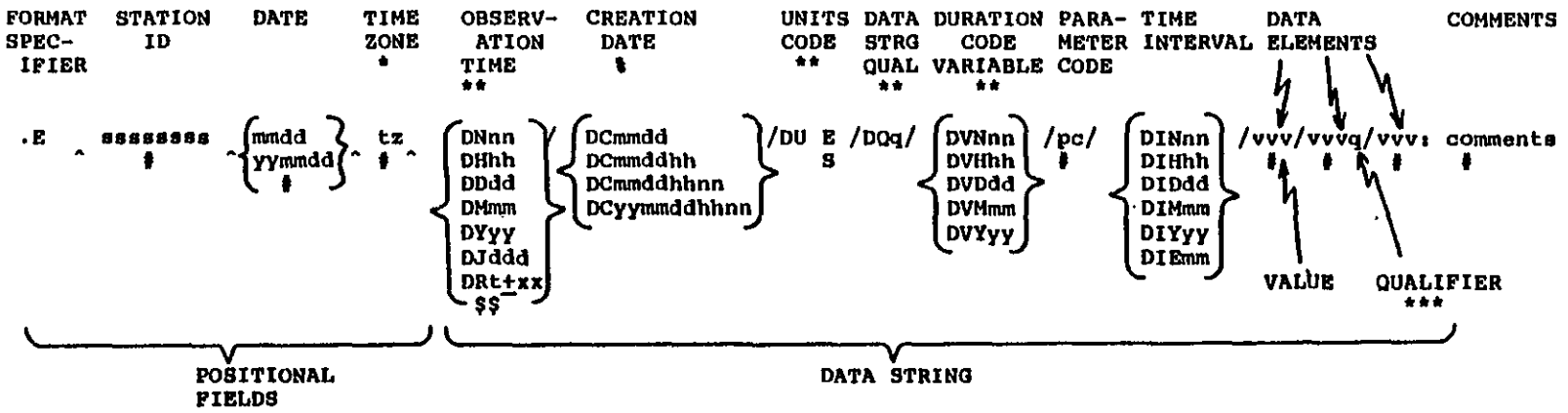
% Denotes mandatory field for forecast data only

= Denotes Date/Data Override 'Dt', t can be any valid 'D' code for this message

EXPANSION TABLE

OBSERVATION TIME	RELATIVE DATE (DRt)
DNnn	DRN±xx
DHhhnn	DRH±xx
DDddhhnn	DRD±xx
DMmmddhhnn	DRM±xx
DYyymmddhhnn	DRY±xx
DJyyddd	DRE±mm

FIGURE 2. GENERALIZED .B FORMAT MESSAGE



* Denotes optional field which defaults to prespecified value and is fixed positionally in the message

** Denotes optional fields which default to prespecified values and can occur anywhere in the data string

*** Denotes optional field which occurs directly after the value

^ Denotes at least 1 mandatory blank used as a delimiter

# Denotes variable length fields

$$ Denotes fields that can be expanded (see table at right)

% Denotes mandatory field for forecast data only

EXPANSION TABLE

OBSERVATION TIME	RELATIVE DATE (DRt)
DNnn	DRN±xx
DHhhnn	DRH±xx
DDddhhnn	DRD±xx
DMmmddhhnn	DRM±xx
DYyymmddhhnn	DRY±xx
DJyyddd	DRE±mm

FIGURE 3. GENERALIZED .E FORMAT MESSAGE

EXAMPLE #1 - SINGLE STATION, MULTI PARAMETER, LOCAL TIME SPECIFICATION

```
WEATHER OBSERVATIONS
NATIONAL WEATHER SERVICE SIOUX FALLS SD
712AM CST MONDAY JANUARY 10 1983

                                24 HR PCPN    NEW SNOW
.A CHES2 830110 C DH0700/    PPP .92   /   SF 6.0     :CHESTER, SD
```

EXAMPLE #2 - MULTI STATION, MULTI PARAMETER, LOCAL TIME SPECIFICATION

```
RIVER AND RAINFALL REPORTS - PART 1
NATIONAL WEATHER SERVICE CONCORDIA KS
500AM CST MON JAN 10 1983
RIVER STAGE REPORTS FOR NORTH CENTRAL KANSAS

                                 STAGE          TREND
.A CNKK1 0110 C DH0500/        HG 1.24/       HI 2 : FALLING
.A NLSK1 0110 C DH0500/        HG 5.4/        HI 2 : FALLING
.A HARN1 0110 C DH0500/        HG 3.8/        HI 0 : STEADY
```

EXAMPLE #3 - MULTI STATION, MULTI PARAMETER, ZULU TIME DEFAULT

```
RIVER REPORT
NATIONAL WEATHER SERVICE COLUMBIA MO
6AM CST MON JAN 10 1983

                                  STAGE        TREND
MISSOURI RIVER
.A JFFM7 0112 DH12/             HG 10.23/    HI 2 : FALLING

BLACKWATER RIVER
.A BLVM7 0112 DH12/             HG  9.42/    HI 0 : STEADY
```

Figure 4. .A Format Examples

EXAMPLE #4 - MULTI STATION, MULTI PARAMETER, ZULU TIME SPECIFICATION

```
.B GTF 830110 DH12/HG/QID/QTD/HL/TX/TN
:MONTANA RIVER STAGES...
: STATION ID  STAGE         STATION ID  STAGE
:MISSOURI RIVER
    TOSM8        M /////,      LDKM8      19.23/////
    OLF       3.97/////,       CLBM8       9.73/////
:YELLOWSTONE RIVER
    CORM8       1.4/////,      LIVM8       1.5 /////
    BILM8       2.8/////,      MILM8       4.3 /////
    SIDM8       9.4/////
:STILLWATER RIVER
    ASSM8       1.1/////
:RESERVOIR DATA
: STATION ID  INFLOW  OUTFLOW  POOL ELEV  TMAX  TMIN
    CYNM8   /   5.4  /   5.6 /  3792.88 /  39 /  28
    FPKM8   /        /  12.0 /  2237.49 /  43 /  31
.END
```

EXAMPLE #5 - MULTI STATION, MULTI PARAMETER, ZULU TIME SPECIFICATION

```
.B DEN 830106 DH17/HG/PPP/HL/QT
FRKC2 +///
LUVC2 1.75/0.0//
BCDC2 //+/
CFDC2 //5432.55/100
EGPC2 //5547.97/
.END
```

EXAMPLE #6 - MULTI STATION, SINGLE PARAMETER, LOCAL TIME SPECIFICATION

```
.B TOP 830110 C DH07/HG/HI

:RIVER AND RAINFALL REPORTS
:NATIONAL WEATHER SERVICE TOPEKA KS
:7AM CST MON JAN 10 1983
:  STATION ID  STAGE    TREND    FS  STA. NAME
:..KANSAS RIVER...
     FRI        5.39   /   2   :  21  FT. RILEY
    WMGK1       4.47   /   0   :  19  WAMEGO
    TPAK1       2.39   /   2   :  21  TOPEKA
    LCPK1       3.40   /   1   :  17  LECOMPTON
    DSOK1       5.66   /   2   :  24  DESOTO
:..MARAIS DES CYGNES RIVER...
    TPOK1       2.45   /   0   :  25  STATE LINE
:..STRANGER CREEK
    TNGK1       5.74   /   1   :  22  TONGANOXIE
.END
```

Figure 5. .B Format Examples

```
EXAMPLE #7 - MULTI STATION, SINGLE PARAMETER AT VARIOUS TIMES

.B BIS 0110 DH12/HG/DRH-6/HG/DRH-12/HG/DRH-18/HG

:NORTH DAKOTA TELEMARK READINGS
:NATIONAL WEATHER SERVICE BISMARCK ND
:600AM CST MON JAN 10 1983

: STATION ID          STAGES                  FS
     BIS        11.45/11.45/12.08/12.21 : 16
    WSBN8       12.40/     /     /       : 22
     ISN        18.30/     /     /       : 20
.END
```

```
EXAMPLE #8 - MULTI STATION SUMMARY OF SYNOPTIC REPORTS

.B DSM 830111 DH12/XW/DH06/TX/DH12/TAIRZP/PPP/SD

:IOWA TEMPERATURE AND PRECIPITATION TABLE
:NATIONAL WEATHER SERVICE DES MOINES IA
:625AM CST TUE JAN 11 1983

:WEATHER AT 6AM...HIGH YESTERDAY...LOW LAST 12 HOURS
:PRECIP FOR 24 HOURS ENDING 6AM CST...6AM SNOW DEPTH

:.....IOWA.....WEATHER   HI   LO  PCPN   SNOW DEPTH
      DSM         22    / 38 / 21 / .05  /          : DES MOINES
      30I         22    / 36 / 25 / .02  /          : LAMONI
      3SE         03    / 33 / 12 /      /     1    : SPENCER
      ALO         03    / 38 / 19 / .03  /          : WATERLOO
      BRL         00    / 38 / 25 / .02  /          : BURLINGTON
      CID         01    / 36 / 20 /  T   /          : CEDAR RAPIDS
      DBQ         00    / 36 / 20 / .05  /          : DUBUQUE
      MCW         22    / 34 / 12 / .07  /     3    : MASON CITY
      OTM         01    / 37 / 23 / .01  /          : OTTUMWA
      SUX         03    / 34 / 17 / .02  /     5    : SIOUX CITY
      AMW         22    / 35 / 25 / .01  /          : AMES
     SLBI4        03    / 31 / 15 /      /     8    : STORM LAKE
.END
```

Figure 6. .B Format Examples

EXAMPLE #10 - MULTI STATION, SINGLE PARAMETER AT VARIOUS INTERVALS

```
630 AM CST TUE JAN 11 1983
KRFM7 KCYM7 TOP FSD A OMA

 STATION ID                      18Z      00Z      06Z      12Z
.E OMAN1 0110 DH18/HG/DIH6/ 14.67 / 14.71 / 14.74 / 14.74
.E BLRN1 0111 DH00/HG/DIH12/          M             /    M
.E BRON1 0111 DH12/HG/DIH6            /                24.24
.E NEBN1 0111 DH00/HG/DIH12/       8.52             /  8.64
.E PTMN1 0111 DH12/HG/DIH6            /                5.59
.E RULN1 0111 DH00/HG/DIH12/       9.47             /  9.34
```

EXAMPLE #11 - GOES DATA

GOES SATELLITE DATA FOR JAN 11 1983

```
.E SCOS2 0110 C DH04/PC/DIH1/: JAMES RIVER NR SCOTLAND - PCPN DATA
:HOUR 04     05     06     07     08     09     10     11
.E1 22.70/22.70/22.70/22.70/22.70/22.70/22.70/22.70/
:HOUR 12     13     14     15     16     17     18     19
.E2 22.70/22.70/22.70/22.70/22.70/22.70/22.70/22.70/
:HOUR 20     21     22     23     00     01     02     03
.E3 22.70/22.70/22.70/22.70/22.70/22.70/22.70/22.70

.E SCOS2 0110 C DH04/HG/DIH1/: JAMES RIVER NR SCOTLAND - GAGE HT DATA
:HOUR 04     05     06     07     08     09     10     11
.E1  5.83/ 5.83/ 5.83/ 5.83/ 5.83/ 5.83/ 5.83/ 5.83/
:HOUR 12     13     14     15     16     17     18     19
.E2  5.83/ 5.83/ 5.83/ 5.83/ 5.83/ 5.83/ 5.83/ 5.83/
:HOUR 20     21     22     23     00     01     02     03
.E3  5.83/ 5.83/ 5.83/ 5.83/ 5.84/ 5.84/ 5.85/ 5.85

.E WKAS2 0110 C DH04/PC/DIH1/: VERMILLION R. NR WAKONDA - PCPN DATA
:HOUR 04     05     06     07     08     09     10     11
.E1 10.19/10.19/10.19/10.19/10.19/10.19/10.19/10.19/
:HOUR 12     13     14     15     16     17     18     19
.E2 10.19/10.19/10.19/10.19/10.19/10.19/10.19/10.19
:HOUR 20     21     22     23     00     01     02     03
.E3 10.19/10.19/10.19/10.19/10.19/10.19/10.19/10.19
```

Fisure 7. .E Format Examples

EXAMPLE #12 - SAMPLE PREFORMAT FOR OBSERVED DATA

```
.B MCI[MMDD]DH12/HG/DRH-6/HG/DRH-12/HG/DRH-18/HG
:STATION ID      RIVER  STAGES              STATION NAME
:              12Z      06Z      00Z      18Z

 KCDM7      [      ]/[      ]/[      ]/[      ]: KANSAS CITY
 SJSM7      [      ]/[      ]/[      ]/[      ]: ST JOSEPH

:                12Z RIVER STAGES
 WVYM7 [      ]///, BOOM7 [      ]///: WAVERLY,BOONVILLE
 MYVM7 [      ]///, AGYM7 [      ]///: MARYVILLE,AGENCY
 SMHM7 [      ]///, SSTM7 [      ]///: SMITHVILLE,SHARPS STATION
 KCKK1 [      ]///, KCCM7 [      ]///: 23RD STREET,BANNISTER RD
 LKCM7 [      ]///, TTNM7 [      ]///: LAKE CITY,TRENTON
 GLLM7 [      ]///, SMNM7 [      ]///: GALLATIN,SUMNER
.END

[NAME        ]
```

EXAMPLE #13 - SAMPLE PREFORMAT FOR OBSERVED DATA

```
.B OMA[MMDD]DH12/HG/HL/QTD/PPP

:STATION ID   STAGE  LAKE ELEVATION  RELEASE   PRECIP  STATION NAME
:               FT       FT,MSL        KCFS      IN

 DCTN1         [    ]/              /           / [    ]: DECATUR
 LOUN1         [    ]/              /           / [    ]: LOUISVILLE 6AM
 LOUN1 DRH-12/[    ]/              /           /       : LOUISVILLE 6PM
 NION1         [    ]/              /           / [    ]: NIOBRARA

 FPKM8              /  [       ]  /  [      ] / [    ]: FORT PECK
 GPDS2              /             /  [      ] / [    ]: GAVINS POINT

.END

[NAME        ]
```

Figure 8. Sample Message Preformats

EXAMPLE #14 - SAMPLE PREFORMAT FOR RIVER FORECASTS

```
.B KRF[MMDD]DH12/[DCMMDDHH]/HG/DRD+1/HGIF/DRD+2/HGIF/DRD+3/HGIF/HX

:[DAY][MON][DA] 1983  NATIONAL WEATHER SERVICE RIVER STAGE FORECASTS

:STATION ID  TODAY [MM/DD][MM/DD][MM/DD] CREST DATE/STG  STA. NAME     FS
 ----------  -----  -----  -----  -----  --------------  ---------     --
:MISSOURI RIVER
  SIOI4      [     ]/[     ]/[     ]/[     ]/[DMMMDDHH/     ]: SIOUX CITY   36
  OMHN1      [     ]/[     ]/[     ]/[     ]/[          /     ]: OMAHA        29
  NEBN1      [     ]/[     ]/[     ]/[     ]/[          /     ]: NEBRSKA CITY 18
  RULN1      [     ]/[     ]/[     ]/[     ]/[          /     ]: RULO         17
  SJSM7      [     ]/[     ]/[     ]/[     ]/[          /     ]: ST. JOSEPH   17
  LVNK1      [     ]/[     ]/[     ]/[     ]/[          /     ]: LEAVENWORTH  19
  KCDM7      [     ]/[     ]/[     ]/[     ]/[          /     ]: KANSAS CITY  22
  WVYM7      [     ]/[     ]/[     ]/[     ]/[          /     ]: WAVERLY      20
  GLSM7      [     ]/[     ]/[     ]/[     ]/[          /     ]: GLASGOW      25
  BOOM7      [     ]/[     ]/[     ]/[     ]/[          /     ]: BOONVILLE    21
  JFFM7      [     ]/[     ]/[     ]/[     ]/[          /     ]: JEFFRSN CITY 23
  HRNM7      [     ]/[     ]/[     ]/[     ]/[          /     ]: HERMANN      21
  SCHM7      [     ]/[     ]/[     ]/[     ]/[          /     ]: ST. CHARLES  25
:KANSAS RIVER
  DSOK1      [     ]/[     ]/[     ]/[     ]/[          /     ]: DESOTO       24
.END

[NAME     ]
```

Fisure 9. Sample Message Preformat

APPENDIX A

OBSERVED HYDROLOGIC PARAMETERS

TRUNCATED CODE	TYPE OF DATA	UNITS	EXPANDED CODE
EP	EVAPORATION, pan, incremental	IN,MM	EPDRZZZ
EV	EVAPORATION, lake (computed)	IN,MM	EVDRZZZ
HG	HEIGHT, river stage	FT,M	HGIRZZZ
HI	STAGE TREND INDICATOR	CODED	HIIRZZZ
HK	HEIGHT, lake above specified datum	FT,M	HKIRZZZ
HN(S)	HEIGHT, river stage, minimum	FT,M	HGIRZNZ
HT	ELEVATION, project tailwater stage	FT,M	HTIRZZZ
HX(S)	HEIGHT, river stage, maximum	FT,M	HGIRZXZ
HY(S)	HEIGHT, river stage at previous 7am	FT,M	HGIRZZZ
IC	ICE COVER, river	%	ICIRZZZ
IE	EXTENT OF ICE from reporting area (+ UPS,- DNS)	MI,KM	IEIRZZZ
IR	ICE REPORT (structure, type, cover)	CODED	IRIRZZZ
LS	LAKE STORAGE, volume	KAF,MCM	LSIRZZZ
MW	MOISTURE, SOIL, percent by weight	%	MWIRZZZ
PC	PRECIPITATION, accumualted	IN,MM	PCIRZZZ
PP	PRECIPITATION, actual increment	IN,MM	PPDRZZZ
PPD	PRECIPITATION, 24-hour increment	IN,MM	PPDRZZZ
PPP	PRECIPITATION, increment since previous 7 am	IN,MM	PPPRZZZ
PPQ	PRECIPITATION, 6-hour increment	IN,MM	PPQRZZZ
PPK	PRECIPITATION, 12-hour increment	IN,MM	PPKRZZZ
PPL	PRECIPITATION, 18-hour increment	IN,MM	PPLRZZZ
PPV	PRECIPITATION, variable duration	IN,MM	PPVRZZZ
PT	PRECIPITATION, type	CODED	PTIRZZZ
PY(S)	PRECIPITATION, increment at previous 7 am	IN,MM	PPDRZZZ
QD	DISCHARGE, canal diversion	KCFS,CMS	QDIRZZZ
QG	DISCHARGE from power generation	KCFS,CMS	QGIRZZZ
QI	DISCHARGE, inflow	KCFS,CMS	QIIRZZZ
QID	DISCHARGE, inflow, mean daily	KCFS,CMS	QIDRZZZ
QP	DISCHARGE, pumping	KCFS,CMS	QPIRZZZ
QPD	DISCHARGE, pumping, mean daily	KCFS,CMS	QPDRZZZ
QR	DISCHARGE, river	KCFS,CMS	QRIRZZZ
QRD	DISCHARGE, river, mean daily	KCFS,CMS	QRDRZZZ
QT	DISCHARGE, computed total project outflow	KCFS,CMS	QTIRZZZ
QTD	DISCHARGE, total project outlow, mean daily	KCFS,CMS	QTDRZZZ
QX(S)	DISCHARGE, maximum flow	KCFS,CMS	QRIRZXZ
SA	SNOW, areal extent of snow cover	%	SAIRZZZ
SD	SNOW, depth	IN,CM	SDIRZZZ
SF	SNOW, depth, new snowfall	IN,CM	SFIRZZZ
SR	SNOW REPORT (structure, type, surface, bottom)	CODED	SRIRZZZ
SS	SNOW, density	IN/IN,MM/MM	SSIRZZZ
SW	SNOW, water equivalent	IN,MM	SWIRZZZ

OBSERVED HYDROLOGIC PARAMETERS (CONT.)

TRUNCATED CODE	TYPE OF DATA	UNITS	EXPANDED CODE
TA	TEMPERATURE, air, dry bulb	DF,DC	TAIRZZZ
TN(S)	TEMPERATURE, air, minimum	DF,DC	TAIRZNZ
TW	TEMPERATURE, water	DF,DC	TWIRZZZ
TX(S)	TEMPERATURE, air, maximum	DF,DC	TWIRZXZ

....NOTE....(S) DENOTES A SEND CODE.

