TIME-DEPENDENT DATA SYSTEM (TDDS)—An Interactive Program to Assemble, Manage, and Appraise Input Data and Numerical Output of Flow/Transport Simulation Models

By R. Steven Regan, Raymond W. Schaffranek, and Robert A. Baltzer

---

U.S. GEOLOGICAL SURVEY
Water-Resources Investigations Report 96-4143

Reston, Virginia
1996

U.S. DEPARTMENT OF THE INTERIOR

BRUCE BABBITT, Secretary

U.S. GEOLOGICAL SURVEY

Gordon P. Eaton, Director

For additional information write to:

U.S. Geological Survey
Chief, Hydrologic Analysis Support Section
437 National Center
Reston, VA 20192
Electronic Mail: h2osoft@usgs.gov

Copies of this report can be purchased from:

U.S. Geological Survey
Branch of Information Services
Box 25286
Denver, CO 80225-0286

PREFACE

This report describes a data-management system, identified as the Time-Dependent Data System (TDDS), developed by the U.S. Geological Survey (USGS) for filing, managing, and appraising time sequences of discrete values of geophysical parameters. The TDDS documented in this report is version 5.2, dated 1996/06/03. Inquiries and specific questions about the TDDS should be directed to:

U.S. Geological Survey
Hydrologic Analysis Software Support Team
437 National Center
Reston, VA 20192
Electronic mail: h2osoft@usgs.gov

Copies of the TDDS program and documentation can be obtained, for a nominal processing and handling fee, upon request to the above address. The program and documentation are also accessible on the World Wide Web (WWW) using Mosaic or similar WWW reader, from the USGS Water Resources Information home page (http://h2o.usgs.gov/), and by anonymous FTP from h2o.usgs.gov (130.11.50.175) in the directory /pub/software/general/tdds.

The TDDS is written in Fortran 77 with graphics capabilities provided by a CalComp graphics library. A library of software routines is provided that translates the CalComp references to Graphical Kernel System (GKS) references. Thus, in general, the TDDS is easily adapted to computer systems having a Fortran compiler and either a CalComp or GKS graphics library.

The TDDS has been thoroughly tested, widely used, and implemented on a variety of computer systems (primarily UNIX- and Microsoft DOS-based systems). However, it is possible that other applications or installations on computer systems for which it has not been verified could reveal erroneous results or implementation problems. If such problems occur, please notify the USGS at the address above.

CONTENTS

FIGURES

TABLES

TIME-DEPENDENT DATA SYSTEM (TDDS)—An Interactive Program to Assemble, Manage, and Appraise Input Data and Numerical Output of Flow/Transport Simulation Models

By R. Steven Regan, Raymond W. Schaffranek, *and* Robert A. Baltzer

ABSTRACT

A system of functional utilities and computer routines, collectively identified as the Time-Dependent Data System (TDDS), has been developed and documented by the U.S. Geological Survey. The TDDS is designed for processing time sequences of discrete, fixed-interval, time-varying geophysical data—in particular, hydrologic data. Such data include various, dependent variables and related parameters typically needed as input for execution of one-, two-, and three-dimensional hydrodynamic/transport and associated water-quality simulation models. Such data can also include time sequences of results generated by numerical simulation models. Specifically, TDDS provides the functional capabilities to process, store, retrieve, and compile data in a Time-Dependent Data Base (TDDB) in response to interactive user commands or preprogrammed directives. Thus, the TDDS, in conjunction with a companion TDDB, provides a ready means for processing, preparation, and assembly of time sequences of data for input to models; collection, categorization, and storage of simulation results from models; and intercomparison of field data and simulation results.

The TDDS can be used to edit and verify prototype, time-dependent data to affirm that selected sequences of data are accurate, contiguous, and appropriate for numerical simulation modeling. It can be used to prepare time-varying data in a variety of formats, such as tabular lists, sequential files, arrays, graphical displays, as well as line-printer plots of single or multiparameter data sets. The TDDB is organized and maintained as a direct-access data base by the TDDS, thus providing simple, yet efficient, data management and access. A single, easily used, program interface that provides all access to and from a particular TDDB is available for use directly within models, other user-provided programs, and other data systems. This interface, together with each major functional utility of the TDDS, is described and documented in this report.

INTRODUCTION

Actual simulation processes, when considered in their most general terms, are similar—one with another—regardless of the particular model used. At a moment in time, boundary-value data, together with predefined or previously computed values of the dependent variables throughout the spatial extent of the model, enable values of unknowns to be computed (simulated) at an advanced moment in time throughout the space domain. To be uniquely defined in both the space and time domains, dynamic mathematical/numerical simulation models must be provided with appropriate sequences of time-varying data at each of the geographical extremities delimiting the areal extent of the waterbody to be modeled. These data, representing unique values of dependent variables at the model (and prototype) boundaries and referred to as boundary-value data, are used to exercise the model and, thereby, to accomplish a particular numerical simulation. With appropriate sequences of boundary values available, the simulation process can be repeated at successive, fixed-time increments, as desired.

Clearly, the success of any numerical simulation depends in large measure on the quality of boundary-value data used to exercise the model. Therefore, collection of time sequences of data for input to simulation models requires careful consideration and planning. Generally, it requires establishment of a network of data-collection sites geographically located throughout the region to be modeled. Locations of data-collection sites depend largely on the following factors:

- ❑ hydrodynamic characteristics of the prototype system,
- ❑ time-dependent data requirements of the simulation model,
- ❑ types and availability of suitable instrumentation, and
- ❑ economical and practical aspects of conducting the field-data acquisition.

At each site, data are collected in some manner as a time sequence of discrete values referenced to a common time, to a standard spatial reference, and perhaps to a value datum, as may be appropriate. For example, water-surface elevations (stages) at a site having known latitude and longitude are commonly recorded at a fixed-time interval with respect to local standard time and referenced either to National Geodetic Vertical Datum of 1929 (NGVD of 1929) or to North American Vertical Datum of 1988 (NAVD88). Accurate time synchronization between the data-collection instrumentation at all sites where boundary-value data are recorded is very important when treating dynamically varying, time-dependent phenomena. Use of even slightly out-of-phase data can cause erroneous simulation results and (or) misleading interpretation of the dynamic conditions being investigated. In other words, the accuracy of simulation results can be no better than the accuracy of the boundary-value data used to exercise the model.

Implementation of a numerical simulation model also typically requires time-dependent data at geographically interior locations within the waterbody for purposes of model calibration and verification. Other types of time-varying data might be needed as well to exercise a particular numerical simulation. Accurate evaluation and time synchronization of these data are equally important. Numerical models also produce numerous time sequences of the simulated results. In terms of the number of data parameters and extent

of the data-collection effort required for the project, the amount of data collected and simulated results generated can be voluminous. An efficient system that can be directly linked to the model is essential to the storage and management of these data. Moreover, if a single system can manage the field-collected data as well as the simulated results, then measured-to-measured, measured-to-simulated, and simulated-to-simulated comparisons of data can be made readily and insightfully. To facilitate implementation of a comprehensive simulation model, a computerized system is needed to process, compile, systematically organize, and manage this often voluminous array of time-dependent data. The system of functional utilities and routines, collectively identified as the Time-Dependent Data System (TDDS) and documented in this report, provides these facilities.

This report describes the TDDS, provides instructions for using the software, and also serves as a programmer's guide to incorporate access to the Time-Dependent Data Base (TDDB) in user-application programs. Examples illustrating data input and use of the TDDS also are provided. An overview of the TDDS outlines the software and describes the TDDB file structure. A description of each TDDS utility, including input and output examples, is provided. The report also presents the data-base structure and detailed programming instructions for use of the Direct-Access Data Input and Output (DADIO) routines that control access to the data base. Appendix A contains a listing of definitions of terminology used throughout the report. The prompting sequence for establishing a TDDB and storing two time sequences of data are presented in Appendix B. Appendixes C and D are installation guides for use of TDDS on UNIX-based and DOS-based computer systems, respectively.

SOFTWARE DESCRIPTION

Background

Initially, the TDDS was developed solely to process large quantities of water-surface elevation data as recorded by Analog-Digital Recorders (ADR) and to organize and make these data readily available for input to simulation models. Field-recorded sets of such data occasionally contain errors or have missing values that require correction or evaluation before use in a model. Therefore, additional capabilities were incorporated in the TDDS to perform the following functions:

- ❑ aid in the detection and identification of errors,
- ❑ provide the means to correct or insert individual and sequences of data values,
- ❑ visually display data for quick and easy analysis and verification, and
- ❑ organize data in a form that makes them readily, selectively, and efficiently usable.

Subsequently, the TDDS was extended to accommodate the processing of simulation results, such as time sequence of water-surface elevations, cross-sectional discharges, conveyance widths, and cross-sectional areas (Schaffranek and Baltzer, 1978; Lai and others, 1978; Schaffranek and others, 1981). With today's increasing complexity of simulation models, such as the addition of water-quality components, the TDDS has been structured to accommodate any user-defined time sequence of discrete data. The system now supports the processing of 100 predefined hydrologic, meteorologic, and water-quality parameters and, moreover, can be easily expanded to include other geophysical parameters as needed.

Design

The TDDS is designed to aid in the analysis and verification of various types of time-dependent data required as input to mathematical/numerical simulation models. It is also intended to facilitate storage, retrieval, and analysis of various types of model-generated output.

The primary attributes incorporated in the TDDS are the following:

- ❑ Flexibility and adaptability:
  - Prepares data in a variety of formats (tabular lists, sequential files, and digital and line-printer plots), including user-specified formats
  - Stores and formats data for use with USGS application programs and models, such as the National Water Data Storage and Retrieval System (WATSTORE) (Hutchison, 1975; Hutchison and others, 1977); the branch-network, unsteady-flow model (BRANCH) (Schaffranek and others, 1981); and the two-dimensional Surface-Water Integrated Flow and Transport model (SWIFT2D) (Leendertse, 1987)
  - Possesses a simple program interface to allow user-application programs to directly store data into, or retrieve data from, a TDDB

- ❑ Ease of use:
  - Provides for interactive execution using menu-driven, conversational prompts
  - Employs self-explanatory prompts to reduce the need to reference written documentation
  - Provides meaningful default values for all prompts (the default response to a yes/no query is always yes)
  - Retains user responses for use as defaults for subsequent prompts of a similar type
  - Employs a logical, data-entry sequence
  - Provides for manual coding of input control files according to fully documented specifications to allow users to bypass interactive prompting
  - Retains generated input control files for subsequent reuse
  - Captures and retains user responses in a file for subsequent recall so that a repetitive task can be duplicated or easily automated
  - Generates output file names based on user-supplied input file names
  - Affords safe and easy abort of interactive sessions; in response to any prompt, the letter "Q" returns control to the TDDS main menu, whereas the letter "X" returns control to the operating system
- ❑ Portability (The TDDS has been used on a wide variety of computers, including DOS-based personal computers and UNIX-based file servers and workstations. Executable versions are available for use on IBM-compatible personal computers, Data General AViiON workstations, and Sun SPARCstations.):
  - Written in a standards-based programming language—Fortran 77
  - Low-level dependency for graphics generation—graphics written based on the CalComp graphics library. (Note that several Fortran compilers, such as the Lahey or Microsoft compliers, support a subset of the CalComp graphics library.) A library is available with the TDDS that translates the CalComp library references to Graphical Kernel System (GKS) references. Thus, the TDDS can be adapted to computer systems having either a CalComp or GKS graphics library.
  - Possesses export capability to create a text (ASCII) version of a TDDB for transfer between computer systems
- ❑ Data type adaptive:
  - Supports a wide range of data parameters—currently 100 predefined data-parameter codes, such as those for water-surface elevation, discharge, wind speed and direction, temperature, conductivity, and salinity
  - Accommodates user revision and definition of data-parameter codes
  - Provides multiple machine (numeric) formats for values in the data base to allow for different storage and data-precision requirements—four storage types: two- and four-byte integer and four- and eight-byte floating-point values

- ❑ Ease-of-management and built-in security:
  - Controls access to a TDDB—access available only through the TDDS, or the four DADIO routines of the program interface, thereby alleviating address inconsistences and storage incompatibilities
  - Provides filter mechanism for accessing data sets stored in a TDDB—the Data-Station Reference (DSR) file defines those data sets (designated by station identifier) that are permitted to be accessed during an execution of the TDDS or user-application program that incorporates the program interface
  - Performs value verification—appropriateness of all interactive user responses are range checked
  - Displays the contents of a TDDB visually—the extent and status of data in a TDDB can be requested at any time
  - Provides backup capability—a partial or total backup copy of a TDDB can be readily created or restored
  - Provides archival capability in text (ASCII) format

Overview

The TDDS consists of eight utility functions and four primary support routines. The principal capabilities of the TDDS are as follows:

- ❑ Process, categorize, store, retrieve, and maintain data in an indexed, direct-access data base—TDDB.
- ❑ Compile, store, retrieve, and maintain station description information comprising a DSR file.
- ❑ Prepare time sequences of data from a TDDB as tabular lists, digital and line-printer plots, and sequential files in USGS user-application program and model-input formats.

Utilities and Routines

There are two processing utilities (DAFILE and DAFIX) to read, edit, store, list, and delete data. Two formatting utilities (DAGET and DAPLOT) are used to retrieve, display, and file data. Four support utilities (ALLOCATE, SUMMARY, BACKUP, and DAOPEN) are used to create and maintain the TDDB and DSR files.

Four Fortran storage and retrieval routines (DADIO, DADI, DADO, and DADSUM) provide the communication link between the TDDB and the TDDS, as well as simulation models or other user-application programs. No other means of direct access to the TDDB is provided because data are maintained in the TDDB in a format unique to these routines. Access to the TDDB is additionally monitored and controlled by information contained in the DSR file. This file specifies, among other attributes, the name used to identify a data-collection site. The DADIO routines and DSR file, when embedded in models, model preprocessors, and (or) other application programs, allow for the secured retrieval of data from, or storage of data into, the TDDB.

Files

The TDDS reads and (or) writes to four files—TDDB, DSR, master, and control files. Additional output files can be created, such as files containing interactive responses, program results, formatted data, and digital graphics. Figure 1 illustrates the links between the TDDS, with embedded DADIO routines, and the associated input and output files. The arrows in figure 1 indicate the direction of data flow. The DSR, control, data, print, log, and master files are text files that can be viewed, created, and modified using a standard text editor. The DAFILE utility allows files of time sequences of values to be stored in a TDDB. Graphics are produced by the DAPLOT utility and text files of time sequences of data are retrieved by the DAGET utility. (Refer to Utilities and Routines section for description of the TDDS utilities.)

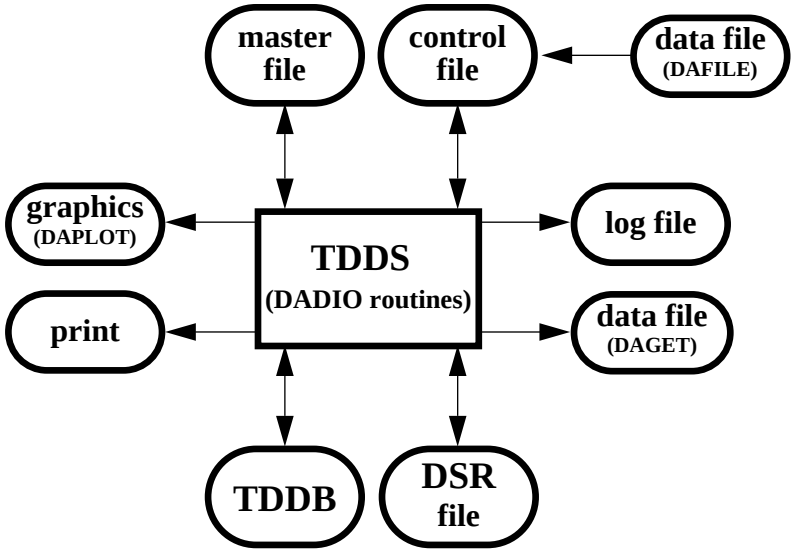

Figure 1. Relationship of files within the TDDS.

TDDB

The TDDB is an indexed, direct-access file written in a format unique to the DADIO routines. Data values are stored as machine code (binary format) to retain precision and to avoid unnecessary alpha-numeric conversions. The TDDB format promotes efficient access to the data for input to, and (or) output from, simulation models and other application programs. The only means of access to data in the TDDB is through incorporation of the DADIO system routines into an application program, such as has been done in the TDDS. Any attempt to examine a TDDB in any other fashion, such as with a text editor, is meaningless and may destroy its integrity. (See the TDDB Description section for more details.)

DSR File

The DSR file contains a list of identifiers used to distinguish data sets in a TDDB. Only data sets assigned an identifier, which is included in the DSR file, can be stored in, or retrieved from, a TDDB. Generally, these identifiers correspond to the geographic name

of data-collection or simulation sites. The DSR file also contains optional reference adjustments to be applied to the data, which are frequently desirable for water-surface elevation data. The purpose of the ALLOCATE utility is to interactively generate and (or) maintain the DSR file.

Master File

The "master" file, default file name of **TDDB_DSR.MTR**, identifies by name the TDDB and DSR files used during the most recent TDDS execution. A master file is automatically generated if one does not initially exist. The purpose of the DAOPEN utility is to maintain the master file.

Control File

A "control" file contains input data, coded in a specific format, defining the task to be performed by a particular TDDS utility. All utilities, except DAOPEN and ALLOCATE, require a control file. Control files, which can preexist or be created interactively, can be previewed before execution of any task. The default name for a control file is the utility name plus the suffix **.rdr**, such as **daget.rdr**. Because control files are text files, they can be created or modified using a text editor provided that they are coded in the format specific for each utility. Use of a text editor to modify an existing control file can oftentimes be quicker than responding to the sequence of prompts needed to create it. This is especially true when only a minor change is needed in a control file; however, care must be exercised to ensure the coding requirements are met. (See the Input subsection of the Utility Descriptions section for a complete description of the control file format of each TDDS utility.)

Output Files

The TDDS generates output file names based on the name of the particular utility's control file. A base name is determined as the control file name minus any, up to three letters, suffix. Each output file name is set to this base name plus a suffix: **.prt** for printer output, **.plt** for plotter output, and **.out** for other filed output, such as text files of time sequences of data. For example, the default control file name for the DAGET utility is **daget.rdr**. Thus, in the default case, the print file is named **daget.prt** and the file of retrieved data is named **daget.out**.

Output file names are generated based on the full pathname specified for the control file. For example, specifying the control file name as **/home/user/daget.rdr** results in a print file with the pathname **/home/user/daget.prt**. Note that due to this naming convention, any existing file having the identical name as the generated output file will be overwritten. Also note that TDDS limits the length of file names to 48 characters.

Each TDDS utility, except DAOPEN, writes messages summarizing the results of its execution to a "print" file. Records (representing lines of printed output) in these files are up to 132 characters in length. Each line contains a Fortran vertical spacing code (in the first column) used to format the results for increased readability. For output devices that recognize and accept them, these codes function as follows: blank = next line (single space), 0 = skip a line (double space), 1 = begin a new page, and + = no line feed (overwrite current line).

The TDDS captures user responses to prompts in a text file named TDDSLOG.DAT. This file can be used to re-execute a particular TDDS session or modified using a text editor to perform a different task(s).

Organization

Figure 2 illustrates the communication links between time-dependent data sources, simulation models and application programs, and the TDDS and TDDB. It also portrays how the TDDS manages this information flow. Notice that the DADIO routines monitor and control all direct interaction with a TDDB. Embedding these routines in a program permits data to be stored into, or retrieved from, a TDDB. Indirect access to data in a TDDB is, of course, possible by using the TDDS to retrieve data from a TDDB (through the embedded DADIO routines) and writing the data in a user-specified format to an intermediate file. This intermediate file can then be used as input to a model or user-application program.

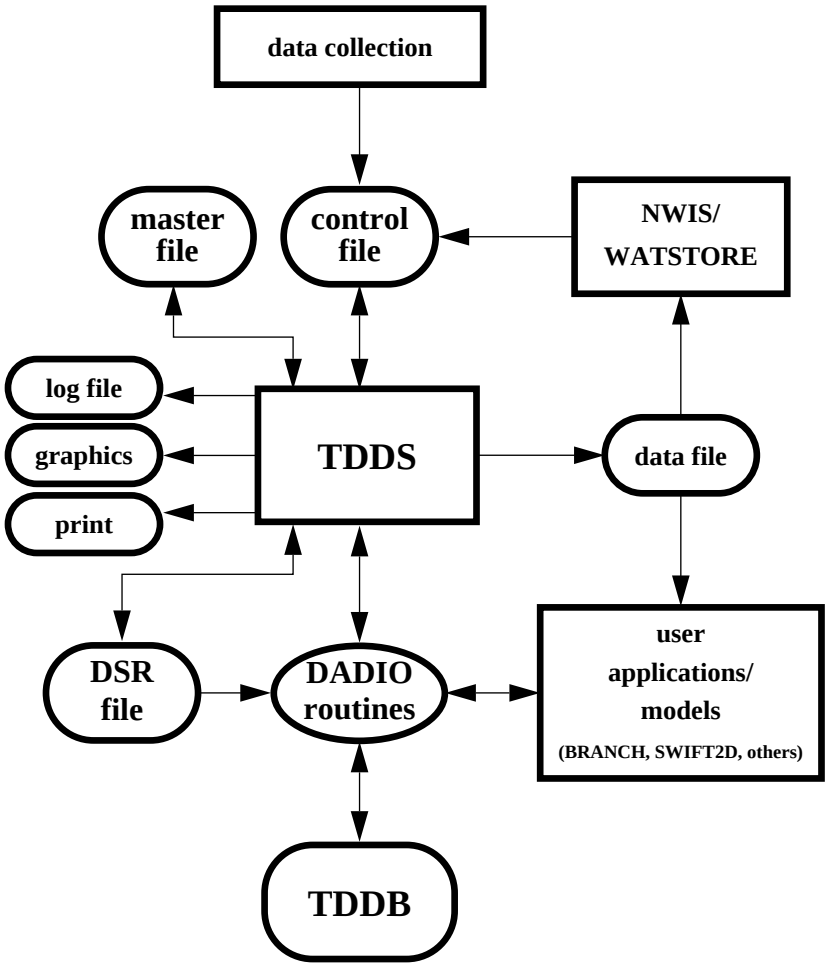

Figure 2. TDDS communication links.