APPENDIX B

ADDITIONAL AGRICULTURAL PARAMETERS

TRUNCATED CODE	TYPE OF DATA	UNITS	EXPANDED CODE
AD	RESERVED FOR STATE DIVISION CODE	-----	
AF	SURFACE FROST INTENSITY	CODED	AFIRZZZ
AM	SURFACE DEW INTENSITY	CODED	AMIRZZZ
AT	TIME BELOW CRITICAL TEMPERATURE, 25 DF	HOURS	ATIRZZZ
AU	TIME BELOW CRITICAL TEMPERATURE, 32 DF	HOURS	AUIRZZZ
AW	LEAF WETNESS	HOURS	AWIRZZZ
GD	FROST DEPTH, depth of frost	IN,CM	GDIRZZZ
GR	FROST REPORT, structure	CODED	GRIRZZZ
GS	STATE OF GROUND	CODED	GSIRZZZ
GT	FROST, depth of frost thawed	IN,CM	GTIRZZZ
MI	MOISTURE, soil index	IN	MIIRZZZ
RI	RADIATION, accumulated incoming solar	LY	RIDRZZZ
RT	RADIATION, sunshine hours	HOURS	RTDRZZZ
TAWPZNM	TEMPERATURE, air, average weekly minimum	DF,DC	TAWPZNM
TAWPZXM	TEMPERATURE, air, average weekly maximum	DF,DC	TAWPZXM
TH	TEMPERATURE, degree days of heating	DF-DA,DC-DA	THSRZZZ
TM	TEMPERATURE, air, wet bulb	DF,DC	TMIRZZZ
TPIRAN	TEMPERATURE, pan water, minimum	DF,DC	TPIRANZ
TRIRAX	TEMPERATURE, pan water, maximum	DF,DC	TPIRAXZ
TSIRZN	TEMPERATURE, soil, minimum	DF,DC	TSIRZNZ
TSIRZX	TEMPERATURE, soil, maximum	DF,DC	TSIRZXZ
UD	WIND, direction	TENS OF DEGREES	UDIRZZZ
ULD	WIND, travel length in 24 hours	MI,KM	ULDRZZZ
US	WIND, speed	MI/HR,M/SEC	USIRZZZ
XP	WEATHER, past	SYNOPTIC CODE	XPQRZZZ
SR	HUMIDITY, relative	%	XRIRZZZ
XW	WEATHER, present	SYNOPTIC CODE	XWIRZZZ

APPENDIX C

SHEF ERRORS

1. This line not decoded.
2. No space in positional data
3. Less than 3 characters in ID or message source
4. TZ code error
5. Date group error
6. Illegal character in ID or message source
7. Error in date code.
8. Observation time error
9. Date relative code error
10. Julian day error
11. Illegal data string qualifier
12. Units code error
13. Not a date or data type element.
14. Not a date or data type element, maybe a missing slash
15. Illegal character in parameter code
16. File read error on SHEFPARM
17. Non-existant parameter code
18. Parameters coded with a send code
19. Continuation of a format does not follow the correct format
20. A format revision continuation follows an original.
21. The format that this is continuing had an error.
22. Year not in the range 1976-1999 for default time zone.

23. Forecast data without creation date
24. Bad date some how.
25. DV not defined for ZZV
26. DV code error
27. DI code error
28. Trace specified for other than PP, PC, SF, SD or SW
29. No time increment specified
30. To many items in .B body line
31. Bad character in the line
32. Not enough items in .B body line
33. No value specified
34. No .END at end of .B
35. Zulu, DR or DI coded with send code QY, PY or HY
36. The explicit date referenced by DRE is not the end of the month.
37. Observation or creation time is between 0201 and 0259 on the date of change from standard to daylight time.

APPENDIX D

PARSHEF SOURCE CODE

```
C      PARSHEF .. DRIVER ROUTINE FOR TESTING SHEF PARSING
C
C-------------------------------------------------------------------C
C   VERSION 1.0 APRIL 1982     GEOFFREY M BONNIN  MBRFC             C
C-------------------------------------------------------------------C
C
C.... SET UP CHANNELS FOR THE INPUT MESSAGE FILE AND
C       THE OUTPUT DATA FILE.
C
       COMMON /LUNS/    LCHN,JCHN,KCHN,MCHN,MREC,ICHER
       LCHN = 26
       JCHN = 27
       KCHN = 54
       MCHN = 55
       ICHER = 56
C
C.... OPEN THE FILES
C
       OPEN LCHN,'TESTFILE',ATT='BL'
       OPEN JCHN,'SHEFOUT'
       OPEN KCHN,'SHEFPARM',LEN=8
       OPEN MCHN,'DOTBTEMP'
       OPEN ICHER,'$LPT.DU'
C
C.... CALL THE PARSING DRIVER ROUTINE
C
       CALL SHDRIVE
C
C.... SET THE OUTPUT FILE INACTIVE
C
       REWIND JCHN
       IACTIV = 0
       WRITE(JCHN) IACTIV
C
C.... CLOSE THE FILES
C
       CLOSE LCHN
       CLOSE JCHN
C
C.... THATS ALL
C
       CALL EXIT
       END
```

```
      SUBROUTINE SHDRIVE
C
C-----------------------------------------------------------------C
C   VERSION 1.0 APRIL 1982    GEOFFREY M BONNIN  MBRFC            C
C-----------------------------------------------------------------C
C
C      THIS ROUTINE IS THE DRIVER FOR PARSING SHEF.
C
      COMMON /ERROR/  NERROR
      COMMON /BUFFER/ IBUF(80),IP,NBLNK
      COMMON /DATIM/   IDATE(3)
      COMMON /LUNS/   LCHN,JCHN,KCHN,MCHN,MREC,ICHER
      COMMON /FORMAT/ ITYPE,LFORM,NERR
C
      NBLNK = 0
      ITYPE = 0
      NERR = 0
      NERROR = 0
C
C.... CHECK IF THE FILE 'SHEFOUT' IS IN USE.
C     IF SO; DELAY A LITTLE, IF NOT; SET THE ACTIVITY FLAG TO 1.
C
   10 READ(JCHN,ERR=20,END=20) IACTIV
C
C     FILE ACTIVE
C
      IF( IACTIV.EQ.0 ) GO TO 20
      CALL FDELY(100)                    ;WAIT 5 SECONDS
      REWIND(JCHN)
      GO TO 10
C
C     FILE INACTIVE; SET IACTIV TO 1
C
   20 REWIND(JCHN)
      IACTIV = 1
      WRITE(JCHN) IACTIV
C
C.... GET THE CURRENT DATE
C
      CALL FGDAY(IDATE(1),IDATE(2),IDATE(3))
      IDATE(3) = IDATE(3) + 1900
      GO TO 25
C
C.... LOOK FOR A SHEF FORMAT AND GET THE TYPE
C
   23 IFLAG = 0
      GO TO 30
   25 IFLAG = 1
   30 LFORM = ITYPE
      CALL SHFOR(ITYPE,IFLAG,$9000)
C
C.... DECODE THIS FORMAT TYPE
      GO TO (100,110,120,130,140,150),IABS(ITYPE)
C
  100 IREV = 0
      CALL SHDECA(IREV,$25,$9000)              ;.A FORMAT
      IFLAG = 0
      GO TO 30
C
  110 IREV = 1
      CALL SHDECA(IREV,$25,$9000)              ;.AR FORMAT
      IFLAG = 0
      GO TO 30
```

```
C
  120 IREV = 0
      CALL SHDECB(IREV,$25,$9000,$23)          ;.B FORMAT
      IFLAG = 1
      GO TO 30
C
  130 IREV = 1
      CALL SHDECB(IREV,$25,$9000,$23)          ;.BR FORMAT
      IFLAG = 1
      GO TO 30
C
  140 IREV = 0
      CALL SHDECE(IREV,$25,$9000)              ;.E FORMAT
      IFLAG = 0
      GO TO 30
C
  150 IREV = 1
      CALL SHDECE(IREV,$25,$9000)              ;.ER FORMAT
      IFLAG = 0
      GO TO 30
C
C
C.... THATS ALL
C
 9000 WRITE(ICHER,9001) NERROR
 9001 FORMAT(5X,'TOTAL NUMBER OF SHEF ERRORS FOR THIS MESSAGE IS',I3)
      RETURN
      END
```

```
      SUBROUTINE SHFOR(ITYPE,IFLAG,$)
C
C-----------------------------------------------------------------C
C   VERSION 1.0 APRIL 1982    GEOFFREY M BONNIN  MBRFC            C
C-----------------------------------------------------------------C
C
C       THIS ROUTINE READS CHARACTERS ONE AT A TIME FROM THE INPUT
C       FILE AND SEARCHES FOR A SHEF FORMAT SPECIFIER.
C       THE FORMAT TYPE IS RETURNED IN ITYPE.
C       IF IFLAG IS SET TO ZERO IT LOOKS AT THE LAST CHARACTER READ.
C
C        SHEF FORMAT SPECIFIER     ITYPE
C                     .A           1
C                     .AR          2
C                     .B           3
C                     .BR          4
C                     .E           5
C                     .ER          6
C
      COMMON /CHAR/ ICHAR
      COMMON /BUFFER/ IBUF(80),IP,NBLNK
      COMMON /CODES/ ICHA,ICHB,ICHC,ICHD,ICHE,ICHF,ICHG,ICHH,ICHI,
     1               ICHJ,ICHK,ICHL,ICHM,ICHN,ICHO,ICHP,ICHQ,ICHR,
     2               ICHS,ICHT,ICHU,ICHV,ICHW,ICHX,ICHY,ICHZ,ICH0,
     3               ICH1,ICH2,ICH3,ICH4,ICH5,ICH6,ICH7,ICH8,ICH9,
     4               IBLNK,ISLASH,ICOLON,IPLUS,IMINUS,IDOT,IARROW,
     5               ICOMMA
C
      IF( IFLAG.EQ.0 ) GO TO 15
C
C.... READ A CHARACTER
C
   10 IP = 81
C
C.... CHECK FOR A '.' IN COLUMN 1
C
      CALL NEXTCH(ICHAR,$10,$9010)
   15 IF( ICHAR.EQ.IDOT ) GO TO 19
      GO TO 10
C
C.... CHECK FOR 'A', 'B' OR 'E'
C
   19 CALL NEXTCH(ICHAR,$10,$9010)
   20 IF( ICHAR.NE.ICHA ) GO TO 30                ;'A'
      ITYPE = 1
      GO TO 50
C
   30 IF( ICHAR.NE.ICHB ) GO TO 40                ;'B'
      ITYPE = 3
      GO TO 50
C
   40 IF( ICHAR.NE.ICHE ) GO TO 9020              ;'E'
      ITYPE = 5
      GO TO 50
C
C.... CHECK FOR AN 'R' .. REVISION
C
   50 CALL NEXTCH(ICHAR,$10,$9010)
   55 IF( ICHAR.NE.ICHR ) GO TO 56
      ITYPE = ITYPE + 1
C
C.... CHECK FOR A CONTINUATION
C
```

```
      CALL NEXTCH(ICHAR,$10,$9010)
   56 CALL IRANG(ICHAR,ICH1,ICH9,$60)
      ITYPE = -ITYPE
      CALL NEXTCH(ICHAR,$10,$9010)
      GO TO 9000
C
C.... MAKE SURE WE HAVEN'T PICKED UP A '.END'
C
   60 IF( ICHAR.EQ.ICHN ) GO TO 9020
C
C.... GOT THE FORMAT TYPE
C
 9000 RETURN
C
C.... FILE ERRORS
C
 9010 RETURN 3
C
 9020 CALL SHERR(1)
      GO TO 10
C
      END
```

```
      SUBROUTINE SHDECA(IREV,$,$)
C
C-----------------------------------------------------------------------C
C   VERSION 1.0 JULY 1982     GEOFFREY M BONNIN  MBRFC                  C
C-----------------------------------------------------------------------C
C
C     DECODE SHEF .A FORMAT
C
C       IREV = 1 FOR REVISED, = 0 FOR NEW DATA
C
      DOUBLE PRECISION VALUE,FACTOR
      COMMON /BUFFER/ IBUF(80),IP,NBLNK
      COMMON /CHAR/   ICHAR
      COMMON /LUNS/   LCHN,JCHN,KCHN,MCHN,MREC,ICHER
      COMMON /FORMAT/ IFORM,LFORM,NERR
      COMMON /DATREL/ MYEAR,MMON,MDAY,MHOUR,MMIN,MEND
      COMMON /DATA/   IDSTN(8),LYEAR,LMON,LDAY,LHOUR,LMIN,
     *                         KYEAR,KMON,KDAY,KHOUR,KMIN,
     *                KODP,KODE,ICODD,IDCODD,KODT,KODS,KODEX,CODP,
     *                KWAL,MSOURCE(8),ITZ,NADJTZ,KODU,VALUE,KFLAG
      COMMON /CODES/ ICHA,ICHB,ICHC,ICHD,ICHE,ICHF,ICHG,ICHH,ICHI,
     1               ICHJ,ICHK,ICHL,ICHM,ICHN,ICHO,ICHP,ICHQ,ICHR,
     2               ICHS,ICHT,ICHU,ICHV,ICHW,ICHX,ICHY,ICHZ,ICHO,
     3               ICH1,ICH2,ICH3,ICH4,ICH5,ICH6,ICH7,ICH8,ICH9,
     4               IBLNK,ISLASH,ICOLON,IPLUS,IMINUS,IDOT,IARROW,
     5               ICOMMA
C
C
C.... INITIALISE
C
      ITZ = 0
      IDOTE = 0
      MSOURCE(1) = IBLNK
      MSOURCE(2) = IBLNK
      MSOURCE(3) = IBLNK
      MSOURCE(4) = IBLNK
      MSOURCE(5) = IBLNK
      MSOURCE(6) = IBLNK
      MSOURCE(7) = IBLNK
      MSOURCE(8) = IBLNK
C
C.... CHECK FOR CONTINUATION
C
      IF( IFORM.GT.0 ) GO TO 10
      IFORM = -IFORM
      IF( (LFORM.NE.1).AND.(LFORM.NE.2) ) GO TO 9020
      IF( (IFORM.EQ.2).AND.(LFORM.EQ.1) ) GO TO 9030
      IF( (IFORM.EQ.1).AND.(LFORM.EQ.2) ) IREV = 1
      IF( NERR.GT.0 ) GO TO 9040
      GO TO 30
C
C.... DEFAULTS
C
   10 KYEAR = 0
      KMON  = 0
      KDAY  = 0
      KHOUR = 0
      KMIN  = 0
      KWAL  = ICHZ
      KODU = 1
      MYEAR = 0
      MMON  = 0
      MDAY  = 0
```

```
      MHOUR = 0
      MMIN  = 0
      MEND = 0
      IDCODD = 5000
      NERR = 0
C
C.... GET THE POSITIONAL DATA
C
   20 CALL SHPOS($9050,$9010)
C
C.... GET THE DATE AND DATA TYPE ELEMENTS
C
   30 CALL SHDTYPE(1,$9000,$9010,$9000)
C
C.... GET THE PARAMETER CODE
C
   40 CALL SHPCODE(KODP,KODE,ICODD,KODT,KODS,KODEX,CODP,FACTOR,
     *                    $9000,$9010,$9000)
      IF( ICHAR.NE.IBLNK ) GO TO 9120
C
C.... TEST FOR CORRECT DURATION
C
      IF( (ICODD.EQ.5003).AND.(IDCODD.EQ.5000) ) GO TO 9080
      IDUR = ICODD
      IF( ICODD.EQ.5003 ) IDUR = IDCODD
C
C.... TEST IF CREATION DATE SPECIFIED FOR FORECAST DATA
C
      IF( (KODT.EQ.ICHF).AND.(KMON.EQ.0) ) GO TO 9060
C
C.... GET THE VALUE
C
      IF( IP.GT.80 ) GO TO 9000
      CALL SHREAL(VALUE,0,NDIG,$43,$9010)
      IEND = 0
      GO TO 45
   43 IEND = 1
   45 IF( NDIG.LE.0 ) GO TO 9100
C
C.... ADJUST FOR TRACE
C
      IF( VALUE.GE.-8D10 ) GO TO 50
      IF( KODP.NE.ICHP ) GO TO 47
          IF( KODE.EQ.ICHC ) GO TO 48
          IF( KODE.EQ.ICHP ) GO TO 48
      GO TO 9090
   47 IF( KODP.NE.ICHS ) GO TO 9090
          IF( KODE.EQ.ICHD ) GO TO 48
          IF( KODE.EQ.ICHF ) GO TO 48
          IF( KODE.EQ.ICHW ) GO TO 48
      GO TO 9090
   48 VALUE = 0.001D0
      GO TO 60
C
C.... CONVERT SI TO ENGLISH UNITS IF NECESSARY
C
   50 IF( KODU.EQ.1 ) GO TO 60
      IVAL = VALUE - 0.01
      IF( IVAL.EQ.-9999 ) GO TO 60
      IF( IVAL.EQ.-9002 ) GO TO 60
      IF( FACTOR.LT.0D0 ) VALUE = VALUE * 1.8D0 + 32.0D0
      IF( FACTOR.GT.0D0 ) VALUE = VALUE * FACTOR
C
```

```
C.... TEST THE QUALIFIER - IT SHOULD BE IN ICHAR ALREADY
C
   60 IF( IEND.EQ.1 ) GO TO 65
      IF( ICHAR.EQ.ICHE ) GO TO 70              ; ESTIMATED
      IF( ICHAR.EQ.ICHR ) GO TO 70              ; REJECTED
      IF( ICHAR.EQ.ICHQ ) GO TO 70              ; QUESTIONED
      IF( ICHAR.EQ.ICHT ) GO TO 70              ; TRIGGERED
      IF( ICHAR.EQ.ICHS ) GO TO 70              ; SCREENED LEVEL 1
      IF( ICHAR.EQ.ICHV ) GO TO 70              ; SCREENED LEVEL 2
      IF( ICHAR.EQ.ICHF ) GO TO 70              ; FLAGGED
      IF( ICHAR.EQ.ICHZ ) GO TO 70              ; FILLER
      IF( ICHAR.EQ.IBLNK  ) GO TO 65
      IF( ICHAR.EQ.ISLASH ) GO TO 65
      GO TO 9120
   65 LWAL = KWAL
      NFLAG = 0
      GO TO 80
   70 LWAL = ICHAR
      NFLAG = 1
C
C.... ADJUST THE OBSERVATION TIME TO 7AM PREVIOUS DAY.
C        CAN'T DO IT IF ZULU TIME OR DATE REL HAS BEEN SPECIFIED
C
   80 NYEAR = LYEAR
      NMON  = LMON
      NDAY  = LDAY
      NHOUR = LHOUR
      NMIN  = LMIN
      IF( KODP.GE.0 ) GO TO 83
      IF( NADJTZ.EQ.0 ) GO TO 9110
      IF( MYEAR.NE.0 ) GO TO 9110
      IF( MMON .NE.0 ) GO TO 9110
      IF( MDAY .NE.0 ) GO TO 9110
      IF( MHOUR.NE.0 ) GO TO 9110
      IF( MMIN .NE.0 ) GO TO 9110
      KODP = -KODP
      IADJ = -1
      IF( NHOUR.LT.7 )
     *       CALL SHTADJ(NYEAR,NMON,NDAY,NHOUR,NMIN,IADJ,2,$9000)
      NHOUR = 7
      NMIN  = 0
      GO TO 85
C
C.... ADJUST FOR DATE RELATIVE FOR YEAR, MONTH AND DAY
C
   83 CALL SHTADJ(NYEAR,NMON,NDAY,NHOUR,NMIN,MDAY ,3,$9000)
      CALL SHTADJ(NYEAR,NMON,NDAY,NHOUR,NMIN,MMON ,4,$9000)
      CALL SHTADJ(NYEAR,NMON,NDAY,NHOUR,NMIN,MYEAR,5,$9000)
      CALL SHTADJ(NYEAR,NMON,NDAY,NHOUR,NMIN,MEND ,6,$9000)
C
C.... TEST FOR LOCAL TIME ZONE AND ADJUST TO ZULU TIME
C
   85 IADJ = 0
      CALL SHLOCL(NYEAR,NMON,NDAY,NHOUR,NMIN,NADJTZ,IADJ,$9000)
      CALL SHTADJ(NYEAR,NMON,NDAY,NHOUR,NMIN,IADJ,1,$9000)
      K1 = KYEAR
      K2 = KMON
      K3 = KDAY
      K4 = KHOUR
      K5 = KMIN
      IF( KMON.EQ.0 ) GO TO 87
      IADJ = 0
      CALL SHLOCL(K1,K2,K3,K4,K5,NADJTZ,IADJ,$9000)
```

```
      CALL SHTADJ(K1,K2,K3,K4,K5,IADJ,1,$9000)
C
C.... ADJUST FOR DATE RELATIVE FOR HOUR AND MINUTE
C
   87 CALL SHTADJ(NYEAR,NMON,NDAY,NHOUR,NMIN,MMIN ,1,$9000)
      CALL SHTADJ(NYEAR,NMON,NDAY,NHOUR,NMIN,MHOUR,2,$9000)
      CALL SHTDAT(NYEAR,NMON,NDAY,$9070)
C
C.... WRITE THE DATA TO FILE
C
   90 WRITE(JCHN) IDSTN,NYEAR,NMON,NDAY,NHOUR,NMIN,
     1                K1,K2,K3,K4,K5,
     2                KODP,KODE,IDUR,KODT,KODS,KODEX,CODP,
     3                VALUE,LWAL,IREV,MSOURCE,IDOTE
      IF( IEND.EQ.1 ) GO TO 9000
C
C.... CLEAR PAST THE NEXT SLASH
C
      IF( NFLAG.EQ.0 ) GO TO 120
  110 CALL NEXTCH(ICHAR,$9000,$9010)
  120 IF( ICHAR.EQ.IBLNK ) GO TO 110
      IF( ICHAR.NE.ISLASH ) GO TO 9120
      CALL NEXTCH(ICHAR,$9000,$9010)
      GO TO 30
C
C.... ERROR RETURN - STOP LOOKING AT THIS FORMAT
C
 9000 RETURN 2
C
C.... ERROR RETURN - STOP LOOKING AT THIS FILE
C
 9010 RETURN 3
C
 9020 NUM = 19
      GO TO 9200
C
 9030 NUM = 20
      GO TO 9200
C
 9040 NUM = 21
      GO TO 9200
C
 9050 NERR = 1
      GO TO 9000
C
 9060 NUM = 23
      NERR = 1
      GO TO 9200
C
 9070 NUM = 24
      NERR = 1
      GO TO 9200
C
 9080 NUM = 25
      NERR = 1
      GO TO 9200
C
 9090 NUM = 28
      NERR = 1
      GO TO 9200
C
 9100 NUM = 33
      NERR = 1
```

```
      GO TO 9200
C
 9110 NUM = 35
      NERR = 1
      GO TO 9200
C
 9120 NUM = 31
      NERR = 1
C
 9200 CALL SHERR(NUM)
      GO TO 9000
C
      END
```

```
      SUBROUTINE SHDECE(IREV,$,$)
C
C-----------------------------------------------------------------------C
C   VERSION 1.0 AUGUST 1982      GEOFFREY M BONNIN  MBRFC               C
C-----------------------------------------------------------------------C
C
C     DECODE SHEF .E FORMAT
C
C       IREV = 1 FOR REVISED, = 0 FOR NEW DATA
C
      DOUBLE PRECISION VALUE,FACTOR
      COMMON /DTYPE/   IEEE
      COMMON /DOTEE/   INUL,NYEAR,NMON,NDAY,NHOUR,NMIN
      COMMON /BUFFER/  IBUF(80),IP,NBLNK
      COMMON /CHAR/    ICHAR
      COMMON /LUNS/    LCHN,JCHN,KCHN,MCHN,MREC,ICHER
      COMMON /FORMAT/  IFORM,LFORM,NERR
      COMMON /DATREL/  MYEAR,MMON,MDAY,MHOUR,MMIN,MEND
      COMMON /DATA/    IDSTN(8),LYEAR,LMON,LDAY,LHOUR,LMIN,
     *                          KYEAR,KMON,KDAY,KHOUR,KMIN,
     *                 KODP,KODE,ICODD,IDCODD,KODT,KODS,KODEX,CODP,
     *                 KWAL,MSOURCE(8),ITZ,NADJTZ,KODU,VALUE,KFLAG
      COMMON /CODES/ ICHA,ICHB,ICHC,ICHD,ICHE,ICHF,ICHG,ICHH,ICHI,
     1               ICHJ,ICHK,ICHL,ICHM,ICHN,ICHO,ICHP,ICHQ,ICHR,
     2               ICHS,ICHT,ICHU,ICHV,ICHW,ICHX,ICHY,ICHZ,ICH0,
     3               ICH1,ICH2,ICH3,ICH4,ICH5,ICH6,ICH7,ICH8,ICH9,
     4               IBLNK,ISLASH,ICOLON,IPLUS,IMINUS,IDOT,IARROW,
     5               ICOMMA
C
C
C.... INITIALISE
C
      MSOURCE(1) = IBLNK
      MSOURCE(2) = IBLNK
      MSOURCE(3) = IBLNK
      MSOURCE(4) = IBLNK
      MSOURCE(5) = IBLNK
      MSOURCE(6) = IBLNK
      MSOURCE(7) = IBLNK
      MSOURCE(8) = IBLNK
C
C.... CHECK FOR CONTINUATION
C
      IF( IFORM.GT.0 ) GO TO 10
      IFORM = -IFORM
      IF( (LFORM.NE.5).AND.(LFORM.NE.6) ) GO TO 9020
      IF( (IFORM.EQ.6).AND.(LFORM.EQ.5) ) GO TO 9030
      IF( (IFORM.EQ.5).AND.(LFORM.EQ.6) ) IREV = 1
      IF( NERR.GT.0 ) GO TO 9040
      GO TO 21
C
C.... DEFAULTS
C
   10 KYEAR = 0
      KMON  = 0
      KDAY  = 0
      KHOUR = 0
      KMIN  = 0
      KWAL  = ICHZ
      KODU = 1
      MYEAR = 0
      MMON  = 0
      MDAY  = 0
```

```
      MHOUR = 0
      MMIN  = 0
      MEND = 0
      IDCODD = 5000
      ITZ = -1
      KFLAG = 0
      NERR = 0
      INUL = 0
C
C.... GET THE POSITIONAL DATA
C
   20 CALL SHPOS($9050,$9010)
      GO TO 30
C
C.... CHECK FOR A NULL
C
   21 IF( INUL.EQ.0 ) GO TO 30
      GO TO 25
   24 CALL NEXTCH(ICHAR,$170,$9010)
   25 IF( ICHAR.EQ.IBLNK ) GO TO 24
      IF( ICHAR.NE.ISLASH ) GO TO 30
      NULL = 1
      GO TO 100
C
C.... GET THE DATE AND DATA TYPE ELEMENTS
C
   30 IEEE = 0
      CALL SHDTYPE(3,$9000,$9010,$165)
      IF( (IEEE.EQ.1).AND.(KFLAG.GT.0) ) KFLAG = 1
      IF( KFLAG.NE.0 ) GO TO 45
C
C.... GET THE PARAMETER CODE
C
   40 CALL SHPCODE(KODP,KODE,ICODD,KODT,KODS,KODEX,CODP,FACTOR,
     *                      $170,$9010,$170)