TDDB Description

A TDDB is an indexed, direct-access, unformatted file consisting of a sequence of fixed-length records. Two factors giving the TDDS its high efficiency are use of unformatted value storage and direct-access input and output (I/O) operations. Storage and retrieval of data without format control, that is, in binary format, eliminates time-consuming format conversions and the loss of precision due to truncation and round-off of values inherent in these conversions. Direct-access read and write operations reduce data access time by allowing a program to read and write records randomly within a file without having to process intervening records. Although no prior knowledge of the TDDB internal file structure is required to use the TDDS, its design permits users to tailor some of its attributes to suit particular project requirements and computer-hardware configurations.

The organization of a TDDB is similar in concept to that of a directory on a computer disk. Typically, a disk directory contains an index with a predefined number of entries (nodes) where the physical location (address) and attributes of each file contained in the directory are maintained. Similarly, a TDDB contains an index with a predefined number of entries where the address and attributes of each data set stored in the TDDB are maintained. The number of index records depends on a user-configurable attribute, MAXENT, that defines the maximum number of index entries, that is, data sets, (default=476) that can be retained in the index. The TDDB index resides in the first few records of a TDDB. The remainder of the TDDB consists of data records containing time sequences of data. Data records are appended to the TDDB as required for data storage. Thus, the size of a TDDB, in terms of storage (disk) space utilized, increases as more data are filed in it. When a data set is added to the TDDB, an index entry is used to retain its address and attributes and the data values are stored in the last physical record and (or) in a new record(s) appended to the TDDB depending on the number of values in the time sequence.

The length of each physical record of a TDDB, both index and data records, depends on a user-configurable attribute, LENREC (default=6232 bytes[1]). Thus, in the default case, each TDDB I/O operation results in the storage or retrieval of 6232 bytes of information. The record length is variable to permit specification of an optimal value based on the characteristics of a particular storage device (such as a magnetic disk) and the size of a typical storage/retrieval request. A record length of sufficient size should be chosen to promote efficient I/O operations as these operations are more time consuming than arithmetic operations.

Data are stored in and retrieved from a TDDB one record at a time until the entire data request is satisfied. Retrieval of data is not constrained to a data set as defined upon storage into the TDDB, that is, a data-retrieval request can span multiple data sets. A retrieval of data that spans multiple data sets can only be requested for data of a common parameter code, storage-type code, station identifier, and recording frequency (values per day). The request will only be fulfilled if the individually stored data sets have no missing values for the specified time span.

[1] The value of 6232 was selected to optimize I/O performance on several computer disks typically in use at the time of the initial development of the TDDS. It was also determined to be an optimal value for storage of data recorded at 15-minute intervals (96 values per day) in sets that typically spanned 30 days. A more appropriate value for disks in use today might be a multiple of 512 bytes, such as 4096. Optimizing the record length in such a way can conserve computer time required to read or write data records.

Index Records

Each index record contains a predefined number of entries, with each entry retaining the address and attributes of a single data set. The number of entries in each index record depends on the index entry length (52 bytes) and the size of a TDDB record (default=6232 bytes). Thus, in the default case, 119 index entries (6232/52) can reside in a single TDDB index record. The number of index records allocated for a TDDB is fixed by the maximum number of index entries assigned (default=476) and the number of index entries per record. Thus, in the default case, four records (476/119) are reserved for the TDDB index.

The address of a data set within the TDDB consists of the record number and byte number within that record at which the first value of the data set is stored. In addition to the data-set address, each index entry contains information that uniquely identifies the following attributes:

- ❑ Station identifier
- ❑ Beginning and ending date and time
- ❑ Number of values per day
- ❑ Data-parameter code
- ❑ Storage-type code
- ❑ Data status
- ❑ Processing date and time

Data Records

Index records are immediately followed by records containing the time sequences of data values.[1] The maximum length of a data set (90 days) and the maximum number of index entries (default=476) establishes the maximum amount of data that can be stored in a TDDB. Thus, for the default case, 42,840 days of data (90*476) can be stored in a TDDB, slightly more than 117 years of data. Note, however, that this calculation is based on optimum storage of maximum length, that is, 90-day data sets, whereas in reality, data sets of shorter length often need to be stored.

A data set can, and frequently does, span multiple data records. In the default case, all data sets larger than 6232 bytes will span at least two data records. For example, a data set 90 days in length with a time interval of 15 minutes (96 values per day) and stored as four-byte floating point values requires 34,560 bytes (90*96*4) of storage space. (See the Storage-Type Codes section for more details on storage types.) Thus, the data set will span at least six consecutive records (90*96*4/6232). The TDDS appends new data to a TDDB beginning at the initial byte following the end of the last data set stored; thus, multiple short data sets can be stored in a single data record.

[1] In the default case, the first data set in the TDDB is stored beginning at byte one of record five. The second data set could be stored beginning at the first available byte within record five or in other subsequent records depending on the number of values in the first data set.

Storage Requirements

Total storage space requirements for a TDDB vary with each application. The amount of required storage space depends on the number of data-parameter types stored for each data location, anticipated time span of a data set, the data-recording frequency, and the amount of storage allotted for each data value. The TDDS supports four storage types: (a) two-byte integer; (b) four-byte integer; (c) four-byte floating point; and (d) eight-byte floating point. (See the Storage-Type Codes section for more details on storage types.) An approximate number of bytes required for a TDDB (based on the TDDS default attributes—record length=6232 bytes and number of index records=4) can be determined from the following equation:

$$B = (Y * 365) * I * (2*S_2 + 4*S_4 + 8*S_8) + 4 * 6232$$

where

B = number of bytes required,

Y = number of years of record (default maximum=25),

I = average number of readings per day for all stations,

S_2 = number of stations of 2-byte data (I2),

S_4 = number of stations of 4-byte data (I4 or R4), and

S_8 = number of stations of 8-byte data (R8).

The four additional records shown in the equation represent the number of records used by the TDDB index, which depends on the number of index entries, record length, and length of an index entry (52 bytes). Thus, a typical project application of 3 years' duration with default TDDB attributes and all data stored as four-byte, 15-minute values (96 values per day) at five locations, requires 342 records—338 data records (3*365*96*5*4/6232) plus 4 index records. Therefore, at the end of the project, the TDDB would require approximately 2 megabytes (342*6232/1,048,576) of storage (disk) space.

Attribute Assignments

Those attributes identifying limits imposed on data to be stored and, thus, defining the variable storage capacity of a TDDB are presented in table 1. The attributes under user control are (a) maximum number of index entries—MAXENT (default=476); (b) the length of each record—LENREC (default= 6232 bytes); (c) maximum number of years of record —MAXYRS (default=25); and (d) maximum number of station identifiers permitted in the DSR file—MAXFLD (default=150). If the default values for these attributes are insufficient for an application, they can be modified easily. To change the assigned values, edit the Fortran include file named **dimpar.cmn**, then recompile and reload the TDDS. The values for MAXFLD and MAXYRS can be changed at any time without recompilation as these do not affect the TDDB record structure.

Table 1. Attributes and associated values for a default TDDB

Attribute	Value	Description
LENIND	52	Length of an index entry in bytes.
LENREC	[1] 6232	Length of a TDDB record in bytes.
MAXDYS	90	Maximum number of days allowed per data set.
MAXENT	[1] 476	Maximum number of entries (data sets).
MAXYRS	[1] 25	Maximum number of years of record in a TDDB.
MAXFLD	[1] 150	Maximum number of station identifiers.

[1] User may change these values.

A TDDB generated using an earlier TDDS version, compiled with a record length and (or) number of index records[1] that are different from the current TDDS version, is **not** usable by the current TDDS version without conversion. To convert a TDDB created by an earlier TDDS version using a incompatible set of **dimpar.cmn** constants, it is necessary to create a backup copy of the old TDDB using the BACKUP utility from the earlier TDDS version; and then restore the backup file to a new TDDB using the BACKUP utility from the current TDDS version.

Data-Parameter Codes

The TDDS distinguishes different types of data using a two-character data-parameter code. Currently, the TDDS predefines 100 data-parameter codes. These codes are identified and defined in table 2. The definitions given in table 2 are used to label plots and tables of retrieved data. Explanatory notes are implied definitions and characteristics of certain parameters that pertain to specific models (that is, BRANCH and SWIFT2D). User revision of these parameter definitions is accommodated and is likely in other user-application programs. If needed, the Fortran source file named **dadini.f** and Fortran include file **datyps.cmn** can be edited to change data-parameter codes, modify definitions, or to add new codes and definitions. (Note that these two files are part of the libutl software distribution.) If the **bldtyp.f** and **datyps.cmn** files are modified, it is necessary to recompile and reload the TDDS to activate the data-parameter code changes.

The current version of the TDDS does not differentiate between systems of measure, such as International System (SI) and U.S. Customary (inch/pound) units. **It is the user's responsibility to use consistent units in storing data in a TDDB**. For example, a digital plot of discharge data produced by the DAPLOT utility will yield a y-axis labeled "CROSS-SECTIONAL DISCHARGE (10**4)" not "CROSS-SECTIONAL DISCHARGE (10**4), IN CUBIC FEET/SECOND".

[1] Note that the number of index records depends on the MAXENT, LENREC, and LENIND (length of an index record) as described above.

Table 2. Predefined time-dependent data-parameter codes

Code	Definition	Notes
Hydrologic Parameters		
A	CROSS-SECTIONAL AREA	Total area of cross section beneath water surface.
AC	CONVEYANCE AREA	Flow-conveying area of cross section.
AS	STORAGE AREA	Nonflow-conveying area of cross section.
B	CROSS-SECTIONAL WIDTH	Total width of cross section at water surface.
BC	CONVEYANCE WIDTH	Top width of flow-conveying cross section.
BS	STORAGE WIDTH	Top width of non-flow-conveying cross section.
Q	CROSS-SECTIONAL DISCHARGE	Positive in the downstream direction.
QL	LATERAL DISCHARGE	In terms of length cubed/time/length.
QT	TRIBUTARY DISCHARGE	Inflow positive.
V	CROSS-SECTIONAL VELOCITY	Positive in downstream direction.
MU	MEAN VELOCITY-U	Velocity component, positive in North direction.
MV	MEAN VELOCITY-V	Velocity component, positive in East direction.
MW	MEAN VELOCITY-W	Velocity component, positive vertically upward.
PU	VELOCITY AT A POINT-U	Velocity component, positive in North direction.
PV	VELOCITY AT A POINT-V	Velocity component, positive in East direction.
PW	POINT VELOCITY-W	Velocity component, positive vertically upward.
UL	LATERAL FLOW VELOCITY	In terms of length/time.
UX	TRANSVERSE VELOCITY	Velocity component in “x” direction, as defined in the model.
VY	LONGITUDINAL VELOCITY	Velocity component in “y” direction, as defined in the model.
WZ	VERTICAL VELOCITY	Velocity component in “z” direction, as defined in the model.
WP	WETTED PERIMETER	Length of intersection of cross section and channel, measured left edge of water to right edge of water.
Z	WATER-SURFACE ELEVATION	Measured positively above reference elevation.
Meteorologic Parameters		
P	PRECIPITATION	In equivalent liquid (that is, melted) depth.
AP	ATMOSPHERIC PRESSURE	
AR	ATMOSPHERIC RADIATION	
R0 to R9	RADIATION	Ten alternate definitions.
SR	SOLAR RADIATION	
AT	AIR TEMPERATURE	
RH	RELATIVE HUMIDITY	
GU	MAX. WIND-GUST VELOCITY-U	Velocity component, positive in North direction.
GV	MAX. WIND-GUST VELOCITY-V	Velocity component, positive in East direction.
WD	WIND DIRECTION	Measured clockwise from true North.
WS	WIND SPEED	
WU	WIND VELOCITY-U	Velocity component, positive in North direction.
WV	WIND VELOCITY-V	Velocity component, positive in East direction.
Water-Quality Parameters		
C	CONDUCTIVITY	Adjusted to standard conditions.
D	DENSITY	
DO	DISSOLVED OXYGEN	
PH	pH	
S	SALINITY	
T	WATER TEMPERATURE	
C0 to C9	DISSOLVED CONCENTRATION	Ten alternate definitions.
K0 to K9	SUSPENDED CONCENTRATION	Ten alternate definitions.
L0 to L9	LATERAL CONCENTRATION	Ten alternate definitions.
M0 to M9	MEASURED CONCENTRATION	Ten alternate definitions.
T0 to T9	TRIBUTARY CONCENTRATION	Ten alternate definitions.

Storage-Type Codes

The TDDS stores data as any one of four storage types (two-byte integer—I2, four-byte integer—I4, four-byte floating point—R4, or eight-byte floating point—R8). Versions of the TDDS prior to January 1, 1995, restricted data-parameter codes to a specific storage type. These early versions stored water-surface elevation (Z), precipitation (P), pH (PH), salinity (S), water temperature (T), v-velocity component (VY), and u-velocity component (UX) as two-byte integer values. All other data types were stored as four-byte integer values.

The current TDDS utilities assume integer values have been multiplied by 100 to retain two decimal places of accuracy prior to storage in the TDDB. For example, the integer value 1012 as stored in a TDDB is interpreted as the floating-point value 10.12 by the TDDS utilities. Floating-point values are interpreted as stored; that is, no multiplier is assumed.

Two-byte, integer water-surface elevation data, as predefined by the data-parameter code ‘ Z’ and storage-type code ‘I2’, are restricted to the range of -999 to 9999 by the TDDS. The reasons for this limited range are to ensure each value is at most four digits and to allow for individual data values to be flagged. (See the Data Flags section for more details.) Users must employ a reference adjustment to keep data values within this integer range. For each station identifier, a reference adjustment can be specified in the DSR file.

Other two-byte, integer values range from -32,768 to 32,767. It is necessary to store data types whose values typically exceed these ranges, or need greater accuracy, using one of the other storage types. Four-byte, integer values range from -2,147,483,648 to 2,147,483,647. Representation of floating-point numbers is computer system dependent.[1] Table 3 summarizes the data storage-type codes.

Table 3. Storage-type codes

Code	Range[1]	Multiplier	Description
I2	-32,768 to 32,767 (-999 to 9999 for water-surface elevation data)	100	Two-byte integer values.
I4	-2,147,483,648 to 2,147,483,647	100	Four-byte integer values.
R4	$\pm 3.4 \times 10^{38}$	1	Four-byte floating-point values.
R8	$\pm 1.7 \times 10^{308}$	1	Eight-byte floating-point values.

[1]The value range for each storage type is computer system dependent.

[1] For information on floating-point implementation, refer to ANSI/IEEE Standard 754-1985. For a Data General AViiON workstation, the range for four-byte floating point numbers is approximately 3.4×10^{38} with the smallest difference possible between any two values being approximately 1.2×10^{-7}. The range for eight-byte floating point numbers is approximately 1.7×10^{308} with the smallest difference between two values being approximately 2.2×10^{-16}.

Data Flags

The TDDS allows for data flags to be assigned to two-byte, integer water-surface elevation values. These data flags are assigned as a numeric offset from the data value. Four data flags (11000, 22000, -11000, and -22000) are recognized. Table 4 defines these data flags. For the offsets to be detected, the two-byte, integer water-surface elevation data values are restricted to the range -999 to 9999. (This range also limits two-byte, integer water-surface elevation values to four digits.) The only data flag assigned by the TDDS is the “update flag” (22000). Data values stored (to fill a gap in the record) or updated (to modify errant value(s)) in the TDDB by the DAFIX utility are offset by 22000 to flag them as not part of an original data set. The other data flags (11000, -11000, and -22000) must be assigned to the data values and stored in the TDDB by a user-application program.

When data values are retrieved from a TDDB by any TDDS utility or user-application program, data sets that contain at least one data value with the 11000, -11000, or -22000 offset are indicated by the DADIO return code equal to 4. A DADIO return code equal to 10 indicates at least one data value includes the 22000 offset. If at least one data value is detected to be offset by 22000 and at least one data value is offset by one of the other data flags, the DADIO return code equals 17. The TDDB index listings indicate those data sets (index summary) and those days (calendar-year summary) contain flagged data. The DAGET utility removes the data flags from retrieved data values prior to processing. For print requests using the DAFIX utility, the four-significant figures format places a code next to flagged data values and the other format prints the data values without removing the data flags. The DAPLOT utility can (1) plot all data sets after removing the data flags; (2) plot only data sets that do not contain valued offset by the data flags; or (3) plot data sets after removing the update data flag (22000) and not plotting data sets that contain data values flagged with the 11000, -11000, or -22000 offsets.

Table 4. Data flags

Flags	Description
11000	Rate of change exceeded.
22000	Updated value, assigned by DAFIX.
-11000	Repeat count exceeded.
-22000	Value is out of range.

HOW TO USE THE TDDS

The following sections describe how to use the TDDS in an interactive computer environment, such as a workstation running the UNIX operating system or a personal computer running a DOS operating system. (The TDDS easily adapts to function in a batch mode, but this capability is not discussed herein. For an example of how to use the TDDS in batch mode, refer to the scripts, included in the software distribution package, that are used to run verification tests of a TDDS installation.) The following discussion assumes the reader has some knowledge of UNIX or DOS terminology and, in particular, the directory-file structure used with such operating systems. References to directory pathnames are given in the UNIX style. To convert pathnames to DOS terminology, substitute a \ in place of each /. For example, the UNIX path specification of **tdds/src** corresponds to **tdds\src** under the DOS operating system. The discussion does not address the input options of TDDS utilities in detail but describes the general use of the TDDS. (See the Utility Descriptions section for the input specifications for all TDDS utilities.)

The TDDS Directory Structure

Figure 3 illustrates the TDDS directory structure resulting from installation of the software via the distribution package. **It is recommended that users not store data files in this directory structure.** The directory **tdds_5.0/bin** contains the TDDS executable. Various documentation files are found in **tdds_5.0/doc** (includes PostScript version of this document and text version of the UNIX man page). Source code and Makefile for compiling and loading the TDDS are found in **tdds_5.0/src**. The directory **tdds_5.0/test** contains scripts to run the verification tests to determine if the TDDS is properly installed. Input data and expected results for each test are in **tdds_5.0/data**.

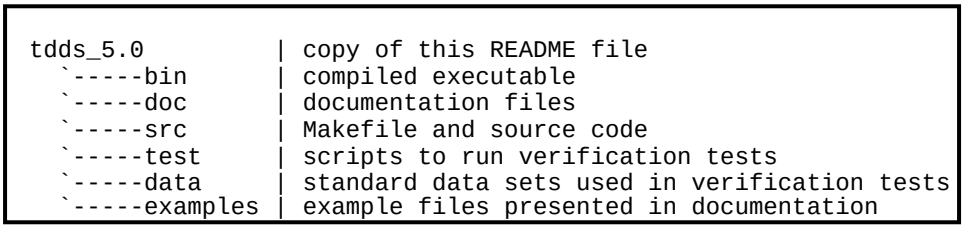

Figure 3. Directory structure of the TDDS.

The TDDS is compiled and loaded using the MAKE program (versions of MAKE are available for UNIX- and DOS-based computer systems). The Makefile (input instructions for the MAKE program) contains operating system specific information and may need customizing for use on your computer system. Appendixes C and D give complete instructions for compiling and loading the TDDS on UNIX- and DOS-based computer systems.

Running the TDDS

The TDDS is an interactive, menu-driven, question and answer program that prompts for all information required to perform each task. Normally, execution is initiated by entering the command "**tdds**", the name of the TDDS executable. (Note: On DOS-based computers, the executable is named tdds.exe.) To allow execution of the TDDS from any directory on the computer system, include the directory **tdds_5.0/bin** in the operating-system path variable or copy the TDDS executable to a directory included in the path variable. If the tdds executable is not included in the path, the full pathname, such as **/usr/opt/wrdapp/tdds/bin/tdds**, must be entered to initiate execution.

Use of the TDDS typically begins soon after the start of a project. The initial TDDS tasks can be done even before starting data collection. These tasks involve selecting names for the TDDB and DSR files, creating a DSR file that contains a station identifier for each data-collection location, and initializing the TDDB. No data can be stored in a TDDB until these initial tasks are completed. If any other TDDS task is attempted, the TDDS interactively forces completion of the initial tasks. To simplify the designation of the name of the TDDB and DSR file to use, the TDDS retains the names of the TDDB and DSR file last used in the current directory in the "master" file. See Appendix B for an example TDDS session to complete the initial tasks and to store several data sets in a TDDB. The general execution sequence of the TDDS follows:

1. **Select master file.** First the TDDS displays a welcome message showing its version and instructions for answering prompts. Then a prompt requests the name of the master file. Enter a valid file name or accept the displayed default file name.