      IF( IP.GT.80 ) GO TO 48
      KFLAG = 1
      IF( ICHAR.EQ.IBLNK  ) GO TO 30
      IF( ICHAR.EQ.ISLASH ) GO TO 30
      GO TO 9130
C
C.... GET THE VALUE
C
   45 CALL SHREAL(VALUE,0,NDIG,$47,$9010)
      IEND = 0
      GO TO 48
   47 IEND = 1
   48 NULL = 0
      IF(  NDIG.GT.0 ) GO TO 49
      IF( (NDIG.EQ.0).AND.(IEND.EQ.0) ) GO TO 9110
      IF( (NDIG.EQ.0).AND.(IEND.EQ.1) ) GO TO 9000
      IF( INUL.EQ.1 ) NULL = 1
      GO TO 100
C
C.... TEST IF CREATION DATE SPECIFIED FOR FORECAST DATA
C
   49 IF( (KODT.EQ.ICHF).AND.(KMON.EQ.0) ) GO TO 9060
C
C.... TEST FOR 7AM PREV DAY
C
      IF( KODP.LT.0 ) GO TO 9120
C
C.... TEST FOR CORRECT DURATION
```

```
C
      IF( (ICODD.EQ.5003).AND.(IDCODD.EQ.5000) ) GO TO 9080
      IDUR = ICODD
      IF( ICODD.EQ.5003 ) IDUR = IDCODD
C
C.... ADJUST FOR TRACE
C
      IF( VALUE.GE.-8D10 ) GO TO 50
      IF( KODP.NE.ICHP ) GO TO 491
          IF( KODE.EQ.ICHC ) GO TO 495
          IF( KODE.EQ.ICHP ) GO TO 495
      GO TO 9090
  491 IF( KODP.NE.ICHS ) GO TO 9090
          IF( KODE.EQ.ICHD ) GO TO 495
          IF( KODE.EQ.ICHF ) GO TO 495
          IF( KODE.EQ.ICHW ) GO TO 495
      GO TO 9090
  495 VALUE = 0.001D0
      GO TO 60
C
C.... CONVERT SI TO ENGLISH UNITS IF NECESSARY
C
   50 IF( KODU.EQ.1 ) GO TO 60
      IVAL = VALUE - 0.01
      IF( IVAL.EQ.-9999 ) GO TO 60
      IF( IVAL.EQ.-9002 ) GO TO 60
      IF( FACTOR.LT.0D0 ) VALUE = VALUE * 1.8D0 + 32D0
      IF( FACTOR.GT.0D0 ) VALUE = VALUE * FACTOR
C
C.... TEST THE QUALIFIER - IT SHOULD BE IN ICHAR ALREADY
C
   60 IF( IEND.EQ.1 ) GO TO 65
      IF( ICHAR.EQ.ICHE ) GO TO 70              ; ESTIMATED
      IF( ICHAR.EQ.ICHR ) GO TO 70              ; REJECTED
      IF( ICHAR.EQ.ICHQ ) GO TO 70              ; QUESTIONED
      IF( ICHAR.EQ.ICHT ) GO TO 70              ; TRIGGERED
      IF( ICHAR.EQ.ICHS ) GO TO 70              ; SCREENED LEVEL 1
      IF( ICHAR.EQ.ICHV ) GO TO 70              ; SCREENED LEVEL 2
      IF( ICHAR.EQ.ICHF ) GO TO 70              ; FLAGGED
      IF( ICHAR.EQ.ICHZ ) GO TO 70              ; FILLER
      IF( ICHAR.EQ.IBLNK  ) GO TO 65
      IF( ICHAR.EQ.ISLASH ) GO TO 65
      GO TO 9130
   65 LWAL = KWAL
      NFLAG = 0
      GO TO 100
   70 LWAL = ICHAR
      NFLAG = 1
C
C-------------------------------------------------------------
C     DO THE DATE RELATIVE AND TIME INCREMENT CALCULATIONS
C-------------------------------------------------------------
C
C.... GET THE TIME INCREMENT VALUE AND UNITS
C
  100 IF( ITZ.EQ.-1 ) GO TO 9100
      IUNIT = ITZ/1000 + 1
      INTVAL = ITZ - (IUNIT-1)*1000
C
C.... ADJUST KFLAG AND SET UP THE TIME VARIABLES
C           1 = NOT YET STARTED A TIME SERIES
C           2 = WITHIN A TIME SERIES
C           3 = FIRST ELEMENT OF A TIME SERIES
```

```
C
      IF( KFLAG.EQ.3 ) KFLAG = 2
      IF( KFLAG.EQ.2 ) GO TO 105
      IF( KFLAG.EQ.1 ) IDOTE = 1
      IF( KFLAG.EQ.1 ) KFLAG = 3
      NYEAR = LYEAR
      NMON  = LMON
      NDAY  = LDAY
      NHOUR = LHOUR
      NMIN  = LMIN
C
C.... ADJUST THE FORECAST DATE TO ZULU
C
  105 K1 = KYEAR
      K2 = KMON
      K3 = KDAY
      K4 = KHOUR
      K5 = KMIN
      IF( KMON.EQ.0 ) GO TO 110
      IADJ = 0
      CALL SHLOCL(K1,K2,K3,K4,K5,NADJTZ,IADJ,$9000)
      CALL SHTADJ(K1,K2,K3,K4,K5,IADJ,1,$9000)
C
C.... DO THE DATE RELATIVE ADJUSTMENT FOR YEAR, MONTH
C     AND DAY (LOCAL TIME) IF NECESSARY
C
  110 IF( KFLAG.EQ.2 ) GO TO 130
      IF( (MYEAR.EQ.0).AND.
     1    (MMON .EQ.0).AND.
     2    (MDAY .EQ.0) ) GO TO 120
      CALL SHTADJ(NYEAR,NMON,NDAY,NHOUR,NMIN,MDAY ,3,$9000)
      CALL SHTADJ(NYEAR,NMON,NDAY,NHOUR,NMIN,MMON ,4,$9000)
      CALL SHTADJ(NYEAR,NMON,NDAY,NHOUR,NMIN,MYEAR,5,$9000)
      CALL SHTADJ(NYEAR,NMON,NDAY,NHOUR,NMIN,MEND ,6,$9000)
C
C.... CONVERT TO ZULU IF NECESSARY
C      - DIHXX OR DINXX
C
      IF( (IUNIT.GE.3).OR.(NADJTZ.GE.0) ) GO TO 130
      IADJ = 0
      CALL SHLOCL(NYEAR,NMON,NDAY,NHOUR,NMIN,NADJTZ,IADJ,$9000)
      CALL SHTADJ(NYEAR,NMON,NDAY,NHOUR,NMIN,IADJ,1,$9000)
      GO TO 130
C
C.... DO THE DATE RELATIVE ADJUSTMENT FOR HOUR AND MINUTE
C      (ZULU TIME) IF NECESSARY
C
  120 IF( (MHOUR.EQ.0).AND.(MMIN.EQ.0) ) GO TO 125
      IADJ = 0
      CALL SHLOCL(NYEAR,NMON,NDAY,NHOUR,NMIN,NADJTZ,IADJ,$9000)
      CALL SHTADJ(NYEAR,NMON,NDAY,NHOUR,NMIN,IADJ,1,$9000)
C
      CALL SHTADJ(NYEAR,NMON,NDAY,NHOUR,NMIN,MMIN, 1,$9000)
      CALL SHTADJ(NYEAR,NMON,NDAY,NHOUR,NMIN,MHOUR,2,$9000)
C
C.... CONVERT BACK TO LOCAL TIME IF NECESSARY
C      DIYXX,DIMXX,DIDXX,DIEXX
C
      IF( (IUNIT.LT.3).AND.(NADJTZ.LE.0) ) GO TO 130
      IADJ = 1
      CALL SHLOCL(NYEAR,NMON,NDAY,NHOUR,NMIN,NADJTZ,IADJ,$9000)
      IADJ = -IADJ
      CALL SHTADJ(NYEAR,NMON,NDAY,NHOUR,NMIN,IADJ,1,$9000)
```

```
      GO TO 130
C
C.... NO DATE RELATIVES, MAKE SURE THE TIME ZONE IS CORRECT
C
  125 IF( (IUNIT.GE.3).OR.(NADJTZ.GE.0) ) GO TO 130
      IADJ = 0
      CALL SHLOCL(NYEAR,NMON,NDAY,NHOUR,NMIN,NADJTZ,IADJ,$9000)
      CALL SHTADJ(NYEAR,NMON,NDAY,NHOUR,NMIN,IADJ,1,$9000)
C
C.... NOW DO THE TIME INCREMENT
C       - WE SHOULD BE IN THE CORRECT TIME ZONE AT THIS STAGE
C
  130 IF( KFLAG.NE.2 ) GO TO 135
      IDOTE = 2
      CALL SHTADJ(NYEAR,NMON,NDAY,NHOUR,NMIN,INTVAL,IUNIT,$9000)
C
C.... IF THIS IS A NULL BYPASS THE REST
C
  135 IF( NULL.EQ.1 ) GO TO 160
C
C.... ADJUST TO ZULU FOR OUTPUT IF NECESSARY
C
      N1 = NYEAR
      N2 = NMON
      N3 = NDAY
      N4 = NHOUR
      N5 = NMIN
      IF( (IUNIT.LT.3).AND.(NADJTZ.LE.0) ) GO TO 140
      IADJ = 0
      CALL SHLOCL(N1,N2,N3,N4,N4,NADJTZ,IADJ,$9000)
      CALL SHTADJ(N1,N2,N3,N4,N5,IADJ,1,$9000)
C
C.... TEST THAT THE DATE IS OK
C
  140 CALL SHTDAT(N1,N2,N3)
C
C.... WRITE THE DATA TO FILE
C
      WRITE(JCHN) IDSTN,N1,N2,N3,N4,N5,
     1                  K1,K2,K3,K4,K5,
     2                 KODP,KODE,IDUR,KODT,KODS,KODEX,CODP,
     3                 VALUE,LWAL,IREV,MSOURCE,IDOTE
      IF( IEND.EQ.1 ) GO TO 9000
C
C.... CLEAR PAST THE NEXT SLASH
C
  145 IF( NFLAG.EQ.0 ) GO TO 160
  150 CALL NEXTCH(ICHAR,$9000,$9010)
  160 IF( ICHAR.EQ.IBLNK ) GO TO 150
      IF( ICHAR.NE.ISLASH ) GO TO 9130
      CALL NEXTCH(ICHAR,$170,$9010)
      INUL = 0
      GO TO 30
C
  165 IF( (IEEE.EQ.1).AND.(KFLAG.GT.0) ) KFLAG = 1
  170 INUL = 1
C
C.... ERROR RETURN - STOP LOOKING AT THIS FORMAT
C
 9000 RETURN 2
C
C.... ERROR RETURN - STOP LOOKING AT THIS FILE
C
```

```
 9010 RETURN 3
C
 9020 NUM = 19
      GO TO 9200
C
 9030 NUM = 20
      GO TO 9200
C
 9040 NUM = 21
      GO TO 9200
C
 9050 NERR = 1
      GO TO 9000
C
 9060 NUM = 23
      NERR = 1
      GO TO 9200
C
 9070 NUM = 24
      NERR = 1
      GO TO 9200
C
 9080 NUM = 25
      NERR = 1
      GO TO 9200
C
 9090 NUM = 28
      NERR = 1
      GO TO 9200
C
 9100 NUM = 29
      NERR = 1
      GO TO 9200
C
 9110 NUM = 33
      NERR = 1
      GO TO 9200
C
 9120 NUM = 35
      GO TO 9200
C
 9130 NUM = 31
      NERR = 1
C
 9200 CALL SHERR(NUM)
      GO TO 9000
C
      END
```

```
      SUBROUTINE SHDECB(IREV,$,$,$)
C
C-----------------------------------------------------------------------C
C   VERSION 1.0 AUGUST 1982     GEOFFREY M BONNIN  MBRFC                C
C-----------------------------------------------------------------------C
C
C     DECODE SHEF .B FORMAT
C
C       IREV = 1 FOR REVISED, = 0 FOR NEW DATA
C
      DOUBLE PRECISION FACTOR,VALUE
      COMMON /DOTBEE/ NELEM,NERROR
      COMMON /BUFFER/ IBUF(80),IP,NBLNK
      COMMON /CHAR/   ICHAR
      COMMON /LUNS/   LCHN,JCHN,KCHN,MCHN,MREC,NCHER
      COMMON /FORMAT/ IFORM,LFORM,NERR
      COMMON /DATREL/ MYEAR,MMON,MDAY,MHOUR,MMIN,MEND
      COMMON /DATA/   MSOURCE(8),LYEAR,LMON,LDAY,LHOUR,LMIN,
     *                          KYEAR,KMON,KDAY,KHOUR,KMIN,
     *                KODP,KODE,ICODD,IDCODD,KODT,KODS,KODEX,CODP,
     *                KWAL,IDSTN(8),ITZ,NADJTZ,KODU,VALUE,KFLAG
      COMMON /CODES/ ICHA,ICHB,ICHC,ICHD,ICHE,ICHF,ICHG,ICHH,ICHI,
     1               ICHJ,ICHK,ICHL,ICHM,ICHN,ICHO,ICHP,ICHQ,ICHR,
     2               ICHS,ICHT,ICHU,ICHV,ICHW,ICHX,ICHY,ICHZ,ICH0,
     3               ICH1,ICH2,ICH3,ICH4,ICH5,ICH6,ICH7,ICH8,ICH9,
     4               IBLNK,ISLASH,ICOLON,IPLUS,IMINUS,IDOT,IARROW,
     5               ICOMMA
C
C
C.... INITIALISE
C
      ITZ = 0
      IDOTE = 0
      LERR = 0
      IBOD = 0
C
C.... CHECK FOR CONTINUATION
C
      IF( IFORM.GT.0 ) GO TO 10
      IFORM = -IFORM
      IF( (LFORM.NE.3).AND.(LFORM.NE.4) ) GO TO 9020
      IF( (IFORM.EQ.4).AND.(LFORM.EQ.3) ) GO TO 9030
      IF( (IFORM.EQ.3).AND.(LFORM.EQ.4) ) IREV = 1
      IF( NERR.GT.0 ) GO TO 9040
      GO TO 30
C
C.... DEFAULTS
C
   10 KYEAR = 0
      KMON  = 0
      KDAY  = 0
      KHOUR = 0
      KMIN  = 0
      KWAL  = ICHZ
      KODU = 1
      MYEAR = 0
      MMON  = 0
      MDAY  = 0
      MHOUR = 0
      MMIN  = 0
      MEND  = 0
      IDCODD = 5000
      NERROR = 0
```

```
      MREC = 0
      NERR = 0
      REWIND MCHN
C
C.... GET THE POSITIONAL DATA
C
   20 CALL SHPOS($9050,$9010)
C
C.... GET THE DATE AND DATA TYPE ELEMENTS
C
   30 CALL SHDTYPE(2,$150,$9010,$130)
C
C.... GET THE PARAMETER CODE
C
   40 CALL SHPCODE(KODP,KODE,ICODD,KODT,KODS,KODEX,CODP,FACTOR,
     *                    $150,$9010,$50)
      IEND = 0
      IF( ICHAR.EQ.IBLNK  ) GO TO 90
      IF( ICHAR.EQ.ISLASH ) GO TO 90
      GO TO 9080
   50 IEND = 1
C
C.... WRITE THE DATA TO THE TEMPORARY FILE
C
   90 WRITE(MCHN)      MSOURCE,LYEAR,LMON,LDAY,LHOUR,LMIN,
     1                           KYEAR,KMON,KDAY,KHOUR,KMIN,
     2                           MYEAR,MMON,MDAY,MHOUR,MMIN,MEND,
     3                 KODP,KODE,ICODD,IDCODD,KODT,KODS,KODEX,CODP,
     4                 KWAL,NADJTZ,KODU,FACTOR,IREV
      MREC = MREC + 1
      IF( IEND.EQ.1 ) GO TO 130
      GO TO 120
C
C.... CLEAR PAST THE NEXT SLASH
C
  110 CALL NEXTCH(ICHAR,$130,$9010)
  120 IF( ICHAR.EQ.IBLNK ) GO TO 110
      IF( ICHAR.NE.ISLASH ) GO TO 9080
      CALL NEXTCH(ICHAR,$130,$9010)
      GO TO 30
C
C.... CHECK NEXT LINE TO SEE IF IT SHOULD BE SEARCHED FOR DATA
C
  125 IP = 81
  130 CALL NEXTCH(ICHAR,$130,$9010)
      IF( ICHAR.NE.IDOT ) GO TO 140
      CALL NEXTCH(ICHAR,$9060,$9010)
      IF( ICHAR.NE.ICHE ) GO TO 135
      CALL NEXTCH(ICHAR,$9060,$9010)
      IF( ICHAR.NE.ICHN ) GO TO 9070
      CALL NEXTCH(ICHAR,$9060,$9010)
      IF( ICHAR.NE.ICHD ) GO TO 9070
      GO TO 900
C
  135 IF( IBOD.EQ.1 ) GO TO 9070
      IF( ICHAR.NE.ICHB ) GO TO 9070
      CALL NEXTCH(ICHAR,$9060,$9010)
      CALL IRANG(ICHAR,ICH1,ICH9,$137)
      GO TO 139
  137 IF( ICHAR.NE.ICHR ) GO TO 9070
      CALL NEXTCH(ICHAR,$9060,$9010)
      CALL IRANG(ICHAR,ICH1,ICH9,$9070)
  139 IP = 0
```

```
      CALL NEXTCH(ICHAR,$9060,$9010)
      RETURN 4
C
  140 IBOD = 1
      CALL SHDOTB($160,$9010)
      LERR = 0
      GO TO 130
C
C.... ERROR CHECKING
C
  150 NERROR = NERROR + 1
  160 IF( LERR.EQ.1 ) GO TO 9000
      LERR = 1
      IF( NERROR.GE.3 ) GO TO 9000
      GO TO 125
C
C.... THATS IT
C
  900 RETURN
C
C.... ERROR RETURN - STOP LOOKING AT THIS FORMAT
C
 9000 RETURN 2
C
C.... ERROR RETURN - STOP LOOKING AT THIS FILE
C
 9010 RETURN 3
C
 9020 NUM = 19
      GO TO 9200
C
 9030 NUM = 20
      GO TO 9200
C
 9040 CALL SHERR(21)
      GO TO 125
C
 9050 NERR = 1
      GO TO 9000
C
 9060 NUM = 34
      GO TO 9200
C
 9070 IP = 0
      CALL NEXTCH(ICHAR,$9060,$9010)
      CALL SHERR(34)
      RETURN 4
C
 9080 CALL SHERR(31)
      NERR = 1
      GO TO 125
C
 9200 CALL SHERR(NUM)
      GO TO 9000
C
      END
```

```
      SUBROUTINE SHDOTB($,$)
C
C-----------------------------------------------------------------C
C   VERSION 1.0 AUGUST 1982     GEOFFREY M BONNIN  MBRFC          C
C-----------------------------------------------------------------C
C
C     Decode the body of the .B format.
C     The header information for each expected data value is
C     stored on file.
C
      DOUBLE PRECISION VALUE,FACTOR
      COMMON /BUFFER/ IBUF(80),IP,NBLNK
      COMMON /CHAR/   ICHAR
      COMMON /LUNS/   LCHN,JCHN,KCHN,MCHN,MREC,ICHER
      COMMON /DOTBEE/ NELEM,NERROR
      COMMON /DATREL/ MDUM1,MDUM2,MDUM3,MDUM4,MDUM5,MDUM6
      COMMON /DATA/   IDSTN(8),LDUM11,LDUM2,LDUM3,LDUM44,LDUM55,
     *                         KDUM1,KDUM2,KDUM3,KDUM44,KDUM55,
     *                KODP,KODE,ICODD,IDUM1,KODT,KODS,KODEX,CODP,
     *                IDUM2,MSOURCE(8),ITZ,NADJTZ,IDUM3,VALUE,KFLAG
      COMMON /CODES/ ICHA,ICHB,ICHC,ICHD,ICHE,ICHF,ICHG,ICHH,ICHI,
     1               ICHJ,ICHK,ICHL,ICHM,ICHN,ICHO,ICHP,ICHQ,ICHR,
     2               ICHS,ICHT,ICHU,ICHV,ICHW,ICHX,ICHY,ICHZ,ICH0,
     3               ICH1,ICH2,ICH3,ICH4,ICH5,ICH6,ICH7,ICH8,ICH9,
     4               IBLNK,ISLASH,ICOLON,IPLUS,IMINUS,IDOT,IARROW,
     5               ICOMMA
C
C.... INITIALISE
C
   10 ITEM = 0
      REWIND MCHN
      NELEM = 0
      IDOTE = 0
      IDSTN(1) = IBLNK
      IDSTN(2) = IBLNK
      IDSTN(3) = IBLNK
      IDSTN(4) = IBLNK
      IDSTN(5) = IBLNK
      IDSTN(6) = IBLNK
      IDSTN(7) = IBLNK
      IDSTN(8) = IBLNK
      LDUM1 = -1
      LDUM2 = -1
      LDUM3 = -1
      LDUM4 = -1
      LDUM5 = -1
      KDUM1 = -1
      KDUM2 = -1
      KDUM3 = -1
      KDUM4 = -1
      KDUM5 = -1
      MDUM1 = -1
      MDUM2 = -1
      MDUM3 = -1
      MDUM4 = -1
      MDUM5 = -1
      MDUM6 = -1
      IDUM1 = -1
      IDUM2 = -1
      IDUM3 = -1
      GO TO 25
C
C.... NOW GET THE STATION ID
```

```
C
   20 CALL NEXTCH(ICHAR,$9000,$9020)
   25 IF( ICHAR.EQ.IBLNK ) GO TO 20
C
C.... READ UP TO 8 CHARS, CHECKING FOR LEGIT ONES
C
      NCHAR = 0
      DO 50 I=1,8
         CALL IRANG(ICHAR,ICHA,ICHZ,$30)
         GO TO 40
   30    CALL IRANG(ICHAR,ICH0,ICH9,$9030)
   40    IDSTN(I) = ICHAR
         NCHAR = NCHAR + 1
         CALL NEXTCH(ICHAR,$190,$9020)
         IF( ICHAR.EQ.IBLNK ) GO TO 60
   50 CONTINUE
      IF( ICHAR.NE.IBLNK ) GO TO 9120
   60 IF( NCHAR.LT.3 ) GO TO 9040
C
C.... TEST FOR TO MANY ITEMS
C
   65 ITEM = ITEM + 1
      IF( ITEM.GT.MREC ) GO TO 70
C
C.... READ THE DATA HEADER
C
      READ(MCHN)       MSOURCE,LYEAR,LMON,LDAY,LHOUR,LMIN,
     1                          KYEAR,KMON,KDAY,KHOUR,KMIN,
     2                          MYEAR,MMON,MDAY,MHOUR,MMIN,MEND,
     3                 KODP,KODE,ICODD,IDCODD,KODT,KODS,KODEX,CODP,
     4                 KWAL,NADJTZ,KODU,FACTOR,IREV
C
C.... GET THE VALUE
C
   70 CALL SHREAL(VALUE,1,NDIG,$74,$9010)
      IEND = 0
      GO TO 77
   74 IEND = 1
   77 IF( (NDIG.EQ.0).AND.(IEND.EQ.0) ) GO TO 150
      IF( (NDIG.EQ.0).AND.(IEND.EQ.1) ) GO TO 190
      IF( NDIG.EQ.-1 ) GO TO 65
      IF( ITEM.GT.MREC ) GO TO 9050
C
C.... CHECK FOR CHANGES IN D TYPE ELEMENTS
C
      IF( LDUM1.NE.-1 ) LYEAR = LDUM1
      IF( LDUM2.NE.-1 ) LMON  = LDUM2
      IF( LDUM3.NE.-1 ) LDAY  = LDUM3
      IF( LDUM4.NE.-1 ) LHOUR = LDUM4
      IF( LDUM5.NE.-1 ) LMIN  = LDUM5
      IF( KDUM1.NE.-1 ) KYEAR = KDUM1
      IF( KDUM2.NE.-1 ) KMON  = KDUM2
      IF( KDUM3.NE.-1 ) KDAY  = KDUM3
      IF( KDUM4.NE.-1 ) KHOUR = KDUM4
      IF( KDUM5.NE.-1 ) KMIN  = KDUM5
      IF( MDUM1.NE.-1 ) MYEAR = MDUM1
      IF( MDUM2.NE.-1 ) MMON  = MDUM2
      IF( MDUM3.NE.-1 ) MDAY  = MDUM3
      IF( MDUM4.NE.-1 ) MHOUR = MDUM4
      IF( MDUM5.NE.-1 ) MMIN  = MDUM5
      IF( MDUM6.NE.-1 ) MEND  = MDUM6
      IF( IDUM1.NE.-1 ) IDCODD = IDUM1
      IF( IDUM2.NE.-1 ) KWAL   = IDUM2
```

```
      IF( IDUM3.NE.-1 ) KODU   = IDUM3
C
C.... ADJUST FOR TRACE
C
      IF( VALUE.GE.-8D10 ) GO TO 80
      IF( KODP.NE.ICHP ) GO TO 78
          IF( KODE.EQ.ICHC ) GO TO 79
          IF( KODE.EQ.ICHP ) GO TO 79
      GO TO 9060
   78 IF( KODP.NE.ICHS ) GO TO 9060
          IF( KODE.EQ.ICHD ) GO TO 79
          IF( KODE.EQ.ICHF ) GO TO 79
          IF( KODE.EQ.ICHW ) GO TO 79
      GO TO 9060
   79 VALUE = 0.001D0
      GO TO 90
C
C.... CONVERT SI TO ENGLISH UNITS IF NECESSARY
C
   80 IF( KODU.EQ.1 ) GO TO 90
      IVAL = VALUE - 0.01
      IF( IVAL.EQ.-9999 ) GO TO 90
      IF( IVAL.EQ.-9002 ) GO TO 90
      IF( FACTOR.LT.0D0 ) VALUE = VALUE * 1.8D0+ 32D0
      IF( FACTOR.GT.0D0 ) VALUE = VALUE * FACTOR
C
C.... TEST THE QUALIFIER - IT SHOULD BE IN ICHAR ALREADY
C
   90 IF( IEND.EQ.1 ) GO TO 95
      IF( ICHAR.EQ.ICHE ) GO TO 100              ; ESTIMATED
      IF( ICHAR.EQ.ICHR ) GO TO 100              ; REJECTED
      IF( ICHAR.EQ.ICHQ ) GO TO 100              ; QUESTIONED
      IF( ICHAR.EQ.ICHT ) GO TO 100              ; TRIGGERED
      IF( ICHAR.EQ.ICHS ) GO TO 100              ; SCREENED LEVEL 1
      IF( ICHAR.EQ.ICHV ) GO TO 100              ; SCREENED LEVEL 2
      IF( ICHAR.EQ.ICHF ) GO TO 100              ; FLAGGED
      IF( ICHAR.EQ.ICHZ ) GO TO 100              ; FILLER
      IF( ICHAR.EQ.IBLNK  ) GO TO 95
      IF( ICHAR.EQ.ISLASH ) GO TO 95
      IF( ICHAR.EQ.ICOMMA ) GO TO 95
      GO TO 9090
   95 LWAL = KWAL
      NFLAG = 0
      GO TO 110
  100 LWAL = ICHAR
      NFLAG = 1
C
C.... ADJUST THE OBSERVATION TIME TO 7AM PREVIOUS DAY IF NECESSARY.
C        CAN'T DO IT IF ZULU TIME OR DATE REL HAS BEEN SPECIFIED
C
  110 NYEAR = LYEAR
      NMON  = LMON
      NDAY  = LDAY
      NHOUR = LHOUR
      NMIN  = LMIN
      IF( KODP.GE.0 ) GO TO 115
      IF( NADJTZ.EQ.0 ) GO TO 9130
      IF(MYEAR.NE.0 ) GO TO 9130
      IF(MMON .NE.0 ) GO TO 9130
      IF(MDAY .NE.0 ) GO TO 9130
      IF(MHOUR.NE.0 ) GO TO 9130
      IF(MMIN .NE.0 ) GO TO 9130
      KODP = -KODP
```

```
      IADJ = -1
      IF( NHOUR.LT.7 )
     *      CALL SHTADJ(NYEAR,NMON,NDAY,NHOUR,NMIN,IADJ,2,$9000)
      NHOUR = 7
      NMIN  = 0
      GO TO 117
C
C.... ADJUST FOR DATE RELATIVE FOR YEAR, MONTH AND DAY
C
  115 CALL SHTADJ(NYEAR,NMON,NDAY,NHOUR,NMIN,MDAY ,3,$9000)
      CALL SHTADJ(NYEAR,NMON,NDAY,NHOUR,NMIN,MMON ,4,$9000)
      CALL SHTADJ(NYEAR,NMON,NDAY,NHOUR,NMIN,MYEAR,5,$9000)
      CALL SHTADJ(NYEAR,NMON,NDAY,NHOUR,NMIN,MEND ,6,$9000)
C
C.... TEST FOR LOCAL TIME ZONE AND ADJUST TO ZULU TIME
C
  117 IADJ = 0
      CALL SHLOCL(NYEAR,NMON,NDAY,NHOUR,NMIN,NADJTZ,IADJ,$9000)
      CALL SHTADJ(NYEAR,NMON,NDAY,NHOUR,NMIN,IADJ,1,$9000)
      K1 = KYEAR
      K2 = KMON
      K3 = KDAY
      K4 = KHOUR
      K5 = KMIN
      IF( KMON.EQ.0 ) GO TO 119
      IADJ = 0
      CALL SHLOCL(K1,K2,K3,K4,K5,NADJTZ,IADJ,$9000)
      CALL SHTADJ(K1,K2,K3,K4,K5,IADJ,1,$9000)
C
C.... ADJUST FOR DATE RELATIVE FOR HOUR AND MINUTE
C
  119 CALL SHTADJ(NYEAR,NMON,NDAY,NHOUR,NMIN,MMIN ,1,$9000)
      CALL SHTADJ(NYEAR,NMON,NDAY,NHOUR,NMIN,MHOUR,2,$9000)
      CALL SHTDAT(NYEAR,NMON,NDAY,$9100)
C
C.... TEST FOR CORRECT DURATION
C
  120 IF( (ICODD.EQ.5003).AND.(IDCODD.EQ.5000) ) GO TO 9070
      IDUR = ICODD
      IF( ICODD.EQ.5003 ) IDUR = IDCODD
C
C.... TEST IF CREATION DATE SPECIFIED FOR FORECAST DATA
C
      IF( (KODT.EQ.ICHF).AND.(KMON.EQ.0) ) GO TO 9080
C
C.... WRITE THE DATA TO FILE
C
  130 WRITE(JCHN) IDSTN,NYEAR,NMON,NDAY,NHOUR,NMIN,
     1                K1,K2,K3,K4,K5,
     2                KODP,KODE,IDUR,KODT,KODS,KODEX,CODP,
     3                VALUE,LWAL,IREV,MSOURCE,IDOTE
      IF( IEND.EQ.1 ) GO TO 190
      GO TO 160
C
C.... CHECK FOR DATE AND DATA TYPE ELEMENTS
C
  150 IF( ICHAR.EQ.ISLASH ) GO TO 65
      LDFL1 = 0
      IF( LDUM1.NE.-1 ) LDFL1 = 1
      LDUM11 = LYEAR
      IF( LDFL1.EQ.1 ) LDUM11 = LDUM1
C
      LDFL4 = 0
```

```
      IF( LDUM4.NE.-1 ) LDFL4 = 1
      LDUM44 = LHOUR
      IF( LDFL4.EQ.1 ) LDUM44 = LDUM4
C
      LDFL5 = 0
      IF( LDUM5.NE.-1 ) LDFL5 = 1
      LDUM55 = LMIN
      IF( LDFL5.EQ.1 ) LDUM55 = LDUM5
C
      LDFK4 = 0
      IF( KDUM4.NE.-1 ) LDFK4 = 1
      KDUM44 = KHOUR
      IF( LDFK4.EQ.1 ) KDUM44 = KDUM4
C
      LDFK5 = 0
      IF( KDUM5.NE.-1 ) LDFK5 = 1
      KDUM55 = KMIN
      IF( LDFK5.EQ.1 ) KDUM55 = KDUM5
C
      CALL SHDTYPE(4,$9010,$9020,$9000)
C
      IF( (LDFL1.EQ.0).AND.(LDUM11.NE.LYEAR) ) LDUM1 = LDUM11
      IF( LDFL1.EQ.1) LDUM1 = LDUM11
C
      IF( (LDFL4.EQ.0).AND.(LDUM44.NE.LHOUR) ) LDUM4 = LDUM44
      IF( LDFL4.EQ.1) LDUM4 = LDUM44
C
      IF( (LDFL5.EQ.0).AND.(LDUM55.NE.LMIN ) ) LDUM5 = LDUM55
      IF( LDFL5.EQ.1) LDUM5 = LDUM55
C
      IF( (LDFK4.EQ.0).AND.(KDUM44.NE.KHOUR) ) KDUM4 = KDUM44
      IF( LDFK4.EQ.1) KDUM4 = KDUM44
C
      IF( (LDFK5.EQ.0).AND.(KDUM55.NE.KMIN ) ) KDUM5 = KDUM55
      IF( LDFK5.EQ.1) KDUM5 = KDUM55
C
      GO TO 170
C
C.... CLEAR PAST THE NEXT SLASH
C
  160 IF( NFLAG.EQ.0 ) GO TO 170
  165 CALL NEXTCH(ICHAR,$190,$9020)
  170 IF( ICHAR.EQ.IBLNK ) GO TO 165
      IF( ICHAR.EQ.ISLASH ) GO TO 175
      IF( ICHAR.EQ.ICOMMA ) GO TO 180
      GO TO 9090
C
  175 IF( NELEM.EQ.0 ) GO TO 65
      NELEM = 0
      GO TO 70
C
C.... CHECK IF ALL THE ITEMS APPEARED
C
  180 LFLAG = 1
      GO TO 200
  190 LFLAG = 0
  200 IF( ITEM.LT.MREC ) GO TO 9110
      IF( IEND.EQ.1 ) GO TO 9000
      IF( LFLAG.EQ.0 ) GO TO 9000
      GO TO 230
C
C.... CLEAR TO A COMMA OR THE END OF THE LINE
C
```

```
  220 CALL NEXTCH(ICHAR,$9000,$9020)
  230 IF( ICHAR.NE.ICOMMA ) GO TO 220
      CALL NEXTCH(ICHAR,$190,$9020)
      GO TO 10
C
 9000 RETURN
C
 9010 IP = S1
      RETURN 1
C
 9020 RETURN 2
C
 9030 NUM = 6
      GO TO 9200
C
 9040 NUM = 3
      GO TO 9200
C
 9050 CALL SHERR(30)
      GO TO 230
C
 9060 NUM = 28
      GO TO 9200
C
 9070 NUM = 25
      GO TO 9200
C
 9080 NUM = 23
      GO TO 9200
C
 9090 NUM = 31
      GO TO 9200
C
 9100 NUM = 24
      GO TO 9200
C
 9110 CALL SHERR(32)
      NERROR = NERROR + 1
      IF( NERROR.GE.3 ) GO TO 9010
      IF( LFLAG.EQ.1 ) GO TO 230
      GO TO 9000
C
 9120 NUM = 2
      GO TO 9200
C
 9130 NUM = 35
C
 9200 CALL SHERR(NUM)
      NERROR = NERROR + 1
      IF( NERROR.GE.3 ) GO TO 9010
      GO TO 180
C
      END
```

```
      SUBROUTINE SHPOS($,$)
C
C-----------------------------------------------------------------C
C   VERSION 1.0 MAY 1982    GEOFFREY M BONNIN  MBRFC              C
C-----------------------------------------------------------------C
C
C     This routine decodes the positional fields of the SHEF
C     format. It returns the positional data in the common block
C     DATA.
C
      DOUBLE PRECISION VALUE
      COMMON /FORMAT/ IFORM,LFORM,NERR
      COMMON /DATIM/ IDATE(3)
      COMMON /CHAR/   ICHAR
      COMMON /DATA/   IDSTN(8),LYEAR,LMON,LDAY,LHOUR,LMIN,
     *                         KYEAR,KMON,KDAY,KHOUR,KMIN,
     *                KODP,KODE,ICODD,IDCODD,KODT,KODS,KODEX,CODP,
     *                KWAL,MSOURCE(8),ITZ,NADJTZ,KODU,VALUE,KFLAG
      COMMON /CODES/ ICHA,ICHB,ICHC,ICHD,ICHE,ICHF,ICHG,ICHH,ICHI,
     1               ICHJ,ICHK,ICHL,ICHM,ICHN,ICHO,ICHP,ICHQ,ICHR,
     2               ICHS,ICHT,ICHU,ICHV,ICHW,ICHX,ICHY,ICHZ,ICH0,
     3               ICH1,ICH2,ICH3,ICH4,ICH5,ICH6,ICH7,ICH8,ICH9,
     4               IBLNK,ISLASH,ICOLON,IPLUS,IMINUS,IDOT,IARROW,
     5               ICOMMA
C
      LHOUR = 12          ;DEFAULT FOR ZULU TIME
      LMIN  = 0
      NADJTZ = 0
      IDSTN(1) = IBLNK
      IDSTN(2) = IBLNK
      IDSTN(3) = IBLNK
      IDSTN(4) = IBLNK
      IDSTN(5) = IBLNK
      IDSTN(6) = IBLNK
      IDSTN(7) = IBLNK
      IDSTN(8) = IBLNK
C
C.... GET THE STATION ID, THERE SHOULD BE A BLANK
C
      IF( ICHAR.NE.IBLNK ) GO TO 260
   10 CALL NEXTCH(ICHAR,$9000,$9010)
      IF( ICHAR.EQ.IBLNK ) GO TO 10
C
C.... READ UP TO 8 CHARACTERS , CHECKING FOR LEGIT ONES
C
      NCHAR = 0
      DO 20 I=1,8
         CALL IRANG(ICHAR,ICHA,ICHZ,$15)          ;CHECK IF A-Z
         GO TO 17
   15    CALL IRANG(ICHAR,ICH0,ICH9,$250)         ;CHECK IF 0-9
   17    IDSTN(I) = ICHAR
         NCHAR = NCHAR + 1
         CALL NEXTCH(ICHAR,$9000,$9010)
         IF( ICHAR.EQ.IBLNK ) GO TO 30
   20 CONTINUE
      IF( ICHAR.NE.IBLNK ) GO TO 260
   30 IF( NCHAR.LT.3 ) GO TO 220
      GO TO 50
C
C.... FIND THE DATE GROUP
C
   40 IF( ICHAR.NE.IBLNK ) GO TO 60
   50 CALL NEXTCH(ICHAR,$9000,$9010)
```

```
      GO TO 40
C
C.... DECODE THE DATE GROUP
C
C     GET THE FIRST TWO DIGITS
C
   60 NDIG = 2
      CALL SHINT(LYEAR,NDIG,0,$9000,$9010,$240)
C
C     GET THE NEXT TWO DIGITS
C
      NDIG = 2
      CALL SHINT(LMON,NDIG,1,$9000,$9010,$240)
C
C     GET THE LAST TWO DIGITS
C
      NDIG = 2
      CALL SHINT(LDAY,NDIG,1,$9000,$9010,$75)
      NFLAG = 1
      GO TO 77
C
C     CHECK FOR ONLY MMDD
C
   75 IF( NDIG.GT.0 ) GO TO 240
      LDAY = LMON
      LMON = LYEAR
      NFLAG = 0
C
C     NOW DEFAULT THE YEAR
C
      LYEAR = IDATE(3) - 1900
      ITEST = LMON - IDATE(1)
      IF( ITEST.GT. 6 ) LYEAR = LYEAR - 1
      IF( ITEST.LT.-6 ) LYEAR = LYEAR + 1
      IF( (ITEST.EQ.-6).AND.(LDAY.LT.IDATE(2)) ) LYEAR = LYEAR + 1
      IF( (ITEST.EQ. 6).AND.(LDAY.GT.IDATE(2)) ) LYEAR = LYEAR - 1
C
C     TEST FOR BAD DATE
C
   77 CALL SHTDAT(LYEAR,LMON,LDAY,$240)
C
C.... FIND THE TIME ZONE GROUP (OPTIONAL)
C
      IF( NFLAG.EQ.0 ) GO TO 78
      CALL NEXTCH(ICHAR,$9000,$9010)
   78 IF( ICHAR.NE.IBLNK ) GO TO 260
   80 CALL NEXTCH(ICHAR,$9000,$9010)
   90 IF( ICHAR.EQ.IBLNK ) GO TO 80
      IF( ICHAR.NE.ICHZ  ) GO TO 95              ;ZULU SPECIFIED
      CALL NEXTCH(ICHAR,$9000,$9010)
      GO TO 205
C
C.... GET THE ACTUAL TIME ZONE
C
  95  IF( ICHAR.NE.ICHC ) GO TO 100              ;CENTRAL TIME
      NADJTZ = 360
      GO TO 180
C
 100  IF( ICHAR.NE.ICHE ) GO TO 110              ;EASTERN TIME
      NADJTZ = 300
      GO TO 180
C
 110  IF( ICHAR.NE.ICHM ) GO TO 120              ;MOUNTAIN TIME
```

```
      NADJTZ = 420
      GO TO 180
C
  120 IF( ICHAR.NE.ICHA ) GO TO 130              ;ATLANTIC TIME
      NADJTZ = 240
      GO TO 180
C
  130 IF( ICHAR.NE.ICHP ) GO TO 140              ;PACIFIC TIME
      NADJTZ = 480
      GO TO 180
C
  140 IF( ICHAR.NE.ICHL ) GO TO 150              ;ALASKAN TIME
      NADJTZ = 600
      GO TO 180
C
  150 IF( ICHAR.NE.ICHY ) GO TO 160              ;YUKON TIME
      NADJTZ = 540
      GO TO 180
C
  160 IF( ICHAR.NE.ICHH ) GO TO 170              ;HAWAIIAN TIME
      NADJTZ = 600
      GO TO 180
C
  170 IF( ICHAR.NE.ICHB ) GO TO 175              ;BERING TIME
      NADJTZ = 660
      GO TO 180
C
  175 IF( ICHAR .NE.ICHN ) GO TO 210             ;NEWFOUNDLAND TIME
      NADJTZ = 210
C
C.... NOW CHECK FOR DAYLIGHT OR STANDARD TIME
C
  180 LCHAR = ICHAR
      CALL NEXTCH(ICHAR,$9000,$9010)
      IF( ICHAR.NE.IBLNK ) GO TO 185
      IF( (LCHAR.EQ.ICHN).OR.(LCHAR.EQ.ICHH) ) GO TO 200
      NADJTZ = -NADJTZ
      GO TO 200
  185 IF( ICHAR.EQ.ICHS )  GO TO 195
  190 IF( ICHAR.NE.ICHD ) GO TO 230
      IF( (LCHAR.EQ.ICHN).OR.(LCHAR.EQ.ICHH) ) GO TO 195
      NADJTZ = NADJTZ - 60
  195 CALL NEXTCH(ICHAR,$9000,$9010)
C
C.... NOW DEFAULT TO 0000
C
  200 LHOUR = 24
C
  205 IF( ICHAR.NE.IBLNK ) GO TO 260
C.....................................................................
C
C.... RETURN OPTIONS
C
  210 RETURN
C
  220 NUM = 3              ;LESS THAN 3 CHARS IN ID OR MSG SOURCE
      GO TO 300
C
  230 NUM = 4              ;TZ CODE ERROR
      GO TO 300
C
  240 NUM = 5              ;DATE GROUP ERROR
      GO TO 300
```

```
C
  250 NUM = 6              ;BAD CHAR IN MSG SOURCE OR STN ID
      GO TO 300
C
  260 NUM = 2
C
  300 CALL SHERR(NUM)
      NERR = 1
C
 9000 RETURN 1
C
C.... FILE READ ERROR
C
 9010 RETURN 2
C
      END
```

```
      SUBROUTINE SHDTYPE(IFLAG,$,$,$)
C
C-----------------------------------------------------------------C
C   VERSION 1.0 JUNE 1982    GEOFFREY M BONNIN  MBRFC              C
C-----------------------------------------------------------------C
C
C      This module decodes the Date and Data Type elements.
C      IFLAG is set to 1 for .A format
C                       2     .B
C                       3     .E
C                       4     .B within the body
C      The routine will take the standard return if the
C      element is not valid.
C
       DOUBLE PRECISION VALUE
       COMMON /DTYPE/   IEEE
       COMMON /CHAR/    ICHAR
       COMMON /FORMAT/  IFORM,LFORM,NERR
       COMMON /DOTBEE/  NELEM,NERROR
       COMMON /DATREL/  MYEAR,MMON,MDAY,MHOUR,MMIN,MEND
       COMMON /DATA/    IDSTN(8),LYEAR,LMON,LDAY,LHOUR,LMIN,
      1                           KYEAR,KMON,KDAY,KHOUR,KMIN,
      2                 KODP,KODE,ICODD,IDCODD,KODT,KODS,KODEX,CODP,
      3                 KWAL,MSOURCE(8),ITZ,NADJTZ,KODU,VALUE,KFLAG
       COMMON /CODES/  ICHA,ICHB,ICHC,ICHD,ICHE,ICHF,ICHG,ICHH,ICHI,
      1                ICHJ,ICHK,ICHL,ICHM,ICHN,ICHO,ICHP,ICHQ,ICHR,
      2                ICHS,ICHT,ICHU,ICHV,ICHW,ICHX,ICHY,ICHZ,ICHO,
      3                ICH1,ICH2,ICH3,ICH4,ICH5,ICH6,ICH7,ICH8,ICH9,
      4                IBLNK,ISLASH,ICOLON,IPLUS,IMINUS,IDOT,IARROW,
      5                ICOMMA
C
      NELEM = 0
C
C.... LOOK FOR THE "D" OF DATA AND DATE TYPE ELEMENTS
C
   20 IF( ICHAR.NE.IBLNK ) GO TO 30
   25 CALL NEXTCH(ICHAR,$9120,$9010)
      GO TO 20
C
   30 IF( ICHAR.EQ.ICHD ) GO TO 35
      IF( ICHAR.NE.ISLASH ) GO TO 600
      IF( IFLAG.EQ.4 ) GO TO 600
      GO TO 25
C
C.... CHECK FOR THE ELEMENT POSSIBILITIES
C
   35 CALL NEXTCH(ICHAR,$9120,$9010)
      NDIG = 2
      NELEM = 1
C
C.... DN - MINUTE
C
      IF( ICHAR.NE.ICHN ) GO TO 40
      CALL SHINT(LMIN,NDIG,1,$9120,$9010,$9030)
      IF( LMIN.GT.59 ) GO TO 9030
      IF( LMIN.LT.0  ) GO TO 9030
      IF( (LHOUR.EQ.24).AND.(LMIN.GT.0) ) GO TO 9030
      GO TO 500
C
C.... DH - HOUR
C
   40 IF( ICHAR.NE.ICHH ) GO TO 50
      CALL SHINT(LHOUR,NDIG,1,$9120,$9010,$9030)
```

```
      IF( LHOUR.GT.24 ) GO TO 9030
      IF( LHOUR.LT.0  ) GO TO 9030
      IF( (LHOUR.EQ.24).AND.(LMIN.GT.0) ) GO TO 9030
      GO TO 110
C
C.... DD - DAY
C
   50 IF( ICHAR.NE.ICHD ) GO TO 60
      CALL SHINT(LDAY,NDIG,1,$9120,$9010,$9030)
      IF( LDAY.GT.31 ) GO TO 9030
      IF( LDAY.LT.0  ) GO TO 9030
      GO TO 100
C
C.... DM - MONTH
C
   60 IF( ICHAR.NE.ICHM ) GO TO 70
      CALL SHINT(LMON,NDIG,1,$9120,$9010,$9030)
      IF( LMON.GT.12 ) GO TO 9030
      IF( LMON.LT.1  ) GO TO 9030
      GO TO 90
C
C.... DY - YEAR
C
   70 IF( ICHAR.NE.ICHY ) GO TO 130
      CALL SHINT(LYEAR,NDIG,1,$9120,$9010,$9030)
      IF( LYEAR.GT.99 ) GO TO 9030
      IF( LYEAR.LT.0  ) GO TO 9030
C
   80 CALL SHINT(NUM,NDIG,1,$9120,$9010,$120)
      LMON = NUM
      IF( LMON.GT.12 ) GO TO 9030
      IF( LMON.LT.1  ) GO TO 9030
   90 CALL SHINT(NUM,NDIG,1,$9120,$9010,$120)
      LDAY = NUM
      IF( LDAY.GT.31 ) GO TO 9030
      IF( LDAY.LT.0  ) GO TO 9030
  100 CALL SHINT(NUM,NDIG,1,$9120,$9010,$120)
      LHOUR = NUM
      IF( LHOUR.GT.24 ) GO TO 9030
      IF( LHOUR.LT.0  ) GO TO 9030
  110 CALL SHINT(NUM,NDIG,1,$9120,$9010,$120)
      LMIN = NUM
      IF( LMIN.GT.59 ) GO TO 9030