   If the specified master file does not exist, or is incomplete, the DAOPEN utility executes to allow specification of the names of the TDDB and DSR file. Enter valid file names or accept the displayed default file names.

   If the specified DSR file did not exist or is empty, the ALLOCATE utility executes to interactively build it.

2. **Select TDDS utility.** Next the TDDS MAIN MENU displays with a prompt requesting the code of the utility to execute. Enter an integer value in the range from one to eight.

3. **Select control file.** Then a prompt requests the name of the control file, except for the DAOPEN and ALLOCATE utilities as they do not require control files. Enter a valid file name or accept the displayed default file name.

4. **Build control file.** If the control file does not exist, prompts display to create it. If the control file exists (either created previously during a TDDS session or manually prepared), or after the control file is created, a prompt requests to begin execution of the selected task, list the contents of the control file, or create the control file. Once the control file is complete, enter a carriage return (or the number 0 and a carriage return) to the prompt to begin the TDDS task.

5. **Review results.** Output files are generated and (or) graphics are displayed. After completion of the task, the print file can be displayed, 22 lines at a time, before returning to the TDDS MAIN MENU.

The following shows the prompting sequence for these steps when the master, DSR, and control files all exist:

```
Welcome to the Time-Dependent Data System - Version: 5.2 1996/06/03

To accept the displayed or default value given in a prompt, press
enter.  The default response to a “yes/no” query is yes.  In
response to TDDS prompts, enter “x” to exit the TDDS or “q” to
return to the main menu.

ENTER THE FILE NAME OF THE MASTER FILE
(OLD) TDDB_DSR.MTR:
<cr>

                          TDDS MAIN MENU
=========================================================================
| Code   Utility                  Function                               |
| ----   -------                  --------                               |
|  1     DAOPEN     Assign Time-Dependent Data Base (TDDB) and            |
|                     Data-Station Reference (DSR) file names             |
|  2     ALLOCATE   Initialize a TDDB or Create or Edit DSR file          |
|  3     DAFILE     Store data in a TDDB                                  |
|  4     DAGET      Retrieve data and output in WATSTORE daily- or        |
|                     unit-values, BRANCH instream-input, SWIFT_IDP       |
|                     input, or user-specified format                     |
|  5     DAPLOT     Plot data on digital or line-printer devices          |
|  6     DAFIX      Modify, delete, and print data                        |
|  7     BACKUP     Create backup file of TDDB or re-create a TDDB        |
|                     from a backup file                                  |
|  8     SUMMARY    Print directory summaries of a TDDB                   |
=========================================================================

ENTER INTEGER VALUE FOR SELECTION CODE (1-8)
CURRENT VALUE =      1:
8<cr>
A control file is required to define the TDDS task.

ENTER THE FILE NAME OF THE CONTROL FILE
(OLD) summary.rdr:
<cr>

Printed output can be found in summary.prt

                  ** EXECUTE MENU **

   Code               Activity
     0  --  BEGIN execution (default)
     1  --  LIST the contents of the control file
     2  --  CREATE the control file
Enter 0, 1, or 2:
<cr>

Please wait while your request is processed.


****   SUMMARY operation completed ****

DO YOU WANT TO LIST THE OUTPUT PRINT FILE [Y,n]?
<cr>

ENTER INTEGER VALUE FOR DISPLAY WIDTH (UP TO 132 CHARACTERS)
CURRENT VALUE =     80:
<cr>
```

Answering TDDS Prompts

Each TDDS prompt indicates what type of value to enter and offers a default value. To accept the default value, simply respond with a carriage return. The default value for many prompts is updated to the value last entered to this or similar prompts during a single TDDS session. The default response to a yes/no query is always **yes**. Responses entered as **Y**, **y**, **YES**, or **yes** are interpreted as **yes**; similarly, responses entered as **N**, **n**, **NO**, or **no** are interpreted as **no**. A response of **Q**, **q**, **QUIT**, or **quit** to a prompt returns control back to the TDDS MAIN MENU. A response of **X**, **x**, **EXIT**, or **exit** to a prompt ends the TDDS session and control returns to the computer operating system. Note that a quit response at the TDDS MAIN MENU prompt is treated as an exit response. If the "break" key is entered, program execution terminates and control returns to the computer operating system. **Do not use the break key when executing a utility that modifies a TDDB (DAFILE, DAFIX, BACKUP, or ALLOCATE) as this can corrupt the data base.** Note that when updating a control file, its contents are initially removed and then generated as prompts are responded to. Thus, if creation of a control file is aborted (quit, exit, or break) before being completed, the control file will generally be incomplete and unusable without additional editing.

Prompts requesting character data, where the uppercase and lowercase letters have the same meaning, can be entered in either case. For example, the following prompt requests character data: ENTER VALUE FOR SELECTION CODE (T, C, D, A, F, OR ?); a response of **t** or **T** has the same effect. When prompted for a floating point number and the response is to be a whole number, the decimal point need not be specified.

UTILITY DESCRIPTIONS

This section describes the TDDS utilities. Although coding formats for the input to each TDDS utility are presented, users can generate this input by responding to the menu-driven, conversational prompts. A table identifying the input specifications and examples showing typical input and output for each utility are included. All examples in this document refer to data sets contained in a single TDDB. Appendix B shows how the first two data sets are stored in the TDDB. Additional data sets were added to this TDDB using the DAFILE utility and the BRANCH model before using in the example. One site used for the examples is referred to as station identifier 1655480. This data-collection site deploys an ADR recording the water-surface elevation of the Potomac River at Indian Head, Md., every 15 minutes. These data, in conjunction with discharge data recorded every 15 minutes at an upstream site (station identifier 1646500), are used as boundary-value data for input to the BRANCH model. This model is used to compute water-surface elevation, discharge, conveyance width, and area at selected cross sections on the Potomac River. Note that the station identifiers in the example DSR file (see fig. 5) beginning with the string “Computed” are data sets computed by the BRANCH model. Station identifiers beginning with the string “Observed” are data sets recorded during prototype data-collection efforts, such as discharge measurements.

DAOPEN—Assigning the TDDB and DSR File Names

The TDDS requires assignment of the TDDB and DSR file names before any other TDDS utility can be executed. These file names are maintained in a “master” file. The default name of the master file is **TDDB_DSR.MTR**. Use the DAOPEN utility to create, update, or display master files. The initial TDDS prompt requests the name of the master file. If the specified master file does not exist, or is incomplete, DAOPEN executes to interactively create it. Also, the master file can be updated using a text editor if the format of the file is maintained. Figure 4 shows an example of a master file. Table 5 gives the format of the master file.

```
Time-Dependent Data Base    : potomac.tdd
Data-Station Reference file : potomac.dsr
```

Figure 4. Example of TDDB_DSR.MTR file.

Table 5. Specifications for the master file

Variable	Default	Position	Format	Description
First record -- TDDB file name record				
LABEL1	-----	2-30	A29	Annotation for the record (not required).
TDDB	TDDB	32-79	A48	File name of the TDDB.
Second record -- DSR file name record				
LABEL2	-----	2-30	A29	Annotation for the record (not required).
DSRFIL	DSRFILE	32-79	A48	File name of the DSR file.

The following describes the execution sequence for DAOPEN. The initial TDDS prompt requests the name of the master file. If the specified file does not exist, or is incomplete, DAOPEN executes to interactively create it. If it is complete, select DAOPEN (code 1) from the TDDS MAIN MENU. The software then displays the contents of the master file (or default names if the master file did not exist) in the following menu:

```
                    ** DAOPEN MENU **

The file names assigned in the master file are:

  Time-Dependent Data Base    : TDDB
  Data-Station Reference file : DSRFILE

   Code               Activity
     0  --  ACCEPT file names as displayed (default)
     1  --  CHANGE the file name for the TDDB
     2  --  CHANGE the file name for the DSR file
Enter 0, 1, or 2:
```

This menu repeatedly displays until the file names are accepted. A prompt will display if a specified TDDB or DSR file does not exist. Once valid file names are entered, the files are opened. If the DSR file does not exist, or is empty, the ALLOCATE utility executes to interactively create it. Once the DSR file is created, control returns to the TDDS MAIN MENU.

ALLOCATE—Maintain the DSR File and Initialize a TDDB

Use the ALLOCATE utility to create or update the DSR file and to initialize a TDDB. The following describes the execution sequence for ALLOCATE. Select ALLOCATE (code 2) from the TDDS MAIN MENU. Then the following menu displays:

```
                    ** ALLOCATE MENU **

   Code               Activity
     0  --  CREATE or UPDATE a DSR file (default)
     1  --  INITIALIZE a TDDB
Enter 0 or 1:
```

After selecting either activity option, the DAOPEN MENU displays to allow verification and (or) update of the TDDB and DSR file names. After accepting the file names, either the DSR FILE UPDATE MENU prompt is displayed or the TDDB is initialized and results of the execution of ALLOCATE are displayed. (ALLOCATE does not use a control file. It does produce the file **allocate.prt** that contains the results of the execution.) The following describes the ALLOCATE tasks and output.

Create or Update a DSR File

The DSR file must be created before any data can be stored in a TDDB because a data set can be stored only if the DSR file contains its station identifier. However, the DSR file can be modified often over the course of a project to add and remove stations as required. The DSR file can be used as a filter to restrict access to a subset of data sets in a TDDB; for example, when a partial backup of the TDDB is needed. Each record of the DSR file identifies a location where data exist or will be available, and optionally, a reference adjustment for data. Station identifiers are specified as a string of up to 16 alphanumeric characters. Reference adjustments apply to water-surface elevation data only and are added to the data values upon retrieval from a TDDB.

Commonly, the station identifier denotes the geographic location of the data, such as the latitude and longitude, site name, or USGS field-station number. A recommended form for identifiers is the seven-digit latitude (dddmmss) and seven-digit longitude (dddmmss) of the station location followed by a two-digit sequence number, where ddd = degrees, mm=minutes, and xx=seconds of latitude or longitude.

The following conventions apply to station identifiers:

- ❑ case of letters is significant
- ❑ leading zeros are treated as blank characters
- ❑ trailing blank characters are removed

To enforce these conventions, the TDDS moves the first nonblank character to position 16 and fills the string from the left with blanks. For example, the identifier " 002 test " equals the identifier "2 test" but does not equal "2 Test". These conventions are necessary to remove the ambiguity on the significance of leading zeros and blanks when comparing identifiers. They also simplify input of station identifiers as leading blanks or zeros need not be specified.

The DSR file can be created, updated, and displayed using ALLOCATE or by using a text editor. The ALLOCATE prompt used to edit the DSR file is the following:

```
                    ** DSR FILE UPDATE MENU **
Prompts are issued to update the values stored in the DSR file.
These values define the valid station identifiers and reference
adjustments for data sets stored in the TDDB.  Use the following
options to update the DSR file.

   Code                Activity
     S  --  SHOW a table of the DSR file contents.
     C  --  CHANGE the values of a table entry, enter its number
             (as shown in the table) and then the new values.
     D  --  DELETE an entry from the DSR file.
     A  --  ADD a new entry to the DSR file.
     F  --  FINISHED (saves changes).
     ?  --  DISPLAY this message.

ENTER VALUE FOR SELECTION CODE (T, C, D, A, F, OR ?)
CURRENT VALUE = "A":
```

After the DSR file is updated, its contents are written to the file **allocate.prt** and displayed prior to returning to the TDDS MAIN MENU. If a text editor is used, the DSR file must be maintained in the format shown in table 6.

Table 6. Specifications for the DSR file

Variable	Default	Position	Format	Description
FLDSTA	blanks	2-17	A16	Station identifier associated with the data location.
DATUM	0.0	26-33	F8.2	Reference adjustment for type Z data only.

Initialize a TDDB

The TDDB must be initialized before data can be stored in it. Use ALLOCATE to create a TDDB and initialize its index. The size of a TDDB generated is equal to the record length times the number of index records. Thus, in the default case, a new TDDB (or initialized existing TDDB) requires 24,928 bytes (6232*4). The size of a TDDB then increases, in terms of storage (disk) space, as data are filed in it.

Output

Output from ALLOCATE consists of a print file named **allocate.prt** and a DSR file or an initialized TDDB. The file **allocate.prt** contains information describing the results of the execution, such as the contents of the DSR file, the TDDB attributes, or any error conditions encountered. Figure 5 shows an example DSR file. This DSR file shows several types of station identifiers including eight-digit USGS field-station numbers, location names, latitude/longitude numbers, and a combination of descriptive text and eight-digit USGS station numbers.

```
         1655480            0.00
         1646500            0.00
   Wilson Bridge          -10.00
 384940077014901            0.00
 384320077020100            0.00
 383627077104602            0.00
 382151077083601            0.00
         1647600            0.00
         1652100            0.00
         1652588            0.00
         1658710            0.00
         1660800            0.00
         1661475            0.00
         1661590            0.00
Computed 1646500          -10.00
Computed 1652588          -10.00
Computed 1655480          -10.00
Observed 1660800            0.00
        11447500            0.00
```

Figure 5. Example DSR file.

Figure 6 shows the TDDB attributes as output to the file **allocate.prt**. These attributes define the structure of the TDDB and identify limits imposed on data to be stored. See the Attribute Assignments section above for descriptions of these attributes and how they can be changed.

```
A TDDB was created or initialized by DADIO - VERSION: 5.6 1996/06/03
with the following attributes as defined in include file dimpar.cmn:

MAXENT =   476 Maximum number of index entries
MAXFLD =   150 Maximum number of station in DSR file
MAXDYS =    90 Maximum number of days per data set
MAXYRS =    25 Maximum number of years per TDDB
LENREC =  6232 Length of each record in the TDDB, in bytes
MAXREC =  6232 TDDB record length based on Fortran I/O
LENIND =    52 Length of each index entry, in bytes

*** TDDB INITIALIZATION COMPLETED ***
```

Figure 6. Example ALLOCATE output—print file showing initialization of a TDDB.

DAFILE—Storing Data in a TDDB

Use the DAFILE utility to store time sequences of data in a TDDB. As input, DAFILE expects a text file of data values or a set of data values to be entered interactively. It accepts data in a variety of formats including predefined, free, and user-defined formats. Free format means data values are arranged sequentially in a file with a comma and (or) a space(s), tab(s), or new line between them. User-defined formats are specified as any Fortran format specification of up to 48 characters, such as (8I10), (8F10.2), or (8D10.4). Therefore, most all data files having consistently formatted records can be read by DAFILE and the data stored in a TDDB without additional user modification.

DAFILE should be used to store only previously verified data. However, unedited data can be stored and then examined graphically for verification using the DAPLOT utility. DAFILE can process time sequences of data edited and verified by other data systems. For example, DAFILE reads data in the WATSTORE unit-values format (Hutchison and others, 1977). Commonly, this format is referred to as the type-2 and type-B record format. The WATSTORE data-parameter types that TDDS recognizes are discharge, water-surface elevation, dissolved oxygen, water temperature, wind speed and direction, conductivity, and salinity. WATSTORE data sets must be complete time sequence of values (no gaps), as DAFILE does not account for WATSTORE missing value indicators. (Note that DAFILE interprets a blank value as a zero value.) Obtain WATSTORE formatted data using the NWIS (National Water Information System) program RETR_UV (Dempster, 1990) or the WATSTORE program UVRETR.

Data stored in a TDDB can be retrieved using the DAGET utility and then stored in another TDDB using DAFILE. For example, to transfer selected data from one TDDB to another, use (1) DAGET to retrieve the data and produce sequential file(s), (2) DAOPEN to change the file name of the current TDDB, and (3) DAFILE to store the data using the DAGET-created sequential file(s) as input. To transfer all data from one TDDB to another, use the BACKUP utility.

Execution Sequence

The following describes the execution sequence for DAFILE. Select DAFILE (code 3) from the TDDS MAIN MENU. At the prompt for the name of the control file, enter a valid file name or accept the displayed default file name (**dafile.rdr**). If the control file exists, the EXECUTE MENU displays with the following options: "BEGIN execution (default)", "LIST the contents of the control file", and "CREATE the control file". If the control file does not exist, or CREATE the control file is selected, prompts request whether TDDB summaries are to be produced before and after execution of DAFILE. Then, the following menu displays:

```
                    ** DAFILE MENU **

 Code            Data Set Characteristics
   1  --  Data values to be entered interactively
   2  --  File of free-formatted values
   3  --  File of 8 values/record with 1 value every 10 characters
   4  --  File created in a user-specified format
   5  --  File created previously by the DAGET utility
   6  --  WATSTORE unit-values file (Z,Q,DO,T,WS,WD,C,S data only)
   7  --  Finished creating control file (default)

ENTER INTEGER VALUE FOR SELECTION CODE (1-7)
CURRENT VALUE =      7:
```

The prompting order for each option follows. Note that data-set attributes refer to the station identifier, data-parameter code, storage-type code, beginning and ending time, number of readings per day, and reference adjustment (for water-surface elevation data only).

Code = 1—Values entered interactively

1. Enter data-set attributes.
2. Are data coded as integer, real, or double precision values?
3. Enter each value.

Code = 2—Free-formatted values in a file

1. Enter data-set attributes.
2. Are data coded as integer, real, or double-precision values?
3. Do any records precede data records in data file?
4. Enter name of the data file.

Code = 3—File of eight values per record with one value every 10 characters

1. Enter data-set attributes.
2. Enter name of the data file.

Code = 4—File of data in a user-specified format

1. Enter data-set attributes.
2. Enter Fortran format specification of the records in the data file.
3. Do any records precede data records in data file?
4. Enter name of the data file.

The prompt for the Fortran format specification is the following:

```
SELECT a data format using the code numbers from the table below
or ENTER a valid Fortran format specification (including
parentheses).  Interpret the formats below using the following
examples: (10I6) means 10 integer values per record with each
value right justified every 6 record positions (columns);
(10F8.2) means 10 decimal values per record with a value
specified every 8 columns (on input, if the decimal point is not
specified for a value, it is assumed to be right justified with
two implied decimal places, that is, “    12345 “ is read as
1234.50); (I6,4X,I6) means two integer values per record, the
first right justified in column 6 and the second in column 16;
(38X,6F7.2) means 6 floating-point values per record specified
every 7 columns beginning at column 39 (positions 1-38 are
ignored); (5D12.5) means 5 double precision values specified
every 12 columns.

PRESS ENTER TO CONTINUE:
<cr>

 Code     Format
 ----     ------
   0  --  (I5) (default)
   1  --  (10I6)
   2  --  (8I10)
   3  --  (10F8.2)
   4  --  (F10.2)
   5  --  (38X,6F7.2)
   6  --  (5D12.5)
   7  --  (4D20.5)

ENTER VALUE FOR A SELECTION CODE OR FORTRAN FORMAT SPECIFICATION
CURRENT VALUE = “(8F10.2)                                      “:
```

Code = 5—File created previously by the DAGET utility

1. Enter name of data file (the **.out** file produced by DAGET).

Code = 6—File of data in WATSTORE unit-values format

1. Are data values coded as integer values times 100 or decimal values?
2. Enter storage type; how are data stored in the TDDB (integer, real, double precision).
3. Enter name of data file.

Prior to DAFILE execution, the time sequence of values are stored in the DAFILE control file, either as read from a data file for options 2 through 6 or as read from the keyboard for option 1. After completion of a DAFILE storage option, the DAFILE menu displays to prompt for additional storage requests. Note that the DAFILE menu appears differently after the initial storage request. If the WATSTORE unit-values option is requested first, then it is the only option available for subsequent requests. If options 1 through 5 are requested first, then the WATSTORE option is not available for subsequent requests. After the control file is created, the EXECUTE MENU displays. Select the BEGIN option to execute DAFILE. After DAFILE finishes, a prompt requests whether to list the print file before returning to the TDDS MAIN MENU.

Input

The DAFILE input control file specifies (1) whether or not TDDB summaries are to be produced; (2) the attributes of the data sets to be stored; and (3) the time sequence of data values to be stored. The control file consists of five record types. The first record, List Index record, controls the production of optional summary listings of the TDDB index before and (or) after execution of DAFILE. The second through fifth record-types are input as a group; each group defines one data set to store. As many groups of these records as desired can be input in a single execution. Table 7 gives the format of the DAFILE control-file records.

There are two restrictions on input of groups. First, if WATSTORE formatted data values are input, then only data sets formatted in WATSTORE format can be input during a single execution; that is, data in a user-defined format cannot also be input. The second restriction applies to time sequence of values that exceed 90 days of record. These time sequences are broken up and stored as multiple data sets of 90 or less days of record by DAFILE. This restriction is necessary because 90 days is the maximum size of a TDDB data set.

Data values can be input to DAFILE from sequential files or the keyboard as either integer, four-byte floating-point (real), or eight-byte floating-point (double-precision) values. The values can subsequently be stored as either integer, real, or double precision. Thus, integer values can be read and then stored as double-precision values. Note that when storing four-byte floating-point values, usually seven significant digits can be retained (a computer system not a TDDS restriction). If more digits are specified, the value will be rounded or truncated to the nearest value that can be represented as a four-byte floating-point value. Computer systems generally retain 14 significant digits for double-precision values. When values are stored as integers (two- or four-byte), they are multiplied by 100 and then rounded to the nearest whole number before being stored in the TDDB. This is true whether the values are input to DAFILE as integer, real, or double-precision values.