      IF( LMIN.LT.0  ) GO TO 9030
      IF( (LHOUR.EQ.24).AND.(LMIN.GT.0) ) GO TO 9030
      GO TO 500
  120 IF( (LHOUR.EQ.24).AND.(LMIN.GT.0) ) GO TO 9030
      IF( NDIG.EQ.0 ) GO TO 510
      GO TO 9020
C
C.... DC - CREATION DATE
C
  130 IF( ICHAR.NE.ICHC ) GO TO 150
      KMIN  = 0
      KHOUR = 12
      CALL SHINT(KMON ,NDIG,1,$9120,$9010,$9020)
      CALL SHINT(KDAY ,NDIG,1,$9120,$9010,$9020)
      CALL SHINT(KHOUR,NDIG,1,$9120,$9010,$135)
      CALL SHINT(NUM ,NDIG,1,$9120,$9010,$145)
      KMIN = NUM
      CALL SHINT(NUM ,NDIG,1,$9120,$9010,$145)
      KYEAR = KMON
      KMON  = KDAY
```

```
      KDAY  = KHOUR
      KHOUR = KMIN
      KMIN = NUM
      JFLAG = 1
      GO TO 147
C
  135 IF( NDIG.NE.0 ) GO TO 9020
      IF( NADJTZ.NE.0 ) KHOUR = 24
C
C     CHECK FOR DEFAULT YEAR
C
  145 IF( NDIG.GT.0 ) GO TO 9020
      KYEAR = LYEAR
      ITEST = KMON - LMON
      IF( ITEST.GT. 6 ) KYEAR = KYEAR - 1
      IF( ITEST.LT.-6 ) KYEAR = KYEAR + 1
      IF( (ITEST.EQ.-6).AND.(KDAY.LT.LDAY) ) KYEAR = KYEAR + 1
      IF( (ITEST.EQ. 6).AND.(KDAY.GT.LDAY) ) KYEAR = KYEAR - 1
      JFLAG = 0
C
C     TEST FOR BAD DATE
C
  147 CALL SHTDAT(KYEAR,KMON,KDAY,$9020)
      IF( (KHOUR.LT.0).OR.(KHOUR.GT.24) ) GO TO 9020
      IF( (KMIN.LT.0).OR.(KMIN.GT.59) ) GO TO 9020
      IF( (KHOUR.EQ.24).AND.(KMIN.GT.0) ) GO TO 9020
      IF( JFLAG.EQ.0 ) GO TO 510
      IF( NDIG.EQ.0 ) GO TO 510
      GO TO 500
C
C.... DJ - JULIAN DATE
C
  150 IF( ICHAR.NE.ICHJ ) GO TO 180
      NDIG = 3
      CALL SHINT(JDAY,NDIG,1,$9120,$9010,$9020)
      NDIG = 2
      CALL SHINT(NUM,NDIG,1,$9120,$9010,$160)
      LYEAR = JDAY/10
      JDAY  = (JDAY - 10*LYEAR)*100 + NUM
      GO TO 170
  160 IF( NDIG.NE.1 ) GO TO 170
      LYEAR = (LYEAR/10)*10
      LYEAR = LYEAR + JDAY/100
      JDAY  = 10*MOD(JDAY,100) + NUM
C
C     CONVERT JULIAN DAY OF LYEAR TO MONTH, DAY
C
  170 CALL SHLEAP(LYEAR,LEAP)
      IF                      (JDAY.GT.366)   GO TO 9050
      IF( (LEAP.EQ.0) .AND. (JDAY.GT.365) ) GO TO 9050
C
      CALL SHCAL(JDAY,LEAP,LMON,LDAY)
      IF( NDIG.NE.2 ) GO TO 510
      GO TO 500
C
C.... DQ - DATA QUALIFIER
C
  180 IF( ICHAR.NE.ICHQ ) GO TO 200
      CALL NEXTCH(ICHAR,$9120,$9010)
      IF( ICHAR.EQ.ICHE ) GO TO 190              ; ESTIMATED
      IF( ICHAR.EQ.ICHR ) GO TO 190              ; REJECTED
      IF( ICHAR.EQ.ICHQ ) GO TO 190              ; QUESTIONED
      IF( ICHAR.EQ.ICHT ) GO TO 190              ; TRIGGERED
```

```
      IF( ICHAR.EQ.ICHS ) GO TO 190              ; SCREENED LEVEL 1
      IF( ICHAR.EQ.ICHV ) GO TO 190              ; SCREENED LEVEL 2
      IF( ICHAR.EQ.ICHF ) GO TO 190              ; FLAGGED
      IF( ICHAR.EQ.ICHZ ) GO TO 190              ; FILLER
      GO TO 9060
C
  190 KWAL = ICHAR
      GO TO 500
C
C.... DU - UNITS CODE
C
  200 IF( ICHAR.NE.ICHU ) GO TO 220
      CALL NEXTCH(ICHAR,$9120,$9010)
      IF( ICHAR.NE.ICHS ) GO TO 210
      KODU = 0                                   ;STANDARD INTERNATIONAL UNITS
      GO TO 500
  210 IF( ICHAR.NE.ICHE ) GO TO 9070
      KODU = 1                                   ;ENGLISH UNITS (US SYSTEM)
      GO TO 500
C
C.... DR - DATE RELATIVE INCREMENT
C
  220 IF( ICHAR.NE.ICHR ) GO TO 290
      MYEAR = 0
      MMON  = 0
      MDAY  = 0
      MHOUR = 0
      MMIN  = 0
      MEND = 0
      CALL NEXTCH(JCHAR,$9120,$9010)             ;UNITS ?
      CALL NEXTCH(NUM  ,$9120,$9010)             ;SIGN ?
      JFLAG = 0
      JSIGN = 1
      IF( NUM.NE.IPLUS  ) GO TO 230
      JFLAG = 1
      GO TO 240
  230 IF( NUM.NE.IMINUS ) GO TO 9075
      JSIGN = -1
      JFLAG = 1
C
  240 NDIG = 2
      CALL SHINT(NUM,NDIG,JFLAG,$9120,$9010,$245)        ;NUMBER ?
  245 IF( NDIG.EQ.0 ) GO TO 9075
      NUM = NUM * JSIGN
      IF( JCHAR.NE.ICHN ) GO TO 250                      ;MINUTES
      MMIN = NUM
      IF( NDIG.EQ.1 ) GO TO 510
      GO TO 500
C
  250 IF( JCHAR.NE.ICHH ) GO TO 260                      ;HOURS
      MHOUR = NUM
      IF( NDIG.EQ.1 ) GO TO 510
      GO TO 500
C
  260 IF( JCHAR.NE.ICHD ) GO TO 270                      ;DAYS
      MDAY = NUM
      IF( NDIG.EQ.1 ) GO TO 510
      GO TO 500
C
  270 IF( JCHAR.NE.ICHM ) GO TO 280                      ;MONTHS
      MMON = NUM
      IF( NDIG.EQ.1 ) GO TO 510
      GO TO 500
```

```
C
  280 IF( JCHAR.NE.ICHY ) GO TO 285                          ;YEARS
      MYEAR = NUM
      IF( NDIG.EQ.1 ) GO TO 510
      GO TO 500
C
  285 IF( JCHAR.NE.ICHE ) GO TO 9075                         ;END OF MONTH
      MEND = NUM
      IF( NDIG.EQ.1 ) GO TO 510
      GO TO 500
C
C.... DV - DURATION CODE
C
  290 IF( ICHAR.NE.ICHV ) GO TO 360
      CALL NEXTCH(ICHAR,$9120,$9010)
C
      IF( ICHAR.NE.ICHN ) GO TO 300
      NUMA = 0
      GO TO 350
C
  300 IF( ICHAR.NE.ICHH ) GO TO 310
      NUMA = 1000
      GO TO 350
C
  310 IF( ICHAR.NE.ICHD ) GO TO 320
      NUMA = 2000
      GO TO 350
C
  320 IF( ICHAR.NE.ICHM ) GO TO 330
      NUMA = 3000
      GO TO 350
C
  330 IF( ICHAR.NE.ICHY ) GO TO 340
      NUMA = 4000
      GO TO 350
C
  340 IF( ICHAR.NE.ICHZ ) GO TO 9095
      IDCODD = 5000
      GO TO 500
C
C     NOW GET THE DURATION
C
  350 NDIG = 2
      JFLAG = 1
      CALL SHINT(NUM,NDIG,JFLAG,$9120,$9010,$9095)
      IDCODD = NUM + NUMA
      GO TO 500
C
C.... DI - TIME INTERVAL FOR .E FORMAT
C
  360 IF( IFLAG.NE.3 )    GO TO 9080
      IF( ICHAR.NE.ICHI ) GO TO 9080
      CALL NEXTCH(ICHAR,$9120,$9010)
C
      IF( ICHAR.NE.ICHN ) GO TO 370
      NUMA = 0
      GO TO 410
C
  370 IF( ICHAR.NE.ICHH ) GO TO 380
      NUMA = 1000
      GO TO 410
C
  380 IF( ICHAR.NE.ICHD ) GO TO 390
```

```
      NUMA = 2000
      GO TO 410
C
  390 IF( ICHAR.NE.ICHM ) GO TO 400
      NUMA = 3000
      GO TO 410
C
  400 IF( ICHAR.NE.ICHY ) GO TO 405
      NUMA = 4000
      GO TO 410
C
  405 IF( ICHAR.NE.ICHE ) GO TO 9085
      NUMA = 5000
C
C     NOW GET THE INTERVAL
C
  410 NDIG = 2
      JFLAG = 1
      CALL SHINT(NUM,NDIG,JFLAG,$9120,$9010,$420)
      ITZ = NUMA + NUM
      GO TO 500
  420 IF( NDIG.EQ.0 ) GO TO 9085
      ITZ = NUMA + NUM
      GO TO 510
C
C.... SCAN PAST BLANKS TO THE NEXT SLASH
C
  500 IEEE = 1
      CALL NEXTCH(ICHAR,$9120,$9010)
  510 IEEE = 1
      IF( IFLAG.EQ.4 ) GO TO 600
      IF( ICHAR.EQ.IBLNK  ) GO TO 500
  520 IF( ICHAR.EQ.ISLASH ) GO TO 25
      GO TO 9090
C
C.... NOT A DATE OR DATA TYPE ELEMENT
C
  600 RETURN
C
C.... ERROR RETURNS
C
 9010 RETURN 3
C
 9020 NUM = 7
      GO TO 700
C
 9030 NUM = 8
      GO TO 700
C
 9050 NUM = 10
      GO TO 700
C
 9060 NUM = 11
      GO TO 700
C
 9070 NUM = 12
      GO TO 700
C
 9075 NUM = 9
      GO TO 700
C
 9080 NUM = 13
      GO TO 700
```

```
C
 9085 NUM = 27
      GO TO 700
C
 9090 NUM = 14
      GO TO 700
C
 9095 NUM = 26
C
  700 CALL SHERR(NUM)
      IF( IFLAG.NE.4 ) NERR = 1
C
 9110 RETURN 2
C
 9120 RETURN 4
C
      END
```

```
      SUBROUTINE SHPCODE(KODP,KODE,ICODD,KODT,KODS,KODEX,CODP,FACTOR,$,$,$)
C
C-----------------------------------------------------------------C
C   VERSION 1.0 JULY 1982    GEOFFREY M BONNIN  MBRFC              C
C-----------------------------------------------------------------C
C
C
C     This routine sets the parameter code, validates it
C     and interprets it.
C
      DOUBLE PRECISION FACTOR
      COMMON /FORMAT/ IFORM,LFORM,NERR
      COMMON /LUNS/   LCHN,JCHN,KCHN,MCHN,MREC,ICHER
      COMMON /CHAR/   ICHAR
      COMMON /CODES/ ICHA,ICHB,ICHC,ICHD,ICHE,ICHF,ICHG,ICHH,ICHI,
     1               ICHJ,ICHK,ICHL,ICHM,ICHN,ICHO,ICHP,ICHQ,ICHR,
     2               ICHS,ICHT,ICHU,ICHV,ICHW,ICHX,ICHY,ICHZ,ICHO,
     3               ICH1,ICH2,ICH3,ICH4,ICH5,ICH6,ICH7,ICH8,ICH9,
     4               IBLNK,ISLASH,ICOLON,IPLUS,IMINUS,IDOT,IARROW,
     5               ICOMMA
C
C.... SET DEFAULTS
C
      KODP  = IBLNK
      KODE  = IBLNK
      KODD  = IBLNK
      KODDD = IBLNK
      KODT  = IBLNK
      KODS  = IBLNK
      KODEX = IBLNK
      KODPR = IBLNK
      ICODD = 0
      CODP = -1.0
      IEND = 0
C
C.... SEARCH FOR A NON BLANK CHARACTER
C
   10 IF( ICHAR.NE.IBLNK ) GO TO 20
      CALL NEXTCH(ICHAR,$9500,$9010)
      GO TO 10
C
C.... IS IT A LETTER ?
C
   20 CALL IRANG(ICHAR,ICHA,ICHZ,$9020)
      JCHAR = ICHAR
C
C.... GET THE SECOND CHARACTER OF 'PE'
C
      CALL NEXTCH(ICHAR,$9500,$9010)
      CALL IRANG(ICHAR,ICHA,ICHZ,$9020)
      KODP = JCHAR
      KODE = ICHAR
C
C.... GET THE 'D' CHARACTER
C
      CALL NEXTCH(ICHAR,$65,$9010)
      IF( ICHAR.EQ.IBLNK  ) GO TO 70
      IF( ICHAR.EQ.ISLASH ) GO TO 70
      CALL IRANG(ICHAR,ICHA,ICHZ,$9020)
      KODD = ICHAR
C
C.... GET THE 'T' OF THE 'TS' CODE
C
```

```
      CALL NEXTCH(ICHAR,$65,$9010)
      IF( ICHAR.EQ.IBLNK  ) GO TO 70
      IF( ICHAR.EQ.ISLASH ) GO TO 70
      CALL IRANG(ICHAR,ICHA,ICHZ,$9020)
      KODT = ICHAR
C
C.... GET THE 'S' OF THE 'TS' CODE
C
      CALL NEXTCH(ICHAR,$65,$9010)
      IF( ICHAR.EQ.IBLNK  ) GO TO 70
      IF( ICHAR.EQ.ISLASH ) GO TO 70
      CALL IRANG(ICHAR,ICHA,ICHZ,$30)
      GO TO 40
   30 CALL IRANG(ICHAR,ICH1,ICH9,$9020)
   40 KODS = ICHAR
C
C.... GET THE 'E' OF THE EXTREMUM CODE
C
      CALL NEXTCH(ICHAR,$65,$9010)
      IF( ICHAR.EQ.IBLNK  ) GO TO 70
      IF( ICHAR.EQ.ISLASH ) GO TO 70
      CALL IRANG(ICHAR,ICHA,ICHZ,$9020)
      KODEX = ICHAR
C
C.... GET THE 'P' OF THE PROBABILITY CODE
C
      CALL NEXTCH(ICHAR,$65,$9010)
      IF( ICHAR.EQ.IBLNK  ) GO TO 70
      IF( ICHAR.EQ.ISLASH ) GO TO 70
      CALL IRANG(ICHAR,ICHA,ICHZ,$50)
      GO TO 60
   50 CALL IRANG(ICHAR,ICH1,ICH9,$9020)
   60 KODPR = ICHAR
      CALL NEXTCH(ICHAR,$65,$9010)
      GO TO 70
C
C.... VALIDATE THE 'PE' CODE
C
   65 IEND = 1
   70 IREC = (KODP-ICHA)*26 + (KODE-ICHA+1)
      READ(KCHN,REC=IREC,ERR=9030) FACTOR
      IF( FACTOR.LT.-1.5D0 ) GO TO 9040
C
C.... CHECK FOR SEND CODES
C
   80 IREC = 1439 + (KODP-ICHA)*26 + (KODE-ICHA+1)
      READ(KCHN,REC=IREC,ERR=9030) IREC
      IF( IREC.LE.0 ) GO TO 90
      IREC = 2116 + 2*(IREC-1)
      READ(KCHN,REC=IREC,ERR=9030) KODP,KODE,KODDD,KODTT
      IF( (KODD.NE.IBLNK).AND.(KODTT.NE.IBLNK) ) GO TO 9050
      IF( KODD.NE.IBLNK ) GO TO 90
      KODD = KODDD
      KODT = KODTT
      IREC = IREC + 1
      READ(KCHN,REC=IREC,ERR=9030) KODS,KODEX,KODPR
C
C.... VALIDATE AND INTERPRET THE DURATION CODE
C
   90 IF( KODD.EQ.IBLNK ) GO TO 140
      IF( KODD. NE.ICHZ  ) GO TO 92
      IF( KODDD.EQ.IBLNK ) GO TO 95
      KODD = KODDD
```

```
   92 IREC = 677 + KODD - ICHA
      READ(KCHN,REC=IREC,ERR=9030) IVAL
      IF( IVAL.EQ.-9 ) GO TO 9040
      ICODD = IVAL
C
C.... VALIDATE THE 'TS' CODE
C
   95 IF( KODT.EQ.IBLNK ) GO TO 140
      IF( KODS.EQ.IBLNK ) KODS = ICHZ
      IF( KODT.NE.ICHZ ) GO TO 98
      IF( KODS.NE.ICHZ ) GO TO 9040
      KODT = ICHR
      GO TO 105
   98 IF( KODT.NE.ICHC ) GO TO 100
      CALL IRANG(KODS,ICH1,ICH9,$100)
      GO TO 105
C
  100 IREC = 702 + (KODT-ICHA)*26 + (KODS-ICHA+1)
      READ(KCHN,REC=IREC,ERR=9030) IVAL
      IF( IVAL.LT.0 ) GO TO 9040
C
C.... VALIDATE THE EXTREMUM CODE
C
  105 IF( KODEX.EQ.IBLNK ) GO TO 160
      IREC = 1379 + (KODEX-ICHA)
      READ(KCHN,REC=IREC,ERR=9030) IVAL
      IF( IVAL.LT.0 ) GO TO 9040
C
C.... VALIDATE AND INTERPRET THE PROBABILITY CODE
C
      IF( KODPR.EQ.IBLNK ) GO TO 9000
      CALL IRANG(KODPR,ICHA,ICHZ,$110)
      IREC = 1405 + (KODPR-ICHA)
      GO TO 120
C
  110 IREC = 1431 + (KODPR-ICH1)
  120 READ(KCHN,REC=IREC,ERR=9030) VALUE
      IF( VALUE.LT.-8.0E9 ) GO TO 9040
      CODP = VALUE
      GO TO 9000
C
C.... REAL DEFAULTS
C
  140 KODT =  ICHR
  150 KODS =  ICHZ
  160 KODEX = ICHZ
C
C.... GOT IT - RETURN
C
 9000 IF( IEND.EQ.1 ) GO TO 9600
      RETURN
C
C.... ERROR RETURNS
C
 9010 RETURN 10
C
 9020 NUM = 15
      GO TO 9400
C
 9030 NUM = 16
      GO TO 9400
C
 9040 NUM = 17
```

```
      GO TO 9400
C
 9050 NUM = 18
C
 9400 CALL SHERR(NUM)
      NERR = 1
 9500 RETURN 9
C
 9600 RETURN 11
C
      END
```

```
      SUBROUTINE SHTADJ(LYEAR,LMON,LDAY,LHOUR,LMIN,KADJ,IADJ,$)
C
C-----------------------------------------------------------------C
C   VERSION 1.0 JULY 1982    GEOFFREY M BONNIN  MRFC               C
C-----------------------------------------------------------------C
C
C      This routine makes adjustments to the current date/time
C      by adding the value of IADJ. The units of LADJ are given
C      in the code IADJ where:
C           1 = minutes
C           2 = hours
C           3 = days
C           4 = months
C           5 = years
C           6 = months; end of month
C
C      It is intended that the range of the value LADJ
C      be plus/minus 99, except for minutes where the range
C      may be plus/minus 1440. Also the value of the year
C      may not change by more than one.
C
      DIMENSION IDAY(12)
      DATA IDAY /31,28,31,30,31,30,31,31,30,31,30,31/
C
C
C.... IS THERE AN ADJUSTMENT TO MAKE?
C
      IF( KADJ.EQ.0 ) GO TO 900
      LADJ = KADJ
C
C.... CHECK IADJ TO SEE WHAT'S BEING ADJUSTED
C
      GO TO (100,200,300,400,500,600), IADJ
C
C.... ADJUSTING MINUTES
C
  100 LMIN = LMIN + LADJ
      IF( LMIN.LT.60 ) GO TO 110
      LADJ = LMIN/60
      LMIN = LMIN - LADJ*60
      GO TO 200
C
  110 IF( LMIN.GE.0 ) GO TO 900
      LADJ = (LMIN-60)/60
      LMIN = LMIN - LADJ*60
      IF( LMIN.NE.60 ) GO TO 200
      LMIN = 0
      LADJ = LADJ + 1
C
C.... ADJUSTING HOURS
C
  200 LHOUR = LHOUR + LADJ
      IF( LHOUR.LT.24 ) GO TO 210
      LADJ = LHOUR/24
      LHOUR = LHOUR - LADJ*24
      GO TO 300
C
  210 IF( LHOUR.GE.0 ) GO TO 900
      LADJ = (LHOUR-24)/24
      LHOUR = LHOUR - LADJ*24
      IF( LHOUR.NE.24 ) GO TO 300
      LHOUR = 0
      LADJ = LADJ + 1
```

```
C
C.... ADJUSTING DAYS
C
C
C     CALCULATE IF THE CURRENT YEAR IS A LEAP YEAR
C
  300 CALL SHLEAP(LYEAR,LEAP)
C
C    CALCULATE THE ORDINATE DAY OF THIS YEAR
C
      IA = 30*(LMON+2) + (55*(LMON+2))/100 - 2*((LMON+10)/13)
     *     - 91 + (LEAP*(LMON+10))/13 + LDAY
C
C     ADJUST IT
C
      IA = IA + LADJ
      IF( IA.LT.(365+LEAP) ) GO TO 310
      LADJ = 1
      IA = IA - 365 - LEAP
      LEAP = 1 - LEAP
      CALL SHCAL(IA,LEAP,LMON,LDAY)
      GO TO 500
C
  310 IF( IA.LT.0 ) GO TO 320
      CALL SHCAL(IA,LEAP,LMON,LDAY)
      GO TO 900
C
  320 LADJ = -1
      LEAP = 1 - LEAP
      IA = IA + 365 + LEAP
      CALL SHCAL(IA,LEAP,LMON,LDAY)
      GO TO 500
C
C.... ADJUSTING MONTHS
C
  400 LMON = LMON + LADJ
      IF( LMON.LT.13 ) GO TO 410
      LADJ = LMON/12
      LMON = LMON - LADJ*12
      IF( LMON.NE.0 ) GO TO 500
      LMON = 12
      LADJ = LADJ - 1
      GO TO 500
C
  410 IF( LMON.GT.0 ) GO TO 900
      LADJ = (LMON-12)/12
      LMON = LMON - LADJ*12
C
C.... ADJUSTING THE YEAR
C
  500 LYEAR = LYEAR + LADJ
      GO TO 900
C
C.... ADJUSTING MONTHS; END OF MONTH
C        IS THIS ACTUALLY THE END OF A MONTH?
  600 LEAP = 0
      IF( LMON.NE.2 ) GO TO 610
      CALL SHLEAP(LYEAR,LEAP)
  610 IA = IDAY(LMON) + LEAP
      IF( IA.NE.LDAY ) GO TO 9000        ;NO IT'S NOT
C
C     YES IT IS - ADJUST IT
C
```

```
      LMON = LMON + LADJ
      IF( LMON.LT.13 ) GO TO 620
      LADJ = LMON/12
      LMON = LMON - LADJ * 12
      IF( LMON.NE.0 ) GO TO 630
      LMON = 12
      LADJ = LADJ - 1
      GO TO 630
C
  620 IF( LMON.GT.0 ) GO TO 640
      LADJ = (LMON-12)/12
      LMON = LMON - LADJ*12
C
C     DO THE YEAR
C
  630 LYEAR = LYEAR + LADJ
C
C     NOW GET THE CORRECT DAY
C
  640 LEAP = 0
      IF( LMON.NE.2 ) GO TO 650
      CALL SHLEAP(LYEAR,LEAP)
  650 LDAY = IDAY(LMON) + LEAP
C
C.... RETURN
C
  900 RETURN
C
 9000 CALL SHERR(36)
      RETURN 8
C
      END
```

```
      SUBROUTINE SHCAL(IORD,LEAP,LMON,LDAY)
C
C-----------------------------------------------------------------------C
C   VERSION 1.0 JULY 1982     GEOFFREY M BONNIN  MRFC                   C
C-----------------------------------------------------------------------C
C
C
C     Calculate the calender month and day given the ordinal
C     day in IORD. LEAP is 1 if a leap year, 0 otherwise.
C
C     IA = IORD + ( (405+IORD-LEAP)/365 ) * (2-LEAP)
C     LMON = ( (IA+91)*100 )/3055 - 2
C     LDAY = IA + 30 - 30*LMON - (56*LMON)/100
C
C     RETURN
C     END
C-----------------------------------------------------------------------
C
      DIMENSION IDAY(12)
      DATA IDAY /31,28,31,30,31,30,31,31,30,31,30,31/
C
      LDAY = IORD
      IDAY(2) = IDAY(2) + LEAP
C
      DO 10 I=1,12
      LMON = I
      IF( LDAY.LE.IDAY(I) ) GO TO 100
      LDAY = LDAY - IDAY(I)
   10 CONTINUE
C
  100 IDAY(2) = IDAY(2) - LEAP
      RETURN
      END
```

```
      SUBROUTINE SHLEAP(LYEAR,LEAP)
C
C-----------------------------------------------------------------------C
C   VERSION 1.0 JULY 1982    GEOFFREY M BONNIN  MRFC                   C
C-----------------------------------------------------------------------C
C
C      Returns LEAP = 1 if LYEAR is a leap year. LEAP = 0 otherwise.
C      LYEAR is the 2 digit year.
C
      LEAP = LYEAR + 1900                  ;WHAT HAPPENS IN YEAR 2000 ?
      IA = (LEAP - (LEAP/4   )*4    + 3  )/4
      IB = (LEAP - (LEAP/100)*100 + 99 )/100
      IC = (LEAP - (LEAP/400)*400 + 399)/400
      LEAP = 1 - IA + IB - IC              ;LEAP IS 1 IF LEAP YEAR
C
      RETURN
      END
```

```
      SUBROUTINE SHTDAT(LYEAR,LMON,LDAY,$)
C
C-----------------------------------------------------------------------C
C    VERSION 1.0 AUGUST 1982     GEOFFREY M BONNIN  MBRFC                C
C-----------------------------------------------------------------------C
C
C       this routine checks to see if a date is valid.
C
C.... TEST THE MONTH
C
      IF( (LMON.GT.12).OR.(LMON.LT.1) ) GO TO 9000
C
C.... TEST THE DAY
C
      MON = LMON
      IF( LDAY.LT.1 ) GO TO 9000
      IF( MON.EQ.2 )  GO TO 10
      IF( MON.GE.8 ) MON = MON + 1
      LEAP = 30 + MON - (MON/2)*2
      IF( LDAY.GT.LEAP ) GO TO 9000
      GO TO 100
C
   10 CALL SHLEAP(LYEAR,LEAP)
      IF( LDAY.GT.(LEAP + 28) ) GO TO 9000
C
  100 RETURN
C
 9000 RETURN 4
C
      END
```

```
      SUBROUTINE SHLOCL(NYEAR,NMON,NDAY,NHOUR,NMIN,NADJTZ,IADJ,$)
C
C-----------------------------------------------------------------C
C   VERSION 1.0 AUGUST 1982    GEOFFREY M BONNIN  MBRFC           C
C-----------------------------------------------------------------C
C
C     This routine adjusts the time difference between the local time
C     and zulu time depending on whether or not daylight
C     savings is in effect.
C     If IADJ comes in as 1 then the time coming in is zulu.
C     If IADJ comes in as 0 then the time coming in is local time.
C     The change over dates for daylight and standard time for the
C     years 1976 through 1999 are contained in the table ITABLE.
C     This information is as supplied by Phil Pasteris NWRFC.
C
      DIMENSION ITABLE(2,24)
C
C     ITABLE(1,I) is the change over date in April to daylight time.
C     ITABLE(2,I) is the change over date in October to standard time.
C     The change is assumed to occur at 2am.
C
      DATA ITABLE
     1           /26,31, 24,30, 30,29, 29,28, 27,26, 26,25,
     2            25,31, 24,30, 29,28, 28,27, 27,26, 26,25,
     3            24,30, 30,29, 29,28, 28,27, 26,25, 25,31,
     4            24,30, 30,29, 28,27, 27,26, 26,25, 25,31/
C
C.... TEST IF ADJUSTMENT NECESARY
C
      IF( NADJTZ.GE.0 ) GO TO 50
C
C.... CHECK WHETHER THE TIME IS IN LOCAL OR ZULU
C
      IA = 2
      IB = 2
      IF( IADJ.NE.1 ) GO TO 5
      IA = 2 - NADJTZ/60
      IB = 1 - NADJTZ/60
C
C.... ADJUST THE YEAR TO SUIT THE BOUNDS OF THE ITABLE
C
    5 IF( (NYEAR.LT.76).OR.(NYEAR.GT.99) ) GO TO 9010
      IY = NYEAR - 75
C
C.... TEST IF DAYLIGHT TIME
C
      IF( (NMON.LT.4) .OR.(NMON.GT.10) ) GO TO 40
      IF( (NMON.GT.4).AND.(NMON.LT.10) ) GO TO 20
C
      IF( NMON.EQ.10 ) GO TO 10
C
C      APRIL
C
      IF( NDAY.LT.ITABLE(1,IY) ) GO TO 40
      IF( NDAY.GT.ITABLE(1,IY) ) GO TO 20
C
      IF( IADJ .EQ.1 ) GO TO 8
      IF( NHOUR.NE.2 ) GO TO 8
      IF( NMIN .NE.0 ) GO TO 9020
C
    8 IF( NHOUR.LT.IA ) GO TO 40
      IF( NHOUR.GT.IA ) GO TO 20
      IF( NMIN.GT.0 ) GO TO 20
```

```
      GO TO 40
C
C       OCTOBER
C
   10 IF( NDAY.GT.ITABLE(2,IY) ) GO TO 40
      IF( NDAY.LT.ITABLE(2,IY) ) GO TO 20
      IF( NHOUR.GT.IB ) GO TO 40
      IF( NHOUR.LT.IB ) GO TO 20
      IF( NMIN.GT.0 ) GO TO 40
C
C.... ADJUST FOR DAYLIGHT TIME
C
   20 IADJ = -(NADJTZ + 60)
      GO TO 100
C
C.... STANDARD TIME
C
   40 IADJ = -NADJTZ
      GO TO 100
C
C.... NO ADJUSTMENT NECESSARY
C
   50 IADJ = NADJTZ
C
C.... THATS IT
C
  100 RETURN
C
C.... ERROR RETURN
C
 9010 NUM = 22
      GO TO 9500
C
 9020 NUM = 37
C
 9500 CALL SHERR(NUM)
      RETURN 8
C
      END
```

```
      SUBROUTINE SHINT(NUM,NDIG,IFLAG,$,$,$)
C
C-------------------------------------------------------------------C
C   VERSION 1.0 APRIL 1982    GEOFFREY M BONNIN  MBRFC              C
C-------------------------------------------------------------------C
C
C      This routine will extract an inteser of NDIG disits from
C      the input file and return it in NUM. If IFLAG is set to 0,
C      it uses the last character read. The number of disits
C      actually read is returned in NDIG.
C
       COMMON /CHAR/ ICHAR
       COMMON /CODES/ ICHA,ICHB,ICHC,ICHD,ICHE,ICHF,ICHG,ICHH,ICHI,
      1               ICHJ,ICHK,ICHL,ICHM,ICHN,ICHO,ICHP,ICHQ,ICHR,
      2               ICHS,ICHT,ICHU,ICHV,ICHW,ICHX,ICHY,ICHZ,ICH0,
      3               ICH1,ICH2,ICH3,ICH4,ICH5,ICH6,ICH7,ICH8,ICH9,
      4               IBLNK,ISLASH,ICOLON,IPLUS,IMINUS,IDOT,IARROW,
      5               ICOMMA
       NUM  = 0
       IDIG = 0
C
C.... GET THE NEXT DIGIT AND CHECK IT
C
       IF( IFLAG.EQ.0 ) GO TO 20
   10  CALL NEXTCH(ICHAR,$9010,$9020)
   20  CALL IRANG(ICHAR,ICH0,ICH9,$9030)
C
C.... BUILD NUMBER
C
       NUM = NUM*10 + (ICHAR-ICH0)
C
C.... CHECK IF ANOTHER DIGIT IS REQUIRED
C
       IDIG = IDIG + 1
       IF( IDIG.GE.NDIG ) GO TO 9000
       GO TO 10
C.....................................................................
C
C.... NORMAL RETURN
C
 9000  RETURN
C
C.... STOP READING THE LINE
C
 9010  RETURN 4
C
C.... ERROR IN READING THE FILE
C
 9020  RETURN 5
C
C.... ERROR IN NUMBER OF DIGITS
C
 9030  NDIG = IDIG
       RETURN 6
C
C
       END
```

```
      SUBROUTINE SHREAL(VALUE,IFLAG,NDIG,$,$)
C
C-----------------------------------------------------------------C
C   VERSION 1.0 JULY 1982    GEOFFREY M BONNIN  MBRFC              C
C-----------------------------------------------------------------C
C
C       This routine reads real numbers with an optional sign.
C       if IFLAG is set to zero the scan starts with the last
C       character read.
C
      DOUBLE PRECISION VALUE,SIGN,X
      COMMON /CHAR/ ICHAR
      COMMON /CODES/ ICHA,ICHB,ICHC,ICHD,ICHE,ICHF,ICHG,ICHH,ICHI,
     1               ICHJ,ICHK,ICHL,ICHM,ICHN,ICHO,ICHP,ICHQ,ICHR,
     2               ICHS,ICHT,ICHU,ICHV,ICHW,ICHX,ICHY,ICHZ,ICH0,
     3               ICH1,ICH2,ICH3,ICH4,ICH5,ICH6,ICH7,ICH8,ICH9,
     4               IBLNK,ISLASH,ICOLON,IPLUS,IMINUS,IDOT,IARROW,
     5               ICOMMA
C
C.... INITIALISE
C
      VALUE = 0D0
      ISIGN = 0
      SIGN  = 1D0
      IDEC  = 0
      NDIG  = 0
      IEND = 0
C
C.... CLEAR PAST BLANKS
C
      IF( IFLAG.EQ.0 ) GO TO 7
    5 CALL NEXTCH(ICHAR,$9010,$9020)
    7 IF( ICHAR.EQ.IBLNK ) GO TO 5
      GO TO 20
C
C.... GET THE NEXT CHARACTER
C
   10 CALL NEXTCH(ICHAR,$9010,$9020)
C
C.... CHECK FOR SIGN
C
   20 IF( (ICHAR.NE.IPLUS).AND.(ICHAR.NE.IMINUS) ) GO TO 30
      IF( ISIGN.NE.0 ) GO TO 100
      ISIGN = 1
      IF( ICHAR.EQ.IMINUS ) SIGN = -1D0
      GO TO 10
C
C.... CHECK FOR DECIMAL
C
   30 IF( ICHAR.NE.IDOT ) GO TO 40
      IF( IDEC.GT.0 ) GO TO 100
      IDEC = 1
      GO TO 10
C
C.... CHECK FOR 'M' - MISSING
C
   40 IF( ICHAR.NE.ICHM ) GO TO 50
      IF( (NDIG.GT.0).OR.(ISIGN.NE.0) ) GO TO 100
      VALUE = -9999D0
      NDIG = 1
      CALL NEXTCH(ICHAR,$9010,$9020)
      GO TO 9000
C
```

```
C.... CHECK FOR 'T' = TRACE = 0.001
C
   50 IF( ICHAR.NE.ICHT ) GO TO 60
      IF( (NDIG.GT.0).OR.(ISIGN.NE.0) ) GO TO 100
      VALUE = -9D10
      NDIG = 1
      CALL NEXTCH(ICHAR,$9010,$9020)
      GO TO 9000
C
C.... CHECK FOR A DIGIT AND ADJUST THE VALUE
C
   60 CALL IRANG(ICHAR,ICH0,ICH9,$90)
      NDIG = NDIG + 1
      IF( IDEC.GT.0 ) GO TO 70
      VALUE = 10D0 * VALUE + (ICHAR-ICH0)
      GO TO 10
C
   70 X = ICHAR - ICH0
      DO 80 I=1,IDEC
         X = X/10D0
   80 CONTINUE
      VALUE = VALUE + X
      IDEC = IDEC + 1
      GO TO 10
C
C.... CHECK FOR / +    = MISSING
C
   90 IF( NDIG.GT.0 ) GO TO 100
      IF( (ICHAR.EQ.ISLASH).AND.(ISIGN.EQ.0) ) GO TO 110
      IF( ISIGN.EQ.0   ) GO TO 100
      IF(  SIGN.LT.0D0 ) GO TO 100
      VALUE = -9999D0
      NDIG = 1
      GO TO 9000
C
C.... THIS IS A NULL ?
C
  110 NDIG = -1
      GO TO 9000
C
C.... MULTIPLY BY THE SIGN
C
  100 VALUE = SIGN * VALUE
      IF( IEND.EQ.1 ) GO TO 9011
C
C.... THAT'S IT
C
 9000 RETURN
C
C.... STOP READING THE LINE
C
 9010 IEND = 1
      GO TO 100
C
 9011 RETURN 4
C
C.... ERROR IN READING FILE
C
 9020 RETURN 5
C
      END
```

```
      SUBROUTINE NEXTCH(ICHAR,$,$)
C
C-----------------------------------------------------------------C
C   VERSION 1.0 APRIL 1982    GEOFFREY M BONNIN  MBRFC            C
C-----------------------------------------------------------------C
C
C       READ LINES INTO A BUFFER AND PROVIDE THE NEXT CHARACTER
C       IP POINTS TO THE LAST CHAR PROVIDED
C       ANY THING AFTER AND INCLUDING A COLON IS IGNORED
C
       DIMENSION IBUF(80)
       COMMON /BUFFER/ IBUF,IP,NBLNK
       COMMON /CHAR/   LASTCH
       COMMON /LUNS/   LCHN,JCHN,KCHN,MCHN,MREC,ICHER
       COMMON /CODES/ ICHA,ICHB,ICHC,ICHD,ICHE,ICHF,ICHG,ICHH,ICHI,
      1               ICHJ,ICHK,ICHL,ICHM,ICHN,ICHO,ICHP,ICHQ,ICHR,
      2               ICHS,ICHT,ICHU,ICHV,ICHW,ICHX,ICHY,ICHZ,ICH0,
      3               ICH1,ICH2,ICH3,ICH4,ICH5,ICH6,ICH7,ICH8,ICH9,
      4               IBLNK,ISLASH,ICOLON,IPLUS,IMINUS,IDOT,IARROW,
      5               ICOMMA
C
C.... UPDATE POINTER AND CHECK IT
C
   10 IP = IP + 1
      IF( IP.EQ.81 ) GO TO 910
      IF( IP.GT.80 ) GO TO 40
C
C.... GET THE CHARACTER, SHIFT TO RIGHT BYTE
C       AND UPDATE LAST CHARACTER.
C
      LCHAR = LASTCH
      ICHAR = IBUF(IP)
C
      ICHAR = ISHFT(ICHAR,-8)
      LASTCH = ICHAR
C
C.... TEST FOR 15 CONSECUTIVE BLANKS, END OF LINE
C
      IF( ICHAR.NE.IBLNK ) GO TO 30
      IF( LCHAR.NE.IBLNK ) GO TO 20
      NBLNK = NBLNK + 1
      IF( NBLNK.LT.15 ) GO TO 30
      IP = 81
      GO TO 910
   20 NBLNK = 1
C
C.... CHECK FOR A COLON
C
   30 IF( ICHAR.NE.ICOLON ) GO TO 900
      IP = 81
      GO TO 910
C
C.... READ THE NEXT LINE
C
   40 DO 45 I=1,80
         IBUF(I) = IBLNK
   45 CONTINUE
      READ(LCHN,50,ERR=920,END=920) IBUF
   50 FORMAT(80A1)
      DO 60 I=80,1,-1
         IF( ISHFT(IBUF(I),-8).NE.IBLNK ) GO TO 70
   60 CONTINUE
      I = 1
```

```
   70 WRITE(ICHER,50) (IBUF(J),J=1,I)
      IP = 0
      GO TO 10
C
C.... GOT IT
C
  900 RETURN
C
C.... 15 CONSECUTIVE BLANKS, END OF LINE
C
  910 NBLNK = 0
      RETURN 2
C
C.... FILE READ ERROR
C
  920 RETURN 3
C
      END
```

```
C
C-----------------------------------------------------------------C
C   VERSION 1.0 MAY 1982    GEOFFREY M BONNIN  MBRFC              C
C-----------------------------------------------------------------C
C
C       Module CODES
C
C       This module sets up ASCII code in the labeled common area CODES.
C
      BLOCKDATA
      COMMON /CODES/ ICHA,ICHB,ICHC,ICHD,ICHE,ICHF,ICHG,ICHH,ICHI,
     1               ICHJ,ICHK,ICHL,ICHM,ICHN,ICHO,ICHP,ICHQ,ICHR,
     2               ICHS,ICHT,ICHU,ICHV,ICHW,ICHX,ICHY,ICHZ,ICHO,
     3               ICH1,ICH2,ICH3,ICH4,ICH5,ICH6,ICH7,ICH8,ICH9,
     4               IBLNK,ISLASH,ICOLON,IPLUS,IMINUS,IDOT,IARROW,
     5               ICOMMA
C
      DATA ICHA     /'<000>A'/
      DATA ICHB     /'<000>B'/
      DATA ICHC     /'<000>C'/
      DATA ICHD     /'<000>D'/
      DATA ICHE     /'<000>E'/
      DATA ICHF     /'<000>F'/
      DATA ICHG     /'<000>G'/
      DATA ICHH     /'<000>H'/
      DATA ICHI     /'<000>I'/
      DATA ICHJ     /'<000>J'/
      DATA ICHK     /'<000>K'/
      DATA ICHL     /'<000>L'/
      DATA ICHM     /'<000>M'/
      DATA ICHN     /'<000>N'/
      DATA ICHO     /'<000>O'/
      DATA ICHP     /'<000>P'/
      DATA ICHQ     /'<000>Q'/
      DATA ICHR     /'<000>R'/
      DATA ICHS     /'<000>S'/
      DATA ICHT     /'<000>T'/
      DATA ICHU     /'<000>U'/
      DATA ICHV     /'<000>V'/
      DATA ICHW     /'<000>W'/
      DATA ICHX     /'<000>X'/
      DATA ICHY     /'<000>Y'/
      DATA ICHZ     /'<000>Z'/
C
      DATA ICHO     /'<000>0'/
      DATA ICH1     /'<000>1'/
      DATA ICH2     /'<000>2'/
      DATA ICH3     /'<000>3'/
      DATA ICH4     /'<000>4'/
      DATA ICH5     /'<000>5'/
      DATA ICH6     /'<000>6'/
      DATA ICH7     /'<000>7'/
      DATA ICH8     /'<000>8'/
      DATA ICH9     /'<000>9'/
C
      DATA IBLNK    /'<000> '/
      DATA ISLASH   /'<000>/'/
      DATA ICOLON   /'<000>:'/
      DATA IPLUS    /'<000>+'/
      DATA IMINUS   /'<000>-'/
      DATA IDOT     /'<000>.'/
      DATA IARROW   /'<000>^'/
      DATA ICOMMA   /'<000>,'/
```

```
C
      END
```

```
      SUBROUTINE IRANG(I,MIN,MAX,$)
C
C     DAVE LEADER    CNRFC
C
C ROUTINE CHECKS IF AN INTEGER IS IN THE RANGE MIN <= I <= MAX
C
      IF(I.LT.MIN) RETURN 4
      IF(I.GT.MAX) RETURN 4
C
      RETURN
C
      END
```

```
      SUBROUTINE SHERR(IER)
C
C-----------------------------------------------------------------C
C    VERSION 1.0 APRIL 1982    GEOFFREY M BONNIN  MBRFC           C
C-----------------------------------------------------------------C
C
      DIMENSION LINE(80)
      COMMON /CODES/ ICHA,ICHB,ICHC,ICHD,ICHE,ICHF,ICHG,ICHH,ICHI,
     1               ICHJ,ICHK,ICHL,ICHM,ICHN,ICHO,ICHP,ICHQ,ICHR,
     2               ICHS,ICHT,ICHU,ICHV,ICHW,ICHX,ICHY,ICHZ,ICH0,
     3               ICH1,ICH2,ICH3,ICH4,ICH5,ICH6,ICH7,ICH8,ICH9,
     4               IBLNK,ISLASH,ICOLON,IPLUS,IMINUS,IDOT,IARROW,
     5               ICOMMA
      COMMON /BUFFER/ IBUF(80),IP,NBLNK
      COMMON /ERROR/ NERROR
      COMMON /LUNS/    LCHN,JCHN,KCHN,MCHN,MREC,ICHER
C
      IF( (IP.LE.0).OR.(IP.GE.81) ) GO TO 30
      DO 10 I=1,IP
         LINE(I) = IBLNK
   10 CONTINUE
      LINE(IP) = IARROW
      WRITE(ICHER,20) (LINE(I),I=1,IP)
   20 FORMAT(80R1)
C
   30 NERROR = NERROR + 1
      WRITE(ICHER,40) IER
   40 FORMAT(20X, 'SHEF ERROR NUMBER',I3)
C
      RETURN
      END
```