Data formatted for input to DAFILE must be carefully prepared as no value checks of the data are performed. If data are coded in the DAFILE default format (I4,T71,5I2), each Data record consists of an integer value coded in columns 1-4 and the date and time (YRMODYHRMN) corresponding to the value coded in columns 71 to 80 as a sequence of two-digit integers (for example, 9412312400). If ICKTIM equals 0 or 2, the date and time associated with each value is read (after or before the data value, respectively) in addition to the data value. The date and time must increase by exactly one recording interval on consecutive records. The first and last date and time must be equal to the beginning and ending date and time on the preceding Data-Definition record. If these conditions are not met, the entire data-storage request is rejected. However, DAFILE will continue to process any valid subsequent requests.

Output

The primary output of the DAFILE execution are data stored in a TDDB. The print file consists of optional TDDB summaries and a summary, including station number, data-parameter code, begin and end time, readings per day, and number of data values for each data set successfully stored in the TDDB. If errors are encountered during an execution, they are identified in the print file. The default name of the print file is **dafile.prt**.

Table 7. Specifications for the DAFILE control file

Variable	Default	Values	Position	Format	Description
List Index record (one required per execution)					
LISTB	1	-1, 0, 1, 2	38-39	I2	Option to list TDDB summaries before task(s) (-1 = list both calendar-year and index summary; 0 = no summaries; 1 = list only index summary; 2 = list valid parameter and storage-type codes with definitions and readings per day).
LISTA	1	-1, 0, 1, 2	46-47	I2	Option to list TDDB summaries after task(s) (-1 = list both calendar-year and index summary; 0 = no summaries; 1 = list only index summary; 2 = list valid parameters and storage-type codes with definitions and readings per day).
WATFMT	0	0, 2	66	A1	Flag to indicate if WATSTORE data values are to be read as in F7.0 or F7.2 format.
WATS	blank	WATSTORE	68-75	A8	Flag to indicate data set is in WATSTORE unit-values format (leave blank if not in WATSTORE format).
STYPE	I2	I2, I4, R4, R8	79-80	A2	Storage-type code (used for WATSTORE data only).
Data-Definition record (one per data-set group)					
STAID	blank	-------	2-17	A16	Station identifier.
BTIME	0	0<N<100	25-38	5(I2,1X)	Beginning date and time for TYPE=FROM, or exact date and time for TYPE=DATE request (YR/MO/DY HR:MN).
ETIME	0	0<N<100	45-58	5(I2,1X)	Ending date and time of edit request for TYPE=FROM (YR/MO/DY HR:MN).
RDPDY	96	look to right for values	62-65	I4	Number of readings per day. Valid values are 1, 8, 12, 24, 48, 96, 144, 240, 288, 720, or 1440.
DTYPE	Z	parameter code	70-71	A2	Data-parameter code (for example, Z = water-surface elevation and Q = cross-sectional discharge).
STYPE	I2	I2, I4, R4, R8	76-77	A2	Storage-type code (I2 = two-byte integer, I4 = four-byte integer, R4 = four-byte floating point, R8 = eight-byte floating point).
Data-Format Specification record (one per data-set group)					
INTREL	INTE	INTE or REAL	11-14	A4	Flag to indicate the type of data values to be input (REAL = floating point, INTE = integer).
DTMCOR	0.0	-------	23-30	F8.3	Offset to be added to data values before storing in TDDB.
NCOM	0	0<N<10	37-38	I2	Number of records preceding the Data record(s). These records are read and listed in the print file.
ICKTIM	0	0, 1, 2	46-47	I2	Option to specify the date and time for each data value on the Data record(s) (0 = after value, 1 = do not specify, 2 = before value).
RDFMT	(I4,T71, 5I2)	-------	54-101	A48	Fortran coded input format of data values.
Comment record(s) (up to 10, only required when NCOM > 0)					
COMMNT	blank	-------	1-80	A80	Comment statement.
Data record(s) (number of records depends on the time span and format of the data-set group)					
DATA[1]	0	-------	1-4	I4	Data value.
TIME[1]	0	-------	71-80	5I2	Date and time of value (YRMODYHRMN).

[1]The Data record description is for the default format specification (I4,T71,5I2). If another format is desired, use the Data-Format Specification record to designate another format or indicate that data are in WATSTORE format. The sequence of input variables on the Data records can be different than as indicated above; however, the Data-Format Specification record must be coded to force the DAFILE utility to assign variables in the above sequence. (This can be accomplished through the Fortran T-format code.) Integer data values must be input with two decimal places assumed; that is, integer water-surface elevation values are assumed to be input in hundredths of the unit and discharge values input in hundredths of cubic units per second.

Examples

Figures 7 and 9 show example sets of DAFILE input records. The first example stores in the TDDB 4 days of hourly water-surface elevation values for stations 384940077014901 and 384320077020100. Figure 8 shows the print file for the first example, which summarizes the attributes of the stored data sets. The second example (fig. 9) shows a set of DAFILE input records to store water-surface elevation data recorded every 15 minutes at station 01646500 for the time span 1986/06/01 00:15 to 1986/06/03 02:00. These values are coded in the WATSTORE unit-values format. Figure 10 shows the print file for the second example.

```
                                        0       0
384940077014901 FROM  79/09/11 09:00  TO  79/09/15 08:00RD=  24 TP= Z ST=R4
        REAL REFADJ=   0.000 NCOM= 0 CKTIM= 1 FMT= (8F8.2)
 13.50   13.45   13.40   13.35   13.32   13.32   13.32   13.30
 13.30   13.30   13.32   13.32   13.30   13.32   13.30   13.30
 13.30   13.30   13.30   13.30   13.20   13.15   13.10   13.05
 13.00   12.93   12.90   12.85   12.83   12.81   12.78   12.75
 12.71   12.68   12.68   12.66   12.62   12.58   12.57   12.55
 12.53   12.50   12.50   12.50   12.50   12.49   12.49   12.49
 12.50   12.50   12.50   12.50   12.50   12.50   12.45   12.45
 12.40   12.45   12.45   12.45   12.45   12.50   12.50   12.50
 12.60   12.60   12.60   12.60   12.65   12.65   12.64   12.62
 12.60   12.72   12.83   13.05   13.41   13.97   14.32   14.52
 14.68   14.46   14.55   14.52   14.55   14.54   14.50   14.52
 14.53   14.76   14.07   14.16   14.42   14.62   14.76   15.88
384320077020100 FROM  79/09/11 09:00  TO  79/09/15 08:00RD=  24 TP= Z ST=R4
        REAL REFADJ=   0.000 NCOM= 0 CKTIM= 1 FMT= (10F8.2)
   15.40     15.24     15.15     15.08     15.04     15.00     14.99     14.90
   14.98     14.98     14.90     14.99     14.90     15.01     14.98     14.98
   14.98     14.97     14.97     14.91     14.71     14.60     14.56     14.30
   14.30     14.20     14.15     14.09     14.06     14.05     14.04     14.03
   13.99     13.96     13.96     13.95     13.91     13.85     13.82     13.80
   13.80     13.80     13.78     13.82     13.75     13.75     13.75     13.74
   13.76     13.70     13.77     13.77     13.85     13.76     13.65     13.40
   13.56     13.56     13.56     13.56     13.55     13.54     13.54     13.60
   13.40     13.61     13.61     13.66     13.66     13.66     13.71     13.66
   13.66     13.96     14.10     14.27     14.43     14.64     14.90     15.22
   15.54     15.78     16.01     16.03     16.09     16.13     16.13     16.13
   16.30     16.58     16.77     17.04     17.38     17.63     17.79     17.92
```

Figure 7. Example DAFILE input—control file with user-specified data format.

```
          NEW TIME-DEPENDENT DATA SETS STORED IN THE DIRECT-ACCESS FILE
             PROCESSED BY DAFILE, VERSION: 5.1 1996/02/15
                          ********
                    PROCESSING DATE : 96/06/03
                          ********


          ==========================================
                      DATA RECORD DESCRIPTION
          ==========================================
          DATA FROM REQUEST NO.                    1
          NO. OF READINGS PER DAY                 24
          STATION ID NUMBER           384940077014901
          DATA-PARAMETER CODE                      Z
          STORAGE-TYPE CODE                       R4
          BEGIN DATE AND TIME          79/09/11 09:00
          END DATE AND TIME            79/09/15 08:00
          REFERENCE ADJUSTMENT APPLIED           0.00
          DATA SET SIZE                           96
          ==========================================

***DATA SET STORED IN TDDB***


          ==========================================
                      DATA RECORD DESCRIPTION
          ==========================================
          DATA FROM REQUEST NO.                    2
          NO. OF READINGS PER DAY                 24
          STATION ID NUMBER           384320077020100
          DATA-PARAMETER CODE                      Z
          STORAGE-TYPE CODE                       R4
          BEGIN DATE AND TIME          79/09/11 09:00
          END DATE AND TIME            79/09/15 08:00
          REFERENCE ADJUSTMENT APPLIED           0.00
          DATA SET SIZE                           96
          ==========================================

***DATA SET STORED IN TDDB***
```

Figure 8. Example DAFILE output—print file for storing two data sets.

```
                                    0       0           FMT=F7.0 WATSTORE   R4
2 01646500                  0006500011
B 01646500      19860601001500   96      2.35   2.28    2.2   2.13   2.07   2.01
B 01646500      19860601014500   96      1.95   1.91   1.89   1.92   2.02   2.19
B 01646500      19860601031500   96      2.38   2.55    2.7   2.82   2.91   2.99
B 01646500      19860601044500   96      3.06   3.12   3.17   3.22   3.27   3.31
B 01646500      19860601061500   96      3.35   3.38    3.4   3.42   3.43   3.44
B 01646500      19860601074500   96      3.44   3.43   3.41   3.37   3.31   3.25
B 01646500      19860601091500   96      3.19   3.13   3.07   3.02   2.96    2.9
B 01646500      19860601104500   96      2.84   2.77    2.7   2.63   2.56   2.49
B 01646500      19860601121500   96      2.41   2.34   2.27    2.2   2.13   2.06
B 01646500      19860601134500   96      1.99   1.93   1.88   1.84   1.84    1.9
B 01646500      19860601151500   96      2.04   2.25   2.44   2.61   2.74   2.85
B 01646500      19860601164500   96      2.94   3.02   3.09   3.15   3.21   3.26
B 01646500      19860601181500   96      3.31   3.35   3.38   3.41   3.43   3.45
B 01646500      19860601194500   96      3.46   3.46   3.46   3.44   3.42   3.37
B 01646500      19860601211500   96      3.31   3.25   3.19   3.13   3.08   3.02
B 01646500      19860601224500   96      2.96    2.9   2.83   2.76   2.69   2.62
B 01646500      19860602001500   96      2.55   2.47    2.4   2.32   2.25   2.18
B 01646500      19860602014500   96      2.11   2.04   1.97    1.9   1.85   1.81
B 01646500      19860602031500   96       1.8    1.8   1.83   1.95   2.11   2.28
B 01646500      19860602044500   96      2.43   2.56   2.67   2.76   2.84   2.91
B 01646500      19860602061500   96      2.99   3.06   3.12   3.17   3.21   3.24
B 01646500      19860602074500   96      3.27   3.29    3.3    3.3   3.27   3.23
B 01646500      19860602091500   96      3.17    3.1   3.03   2.97    2.9   2.84
B 01646500      19860602104500   96      2.77    2.7   2.63   2.56   2.48    2.4
B 01646500      19860602121500   96      2.32   2.24   2.16   2.08      2   1.92
B 01646500      19860602134500   96      1.84    1.8   1.76   1.62   1.55    1.5
B 01646500      19860602151500   96      1.47   1.51   1.61   1.82      2   2.15
B 01646500      19860602164500   96      2.31   2.45   2.58   2.69   2.78   2.87
B 01646500      19860602181500   96      2.96   3.04   3.11   3.17   3.22   3.26
B 01646500      19860602194500   96      3.29   3.32   3.34   3.35   3.36   3.37
B 01646500      19860602211500   96      3.37   3.36   3.33   3.29   3.23   3.16
B 01646500      19860602224500   96      3.09   3.03   2.97   2.91   2.84   2.77
B 01646500      19860603001500   96       2.7   2.63   2.56   2.48    2.4   2.32
B 01646500      19860603014500   96      2.23   2.15
```

Figure 9. Example DAFILE input—control file with WATSTORE unit-values data.

```
            NEW TIME-DEPENDENT DATA SETS STORED IN THE DIRECT-ACCESS FILE
                   PROCESSED BY DAFILE, VERSION: 5.1 1996/02/15
                                 ********
                            PROCESSING DATE : 96/06/03
                                 ********


***NOTE*** THIS 2 RECORD BEING PROCESSED
2 01646500                  0006500011


                  ==========================================
                              DATA RECORD DESCRIPTION
                  ==========================================
                  DATA FROM REQUEST NO.                    1
                  NO. OF READINGS PER DAY                 96
                  STATION ID NUMBER                  1646500
                  DATA-PARAMETER CODE                      Z
                  STORAGE-TYPE CODE                       R4
                  BEGIN DATE AND TIME          86/06/01 00:15
                  END DATE AND TIME            86/06/03 02:00
                  REFERENCE ADJUSTMENT APPLIED           0.00
                  DATA SET SIZE                           200
                  ==========================================

***DATA SET STORED IN TDDB***

*NOTE*    34 B RECORDS PROCESSED WITH LAST 2 RECORD
```

Figure 10. Example DAFILE output—print file for storing WATSTORE data.

DAGET—Formatting Data for Other Uses

Use the DAGET utility to retrieve time sequences of data from a TDDB and output the data to a text file in a variety of formats. DAGET facilitates formatting data for use with models and other application programs or for transfer to other data-base systems. Data values can be output with or without the corresponding date and time assigned. In a single execution, DAGET can output any number of time sequences in selected formats.

Several predefined output formats are selectable. Also, data can be output in any format that can be described by a Fortran format specification. Included among the predefined formats are the input formats to the BRANCH model, input-data processor for the SWIFT2D model, and the WATSTORE daily- and unit-values file formats (Hutchinson and others, 1977). For example, DAGET can be used to retrieve a time sequence of discharges computed and stored in the TDDB by the BRANCH model, average them over a day, and then write these daily discharge values to a file in a format compatible with the WATSTORE program DVINPUT (Hutchinson, 1975).

Execution Sequence

The following describes the execution sequence for DAGET. Select DAGET (code 4) from the TDDS MAIN MENU. At the prompt for the name of the control file, enter a valid file name or accept the displayed default file name (**daget.rdr**). If the control file exists, the EXECUTE MENU displays with the following options: “BEGIN execution (default)”, “LIST the contents of the control file”, and “CREATE the control file”. If the control file does not exist, or “CREATE the control file” is selected, prompts request whether TDDB summaries are to be produced before and after execution of DAGET. Then, the following menu displays:

```
                    ** DAGET MENU **

DAGET retrieves and outputs sequential records of data values
and optional date and time from the TDDB according to user-
specifications.  (If the output data set (.out file) is to be
used with DAFILE, use option 5 or 6.)

Code       Data set output format
  1  --  BRANCH model input
  2  --  WATSTORE daily-values
  3  --  WATSTORE unit-values
  4  --  SWIFT2D/IDP model input
  5  --  User-defined
  6  --  10 integer values per record (default)

ENTER INTEGER VALUE FOR SELECTION CODE (1-6)
CURRENT VALUE =      6:
```

The prompt order and the option codes associated with describing output formats follow.

Code = 2 or 3—Retrieval to produce a WATSTORE formatted file

1. Enter the number of decimal digits for data values, must be 0, 1, or 2. Each value can be output with up to six digits including one digit for the decimal period. Up to two of these six digits can be used for decimal precision.

Code = 5 or 6—User-defined and default output formats

1. Enter the number of Comment records to be added to the output file. If no comments are to be input, a prompt requests whether to include two records defining the attributes and format of the time sequence as the first two records in the output file. If Comment records are to be input, a prompt requests whether to include the two attribute records, to include them before Comment records, or to include them after Comment records in the output file. Select to include the two attribute records before Comment records in the output file, if this output file is to be used as input to the DAFILE utility. Next, prompts request the specified number of Comment records.

   For option 5 only, the following prompt displays to request the output format:

```
SELECT a data format using the code numbers from the table below
or ENTER a valid Fortran format specification (including
parentheses).  Interpret the formats below using the following
examples: (10I6) means 10 integer values per record with each
value right justified every 6 record positions (columns);
(10F8.2) means 10 decimal values per record with a value
specified every 8 columns (on input, if the decimal point is not
specified for a value, it is assumed to be right justified with
two implied decimal places, that is, “    12345 “ is read as
1234.50); (I6,4X,I6) means two integer values per record, the
first right justified in column 6 and the second in column 16;
(I6,4X,5I2.2) means one integer value per record right justified
in column 6 and the date and time in columns 11-20 as YRMODYHRMN,
that is, “  1157    9112011015”.  Note that I2.2 means to output
leading zeros in each date field.  To output dates in the form
YR/MO/DY HR:MN, use a format such as: (I6,4X,2(I2,’/’),I2,I3
,’:’,I2).  The date and time can be output as the decimal value
equal to (YR*1000+JULIAN_DAY+(HR*60+MN)/1440).  This value is
output with the specification F9.3.  Note that codes 11-15 output
date and time before data values, codes 5-10 after.
PRESS ENTER TO CONTINUE:
<cr>

      Code   Data Only     Code    Data and Time
      ----   ---------     ----    --------------
        0  (10I6) (default)  8     (I6,I6,2(’/’,I2),I3,’:’,I2)
        1  (I5)              9     (F10.2,5X,F9.3)
        2  (8I10)            10    (4(F10.2,2X,F9.3)
        3  (8F10.2)          11    (5I2,5X,I8)
        4  (F10.2)           12    (5I2,5X,F10.2)
        5  (I10,60X,5I2.2)   13    (2(I2,’/’),I2,I3,’:’,I2,I13)
        6  (4(I8,2X,5I2.2))  14    ((2(I2,’/’),I2,I3,’:’,I2,F10.2)
        7  (I6,4X,5I2.2)     15    (F9.3,5X,F10.2)

ENTER VALUE FOR A SELECTION CODE OR FORTRAN FORMAT SPECIFICATION
CURRENT VALUE = “0                                              “:
```

Code = 1 through 6

1. Enter data-set attributes. (Note that this is the second prompt for code = 2, 3, 5, or 6.) Do this by either selecting a data set from the TDDB index or by specifying the attributes in a sequence of prompts. Respond to the following prompt to make this choice:

```
        ** TIME-SEQUENCE DEFINITION SETUP MENU **

   Code               Activity
     0  --  Select a data set from the TDDB (default)
     1  --  Specify the attributes of the time sequence
 Enter 0 or 1:
```

2. If selection of a data set is chosen, a prompt similar to the following displays:

```
DATA|     STATION    |PARM| START OF ENTRY |  END OF ENTRY  |RDS/|NO OF|STOR|
SET |   IDENTIFIER   |CODE| YR MO DY HR MN | YR MO DY HR MN | DAY|  RDS|TYPE|
=================================================================================
  1 |         1646500|  Q | 80/08/04 06:00 | 80/08/08 19:00 |  96|  437| I4 |
  2 |         1646500|  Q | 80/11/21 01:00 | 80/12/10 24:00 |  24|  480| I4 |
  3 |         1646500|  Q | 81/07/20 06:00 | 81/08/04 06:00 |  24|  361| I4 |
  4 |         1646500|  Q | 81/08/14 24:00 | 81/09/13 24:00 |  24|  721| I4 |
  5 |         1646500|  Q | 81/09/23 00:30 | 81/09/23 24:00 |  96|   95| I4 |
  6 |         1646500|  Z | 86/06/01 00:15 | 86/06/03 02:00 |  96|  200| I2 |
  7 |         1646500|  Z | 86/06/01 00:15 | 86/06/03 02:00 |  96|  200| R4 |
  8 |         1647600|  Z | 81/08/19 00:15 | 81/10/01 24:00 |  96| 4224| I2 |
  9 |         1652100|  Z | 81/09/19 00:15 | 81/10/01 24:00 |  96| 1248| I2 |
 10 |         1652588|  Z | 80/08/04 06:00 | 80/08/08 19:00 |  96|  437| I2 |

   Code               Function
     #  --  Enter data-set number from table
     T  --  Go to first 10 index entries
     R  --  Return to DATA-DEFIINITION SETUP menu
     S  --  Start table at user-input data-set number
     N  --  Go to next 10 index entries

ENTER VALUE FOR SELECTION CODE (DATA-SET NUMBER, T, R, S, OR N)
CURRENT VALUE = “N   “:
```

If specification of attributes is chosen, a sequence of prompts request the station identifier, data-parameter code, storage-type code, begin date and time, end date and time, readings per day, and if data-parameter code = Z, a reference adjustment.

Code = 4—Retrieval to produce SWIFT2D formatted data

3. Enter the date and time of the end of the warmup period.
4. Enter the initial water-surface elevation used in the warmup period (defaults to the average value of the warmup period).
5. Specify whether this is a type A or B side.
6. Enter the tide-opening number.

After completion of an option, a prompt displays to determine if additional retrieval requests (in the specified format) are desired. If yes, the TIME-SEQUENCE DEFINITION MENU displays. After the DAGET control file is created, the EXECUTE MENU displays. Select the BEGIN option to execute DAGET. After DAGET is finished executing, a prompt requests whether to list the print file before returning to the TDDS MAIN MENU.

Input

The DAGET input control file consists of six record types. Three types are required for all executions. Three other record types are optional and their input depends on whether a user-specified format, Comment records, or data in SWIFT2D format are desired. The first record, List Index record, controls the production of TDDB index summaries before and (or) after execution of DAGET. The second input record, Output Control record, specifies the type of output format, number of Comment records, and whether to output the date and time of each value. The third record type, Warmup record, is input only when SWIFT2D output is selected. The fourth record type, Format Specification record, describes (in Fortran code) the output format of each data record. This record is not included when the default-output format is selected. The fifth record type, Comment record(s), is optional. Comment records are used to annotate the output file of data values. The last records in the DAGET control file are the Data-Definition records. One such record is required to specify the time sequence of values to be retrieved and output. The format of all record types is shown in table 8.

As many Data-Definition records can be input as desired. By default, DAGET assumes all retrieval requests to be of the same parameter type, storage type, time span, recording frequency, and to be subject to a common reference adjustment. Therefore, after the first Data-Definition record is defined, subsequent records need only specify a station identifier if appropriate. However, these default conditions can be overridden by specifying any of the Data-Definition variables on subsequent records. The convention is: if a variable is left blank on a Data-Definition record, excluding the station identifier, DAGET assigns to that variable, the value last specified, as other than blank on a previous record. If the variable has not been defined previously, it is assigned the default value shown in table 8. For example, if on the first Data-Definition record, the reference adjustment field is left blank, the reference adjustment is set to the default value from table 8 (0.0). If on the second Data-Definition record the reference adjustment is specified as -5.3, then the default value for the reference adjustment for the next Data-Definition record is -5.3.

Output

The primary result from DAGET is a text file of data records in the specified format. The default name of this data file is **daget.out**. The data records for each station are preceded by the Comment records (if any were input and requested to be included in the output) and a Data-Definition record giving the station number, beginning and ending times, number of data recorded per day, the reference adjustment (if any), and the number of data. For WATSTORE formatted requests, the data file contains type-2 and type-3 records (daily values) or type-2 and type-B records (unit values) in a format compatible with the WATSTORE DVINPT or UVINPT programs, respectively (WATSTORE, volume 1). WATSTORE formatted files may need to be edited, for instance, to provide estimated values for incomplete days of record.

Table 8. Specifications for the DAGET control file

Variable	Default	Values	Position	Format	Description
List Index record (one required per execution)					
LISTB	1	-1, 0, 1, 2	38-39	I2	Option to list TDDB summaries before task(s) (-1 = list both calendar-year and index summary; 0 = no summaries; 1 = list only index summary; 2 = list valid parameter and storage-type codes with definitions and readings per day).
LISTA	1	-1, 0, 1, 2	46-47	I2	Option to list TDDB summaries after task(s) (-1 = list both calendar-year and index summary; 0 = no summaries; 1 = list only index summary; 2 = list valid parameter and storage-type codes with definitions and readings per day).
Output Control record (one required per execution)					
DFPUN	TRUE	TRUE FALSE	2-6	L5	Flag to select the type of output format of the data (TRUE = use DAGET-default format (10I6), FALSE = use BRANCH, WATSTORE, SWIFT2D, or user-specified format as specified on Format Specification record).
DFCRD	0	0, 1, 2, 3	14	I1	Option to output a Data-Definition record (0 = output before any Comment records, 1 = do not output, 2 = output after any Comment records, 3 = output in BRANCH model format).
NCCRDS	0	0 to 10	23-24	I2	Number of Comment records input. Not applicable for BRANCH, SWIFT2D, WATSTORE, or DAGET default output formats.
NDATPT	10	1 to 80	33-34	I2	Number of data output per record. For BRANCH model and WATSTORE daily-value output NDATPT = 8. For SWIFT2D model and WATSTORE unit-value output NDATPT = 6.
WATSTR	0	0 or 1	43	I1	Flag to signal output of data in WATSTORE daily-values format, input format for program DVINPUT (0 = no, 1 = output in DVINPUT format).
TIMOUT	0	0, 1, 2, 3, 4	52	I1	Option to output date and time along with data values for user-specified output format (0 = output data only, 1 = integer date and time after data value, 2 = decimal date and time after data value, 3 = integer date and time before data value, 4 = decimal date and time before data value).
SWIFTF	0	0 or 1	61	I1	Flag to signal output of data in SWIFT2D/IDP input format (0 = no, 1 = yes).
Warmup record (only required when SWIFTF=1)					
BDTIME	0	-------	25-38	5(I2,1X)	Beginning of warmup period (YR/MO/DY HR:MN). The DAGET output file will begin with constant data values for the warmup time period.
WMTIME	0	-------	45-58	5(I2,1X)	End of warmup period (YR/MO/DY HR:MN).
SEINV	0.0	-------	65-72	F8.3	Constant elevation for warmup period (0.0 = the average of the data values as retrieved from the TDDB for the warmup time period, otherwise a constant value should be specified for use at all tide openings).
Format Specification record (only required when DFPUN = FALSE)					
FMT	(10I6)	-------	1-48	A48	Fortran coded output format including outside parentheses. For BRANCH model, FMT=(8F10.2). For SWIFT2D model, FMT= (A1,2X,5I2,2X,6F8.3). For WATSTORE data, each data value is output as F7.0, F7.1, or F7.2 depending on the number of decimal places desired (for unit-value data with zero decimal places data are output as I7). For daily-value data, FMT = ('3',1X,A14,I4,A2,'0', I1,7X, 8F7.0). For unit-value data, FMT = ('B',1X, A14, 6I2.2,'00',I5, 3X, 6F7.0).
FTYPE	INTE	INTE or REAL	73-76	A4	Flag to signal to output data as integer or decimal values (INTE = integer, REAL = floating point). For BRANCH and SWIFT2D models, FTYPE=REAL. For WATSTORE data, FTYPE=REAL, except for unit-value data with zero decimal places, specify FTYPE=INTE.

Table 8. Specifications for the DAGET control file—Continued

Variable	Default	Values	Position	Format	Description
Comment record(s) (up to 10, only required when NCCRDS > 0)					
COMMNT	blank	-------	1-80	A80	Comment statement.
Data-Definition record (one required, others optional)					
STAID	blank	-------	2-17	A16	Station identifier.
OPENCD	A	A or B	19	A1	SWIFT2D/IDP tide-opening code.
OPENNO	1	0 to 99	20-21	I2	SWIFT2D/IDP tide-opening number.
BTIME	0	0<N<100	25-38	5(I2,1X)	Beginning date and time of request (YR/MO/DY HR:MN).
ETIME	0	0<N<100	45-58	5(I2,1X)	Ending date and time of request (YR/MO/DY HR:MN).
DTYPE	Z	parameter codes	60-61	A2	Data-parameter code (for example: Z = water-surface elevation and Q = cross-sectional discharge).
RDPDY	96	look to right for values	62-65	I4	Number of readings per day. Valid values are 1, 8, 12, 24, 48, 96, 144, 240, 288, 720, or 1440.
DATUM	0.0	-------	66-73	F8.3	Reference adjustment to be added to data values.
PARMCD	00065	-------	74-78	I5	WATSTORE data-parameter code (for example, water-surface elevation = 00065, discharge = 00060, dissolved oxygen = 00300, water temperature = 00010, wind speed = 00035, wind direction = 00036, and conductivity = 00095).
STYPE	I2	I2, I4, R4, R8	79-80	A2	Storage-type code (I2 = two-byte integer, I4 = four-byte integer, R4 = four-byte floating point, R8 = eight-byte floating point).

The DAGET print file consists of a summary of the execution and of the retrieval requests for each station (excluding Comment records). This is in addition to TDDB summary listings, if requested. For WATSTORE formatted daily-value requests, the print file also identifies days for which daily values were not computed due to lack of data and includes a table of the daily-mean values computed. The default name of the print file is **daget.prt**.

Examples

Figure 11 shows DAGET input records to retrieve water-surface elevation data for stations 384940077014901 and 384320077020100 and output the data in a user-specified format (10, eight-character wide, floating-point numbers per record). One Comment record is included in the input records and output with each data set. Figure 12 shows a listing of the output file generated by DAGET of these data sets. Input data records for executing DAGET for the same two stations, without Comment records and using the default output format, are shown in figure 13. Figure 14 shows the output data file for this execution. Note that DAGET output integer values with two implied decimal places (multiplied by 100).

```
                                        0       0
 FALSE DFCRD=2 NCCRDS= 1 NDATPT=10 WATSTR=0 TIMOUT=0 SWIFTF=0
(10F8.2)                                                                REAL
Example to show user-defined output format for DAGET.
  384940077014901 FROM  79/09/11 09:00  TO  79/09/15 08:00  Z  24   0.000    0R4
  384320077020100 FROM  79/09/11 09:00  TO  79/09/15 08:00  Z  24   0.000    0R4
```

Figure 11. Example DAGET input—control file with user-specified format.

```
Example to show user-defined output format for DAGET.
  384940077014901 FROM  79/09/11 09:00  TO  79/09/15 08:00RD=  24 TP= Z ST=R4 KT=   96
INT/REAL= REAL REFADJ=   0.000 NCOM= 1 CKTIM= 1 FMT= (10F8.2)
   13.50   13.45   13.40   13.35   13.32   13.32   13.32   13.30   13.30   13.30
   13.32   13.32   13.30   13.32   13.30   13.30   13.30   13.30   13.30   13.30
   13.20   13.15   13.10   13.05   13.00   12.93   12.90   12.85   12.83   12.81
   12.78   12.75   12.71   12.68   12.68   12.66   12.62   12.58   12.57   12.55
   12.53   12.50   12.50   12.50   12.50   12.49   12.49   12.49   12.50   12.50
   12.50   12.50   12.50   12.50   12.45   12.45   12.40   12.45   12.45   12.45
   12.45   12.50   12.50   12.50   12.60   12.60   12.60   12.60   12.65   12.65
   12.64   12.62   12.60   12.72   12.83   13.05   13.41   13.97   14.32   14.52
   14.68   14.46   14.55   14.52   14.55   14.54   14.50   14.52   14.53   14.76
   14.07   14.16   14.42   14.62   14.76   15.88
Example to show user-defined output format for DAGET.
  384320077020100 FROM  79/09/11 09:00  TO  79/09/15 08:00RD=  24 TP= Z ST=R4 KT=   96
INT/REAL= REAL REFADJ=   0.000 NCOM= 1 CKTIM= 1 FMT= (10F8.2)
   15.40   15.24   15.15   15.08   15.04   15.00   14.99   14.90   14.98   14.98
   14.90   14.99   14.90   15.01   14.98   14.98   14.98   14.97   14.97   14.91
   14.71   14.60   14.56   14.30   14.30   14.20   14.15   14.09   14.06   14.05
   14.04   14.03   13.99   13.96   13.96   13.95   13.91   13.85   13.82   13.80
   13.80   13.80   13.78   13.82   13.75   13.75   13.75   13.74   13.76   13.70
   13.77   13.77   13.85   13.76   13.65   13.40   13.56   13.56   13.56   13.56
   13.55   13.54   13.54   13.60   13.40   13.61   13.61   13.66   13.66   13.66
   13.71   13.66   13.66   13.96   14.10   14.27   14.43   14.64   14.90   15.22
   15.54   15.78   16.01   16.03   16.09   16.13   16.13   16.13   16.30   16.58
   16.77   17.04   17.38   17.63   17.79   17.92
```

Figure 12. Example DAGET output—data file of two time sequences in user-specified format.

```
                                        0       0
 TRUE  DFCRD=0 NCCRDS= 0 NDATPT=10 WATSTR=0 TIMOUT=0 SWIFTF=0
  384940077014901 FROM  79/09/11 09:00  TO  79/09/15 08:00  Z  24   0.000    0R4
  384320077020100 FROM  79/09/11 09:00  TO  79/09/15 08:00  Z  24   0.000    0R4
```

Figure 13. Example DAGET input—control file with default format.

```
  384940077014901 FROM  79/09/11 09:00  TO  79/09/15 08:00RD=  24 TP= Z ST=R4 KT=   96
INT/REAL= INTE REFADJ=   0.000 NCOM= 0 CKTIM= 1 FMT= (10I6)
  1350  1345  1340  1335  1332  1332  1332  1330  1330  1330
  1332  1332  1330  1332  1330  1330  1330  1330  1330  1330
  1320  1315  1310  1305  1300  1293  1290  1285  1283  1281
  1278  1275  1271  1268  1268  1266  1262  1258  1257  1255
  1253  1250  1250  1250  1250  1249  1249  1249  1250  1250
  1250  1250  1250  1250  1245  1245  1240  1245  1245  1245
  1245  1250  1250  1250  1260  1260  1260  1260  1265  1265
  1264  1262  1260  1272  1283  1305  1341  1397  1432  1452
  1468  1446  1455  1452  1455  1454  1450  1452  1453  1476
  1407  1416  1442  1462  1476  1588
  384320077020100 FROM  79/09/11 09:00  TO  79/09/15 08:00RD=  24 TP= Z ST=R4 KT=   96
INT/REAL= INTE REFADJ=   0.000 NCOM= 0 CKTIM= 1 FMT= (10I6)
  1540  1524  1515  1508  1504  1500  1499  1490  1498  1498
  1490  1499  1490  1501  1498  1498  1498  1497  1497  1491
  1471  1460  1456  1430  1430  1420  1415  1409  1406  1405
  1404  1403  1399  1396  1396  1395  1391  1385  1382  1380
  1380  1380  1378  1382  1375  1375  1375  1374  1376  1370
  1377  1377  1385  1376  1365  1340  1356  1356  1356  1356
  1355  1354  1354  1360  1340  1361  1361  1366  1366  1366
  1371  1366  1366  1396  1410  1427  1443  1464  1490  1522
  1554  1578  1601  1603  1609  1613  1613  1613  1630  1658
  1677  1704  1738  1763  1779  1792
```

Figure 14. Example DAGET output—data file showing two time sequences in default format.

Figure 15 shows the input records to retrieve a time sequence of water-surface elevation data, compute daily values, and format the computed values for input to the WATSTORE program DVINPUT. A listing of the WATSTORE formatted output file generated by DAGET for this execution is shown in figure 16. Figure 17 shows a partial listing of the retrieval summary and the calendar-year table of daily values computed. Note that the first and last day of the time sequence is incomplete; therefore, a daily value is not included in the output file for those days. DAGET does print the average data value for a partial day as an estimate of the daily value.

```
                                        0       0
 FALSE DFCRD=0 NCCRDS= 0 NDATPT= 8 WATSTR=1 TIMOUT=0 SWIFTF=0
(F7.2)                                                                  REAL
          1655480 FROM  81/08/04 12:15  TO  81/10/03 12:00  Z  96   0.000   65I2
```

Figure 15. Example DAGET input—control file with daily-values format.

```
2 01655480                  0006500003                ENT
3 01655480      19810801                              10.83  10.68  11.02  11.53
3 01655480      19810802  10.89  10.75  10.83  10.63  10.48  10.46  10.79  10.64
3 01655480      19810803  10.26  10.59  10.98  11.52  11.99  11.63  10.98  11.06
3 01655480      19810804  10.60  11.50  11.55  11.23  11.21  11.45  11.42
3 01655480      19810901  11.50  11.60  11.81  11.95  11.88  11.75  11.78  11.81
3 01655480      19810902  11.04  11.34  11.11  10.89  10.99  11.10  10.97  10.84
3 01655480      19810903  11.31  11.24  10.49  10.68  10.29  10.74   9.72   9.48
3 01655480      19810904   9.97  10.80  11.04  10.37  10.29  10.68
3 01655480      19811001  10.59   9.78
```

Figure 16. Example DAGET output—data file showing one time sequence in daily-values format.

```
==================================================
      DATA-RETRIEVAL DESCRIPTION NUMBER      1
==================================================
      STATION ID NUMBER #             1655480
      BEGIN DATE AND TIME       81/08/04 12:15
      END DATE AND TIME         81/10/03 12:00
      DATA PARAMETER CODE                    Z
      NO. OF READINGS PER DAY               96
      REFERENCE ADJUSTMENT APPLIED       0.000
      STORAGE-TYPE CODE                     I2
==================================================
*** THE FIRST DAY HAS    48 VALUES (MEAN =         9.654375    )
    AND IS NOT INCLUDED IN THE OUTPUT.

*** THE LAST DAY HAS    48 VALUES (MEAN =         8.278125    )
    AND IS NOT INCLUDED IN THE OUTPUT.

          A CALENDAR-YEAR SUMMARY OF DAILY-MEAN VALUES CALCULATED FROM UNIT DATA
                              PROCESSING DATE 96/06/03

                              WATER-SURFACE ELEVATION

  CALENDAR YEAR - 1981                                    STATION NO.  01655480
                                      MONTH
```

DAY	JAN	FEB	MAR	APR	MAY	JUN	JUL	AUG	SEP	OCT	NOV	DEC
1									10.50	9.59		
2									10.60	8.78		
3									10.81			
4									10.95			
5								9.83	10.88			
6								9.68	10.75			
7								10.02	10.78			
8								10.53	10.81			
9								9.89	10.04			
10								9.75	10.34			
11								9.83	10.11			
12								9.63	9.89			
13								9.48	9.99			
14								9.46	10.10			
15								9.79	9.97			
16								9.64	9.84			
17								9.26	10.31			
18								9.59	10.24			
19								9.98	9.49			
20								10.52	9.68			
21								10.99	9.29			
22								10.63	9.74			
23								9.98	8.72			
24								10.06	8.48			
25								9.60	8.97			
26								10.50	9.80			
27								10.55	10.04			
28								10.23	9.37			
29								10.21	9.29			
30								10.45	9.68			
31								10.42				

Figure 17. Example DAGET output—print file of daily values.

DAPLOT—Plotting Data in the TDDB

Use the DAPLOT utility to visually display selected time sequences of data on either a digital plotter or printer. The plots provide a simple method to inspect and compare data in a TDDB. DAPLOT can plot as many as seven different data sets together on a digital plotter and up to three on a printer. For example, it is possible to simultaneously plot water-surface elevation data and measured discharge data at one location together with water-surface elevation data from a second location. When multiple data sets are plotted together, they all must span the same beginning-to-end time interval. However, they may have different recording frequencies if all frequencies are multiples of the highest frequency. For example, data sets having 5-, 10-, and 15-minute recording intervals can be plotted together, whereas, data sets having 10- and 15-minute intervals cannot. All data sets may be adjusted to a single common reference, or each data set may be adjusted to its own reference, as may be appropriate.

Printer plots are fixed in size. DAPLOT automatically determines scaling. The y-axis is divided into 100 data units with a scale based on multiples of 1, 1.5, 2, 3, 4, 5, 6, or 8. The x-axis (time axis) scaling is fixed at 48 time increments (based on the recording interval of the first data set) per page. The number of pages for each plot request is then the length in time of the plot request divided by 48 times the recording interval. DAPLOT lists the data values along the time axis in addition to plotting them.

The size of the plots generated for digital plotters is selectable. The default size is rectangular, about 15 by 12 inches. A size factor can be specified to adjust the plot size. To reduce the plot size, set the size factor to a value between zero and one. The time-axis scaling is selectable with a range of from 1 to 9 days of record per plot page. The number of plot pages is then the length in days of the plot request divided by the number of days of record per plot page. DAPLOT automatically determines the y-axis scale based on multiples of 1, 1.5, 2, 3, 4, 5, 6, or 8.

The maximum number of days plotted in a plot request is 8640 divided by the number of readings per day (the 8640 limit is easily changed by editing a Fortran include file named **dimmax.cmn** and then recompiling and reloading the TDDS). Thus, up to 30 days of 5-minute data (288 readings per day) or 90 days of 15-minute data (96 readings per day) can be plotted per individual plot request. There is no limit on the number of plot requests allowed per execution.

For multistation, digital plots, a secondary graph is optionally generated showing difference(s) between data for the first station and all subsequent stations. The time-axis scale of this graph is the same as that of the primary graph. The y-axis (difference) scale is program determined.

DAPLOT derives its graphics capabilities from references to a sub-set of the CalComp graphics library. If such a library is not available, DAPLOT can be easily adapted for other software that accepts the same or a similar set of subroutines (PLOTS, PLOT, FACTOR, NEWPEN, SYMBOL, and NUMBER). A library part of the LIBUTL software is provided that translates the CalComp library references to Graphical Kernel System (GKS) references. The use of LIBUTL, in combination with the Prior GKS library on Data General workstations, allows digital plots to be generated on a variety of output devices. The Prior GKS library produces graphics in formats such as HPGL

(Hewlett Packard Graphics Language), Postscript, or CGM (Computer Graphics Metafile). These formats can be imported to graphics tools such as Collage or word processors such as FrameMaker, MS-Word, or WordPerfect. The LIBUTL software provides a second library (SYMBOL) that includes the routines AXIS, SYMBOL, and NUMBER. These routines produce stroked characters (sequence of straight lines to generate each ASCII character). The combination of the SYMBOL library and the Lahey or Microsoft Fortran graphics libraries can be used to produce digital plots on video-display terminals of DOS-based personal computers.