APPENDIX E

SUPPORT SOFTWARE

```
C      OUTPUT
C
C-----------------------------------------------------------------C
C   VERSION 1.0 JULY 1982      GEOFFREY M BONNIN  MBRFC           C
C-----------------------------------------------------------------C
C
C       READ THE SHEFOUT FILE AND PRINT IT FOR DEBUG PURPOSES
       DOUBLE PRECISION VALUE
       DIMENSION IDSTN(8),MSOURCE(8)
C
C.... OPEN THE FILES
C
       OPEN 10,'SHEFOUT'
       OPEN 11,'OUTPUTFIL',ATT='P'
C
C.... THROW AWAY THE ACTIVE FLAG
C
       READ (10) IACTIVE
C
C.... WRITE THE PAGE HEADER
C
    10 WRITE(11,20)
    20 FORMAT(1H1/1X,' STN ID       OBS DATE       CREATION DATE',
      1            '  PE   DURATION  TSE  PROBABILITY  VALUE',
      2            '     QUAL  REV  SOURCE  IDOTE'//)
C
C.... NOW READ AND PRINT A PAGE
C
       DO 100 I=1,30
       READ(10,END=900) IDSTN,NYEAR,NMON,NDAY,NHOUR,NMIN,
      1                        KYEAR,KMON,KDAY,KHOUR,KMIN,
      2                 KODP,KODE,IDUR,KODT,KODS,KODEX,CODP,
      3                 VALUE,LWAL,IREV,MSOURCE,IDOTE
        IDSTN(1) = ISHFT(IDSTN(1),8)
        IDSTN(2) = ISHFT(IDSTN(2),8)
        IDSTN(3) = ISHFT(IDSTN(3),8)
        IDSTN(4) = ISHFT(IDSTN(4),8)
        IDSTN(5) = ISHFT(IDSTN(5),8)
        IDSTN(6) = ISHFT(IDSTN(6),8)
        IDSTN(7) = ISHFT(IDSTN(7),8)
        IDSTN(8) = ISHFT(IDSTN(8),8)
        KODP     = ISHFT(KODP,    8)
        KODE     = ISHFT(KODE,    8)
        KODT     = ISHFT(KODT,    8)
        KODS     = ISHFT(KODS,    8)
        KODEX    = ISHFT(KODEX,   8)
        LWAL     = ISHFT(LWAL,    8)
        MSOURCE(1) = ISHFT(MSOURCE(1),8)
        MSOURCE(2) = ISHFT(MSOURCE(2),8)
        MSOURCE(3) = ISHFT(MSOURCE(3),8)
        MSOURCE(4) = ISHFT(MSOURCE(4),8)
        MSOURCE(5) = ISHFT(MSOURCE(5),8)
        MSOURCE(6) = ISHFT(MSOURCE(6),8)
        MSOURCE(7) = ISHFT(MSOURCE(7),8)
        MSOURCE(8) = ISHFT(MSOURCE(8),8)
        WRITE(11,30) IDSTN,NYEAR,NMON,NDAY,NHOUR,NMIN,
      1                        KYEAR,KMON,KDAY,KHOUR,KMIN,
      2                 KODP,KODE,IDUR,KODT,KODS,KODEX,CODP,
      3                 VALUE,LWAL,IREV,MSOURCE,IDOTE
    30 FORMAT(1X,8A1,2(2X,4(I2,'/'),I2),2X,2A1,I7,6X,3A1,F10.3,
      1          2X,G10.3,4X,A1,4X,I1,3X,8A1,I3/)
   100 CONTINUE
       GO TO 10
```

```
C
C     THATS IT
C
  900 WRITE(11,110)
  110 FORMAT(1H1)
      CALL EXIT
      END
```

```
C      CPARM
C
C-----------------------------------------------------------------C
C   VERSION 1.0 JULY 1982    GEOFFREY M BONNIN  MBRFC             C
C-----------------------------------------------------------------C
C
C      CREATE THE SHEF PARAMETER FILE 'SHEFPARM' FROM THE
C      THE FILE 'INPUTPARM'.
C
       DOUBLE PRECISION VALUE
       DIMENSION IBUF(2),ISEND(7)
       DATA IAST/'<000>*'/
       COMMON /CODES/ ICHA,ICHB,ICHC,ICHD,ICHE,ICHF,ICHG,ICHH,ICHI,
      1               ICHJ,ICHK,ICHL,ICHM,ICHN,ICHO,ICHP,ICHQ,ICHR,
      2               ICHS,ICHT,ICHU,ICHV,ICHW,ICHX,ICHY,ICHZ,ICHO,
      3               ICH1,ICH2,ICH3,ICH4,ICH5,ICH6,ICH7,ICH8,ICH9,
      4               IBLNK,ISLASH,ICOLON,IPLUS,IMINUS,IDOT,IARROW,
      5               ICOMMA
C
C.... OPEN FILES
C
       OPEN 20,'SHEFPARM',LEN=8
       OPEN 21,'INPUTPARM'
C
C      INITIAL SETUP OF THE SHEF PARAMETER FILE 'SHEFPARM'.
C           .... NOTHING EXISTS!
C
       VALUE = -9D9
       IVAL  = -9
C
       DO 2 I=1,676
          WRITE(20,REC=I) VALUE
    2  CONTINUE
C
       DO 4 I=677,1404
          WRITE(20,REC=I) IVAL
    4  CONTINUE
C
       DO 6 I=1405,1439
          WRITE(20,REC=I) VALUE
    6  CONTINUE
C
       DO 8 I=1440,2115
          WRITE(20,REC=I) IVAL
    8  CONTINUE
C
C.... READ A RECORD
C
   10  READ(21,20,END=9000) IBUF
   20  FORMAT(2A1)
       IBUF(1) = ISHFT(IBUF(1),-8)
       IBUF(2) = ISHFT(IBUF(2),-8)
C
C.... INTERPRET IT
C
   25  IF( IBUF(1).NE.IAST ) GO TO 9010
C
C.... PE CODE
C
       IF( IBUF(2).NE.ICH1 ) GO TO 50
   30  READ(21,40,END=9000) IBUF,VALUE
   40  FORMAT(2A1,G20.0)
       IBUF(1) = ISHFT(IBUF(1),-8)
```

```
      IBUF(2) = ISHFT(IBUF(2),-8)
      CALL IRANG(IBUF(1),ICHA,ICHZ,$25)
      CALL IRANG(IBUF(2),ICHA,ICHZ,$9010)
      IREC = (IBUF(1)-ICHA)*26 + IBUF(2)-ICHA+1
      WRITE(20,REC=IREC) VALUE
      GO TO 30
C
C.... DURATION
C
   50 IF( IBUF(2).NE.ICH2 ) GO TO 65
   60 READ(21,67,END=9000) IBUF,IVAL
   67 FORMAT(2A1,1X,I5)
      IBUF(1) = ISHFT(IBUF(1),-8)
      IBUF(2) = ISHFT(IBUF(2),-8)
      CALL IRANG(IBUF(1),ICHA,ICHZ,$25)
      IF( IBUF(2).NE.IBLNK ) GO TO 9010
      IREC = 677 + IBUF(1)-ICHA
      WRITE(20,REC=IREC) IVAL
      GO TO 60
C
C.... TS CODE
C
   65 IF( IBUF(2).NE.ICH3 ) GO TO 90
   70 READ(21,77,END=9000) IBUF,IVAL
   77 FORMAT(2A1,1X,I2)
      IBUF(1) = ISHFT(IBUF(1),-8)
      IBUF(2) = ISHFT(IBUF(2),-8)
      CALL IRANG(IBUF(1),ICHA,ICHZ,$25)
      CALL IRANG(IBUF(2),ICHA,ICHZ,$9010)
      IREC = 702 + (IBUF(1)-ICHA)*26 + IBUF(2)-ICHA+1
      WRITE(20,REC=IREC) IVAL
      GO TO 70
C
C.... EXTREMUM
C
   90 IF( IBUF(2).NE.ICH4 ) GO TO 110
  100 READ(21,77,END=9000) IBUF,IVAL
      IBUF(1) = ISHFT(IBUF(1),-8)
      IBUF(2) = ISHFT(IBUF(2),-8)
      CALL IRANG(IBUF(1),ICHA,ICHZ,$25)
      IF( IBUF(2).NE.IBLNK ) GO TO 9010
      IREC = 1379 + IBUF(1)-ICHA
      WRITE(20,REC=IREC) IVAL
      GO TO 100
C
C.... PROBABILITY
C
  110 IF( IBUF(2).NE.ICH5 ) GO TO 130
  120 READ(21,40,END=9000) IBUF,VALUE
      IBUF(1) = ISHFT(IBUF(1),-8)
      IBUF(2) = ISHFT(IBUF(2),-8)
      CALL IRANG(IBUF(1),ICHA,ICHZ,$125)
      IREC = 1405 + IBUF(1)-ICHA
      GO TO 126
  125 CALL IRANG(IBUF(1),ICH1,ICH9,$25)
      IREC = 1431 + IBUF(1)-ICH1
  126 IF( IBUF(2).NE.IBLNK ) GO TO 9010
      WRITE(20,REC=IREC) VALUE
      GO TO 120
C
C.... SEND CODES OR DEFAULT DURATION
C
  130 IF( IBUF(2).NE.ICH6 ) GO TO 9010
```

```
  140 READ(21,150,END=9000) IBUF,ISEND,NFLAG
  150 FORMAT(2A1,1X,7A1,1X,I1)
      IBUF(1) = ISHFT(IBUF(1),-8)
      IBUF(2) = ISHFT(IBUF(2),-8)
      ISEND(1) = ISHFT(ISEND(1),-8)
      ISEND(2) = ISHFT(ISEND(2),-8)
      ISEND(3) = ISHFT(ISEND(3),-8)
      ISEND(4) = ISHFT(ISEND(4),-8)
      ISEND(5) = ISHFT(ISEND(5),-8)
      ISEND(6) = ISHFT(ISEND(6),-8)
      ISEND(7) = ISHFT(ISEND(7),-8)
      CALL IRANG(IBUF(1),ICHA,ICHZ,$25)
      CALL IRANG(IBUF(2),ICHA,ICHZ,$9020)
      IF( NFLAG.EQ.1 ) ISEND(1) = -ISEND(1)
      IREC = 1439 + (IBUF(1)-ICHA)*26 + IBUF(2)-ICHA+1
C
C     NOW GET ITS GROUP # BY CHECKING HOW MANY SEND CODES ARE ON FILE
C
      READ(20,REC=IREC) IVAL
      IF( IVAL.LE.0 ) GO TO 155
      JREC = IVAL
      GO TO 175
  155 JREC = 2116
  160 READ(20,REC=JREC,END=170) IDUM
      JREC = JREC + 1
      GO TO 160
C
  170 JREC = JREC - 2115
      JREC = JREC/2 + 1
      WRITE(20,REC=IREC) JREC
  175 JREC = 2116 + (JREC-1)*2
      WRITE(20,REC=JREC) ISEND(1),ISEND(2),ISEND(3),ISEND(4)
      JREC = JREC + 1
      WRITE(20,REC=JREC) ISEND(5),ISEND(6),ISEND(7)
      GO TO 140
C
C..... THATS IT
C
 9000 STOP
C
C.... INPUT ERROR
C
 9010 WRITE(10,9011) IBUF,VALUE
 9011 FORMAT(5X,'ERROR',5X,2R1,5X,G20.7,'<015>')
      GO TO 9000
C
C.... INPUT ERROR - SEND CODES ETC
C
 9020 WRITE(10,9021) IBUF,ISEND
 9021 FORMAT(5X,'ERROR',5X,2R1,5X,7R1,'<015>')
C
      GO TO 9000
C
      END
```

```
C       LSTSETUP
C
C      LISTS THE SHEF PARAMETER FILE
C
C-------------------------------------------------------------------C
C   VERSION 1.0 JULY 1982    GEOFFREY M BONNIN  MBRFC               C
C-------------------------------------------------------------------C
C
      DOUBLE PRECISION VALUE
      DIMENSION ISEND(7)
      COMMON /CODES/ ICHA,ICHB,ICHC,ICHD,ICHE,ICHF,ICHG,ICHH,ICHI,
     1               ICHJ,ICHK,ICHL,ICHM,ICHN,ICHO,ICHP,ICHQ,ICHR,
     2               ICHS,ICHT,ICHU,ICHV,ICHW,ICHX,ICHY,ICHZ,ICH0,
     3               ICH1,ICH2,ICH3,ICH4,ICH5,ICH6,ICH7,ICH8,ICH9,
     4               IBLNK,ISLASH,ICOLON,IPLUS,IMINUS,IDOT,IARROW,
     5               ICOMMA
C
C
C.... SET UP FILES
C
      OPEN 20,'SHEFPARM',LEN=8
      OPEN 21,'$LPT.DU'
C
C.... PE CODES
C
      WRITE(21,10)
   10 FORMAT('<O15>',' *** PE CODES  -  CONVERSION FROM SI TO ENGLISH ***')
C
      DO 40 I=1,26
         JCH1 = ICHA+I-1
         DO 30 J=1,26
            JCH2 = ICHA+J-1
            IREC = (JCH1-ICHA)*26 + (JCH2-ICHA+1)
            READ(20,REC=IREC) VALUE
            WRITE(21,20) JCH1,JCH2,VALUE
   20       FORMAT(5X,2R1,5X,G22.15,5X,10('_') )
   30    CONTINUE
         WRITE(21,35)
   35    FORMAT(1X,1X)
   40 CONTINUE
C
C.... DURATION CODE
C
      WRITE(21,50)
   50 FORMAT('<O15>',' *** DURATION CODE ***' )
      DO 70 I=1,26
         JCH1 = ICHA+I-1
         IREC = 677 + JCH1-ICHA
         READ(20,REC=IREC) IVAL
         WRITE(21,60) JCH1,IVAL
   60    FORMAT(5X,R1,5X,I10,5X,10('_') )
   70 CONTINUE
C
C.... TS CODES
C
      WRITE(21,80)
   80 FORMAT('<O15>',' *** TS CODE ***')
C
      DO 120 I=1,26
         JCH1 = ICHA+I-1
         DO 100 J=1,26
            JCH2 = ICHA+J-1
            IREC = 702 + (JCH1-ICHA)*26 + (JCH2-ICHA+1)
```

```
            READ(20,REC=IREC) IVAL
            WRITE(21,90) JCH1,JCH2,IVAL
 90         FORMAT(5X,2R1,5X,I10,5X,10('_') )
100      CONTINUE
         WRITE(21,110)
110      FORMAT(1X,1X)
120   CONTINUE
C
C.... EXTREMUM CODE
C
      WRITE(21,230)
230   FORMAT('<015>',' *** EXTREMUM CODE ***' )
      DO 250 I=1,26
         JCH1 = ICHA+I-1
         IREC = 1379 + JCH1-ICHA
         READ(20,REC=IREC) IVAL
         WRITE(21,240) JCH1,IVAL
240      FORMAT(5X,R1,5X,I10,5X,10('_') )
250   CONTINUE
C
C.... PROBABILITY CODE
C
      WRITE(21,200)
200   FORMAT('<015>',' *** PROBABILITY CODE  ***' )
      DO 220 I=1,26
         JCH1 = ICHA+I-1
         IREC = 1405 + JCH1-ICHA
         READ(20,REC=IREC) VALUE
         WRITE(21,210) JCH1,VALUE
210      FORMAT(5X,R1,5X,G10.3,5X,10('_') )
220   CONTINUE
      DO 225 I=1,9
         JCH1 = ICH1+I-1
         IREC = 1430 + I
         READ(20,REC=IREC) VALUE
         WRITE(21,223) JCH1,VALUE
223      FORMAT(5X,R1,5X,G10.3,5X,10('_') )
225   CONTINUE
C
C.... SEND CODES OR DEFAULT DURATIONS
C
      WRITE(21,260)
260   FORMAT('<015>',' *** SEND CODES OR DURATION DEFAULTS ***')
C
      DO 350 I=1,26
         JCH1 = ICHA+I-1
         DO 330 J=1,26
            JCH2 = ICHA+J-1
            IREC = 1439 + (JCH1-ICHA)*26 + (JCH2-ICHA+1)
            READ(20,REC=IREC) IVAL
            IF( IVAL.GT.0 ) GO TO 280
            WRITE(21,270) JCH1,JCH2,IVAL
270         FORMAT(5X,2R1,5X,I5,5X,7('_') )
            GO TO 330
280         IREC = 2116 + (IVAL-1)*2
            READ(20,REC=IREC) ISEND(1),ISEND(2),ISEND(3),ISEND(4)
            IREC = IREC + 1
            READ(20,REC=IREC) ISEND(5),ISEND(6),ISEND(7)
            IF( ISEND(1).GE.0 ) GO TO 310
            ISEND(1) = -ISEND(1)
            WRITE(21,300) JCH1,JCH2,IVAL,ISEND
300         FORMAT(5X,2R1,5X,I5,5X,7R1,'  PREVIOUS 7AM')
            GO TO 330
```

```
C
  310        WRITE(21,320) JCH1,JCH2,IVAL,ISEND
  320        FORMAT(5X,2R1,5X,I5,5X,7R1)
  330     CONTINUE
          WRITE(21,340)
  340     FORMAT(1X,1X)
  350 CONTINUE
      END
```

APPENDIX F

INPUTPARM FILE

PE CODES

```
*1
AD 1.0
AF 1.0
AM 1.0
AT 1.0
AU 1.0
AW 1.0
EA 0.0393701
ED 0.0393701
EM 0.0393701
EP 0.0393701
ER 0.0393701
ET 0.0393701
EV 0.0393701
FA 1.0
FB 1.0
FC 1.0
FE 1.0
FK 1.0
FL 1.0
FP 1.0
FS 1.0
FT 1.0
FZ 1.0
GD 0.3937008
GR 1.0
GS 1.0
GT 0.3937008
HA 3.2808399
HB 3.2808399
HC 3.2808399
HD 3.2808399
HE 3.2808399
HF 3.2808399
HG 3.2808399
HH 3.2808399
HI 1.0
HJ 3.2808399
HK 3.2808399
HL 3.2808399
HM 3.2808399
HN 3.2808399
HP 3.2808399
HR 3.2808399
HS 3.2808399
HT 3.2808399
HW 3.2808399
HX 3.2808399
HY 3.2808399
HZ 3.2808399
IC 1.0
IE 0.6213712
IO 3.2808399
IR 1.0
IT 0.3937008
LA 247.10541
LC 0.8107131
LS 0.8107131
MI 1.0
ML 0.3937008
MM 1.0
MS 0.0393701
MT -1.0
```

```
MU 0.3937008
MW 1.0
NG 3.2808399
NN 1.0
PA 0.296134
PC 0.0393701
PP 0.0393701
PR 0.0393701
PT 1.0
PY 0.0393701
QA 0.0353147
QC 0.8107131
QD 0.0353147
QG 0.0353147
QI 0.0353147
QL 0.0353147
QM 0.0353147
QN 0.0353147
QP 0.0353147
QR 0.0353147
QS 0.0353147
QT 0.0353147
QU 0.0353147
QV 0.8107131
QX 0.0353147
QY 0.0353147
RA 1.0
RC 1.0
RI 1.0
RP 1.0
RT 1.0
SA 1.0
SD 0.3937008
SF 0.3937008
SL 0.00328084
SR 1.0
SS 1.0
SW 0.0393701
TA -1.0
TC -1.0
TD -1.0
TF -1.0
TH -1.0
TM -1.0
TN -1.0
TP -1.0
TS -1.0
TW -1.0
TX -1.0
UC 0.6213712
UD 1.0
UL 0.6213712
US 2.2369363
VB 1.0
VC 1.0
VE 1.0
VG 1.0
VH 1.0
VJ 1.0
VK 1.0
VL 1.0
VM 1.0
VP 1.0
```

```
VQ 1.0
VR 1.0
VS 1.0
VT 1.0
VU 1.0
VW 1.0
WA 1.0
WC 1.0
WG .0393701
WH 1.0
WL 1.0
WO 1.0
WP 1.0
WT 1.0
WV 3.2808399
XG 1.0
XL 1.0
XP 1.0
XR 1.0
XU 2.2883564
XV 0.6213712
XW 1.0
*2                                    DURATION CODES
A    1008
B    1002
C    0015
D    2001
H    1001
I    0000
J    0030
K    1012
L    1018
M    3001
P    5004
Q    1006
R    5002
S    5001
T    1003
U    0001
V    5003
W    2007
X    5005
Y    4001
Z    5000
*3                                    TS CODES
CA   1
CB   1
CC   1
CD   1
CE   1
CF   1
CG   1
CH   1
CI   1
CJ   1
CK   1
CL   1
CM   1
CN   1
CO   1
CP   1
CQ   1
CR   1
```

CS 1
CT 1
CU 1
CV 1
CW 1
CX 1
CY 1
CZ 1
FA 1
FB 1
FC 1
FD 1
FE 1
FM 1
FN 1
FP 1
FQ 1
FU 1
FV 1
FW 1
FX 1
FZ 1
HA 1
HB 1
HC 1
HD 1
HE 1
HF 1
HG 1
HH 1
HI 1
HJ 1
HK 1
HL 1
HM 1
HN 1
HO 1
HP 1
HQ 1
HR 1
HS 1
HT 1
HU 1
HV 1
HW 1
HX 1
HY 1
HZ 1
PA 1
PB 1
PC 1
PD 1
PE 1
PF 1
PG 1
PH 1
PI 1
PJ 1
PK 1
PL 1
PM 1
PN 1
PO 1

```
PP  1
PQ  1
PR  1
PS  1
PT  1
PU  1
PV  1
PW  1
PX  1
PY  1
PZ  1
RA  1
RB  1
RC  1
RD  1
RG  1
RM  1
RP  1
RR  1
RS  1
RT  1
RV  1
RW  1
RX  1
RZ  1
ZZ  1
*4                          EXTREMUM CODES
J   1
K   1
L   1
M   1
N   1
P   1
T   1
U   1
V   1
W   1
X   1
Y   1
Z   1
*5                          PROBABILITY CODES
A   0.002
B   0.004
C   0.01
D   0.02
E   0.04
F   0.05
G   0.25
H   0.75
J   0.0013
K   0.0228
L   0.1587
M   -0.5
N   0.8413
P   0.9772
Q   0.9987
T   0.95
U   0.96
V   0.98
W   0.99
X   0.996
Y   0.998
Z   -1.0
```

```
1  0.1
2  0.2
3  0.3
4  0.4
5  0.5
6  0.6
7  0.7
8  0.8
9  0.9
*6                          SEND CODES OR DURATION DEFAULTS
AD ADZZZZZ
AT ATD
AU AUD
AW AWD
EA EAD
EM EMD
EP EPD
ER ERD
ET ETD
EV EVD
HY HGIRZZZ 1
HN HGIRZNZ
HX HGIRZXZ
LC LCD
PP PPD
PR PRD
PY PPDRZZZ 1
QC QCD
QN QRIRZNZ
QV QVZ
QX QRIRZXZ
QY QRIRZZZ 1
RI RID
RP RPD
RT RTD
TC TCS
TF TFS
TH THS
TN TAIRZNZ
TX TAIRZXZ
UC UCD
UL ULD
XG XGJ
XP XPQ
**
```

CENTRAL REGION TECH MEMOS

(continued from front inside cover)

NWS CR 48 Manual of Great Lakes Ice Forecasting. C. Robert Snider December 1971 (COM-72-10143)

NWS CR 49 A Preliminary Transport Wind and Mixing Height Climatology, St. Louis, Missouri. Donald E. Wuerch, Albert J. Courtois, Carl Ewald, Gary Ernst. June 1972 (COM-72-10859)

NWS CR 50 An Objective Forecast Technique for Colorado Downslope Winds Wayne E. Sangster. December 1972 (COM-73-10280)

NWS CR 51 Effect on Temperature and Precipitation of Observation Site Change at Columbia, Missouri. Warren M. Wisner. March 1973 (COM-73-10734)

NWS CR 52 Cold Air Funnels. Jack R. Cooley and Marshall E. Soderberg September 1973. (COM-73-11753)

NWS CR 53 The Frequency of Potentially Unfavorable Temperature Conditions in St. Louis, Missouri. Warren M. Wisner. October 1973

NWS CR 54 Objective Probabilities of Severe Thunderstorms Using Predictors from FOUS and Observed Surface Data. Clarence A. David. May 1974 (COM-74-11258)

NWS CR 55 Detecting and Predicting Severe Thunderstorms Using Radar and Sferics. John V. Graff and Duane C. O'Malley. June 1974 (COM-74-11335)

NWS CR 56 The Prediction of Daily Drying Rates. Jerry D. Hill. Nov. 1974. (COM-74-11806)

NWS CR 57 Summer Radar Echo Distribution Around Limon, Colorado. Thomas D. Karr and Ronald L. Wooten. Nov. 1974. (COM-75-10076)

NWS CR 58 Guidelines for Flash Flood and Small Tributary Flood Prediction. Lawrence A. Hughes and Lawrence L. Longsdorf. October 1975 (PR247569/AS) NWS CR 58rev (revised) March 1978 (PB281461/AS)

NWS CR 59 Hourly Cumulative Totals of Rainfall - Black Hills Flash Flood June 9-10, 1972. Don K. Halligan and Lawrence L. Longsdorf (PB256087)

NWS CR 60 Meteorological Effects on the Drift of Chemical Sprays. J. D. Hill. July 1976. (PB259593)

NWS CR 61 An Updated Objective Forecast Technique for Colorado Downslope Winds. Wayne E. Sangster. March 1977. (PB266966)

NWS CR 62 Design Weather Conditions for Prescribed Burning. Ronald E. Haug. April 1977. (PB268034)

NWS CR 63 A Program of Chart Analysis (with some diagnostic and forecast implications). Lawrence A. Hughes. March 1978 (PB279866)

NWS CR 64 Warm Season Nocturnal Quantitative Precipitation Forecasting for Eastern Kansas Using the Surface Geostrophic Wind Chart. Wayne E. Sangster. April 1979 (PB295982)

NWS CR 65 The Utilization of Long Term Temperature Data in the Description of Forecast Temperatures. Nov. 1981. Arno Perlow, WSO CBI

NWS CR 66 The Effect of Diurnal Heating on the Movement of Cold Fronts Through Eastern Colorado. August 1982. James L. Wiesmueller, WSFO DEN

NOAA SCIENTIFIC AND TECHNICAL PUBLICATIONS

The National Oceanic and Atmospheric Administration was established as part of the Department of Commerce on October 3, 1970. The mission responsibilities of NOAA are to assess the socioeconomic impact of natural and technological changes in the environment and to monitor and predict the state of the solid Earth, the oceans and their living resources, the atmosphere, and the space environment of the Earth.

The major components of NOAA regularly produce various types of scientific and technical information in the following kinds of publications:

PROFESSIONAL PAPERS — Important definitive research results, major techniques, and special investigations.

CONTRACT AND GRANT REPORTS — Reports prepared by contractors or grantees under NOAA sponsorship.

ATLAS — Presentation of analyzed data generally in the form of maps showing distribution of rainfall, chemical and physical conditions of oceans and atmosphere, distribution of fishes and marine mammals, ionospheric conditions, etc.

TECHNICAL SERVICE PUBLICATIONS — Reports containing data, observations, instructions, etc. A partial listing includes data serials; prediction and outlook periodicals; technical manuals, training papers, planning reports, and information serials; and miscellaneous technical publications.

TECHNICAL REPORTS — Journal quality with extensive details, mathematical developments, or data listings.

TECHNICAL MEMORANDUMS — Reports of preliminary, partial, or negative research or technology results, interim instructions, and the like.

Information on availability of NOAA publications can be obtained from:

ENVIRONMENTAL SCIENCE INFORMATION CENTER
ENVIRONMENTAL DATA AND INFORMATION SERVICE
NATIONAL OCEANIC AND ATMOSPHERIC ADMINISTRATION
U.S. DEPARTMENT OF COMMERCE

Rockville, MD 20852