Execution Sequence

The following describes the execution sequence for DAPLOT. Select DAPLOT (code 5) from the TDDS MAIN MENU. At the prompt for the name of the control file, enter a valid file name or accept the displayed default file name (**daplot.rdr**). If the control file exists, the EXECUTE MENU prompt displays with the following options: "BEGIN execution (default)", "LIST the contents of the control file", and "CREATE the control file". If the control file does not exist or "CREATE the control file" is selected, prompts request whether TDDB summaries are to be produced before and after execution of DAPLOT. Then the following menu displays:

```
                    ** DAPLOT MENU **

   Code                Activity
     0  --  Digital plot (default)
     1  --  Printer plot
Enter 0 or 1:
```

The prompting order and options associated with each prompt follows.

Code = 0—Digital plot

1. Enter the number of days each plot page represents (1-9). Enter a scale factor—used to enlarge or reduce size of plots produced.

Code = 0 and 1

1. Enter the number of curves (time sequence of values) drawn (1-7) on each plot. (Note that this is the second prompt for code=0.)

   For digital plots and number of curves greater than one, a prompt requests whether to also produce a secondary difference plot.

2. Enter a datum adjustment to be added to each value (valid for water-surface elevation data only).

3. Enter data-set attributes for first time sequence. Do this by either selecting a data set from the TDDB index or specifying the attributes as response to a sequence of prompts. Respond to the following prompt to make this choice:

```
          ** TIME-SEQUENCE DEFINITION SETUP MENU **

   Code                Activity
     0  --  Select a data set from the TDDB (default)
     1  --  Specify the attributes of the time sequence
Enter 0 or 1:
```

4. If selection of a data set is chosen, a prompt similar to the following displays:

```
DATA|     STATION    |PARM| START OF ENTRY |  END OF ENTRY  |RDS/|NO OF|STOR|
SET |   IDENTIFIER   |CODE| YR MO DY HR MN | YR MO DY HR MN | DAY|  RDS|TYPE|
=============================================================================
  1 |         1646500|  Q | 80/08/04 06:00 | 80/08/08 19:00 |  96|  437| I4 |
  2 |         1646500|  Q | 80/11/21 01:00 | 80/12/10 24:00 |  24|  480| I4 |
  3 |         1646500|  Q | 81/07/20 06:00 | 81/08/04 06:00 |  24|  361| I4 |
  4 |         1646500|  Q | 81/08/14 24:00 | 81/09/13 24:00 |  24|  721| I4 |
  5 |         1646500|  Q | 81/09/23 00:30 | 81/09/23 24:00 |  96|   95| I4 |
  6 |         1646500|  Z | 86/06/01 00:15 | 86/06/03 02:00 |  96|  200| I2 |
  7 |         1646500|  Z | 86/06/01 00:15 | 86/06/03 02:00 |  96|  200| R4 |
  8 |         1647600|  Z | 81/08/19 00:15 | 81/10/01 24:00 |  96| 4224| I2 |
  9 |         1652100|  Z | 81/09/19 00:15 | 81/10/01 24:00 |  96| 1248| I2 |
 10 |         1652588|  Z | 80/08/04 06:00 | 80/08/08 19:00 |  96|  437| I2 |

   Code               Function
     #  --  Enter data-set number from table
     T  --  Go to first 10 index entries
     R  --  Return to DATA-DEFIINITION SETUP menu
     S  --  Start table at user-input data-set number
     N  --  Go to next 10 index entries

ENTER VALUE FOR SELECTION CODE (DATA-SET NUMBER, T, R, S, OR N)
CURRENT VALUE = “N   “:
```

If attributes are to be specified, a sequence of prompts request the station identifier, data-parameter code, storage-type code, time span of plot request, readings per day, and if data-parameter code = Z, a reference adjustment.

5. Enter a name up to 40 characters used in the legend of each plot.

6. If plotting two-byte integer, water-surface elevation data, a prompt requests if (a) data flags are to be stripped from data values before plotting; (b) strip update flags (22000) but **not** plot data sets containing values with other flags (11000, -11000, -22000); or (c) data flags are **not** stripped and data sets containing flagged data are **not** plotted.

7. If digital plot is selected, a prompt requests a number to assign to the curve. This number designates a line type (such as solid, dashed, and dotted) and (or) color (depending on output graphic device).

8. If multiple curves were chosen, prompts request the data-set attributes of those curves. The attributes requested are station identifier; data-parameter code; storage-type code; readings per day; and if data-parameter code = Z only, a reference adjustment. Multiple curves must be for the same time span, thus prompts for the begin and end time are not issued. After these attributes are entered, return to step 5 in above prompt sequence (station name for plot legend).

After completion of an option, a prompt requests if additional plots are desired. If yes, the DAPLOT MENU displays and the prompting sequence repeats. After the control file is created, the EXECUTE MENU displays. Select the BEGIN option to execute the DAPLOT utility. When finished, a prompt requests whether to list print file before returning to the TDDS MAIN MENU.

Input

DAPLOT input consists of four types of records. The first record, List Index record, controls the production of optional summary listings of the TDDB index before and (or) after execution of DAPLOT. The second through fourth record types are input as a group, with each group defining one plot request. The second record type, Plot Request record, describes a plot request by specifying the plot-request number, the number of stations to be plotted, the size factor for digital plots, and a reference adjustment value if needed. The other two record types, Station Definition and Data Definition records, are input in pairs, one pair for each data set of the plot request. The Station Definition record specifies the station name. The Data Definition record specifies the type of plot (printer or digital), station number, beginning and ending date and time, number of readings per day, reference adjustment, whether or not to plot flagged data, and selection of the color number. Table 9 gives the format of these records.

Output

The primary results are either printer or digital plots. In either case, the print file contains a summary of the input records, the mean value of each data set, and messages to identify if any data sets contain flagged values. If errors are detected in the input records or TDDB, appropriate messages are listed in the print file.

Examples

The input records required to produce a single-day printer and digital plot and a 7-day digital plot of water-surface elevation data for three stations are shown in figure 18. Figure 19 shows the first page of the printer plot produced, whereas figures 20 and 21 show single-day and 7-day digital plots.

```
                                        0       0
   1   3  0.500       0.00 1
Potomac River
PRT PLOT         165548081/09/23 00:15      81/09/23 24:00  Z  96   0.000 -1 1I2
Potomac River
PRT PLOT         165258881/09/23 00:15      81/09/23 24:00  Z  96   0.000 -1 1I2
Potomac River
PRT PLOT         165210081/09/23 00:15      81/09/23 24:00  Z  96   0.000 -1 1I2
   2   3  0.500       0.00 0
Potomac River
DIG PLOT         165548081/09/23 00:15      81/09/23 24:00  Z  96   0.000 -1 1I2
Potomac River
DIG PLOT         165258881/09/23 00:15      81/09/23 24:00  Z  96   0.000 -1 1I2
Potomac River
DIG PLOT         165210081/09/23 00:15      81/09/23 24:00  Z  96   0.000 -1 1I2
   3   3  0.500       0.00 0
Potomac River
DIG 7DAY         165548081/09/22 00:15      81/09/28 24:00  Z  96   0.000 -1 1I2
Potomac River
DIG 7DAY         165258881/09/22 00:15      81/09/28 24:00  Z  96   0.000 -1 1I2
Potomac River
DIG 7DAY         165210081/09/22 00:15      81/09/28 24:00  Z  96   0.000 -1 1I2
```

Figure 18. Example DAPLOT input—control file with three plot requests.

Table 9. Specifications for the DAPLOT control file

Variable	Default	Values	Position	Format	Description
List Index record (one required per execution)					
LISTB	1	-1, 0, 1, 2	38-39	I2	Option to list TDDB summaries before task(s) (-1 = list both calendar-year and index summary; 0 = no summaries; 1 = list only index summary; 2 = list valid parameter and storage-type codes with definitions and readings per day).
LISTA	1	-1, 0, 1, 2	46-47	I2	Option to list TDDB summaries after task(s) (-1 = list both calendar-year and index summary; 0 = no summaries; 1 = list only index summary; 2 = list valid parameter and storage-type codes with definitions and readings per day).
Plot Request record (one required, others optional)					
IIREQ	0	1 to 9999	1-4	I4	Plot request number.
NSPPT	0	1 to 6	5-8	I4	Number of stations to be plotted (up to three for printer plots and up to seven for digital plots).
FACT	1.0	0 to 10	11-15	F5.3	Scaling factor of plots to reduce or enlarge plots. To reduce, use FACT < 1.0.
CDATUM	0.0	-------	20-26	F7.3	Common reference adjustment to annotate plot.
IDIFF	0	0 or 1	27-28	I2	Flag for difference plot (0 = produce difference plot for multistation plot requests, 1 = do not produce difference plot).
Station Definition record (one required per plot request)					
NAME	blank	-------	1-40	A40	Name to describe station.
Data Definition record (one required per plot request)					
OPTION	PRT	PRT or DIG	1-4	A4	Plotter type (PRT = printer, DIG = digital).
PTYPE	PLOT	PLOT, WEEK, or nDAY	5-8	A4	Plot type (PLOT = single day plot, only option for printer plots, 2DAY = 2-day plot, 3DAY = 3-day plot, ..., 9DAY = 9-day plot).
STAID	blank	-------	9-24	A16	Station identifier.
BTIME	0	0<N<100	25-38	5(I2,1X)	Beginning date and time of request (YR/MO/DY HR:MN).
ETIME	0	0<N<100	45-58	5(I2,1X)	Ending date and time of request (YR/MO/DY HR:MN).
DTYPE	‘ Z’	parameter codes	60-61	A2	Data-parameter code (for example, Z = water-surface elevation and Q = cross-sectional discharge).
RDPDY	96	look to right for values	62-65	I4	Number of readings per day. Valid values are 1, 8, 12, 24, 48, 96, 144, 240, 288, 720, or 1440.
DATUM	0.0	-------	66-73	F8.3	Reference adjustment to be added to data values.
STRIP	0	-1, 0, 2	75-76	I2	Option to strip data flags from values (-1 = strip flags—plot all data, 0 = do not strip flags—only plot data sets without flagged values, 1 = strip update flags (22000)—do not plot data sets containing values with other flags (11000, -11000, -22000).
IPENNO	1	1, 2, 3, 4	77-78	I2	Option to select color number for color plots.
STYPE	I2	I2, I4, R4, R8	79-80	A2	Storage-type code (I2 = two-byte integer, I4 = four-byte integer, R4 = four-byte floating point, R8 = eight-byte floating point).

A PLOT OF SELECTED DATA SETS CONTAINED IN THE TDDB
PROCESSED BY "DAPLOT" VERSION: 5.0 1996/02/15
# # # #
PROCESSING DATE 96/06/03 START= 81/09/23 00:15
PAGE 1 BEGINS 81/09/23 00:15 (A) 1655480(Z) (B) 1652588(Z) (C) 1652100(Z) END= 81/09/23 24:00

WATER-SURFACE ELEVATION 7.20 7.80 8.40 9.00 9.60 10.20 10.80 11.40 12.00 12.60 13.20

TIME	(A)	(B)	(C)
00:15	10.2	12.8	10.8
00:30	10.3	12.8	11.0
00:45	10.3	12.8	11.2
01:00	10.4	12.8	11.3
01:15	10.4	12.8	11.4
01:30	10.5	12.8	11.6
01:45	10.5	12.8	11.7
02:00	10.5	12.8	11.8
02:15	10.6	12.8	11.9
02:30	10.6	12.8	12.0
02:45	10.5	12.8	12.0
03:00	10.5	12.8	12.0
03:15	10.5	12.8	12.0
03:30	10.4	12.8	12.0
03:45	10.3	12.8	11.9
04:00	10.2	12.8	11.8
04:15	10.1	12.8	11.7
04:30	9.99	12.8	11.6
04:45	9.87	12.8	11.5
05:00	9.73	12.8	11.4
05:15	9.64	12.8	11.2
05:30	9.52	12.8	11.1
05:45	9.43	12.8	10.9
06:00	9.33	12.8	10.8
06:15	9.21	12.8	10.6
06:30	9.12	12.8	10.5
06:45	9.04	12.8	10.3
07:00	8.94	12.8	10.1
07:15	8.84	12.8	9.97
07:30	8.75	12.8	9.84
07:45	8.66	12.8	9.67
08:00	8.57	12.8	9.54
08:15	8.52	12.8	9.41
08:30	8.43	12.8	9.29
08:45	8.34	12.8	9.21
09:00	8.27	12.8	9.11
09:15	8.23	12.8	9.00
09:30	8.17	12.8	8.90
09:45	8.16	12.7	8.83
10:00	8.13	12.7	8.73
10:15	8.09	12.6	8.67
10:30	8.10	12.4	8.63
10:45	8.14	12.3	8.57
11:00	8.13	12.1	8.55
11:15	8.16	12.0	8.53
11:30	8.20	11.8	8.49
11:45	8.16	11.7	8.46
12:00	8.21	11.5	8.50

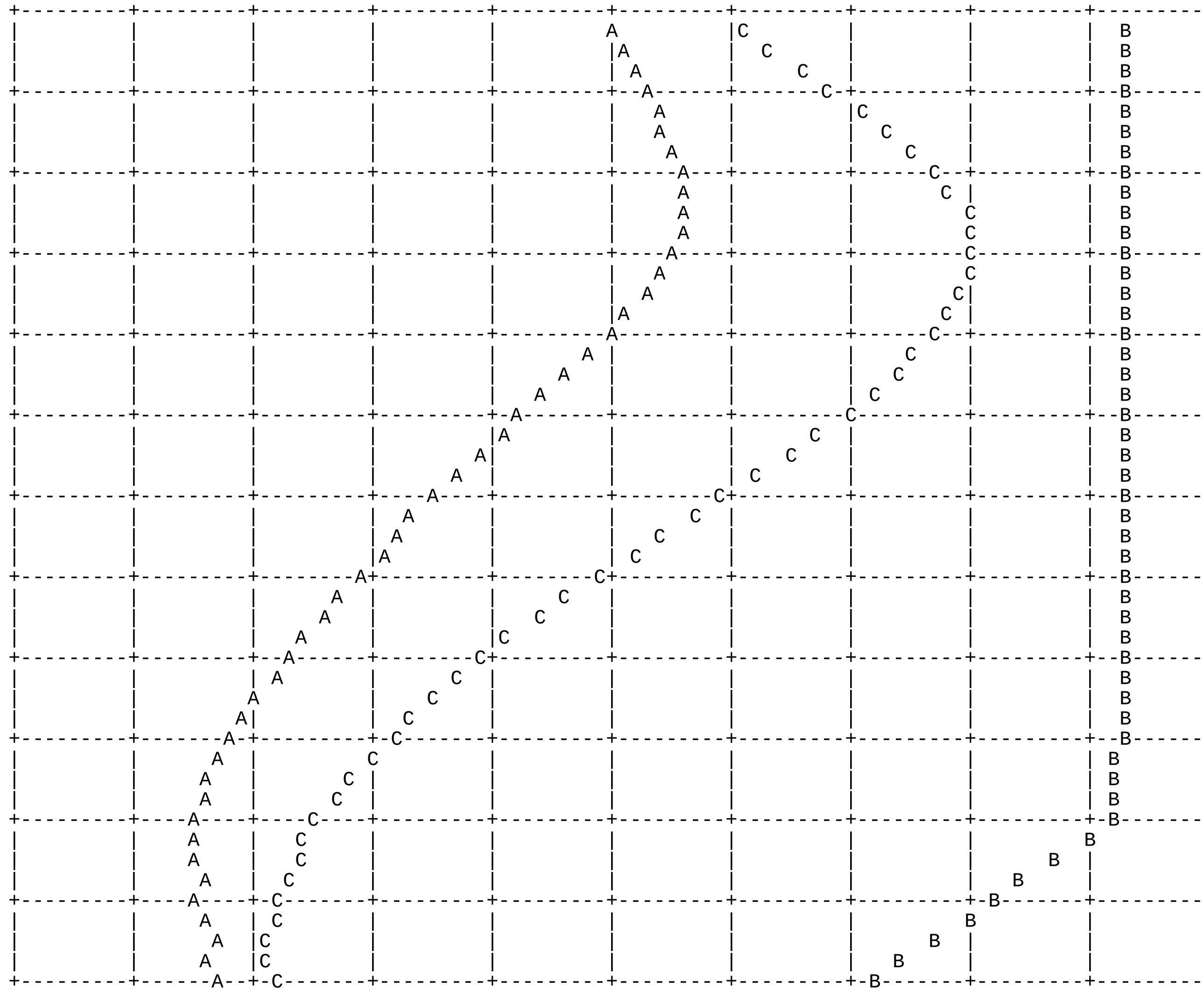

Figure 19. Example DAPLOT output—print file of printer plot.

DAFIX—Modifying, Deleting, and Printing Data in a TDDB

Use the DAFIX utility to change an individual data value, to add a constant value or apply a scale factor to a time sequence of values, to delete single or time sequence of values, and to produce concise listings of data in the TDDB. Data can be replaced in a TDDB either by changing individual values or by deleting values and storing new values. **The TDDS does not allow data to be stored over existing data in the TDDB; therefore, existing data must be deleted before it can be replaced.** If many changes are needed to a data set, it can be faster to delete the entire data set and then store a modified version than to change individual values using DAFIX. For example, to replace an entire time sequence of values, (1) obtain corrected data (either as a new data set or retrieve the original data using DAGET and then edit this data with a text editor); (2) delete the original data set from the TDDB using DAFIX; and (3) store the corrected data set using DAFILE.

A DAFIX delete request does not remove the time sequence of values from the TDDB. Instead, the index entries associated with the time sequence are removed if the sequence covers complete data sets. Otherwise, the index entries are modified to account for the time sequence deleted. Thus, the TDDB does not decrease in size after a delete request. To recover the space occupied by deleted data, use the BACKUP utility twice, first to create a binary backup copy of the TDDB and second to re-create the TDDB. See the BACKUP—Maintenance of the TDDB section for more details.

When two-byte, integer water-surface elevation data are changed using DAFIX, an update flag (22000) is added to each modified value to reflect its revised status. Data sets containing updated two-byte, integer water-surface elevation values are identified in the index and calendar-year summaries. Also, DAFIX printout requests identify updated individual values. Update flags are not available for other combinations of storage-type and data-parameter codes.

DAFIX can be used to produce concise listings of selected data. Two formats are available: a four-digit daily table of values shown scaled to four-significant figures in an hour by minute format (shown in fig. 25); and a six-column daily table of values that includes a list of the minimum, maximum, and average values for each day and the minimum, maximum, median, and average values for the time span specified (shown in fig. 23). Update flags associated with two-byte integer water-surface elevation data **are not** stripped (values greater than 21,000 (-999+22000) are updated values) in the second printout format. However, update flags **are** stripped and flagged data values are marked by an asterisk in the first print format. The six-column format is the only means within the TDDS to list the data values as stored in the TDDB. Other TDDS utilities strip update flags.

Execution Sequence

The following describes the execution sequence for DAFIX. Select DAFIX (code 6) from the TDDS MAIN MENU. At the prompt for the name of the control file, enter a valid file name or accept the displayed default file name (**dafix.rdr**). If the control file exists, the EXECUTE MENU prompt displays with the following options: "BEGIN execution (default)", "LIST the contents of the control file", and "CREATE the control file". If the control file does not exist or "CREATE the control file" is selected, prompts request whether TDDB summaries are to be produced before and after execution of DAFIX. Then the following menu displays:

```
                  ** DAFIX MENU **

 Code                  Activity
   1  --  Modify a data value
   2  --  Remove update flag from a two-byte, integer
          water-surface elevation value
   3  --  Apply increment/decrement to data
   4  --  Delete data from the TDDB
   5  --  Print data from the TDDB
   6  --  Finished creating control file (default)

ENTER INTEGER VALUE FOR SELECTION CODE (1-6)
CURRENT VALUE =      6:
```

The prompting sequence for each activity follows.

Code = 1—Modify a data value

1. Enter data attributes (station identifier, data-parameter code, storage-type code, date and time).
2. Is a printout of the data desired after the change is made (this print is in the four-digit format and shows the values for the day of the change request)?
3. After the control file is created and BEGIN is selected from the execute menu, DAFIX prompts for a new value. Enter the value from the keyboard as an integer, floating-point, or double-precision number (depending on storage type). The prompt shows the current value (minus any update flag) as stored in the TDDB, which also acts as the default value. Then the following prompts display to request whether it is okay to make the change as specified:

```
 Modify request for station = “         1647600”
 Date = 81/09/22 09:30 Parameter = “ Z” Storage type = “I2”

 ENTER INTEGER VALUE FOR REPLACING VALUE IN THE TDDB
 CURRENT VALUE =   5132:
921

 ARE YOU SURE YOU WANT TO MAKE THIS CHANGE [Y,n]?
```

Code = 2—Remove update flags from a two-byte integer water-surface elevation value

1. Enter data attributes (station identifier, date, and time). Note that the data-parameter code is set to “ Z” and storage-type code is set to “I2”.
2. Is a printout of the data set desired after changes are made (this print is in the four-digit format and shows the values for the day of the change request)?

Code = 3, 4, or 5—Apply increment/decrement, delete, or print values

1. For option 3 only, a prompt requests if a printout of the data set is desired after the changes are made (this print is in the four-digit format and shows the values for the day of the change request)?

   For option 5 only, a prompt requests whether the four-significant figures or six-column format is wanted.

2. The following prompt displays:

```
Code                Activity
  0  --  Select an entire data set from a table (default)
  1  --  Specify attributes for a single data value
  2  --  Specify attributes for a time sequence of values
Enter 0, 1, or 2:
```

3. If an entire data set is to be selected, a prompt similar to the following displays:

```
DATA|     STATION    |PARM| START OF ENTRY |  END OF ENTRY  |RDS/|NO OF|STOR|
SET |   IDENTIFIER   |CODE| YR MO DY HR MN | YR MO DY HR MN | DAY|  RDS|TYPE|
================================================================================
  1 |         1646500|  Q | 80/08/04 06:00 | 80/08/08 19:00 |  96|  437| I4 |
  2 |         1646500|  Q | 80/11/21 01:00 | 80/12/10 24:00 |  24|  480| I4 |
  3 |         1646500|  Q | 81/07/20 06:00 | 81/08/04 06:00 |  24|  361| I4 |
  4 |         1646500|  Q | 81/08/14 24:00 | 81/09/13 24:00 |  24|  721| I4 |
  5 |         1646500|  Q | 81/09/23 00:30 | 81/09/23 24:00 |  96|   95| I4 |
  6 |         1646500|  Z | 86/06/01 00:15 | 86/06/03 02:00 |  96|  200| I2 |
  7 |         1647600|  Z | 81/08/19 00:15 | 81/10/01 24:00 |  96| 4224| I2 |
  8 |         1652100|  Z | 81/09/19 00:15 | 81/10/01 24:00 |  96| 1248| I2 |
  9 |         1652588|  Z | 80/08/04 06:00 | 80/08/08 19:00 |  96|  437| I2 |
 10 |         1652588|  Z | 81/08/19 00:15 | 81/10/01 24:00 |  96| 4224| I2 |

 Code                 Function
   #  --  Enter data-set number from table
   T  --  Go to first 10 index entries
   D  --  Disregard current data-set selection
   S  --  Start table at user-input data-set number
   N  --  Go to next 10 index entries (default)

ENTER VALUE FOR SELECTION CODE (DATA-SET NUMBER, T, D, S, OR N)
CURRENT VALUE = “N   “:
```

If a single value is to be specified, prompts request the data-set attributes (station identifier, data-parameter code, storage-type code, and individual date and time).

If a time sequence of values are to be specified, prompts request the data-set attributes (station identifier, data-parameter code, storage-type code, and begin and end date and time of the request).

4. For option 3 only, enter the reference adjustment to add to and scale factor to multiply by the time sequence of values before restoring to the TDDB.

   For option 5 only, before printing, note that the data values in the TDDB are unaffected; factor only applied to printed values.

After completion of an option, the DAFIX menu displays a prompt for additional requests. After the control file is created, the EXECUTE MENU displays. Select the BEGIN option to execute the DAFIX utility. When finished, a prompt requests whether to list the print file before returning to the TDDS MAIN MENU.

Input

Input to DAFIX consists of two record types. The first record, List Index record, controls the production of optional summary listings of the TDDB index before and (or) after execution of DAFIX. The second and subsequent records, Data Correction record(s), specify the operation (modify, delete, and printout) to be performed on the selected data. Table 10 shows the format of these records.

Data Correction records specify instructions to validate, increment, decrement, scale, print, or delete a time sequence of data or an individual data value. They can be input in any order. Care must be taken to specify the correct beginning and ending dates and times for modify and delete requests to avoid mistakenly affecting valid data. As protection against this possibility, requests are processed only if the specified time span is totally contained in a singular data set (single TDDB index entry). If a time span to modify or delete data values extends beyond a single data set, a separate Data Correction record is required for each data set affected.

Prior to processing, DAFIX sorts the Data Correction records by type of request, station number, beginning date and time, data-parameter code, and storage-type code. Processing begins with printout requests, thereby showing the data-set status prior to any modifications requested for the DAFIX execution. These are then followed by delete and modify requests, respectively. The sorting process requires that Data Correction records be coded consistently. This is especially true for time values and station identifiers. Requests are sorted as character fields. Therefore, if leading zeros are coded for some time or station identifiers and not for others, requests would be improperly sorted creating unpredictable processing results. By convention, code single-digit date and time values, right justified in the two-digit fields with a leading zero (that is, 94/12/01 05:00). The interactive procedure for creating the DAFIX input records codes these values in a consistent manner to ensure a correctly sorted processing order.

Output

The primary results of DAFIX are modified data in the TDDB. The print file contains a listing of each Data Correction record input along with a brief statement indicating the action taken or error condition encountered and time sequence of values or TDDB index summary printout, if requested. As an option, an auxiliary listing can be produced for modify requests. This listing is in the four-digit daily table format and shows data values for the specified time span after the requested modification(s). The default name of the print file is dafix.prt.

Table 10. Specifications for the DAFIX control file

Variable	Default	Values	Position	Format	Description
List Index record (one required per execution)					
LISTB	1	-1, 0, 1, 2	38-39	I2	Option to list TDDB summaries before task(s) (-1 = list both calendar-year and index summary; 0 = no summaries; 1 = list only index summary; 2 = list valid parameter and storage-type codes with definitions and readings per day).
LISTA	1	-1, 0, 1, 2	46-47	I2	Option to list TDDB summaries after task(s) (-1 = list both calendar-year and index summary; 0 = no summaries; 1 = list only index summary; 2 = list valid parameter and storage-type codes with definitions and readings per day).
Data Correction record(s) (one required per execution, others optional)					
KEYWRD	(None)	U, P, T, or S	1	A1	Identifies kind of data-processing task to be performed (M = modify request, P = printout request, T = six-digit table printout request, and D = delete request).
STAID	blank	-------	2-17	A16	Station identifier.
TYPE	(None)	FROM or DATE	19-22	A4	Type of editing to be performed (FROM = modify, print, or delete a sequence of values, DATE = modify, print, or delete a single value).
BTIME	0	0<N<100	25-38	5(I2,1X)	Beginning date and time of task for TYPE=FROM, or exact date and time of task for TYPE=DATE request (YR/MO/DY HR:MN).
COK	blank	OK	40-41	A2	Flag to indicate that data value is to be unchanged and to remove update flag (valid for TYPE=DATE, KEYWRD=U requests, and DTYPE=Z).
ETIME	0	0<N<100	45-58	5(I2,1X)	Ending date and time of edit for TYPE=FROM (YR/MO/DY HR:MN).
DTYPE	Z	parameter codes	60-61	A2	Data-parameter code (for example, Z = water-surface elevation and Q = cross-sectional discharge).
STYPE	I2	I2, I4, R4, R8	63-64	A2	Storage-type code (I2 = two-byte integer, I4 = four-byte integer, R4 = four-byte floating point, R8 = eight-byte floating point).
DSHIFT	0.0	-------	66-72	F7.2	Offset to be added to data values. DSHIFT is applied before SCFACT. Values in the TDDB are unaffected for KEYWRD=P or R request. For these requests, data are retrieved, DSHIFT is added to each value, and then the new values are listed. Values in the TDDB are changed for TYPE=FROM, KEYWRD=U requests. For these requests, data are retrieved, DSHIFT is added, then the new values are stored in the TDDB.
SCFACT	0.0	-------	73-79	F7.2	Multiplier to be applied to data values. DSHIFT is applied before SCFACT. Values in the TDDB are unaffected for KEYWRD=P or R request. For these requests, data are retrieved, each value is multiplied by SCFACT, and then the new values are listed. Values in the TDDB are changed for KEYWRD=U requests. For these requests, data are retrieved, each value is multiplied by Scfact, then the new values are stored in the TDDB.
PRINT	blank	P	80	A1	Flag to indicate that a printout is desired after KEYWRD=U requests (blank=no printout, P=print changed data values).

Examples

Figure 22 shows DAFIX input records used to produce the output shown in figures 23-25. The purpose of each input record, in order of input, is as follows:

1. List TDDB index summary after processing Data Correction records.
2. List data values from a data set to show the six-column daily format.
3. Apply a datum correction by adding 0.02 to the values of a data set, and list the data values after correction to show the four-digit daily format.
4. Change a value from 51.32 to 9.21 for 81/09/22 09:30.
5. Change a value from 85.68 to 9.20 for 81/09/22 09:45.
6. Delete a data set.

```
                                        0       1
TComputed 1652588 FROM  80/11/30 08:00  TO  80/12/01 15:15  Q I4    0.00   1.00
M         1652100 FROM  81/09/19 00:15  TO  81/10/01 24:00  Z I2    0.02   1.00P
M         1647600 DATE  81/09/22 09:30                       Z I2
M         1647600 DATE  81/09/22 09:45                       Z I2               P
D         1647600 FROM  81/09/21 16:30  TO  81/09/22 09:15  Z I2    0.00   1.00
```

Figure 22. Example DAFIX input—control file to print, modify, and delete data.

Figure 23 shows the six-column table listing of the selected data. Figure 24 shows a summary of the input records and the processes performed. Notice that DADIO warning messages were issued; because these were nonfatal, all requests were processed. If the DADIO routines detect a fatal error, a message is included in the print file stating that the request was ignored and DAFIX tries to process any remaining requests. Note that the input records as shown in figure 22 were sorted by DAFIX to produce the processing order as shown in figure 24. Thus, the print requests were processed first, the delete request next, and then the modify requests (modify requests were sorted by station number and time period causing the datum correction request to be performed last). Also, note that for a modify request of an individual value (records 4 and 5 above), DAFIX displays the value stored in the TDDB and then requests a new value.

When a value(s) is modified and the PRINT variable is specified as “P”, as for the third modify request shown in figure 22, all data for that particular day are shown in the four-digit table format. If multiple modify requests are made, with PRINT=“P”, for data of the same day, a single four-digit table for that day is produced. If the requests extend to multiple days, output consists of a four-digit table for only those days for which a data value was modified.

Figure 25 shows the four-digit table for the data set after the two values on 81/09/22 are corrected and a portion of the four-digit table after the 0.02 datum correction is applied. Notice that an asterisk preceding a data value indicates its value has been changed from its original value when it was initially stored in the TDDB. This is true for the first two values shown for station 1647600 and all values for station 1652100.

The data values deleted are those for station 1647600 that were flagged as possibly being in error. Because these values are a subset of a single data set, DAFIX splits the original data set in two, leaving a gap in the original time sequence. (This gap can then be filled using DAFILE to store corrected data.) As a result, the index summary after this execution of DAFIX (not shown) has one more index entry.

```
STATION IDENTIFIER = "Computed 1652588"  DATA-PARAMETER CODE = " Q" PARAMETER = CROSS-SECTIONAL DISCHARGE  DATE = 80/11/30

  TIME     VALUE     TIME     VALUE     TIME     VALUE     TIME     VALUE     TIME     VALUE
 00:15     -----    04:15     -----    08:15  16432.80    12:15 -21490.54    16:15  25030.96    20:15  17939.11
 00:30     -----    04:30     -----    08:30  15459.97    12:30 -20790.79    16:30  26125.87    20:30  16856.87
 00:45     -----    04:45     -----    08:45  14627.66    12:45 -18919.93    16:45  26629.72    20:45  15667.77
 01:00     -----    05:00     -----    09:00  13862.73    13:00 -16052.32    17:00  26673.34    21:00  14316.66
 01:15     -----    05:15     -----    09:15  13099.55    13:15 -12548.41    17:15  26440.55    21:15  12751.41
 01:30     -----    05:30     -----    09:30  12254.06    13:30  -8913.69    17:30  26001.63    21:30  10841.29
 01:45     -----    05:45     -----    09:45  11254.87    13:45  -5400.20    17:45  25388.92    21:45   8321.27
 02:00     -----    06:00     -----    10:00  10026.47    14:00  -1962.31    18:00  24717.98    22:00   4846.70
 02:15     -----    06:15     -----    10:15   8365.37    14:15   1486.43    18:15  24114.44    22:15     47.62
 02:30     -----    06:30     -----    10:30   5925.47    14:30   4894.80    18:30  23576.90    22:30  -6187.35
 02:45     -----    06:45     -----    10:45   2389.46    14:45   8249.68    18:45  23003.14    22:45 -12909.01
 03:00     -----    07:00     -----    11:00  -2370.60    15:00  11624.69    19:00  22329.53    23:00 -18735.77
 03:15     -----    07:15     -----    11:15  -8045.75    15:15  14974.77    19:15  21569.88    23:15 -22688.32
 03:30     -----    07:30     -----    11:30 -13669.95    15:30  18165.51    19:30  20751.51    23:30 -24695.82
 03:45     -----    07:45     -----    11:45 -18130.29    15:45  21012.97    19:45  19882.12    23:45 -25185.61
 04:00     -----    08:00  17582.28    12:00 -20730.62    16:00  23325.93    20:00  18947.70    24:00 -24558.67

Minimum:-0.25186D+05   Maximum: 0.26673D+05   Average:  0.6520D+04

STATION IDENTIFIER = "Computed 1652588"  DATA-PARAMETER CODE = " Q" PARAMETER = CROSS-SECTIONAL DISCHARGE  DATE = 80/12/01

  TIME     VALUE     TIME     VALUE     TIME     VALUE     TIME     VALUE     TIME     VALUE
 00:15 -23169.49    04:15  18013.49    08:15  23676.00    12:15 -27766.74    16:15     -----    20:15     -----
 00:30 -21375.98    04:30  21783.84    08:30  22878.84    12:30 -27340.67    16:30     -----    20:30     -----
 00:45 -19496.73    04:45  24859.88    08:45  22048.52    12:45 -25899.98    16:45     -----    20:45     -----
 01:00 -17747.39    05:00  27160.38    09:00  21149.63    13:00 -23867.25    17:00     -----    21:00     -----
 01:15 -16201.51    05:15  28811.98    09:15  20103.22    13:15 -21660.92    17:15     -----    21:15     -----
 01:30 -14750.75    05:30  29937.09    09:30  18758.00    13:30 -19655.70    17:30     -----    21:30     -----
 01:45 -13220.55    05:45  30573.92    09:45  16976.45    13:45 -18037.07    17:45     -----    21:45     -----
 02:00 -11501.27    06:00  30762.29    10:00  14681.62    14:00 -16667.60    18:00     -----    22:00     -----
 02:15  -9564.88    06:15  30568.30    10:15  11752.26    14:15 -15259.47    18:15     -----    22:15     -----
 02:30  -7385.62    06:30  30056.74    10:30   7878.58    14:30 -13631.97    18:30     -----    22:30     -----
 02:45  -4872.42    06:45  29311.93    10:45   2616.94    14:45 -11762.78    18:45     -----    22:45     -----
 03:00  -1939.75    07:00  28433.77    11:00  -4240.92    15:00  -9671.83    19:00     -----    23:00     -----
 03:15   1444.64    07:15  27483.15    11:15 -11899.86    15:15  -7349.27    19:15     -----    23:15     -----
 03:30   5271.54    07:30  26475.49    11:30 -18804.95    15:30     -----    19:30     -----    23:30     -----
 03:45   9458.41    07:45  25455.22    11:45 -23838.91    15:45     -----    19:45     -----    23:45     -----
 04:00  13805.61    08:00  24511.79    12:00 -26741.83    16:00     -----    20:00     -----    24:00     -----

Minimum:-0.27767D+05   Maximum: 0.30762D+05   Average:  0.2645D+04

Number of values in retrieval:   126
Minimum value:-0.27767D+05   Median value: 0.14978D+04
Maximum value: 0.30762D+05   Average value: 0.46443D+04
```

Figure 23. Example DAFIX output—print file showing the six-column table format.

```
                          CONTROL-RECORD SUMMARY

                                 #  #  #  #

REQUEST |TComputed 1652588 FROM  80/11/30 08:00  TO  80/12/01 15:15  Q I4    0.00   1.00 | PROCESSED

         ***DADIO ERROR, RETURN CODE = 17***
REQUESTED DATA CONTAINS PREVIOUSLY FLAGGED VALUES
(WARNING CODES 11000, -11000, or -22000)
REQUESTED DATA CONTAINS PREVIOUSLY UPDATED VALUES
(WARNING CODE 22000 ASSIGNED BY DAFIX)
STATION IDENTIFIER = “         1647600”
TIME SPAN = 81/08/19 00:15 TO 81/10/01 24:00
DATA-PARAMETER CODE = “ Z”    READINGS/DAY=  96    STORAGE-TYPE CODE = “I2”


REQUEST |D        1647600 FROM  81/09/21 16:30  TO  81/09/22 09:15  Z I2    0.00   1.00 | PROCESSED


         ***DADIO WARNING, RETURN CODE = 10***
REQUESTED DATA CONTAINS PREVIOUSLY UPDATED VALUES
(WARNING CODE 22000 ASSIGNED BY DAFIX)
STATION IDENTIFIER = “         1647600”
TIME SPAN = 81/09/22 09:30 TO 81/10/01 24:00
DATA-PARAMETER CODE = “ Z”    READINGS/DAY=  96    STORAGE-TYPE CODE = “I2”


REQUEST |M        1647600 DATE  81/09/22 09:30                    Z I2               | VALIDATED

REQUEST |M        1647600 DATE  81/09/22 09:45                    Z I2              P| VALIDATED
DATA SET|M        1647600 FROM  81/09/22 09:30  TO  81/09/22 09:45                   P| PROCESSED


REQUEST |M        1652100 FROM  81/09/19 00:15  TO  81/10/01 24:00  Z I2    0.02  1.00P| VALIDATED
DATA SET|M        1652100 FROM  81/09/19 00:15  TO  81/10/01 24:00                  P| PROCESSED


                                 #  #  #  #
```

Figure 24. Example DAFIX output—print file showing the input record summary.

```
STATION IDENTIFIER = "          1647600"    " Z" X 10** 02   PARAMETER = WATER-SURFACE ELEVATION      DATE = 81/09/22
HOUR |  OO   01   02   03   04   05   06   07   08   09   10   11   12   13   14   15   16   17   18   19   20   21   22   23   24
  MIN +----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+
   00 | ---- ---- ---- ---- ---- ---- ---- ----  922  966 1061 1138 1182 1202 1192 1146 1082 1028  989  955  930  970 1064
   15 | ---- ---- ---- ---- ---- ---- ---- ----  928  986 1084 1153 1190 1202 1182 1127 1068 1016  984  948  928  996 ---- 
   30 | ---- ---- ---- ---- ---- ---- ---- ----* 921  936 1009 1104 1163 1195 1202 1172 1119 1055 1004  975  942  931 1022 ---- 
   45 | ---- ---- ---- ---- ---- ---- ---- ----* 920  949 1036 1123 1173 1199 1200 1159 1101 1041  994  964  935  946 1044 ---- 

                                                CODES
      > = VALUE IS OUT OF RANGE;  # = REPEAT COUNT EXCEEDED;  % = RATE OF CHANGE EXCEEDED;  * = UPDATED

STATION IDENTIFIER = "          1652100"    " Z" X 10** 02   PARAMETER = WATER-SURFACE ELEVATION      DATE = 81/09/19
HOUR |  OO   01   02   03   04   05   06   07   08   09   10   11   12   13   14   15   16   17   18   19   20   21   22   23   24
  MIN +----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+
   00 | ----*1183*1112*1041* 984* 942* 919* 919* 969*1057*1123*1157*1163*1128*1071*1011* 961* 924* 909* 906* 938*1030*1120*1187*1228
   15 |*1227*1160*1094*1025* 971* 934* 916* 925* 989*1077*1135*1164*1158*1118*1058* 998* 950* 910* 908* 910* 958*1056*1139*1200 ----
   30 |*1214*1148*1074*1011* 960* 928* 915* 936*1009*1093*1144*1167*1152*1100*1051* 982* 940* 914* 907* 914* 980*1079*1156*1212 ----
   45 |*1200*1131*1058* 996* 950* 923* 916* 951*1034*1108*1152*1166*1141*1083*1025* 972* 930* 911* 906* 923*1004*1100*1172*1221 ----

STATION IDENTIFIER = "          1652100"    " Z" X 10** 02   PARAMETER = WATER-SURFACE ELEVATION      DATE = 81/09/20
HOUR |  OO   01   02   03   04   05   06   07   08   09   10   11   12   13   14   15   16   17   18   19   20   21   22   23   24
  MIN +----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+
   00 | ----*1230*1194*1139*1078*1024* 978* 948* 940* 971*1054*1134*1178*1192*1163*1106*1043* 989* 947* 919* 915* 962*1059*1148*1207
   15 |*1232*1224*1183*1124*1064*1011* 969* 944* 943* 988*1077*1140*1185*1188*1152*1091*1028* 977* 939* 916* 921* 984*1084*1167 ----
   30 |*1234*1216*1168*1108*1051*1000* 961* 941* 948*1009*1099*1159*1190*1182*1137*1075*1014* 965* 931* 914* 931*1008*1107*1183 ----
   45 |*1233*1205*1153*1093*1038* 989* 953* 940* 957*1031*1116*1169*1192*1173*1122*1059*1001* 954* 925* 914* 945*1033*1129*1196 ----

STATION IDENTIFIER = "          1652100"    " Z" X 10** 02   PARAMETER = WATER-SURFACE ELEVATION      DATE = 81/09/21
HOUR |  OO   01   02   03   04   05   06   07   08   09   10   11   12   13   14   15   16   17   18   19   20   21   22   23   24
  MIN +----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+
   00 | ----*1228*1215*1165*1100*1034* 976* 932* 903* 886* 896* 959*1044*1104*1131*1124*1082*1021* 961* 912* 882* 880* 932*1034*1125
   15 |*1216*1228*1205*1149*1083*1018* 963* 923* 900* 884* 906* 980*1062*1115*1134*1116*1068*1006* 947* 902* 879* 887* 956*1060 ----
   30 |*1223*1227*1193*1133*1067*1004* 951* 915* 893* 885* 920*1002*1079*1124*1132*1105*1052* 990* 934* 893* 877* 898* 982*1084 ----
   45 |*1226*1222*1180*1117*1051* 989* 941* 900* 889* 880* 938*1023*1092*1120*1130*1094*1037* 974* 922* 887* 877* 913*1000*1105 ----

STATION IDENTIFIER = "          1652100"    " Z" X 10** 02   PARAMETER = WATER-SURFACE ELEVATION      DATE = 81/09/22
HOUR |  OO   01   02   03   04   05   06   07   08   09   10   11   12   13   14   15   16   17   18   19   20   21   22   23   24
  MIN +----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+
   00 | ----*1184*1215*1212*1178*1123*1061*1006* 962* 930* 925* 968*1056*1134*1178*1197*1185*1140*1088*1025* 988* 956* 936* 976*1064
   15 |*1143*1195*1217*1206*1165*1103*1046* 994* 952* 925* 933* 988*1079*1140*1187*1190*1175*1127*1068*1014* 978* 949* 934* 997 ----
   30 |*1158*1204*1217*1198*1153*1092*1032* 982* 943* 922* 939*1010*1100*1160*1191*1195*1165*1113*1052*1003* 971* 944* 943*1020 ----
   45 |*1172*1211*1216*1189*1138*1076*1019* 972* 936* 923* 952*1033*1118*1169*1194*1191*1153*1100*1038* 990* 964* 938* 950*1043 ----

STATION IDENTIFIER = "          1652100"    " Z" X 10** 02   PARAMETER = WATER-SURFACE ELEVATION      DATE = 81/09/23
HOUR |  OO   01   02   03   04   05   06   07   08   09   10   11   12   13   14   15   16   17   18   19   20   21   22   23   24
  MIN +----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+
   00 | ----*1131*1181*1200*1182*1138*1077*1014* 954* 911* 873* 855* 850* 877* 912* 931* 946* 944* 928* 913* 886* 852* 832* 830* 868
   15 |*1084*1144*1190*1200*1172*1124*1060* 997* 941* 900* 867* 853* 856* 886* 910* 936* 945* 930* 925* 907* 877* 845* 830* 834 ----
   30 |*1100*1157*1198*1197*1162*1109*1045* 984* 929* 890* 863* 849* 861* 895* 923* 940* 944* 936* 923* 901* 868* 840* 829* 843 ----
   45 |*1110*1170*1200*1189*1151*1093*1030* 967* 921* 883* 857* 840* 869* 904* 924* 943* 945* 933* 918* 893* 860* 835* 829* 853 ----

STATION IDENTIFIER = "          1652100"    " Z" X 10** 02   PARAMETER = WATER-SURFACE ELEVATION      DATE = 81/09/24
HOUR |  OO   01   02   03   04   05   06   07   08   09   10   11   12   13   14   15   16   17   18   19   20   21   22   23   24
  MIN +----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+
   00 | ----* 940*1019*1066*1091*1086*1040*1001* 947* 898* 855* 822* 804* 812* 845* 903* 966*1013*1029*1017* 978* 930* 882* 837* 820
   15 |* 885* 968*1033*1075*1094*1079*1038* 988* 935* 885* 845* 817* 802* 815* 858* 921* 978*1020*1030*1010* 967* 919* 870* 830 ----
   30 |* 906* 986*1046*1082*1094*1070*1026* 978* 923* 875* 835* 810* 802* 821* 873* 936* 990*1024*1027* 999* 954* 907* 858* 825 ----
   45 |* 928*1003*1056*1087*1091*1060*1014* 961* 910* 865* 828* 808* 806* 831* 888* 954*1003*1027*1024* 990* 942* 895* 847* 822 ----
```

Figure 25. Example DAFIX output—print file showing the four-digit table format.

BACKUP—Maintenance of the TDDB

Use the BACKUP utility to copy all or part of a TDDB to an archive file or to restore all or part of an archive file to a TDDB. The archive file is a sequential file written in either of two formats (binary or text) specific to BACKUP. In most instances, the binary archive format should be used as this format retains data values as stored in the TDDB. When floating-point values are stored in text format, they are rounded to a finite precision (8 significant digits for four-byte values and 13 significant digits for eight-byte values). Binary format is processed in less time than the text format as there is no need for intermediate data conversions. Binary format also requires about one-half the storage space of text format. The binary archive file and the TDDB (it is also a binary file) can be used on different computer systems only if the internal machine-code representation is identical on both systems. This is often not the case. Text archive format allows portability between computer systems and provides a means to examine values in a TDDB using a text editor. Do not edit a binary archive file or a TDDB with a text editor, as this can destroy the integrity of these files. A text archive file can be modified using a text editor only if its format is not changed.

Creation of archive files protects against accidental loss or errant alteration of data in a TDDB. BACKUP also allows for selectively building one TDDB from another, retaining the status of a TDDB at a certain point in time, porting a TDDB to another computer platform, and updating a TDDB created by a different version of the TDDS. BACKUP also is used to “clean up” a TDDB. Clean up of a TDDB is necessary to recover disk space taken up by data deleted from a TDDB using DAFIX or when the number of index entries used is close to the maximum allowed (MAXENT, default=476). When data are deleted, the TDDB index is altered; but the data still occupy space in the TDDB though there is no means to access it. Using BACKUP to create an archive file and then restoring it to a new TDDB removes the unwanted data and frees the used storage space.

BACKUP can be used to save data that are no longer needed in a readily available form. For example, an archive file of data from previous water years can be saved for possible future reference or only data for particular stations can be archived. BACKUP can restore the TDDB from an archival file selectively by date and from a list of station identifiers. The archive file will contain only data sets for the station identifiers specified. Multiple DSR files can be input or station identifiers can be included in the control file (in addition to those specified in the DSR file named in the **TDDB_DSR.MTR** master file).

During a restore operation, BACKUP optionally combines contiguous data sets thereby reducing the number of index entries used. This is useful if there are many small data sets resulting from the filling in of missing data between larger data sets. However, it may be important to be able to differentiate these filled-in data sets from field-recorded data sets. It is not recommended that data sets be combined for this reason. To combine contiguous data sets, BACKUP requires several criteria be satisfied. The data sets must have the same station identifier, data-parameter code, storage-type code, and readings per day; and the total length of the combined data set cannot exceed 90 calendar days. A date and time can be specified to limit consideration for combination of data sets to those with a time span that falls after the specified date and time.

In using BACKUP to restore data, data sets can be stored in an existing or new TDDB. Data sets can be selectively stored in a TDDB by station number and date. Therefore, it is possible to create a new TDDB with only the most recent data and for a subset of stations. If partial data bases are created, be sure to save an archive file of the entire original TDDB until such time as all data are no longer needed.

If data are restored to an existing TDDB (one with data sets), only those data sets in the archive file are restored that do not overlap in time (other data-set attributes being equal) of data sets in the existing TDDB. This is because the restoration of a TDDB archive file is an update process, not an overwrite process. After data have been restored, old TDDB's should be deleted and the archive files transferred to an off-line storage media.

Use of BACKUP should be supplemented by periodically copying the TDDB to magnetic tape, floppy disk, or some other off-line storage medium. For example, the TDDB could be copied to a floppy disk anytime a major change is made.

To move a TDDB between computer systems that are not binary compatible (that is, have different machine-code representations), use BACKUP to create a text archive file of the old TDDB, copy this archive file and DSR file to the target computer, then restore the archive file to a new TDDB on the target computer using its installed TDDS.

Similarly, to transfer a TDDB from one TDDS version to a later version, first select the BACKUP utility from the source TDDS that generated the TDDB to create the archive file (in either binary or text format), then run the BACKUP utility from the target TDDS to create a new TDDB. The format of the TDDB is compatible between TDDS versions except in two instances: if the TDDB was created using a TDDS that referenced stations using the old USGS 8-digit station number format as opposed to the 16-digit station identifiers used now (change occurred 12/90); and if a change was made to the TDDB record length or number of index records (based on record length, number of index entries, and index entry length) from their original values. See the Attribute Assignments section above for information on these TDDB attributes.

Execution Sequence

In general, use of the BACKUP utility requires two executions of BACKUP and possibly one execution of the ALLOCATE utility. The following is a typical execution sequence:

1. BACKUP is used to create an archive file of the TDDB for the desired stations and to optionally combine data sets.
2. Optionally, ALLOCATE is used to initialize an old TDDB.
3. BACKUP is used to restore the data (either all or selected stations and (or) data sets with time spans that fall after a specified time) from the archive file to the TDDB.

Note that BACKUP automatically initiates a TDDB that does not already exist; therefore, the ALLOCATE step is necessary only when an "existing" TDDB needs initialization. Also, note that the target TDDB may contain data sets; in this case the archive file of a TDDB and the target (existing) TDDB are merged.

The following describes the execution sequence for BACKUP. Select BACKUP (code 7) from the TDDS MAIN MENU. The DAOPEN utility is executed to force verification of the file names for the TDDB and DSR files. After verification, a prompt requests the name of the control file. Enter a valid file name or accept the displayed default file name (**backup.rdr**). If the control file exists, the EXECUTE MENU prompt displays with the following options: "BEGIN execution (default)", "LIST the contents of the control file", and "CREATE the control file". If the control file does not exist or "CREATE the control file" is selected, the following menu displays:

```
                    ** BACKUP MENU **

   Code                 Activity
     1  --  Print TDDB directory summary
     2  --  Create BINARY file copy of TDDB
     3  --  Restore TDDB from a archive file created by option #2
     4  --  Create TEXT file copy of TDDB
     5  --  Restore TDDB from a archive file created by option #4

ENTER INTEGER VALUE FOR SELECTION CODE (1-5)
CURRENT VALUE =      2:
```

The prompting sequence with associated activities identified follows:

Code = 1, 2, 3, 4, or 5

1. Select whether the index summary only, or the index summary and calendar-year summary, is to be produced (before execution for options 1, 2, and 4 and after execution for options 3 and 5). Note that option 1 permits generation of directory summaries using multiple DSR files or station identifiers entered in addition to those in the active DSR file. The SUMMARY utility permits generation of directory summaries using only the active DSR file.

Code = 2 or 4—Create an archive file

2. A prompt requests whether data sets should be combined. If they are, enter the date and time after which data sets with a time span after the specified date and time are to be combined, if possible.

Code = 3 or 5—Restore an archive file

2. A prompt requests a date and time limit for restoring data sets. BACKUP restores only those data sets with beginning date and times after this specified date and time. Then the following prompt requests the TDDS version that produced the archive file:

```
                    ** BACKUP VERSION **

   Code                 Activity
     0  --  Current TDDS version (1994 - today)
     1  --  16-digit TDDS version (4/89 - 12/93)
     2  --  8-digit TDDS version (1986 - 3/89)
     3  --  8-digit TDDS version (pre-1985)

ENTER INTEGER VALUE FOR SELECTION CODE (1-4)
CURRENT VALUE =      0:
```

Code = 2, 3, 4 or 5

3. A prompt requests where the valid station identifiers are to be read from. BACKUP only restores data sets specified in a list of station identifiers. This list can consist of (a) the active DSR file only (as named in the **TDDB_DSR.MTR** master file); (b) the active DSR file and other DSR files (file names to be specified); or (c) the active DSR file and station identifiers to be specified.

Code = 1, 2, 3, 4 or 5

4. A prompt requests whether to verify of the contents of the active DSR file. If yes, the ALLOCATE utility is executed in the DSR file update mode. Note that this is the second prompt for code=1.

After completion of an option, the EXECUTE MENU displays. Select the BEGIN option to execute the BACKUP utility. A prompt then requests the file name of the archive file. When BACKUP finishes, a prompt requests whether to list the print file before returning to the TDDS MAIN MENU.

Input

Control file input to BACKUP consists of three types of records. Table 11 gives the format of these records. Other input can be data in a TDDB, DSR, or archive file. Only data for stations as specified in the DSR file or as identified in the control file are archived or restored.

The first input record, List Index record, controls the production of optional summary listings of the TDDB index before and (or) after execution of BACKUP. The function of variables LISTB and LISTA on this record are constrained by the BACKUP option performed. LISTA is valid only when data are restored, whereas LISTB is valid only when creating an archive file.

The second input record, Process Control record, specifies the option to be performed and the processing date and time before which no data are combined (options 2 and 4), or the processing date and time after which data are restored to the TDDB (options 3 and 5). The variable, LPTIME, on this record has a different meaning depending on which BACKUP option is selected. For example, consider a set of data from June 10, 1982, at 13:00 to July 17, 1982, at 11:00 that was stored in the TDDB on August 9, 1982, at 17:22. For option 2 or 4, if LPTIME is specified with a value before 82/08/09 17:22, BACKUP attempts to combine the above-mentioned data set with any preceding and (or) following (in time) sets of data of the same station identifier, data-parameter code, storage type, and recording frequency. For option 3 or 5, if LPTIME is specified with a value before 82/07/17 11:00, BACKUP restores the above-mentioned data set to the new TDDB; otherwise it is not restored.

The third control record, Station record(s), is optional. Station records are used to specify station identifiers, other than those designated in the DSR file, that are to have data archived, restored, or summarized. BACKUP first reads the station numbers out of the DSR file and then from any Station records.

Table 11. Specifications for the BACKUP control file

Variable	Default	Values	Position	Format	Description
List Index record (one required per execution)					
LISTB	1	-1, 0, 1, 2	38-39	I2	Option to list TDDB summaries before task(s) (-1 = list both calendar-year and index summary; 0 = no summaries; 1 = list only index summary; 2 = list valid parameter and storage-type codes with definitions and readings per day).
LISTA	1	-1, 0, 1, 2	46-47	I2	Option to list TDDB summaries after task(s) (-1 = list both calendar-year and index summary; 0 = no summaries; 1 = list only index summary; 2 = list valid parameter and storage-type codes with definitions and readings per day).
Process Control record (one required per execution)					
PROCESS	1	1, 2, 3	10-11	I2	Option to be performed (1 = directory summaries of TDDB index for list of station identifiers; 2 = create binary archive file of a TDDB; 3 = restore data sets in a binary archive file to a TDDB; 4 = create text archive file of a TDDB; 5 = restore data sets in a text archive file to a TDDB).
IOLD	0	0, 1, 2, 3	12-13	I2	Flag to indicate the version of TDDS that created the archive file (0 = 1/94 to present; 1 = 16-digit version 4/89 to 12/93; 2 = 8-digit version 1/85 to 3/89; 3 = 8-digit version pre-1985).
LPTIME	00/01/01 01:00	-------	29-42	5(I2,1X)	Date and time (YR/MO/DY HR:MN). For option 2 or 4: only data sets stored in the TDDB after this date and time will be considered for combining with contiguous data sets. For option 3 or 5: only data sets with data after this date and time will be restored.
Station record(s) (optional, up to MAXFLD-number of stations in DSR file)					
FLDSTA	blank	-------	2-17	A16	Station identifier.

Output

The nature of results generated by BACKUP depends upon the option selected. The primary output for options 2 and 4 is an archive file (default file name of BACKUP.TDD), whereas for options 3 and 5 an archive file is restored to a TDDB. The print file consists of selected TDDB directory summaries, and depending upon the option performed, a table of station numbers, a table of data sets combined, and error and warning messages if problems are encountered during execution.

Examples

The input records required to execute BACKUP to create a sequential, binary copy of a TDDB are shown in figure 26. The output print file from this execution (not shown) includes an introductory page giving the date and time of execution and the BACKUP option performed, a list of the stations identifiers for which data were archived, a TDDB index summary before execution of BACKUP, and a summary of the options performed by BACKUP. If data sets are combined or not stored in an archive file (station identifier not in DSR file or not specified on Station records), an index summary of those data sets combined and excluded is produced. In this example, no data sets were combined or not included in the archive file.

```
                                        1       0
2                    00/01/01 01:00
```

Figure 26. Example BACKUP input—control file to create binary archive file.

Figure 27 shows the input records to create a TDDB with data recorded after 80/01/01 01:00 and found in the archive file created in the previous example. The output from this execution (not shown) includes an introductory output page giving the date and time of execution, a summary of TDDB attributes since a new TDDB was created, a list of station identifiers, a summary of data sets not restored to the new TDDB, and an index summary of the new TDDB.

```
                                        0       1
 3 0                 80/01/01 01:00
```

Figure 27. Example BACKUP input—control file to restore TDDB from a binary archive file.

SUMMARY—Checking the Status of Data in a TDDB

Use the SUMMARY utility to determine the availability and status of data in a TDDB for those stations identified in the DSR file. The SUMMARY utility provides two complementary means for reviewing the contents of a TDDB, a calendar-year summary, and an index summary. A third "conventions" summary gives the valid data-parameter codes, storage-type codes, and readings per day of data that can be stored in a TDDB. The TDDS utilities BACKUP, DAFILE, DAFIX, DAGET, and DAPLOT can produce summaries before and (or) after performing their specific processing task. Thus, data status can be checked along with other data-processing tasks. For example, an index summary can be produced by DAFIX both before and after a data set is deleted to visually verify that the task was performed successfully. SUMMARY simply provides for obtaining these summaries without performing any other task.

Execution Sequence

The following describes the execution sequence for the SUMMARY utility. From the TDDS MAIN MENU select SUMMARY (code 8). At the prompt for the file name of the control file, enter a valid file name or accept the displayed default file name (**summary.rdr**). If the control file exists, the EXECUTE MENU prompt displays with the following options: "BEGIN execution (default)", "LIST the contents of the control file", and "CREATE the control file". If the control file does not exist or "CREATE the control file" is selected, the following menu displays:

```
                    **SUMMARY MENU**

   Code                Activity
     0  --  Index summary (default)
     1  --  Index summary and calendar-year summary
     2  --  Valid parameters, storage types, and readings per day
Enter 0, 1, or 2:
```

After selecting a SUMMARY option, the EXECUTE MENU displays. Select the BEGIN option to execute the SUMMARY utility. When SUMMARY is finished executing, a prompt requests whether to list the print file before returning to the TDDS MAIN MENU.

Input

The control file for SUMMARY consists of a single data record. Table 12 gives the format of this record.

Table 12. Specifications for the SUMMARY control file

Variable	Default	Values	Position	Format	Description
LIST	1	-1, 0, 1, 2	38-39	I2	Option to list TDDB summaries (-1 = list both calendar-year and index summary; 0 = no summaries; 1 = list only index summary; 2 = list valid parameter and storage-type codes with definitions and readings per day).

Output

Output from SUMMARY consists of a print file, default file name of **summary.prt**, containing either an index summary, an index summary and a calendar-year summary, or a conventions summary. Two-byte, integer water-surface elevation data sets that contain flagged values, such as values updated using DAFIX, are identified in the calendar-year and index summaries. The index summary is a table, 114 characters wide, showing the attributes of each data set. Data sets are sorted alphabetically first by station identifier, then by data-parameter code, storage-type code, and finally by beginning date and time. Figure 28 shows the index summary produced for a sample TDDB. The following attributes are given in the table for each data set:

1. Data-set sequence number.
2. Station identifier.
3. Data-parameter code (for example: Z=water-surface elevation and Q=cross-sectional discharge).
4. Starting date and time (year, month, day, hour, minute).
5. Ending date and time (year, month, day, hour, minute).
6. Number of readings per day.
7. Data-set size (number of readings).
8. Storage-type code (I2, I4, R4, or R8).
9. Indicator as to whether or not the data set includes flagged data.
10. Date and time that the data set was entered into the TDDB (year, month, day, hour, minute).
11. File address of the initial value of the data set in terms of record number and byte number of that record.

The calendar-year summary is a compact printout, 132 characters wide, showing data availability in a given calendar year by day for each particular station identifier, data-parameter code, and storage-type code contained in the TDDB. The calendar-year summary is in a grid format with a box allotted for each day in the year. Coded characters in each box identify the recording interval and completeness of the data stored for that

day, as well as whether the data for that day include flagged values. The calendar-year summaries are sorted by the order of station identifiers in the DSR file, then by calendar year, data-parameter code, and finally by storage-type code. Figure 29 is one page of a calendar-year summary.

The conventions summary identifies the valid storage-type codes, readings per day, and data-parameter codes with definitions that are allowed to be stored in a TDDB. Figure 30 shows an example of a conventions summary.

INDEX SUMMARY OF DATA SETS STORED IN THE TDDB
PROCESSED BY DADIO - VERSION: 5.6 1996/06/03
PROCESSING DATE AND TIME 96/06/03 12:34

DATA SET	STATION IDENTIFIER	PARM CODE	START OF ENTRY YR MO DY HR MN	END OF ENTRY YR MO DY HR MN	RDS/ DAY	NO OF RDS	STOR TYPE	DATA FLAG	PROCESSING TIME YR MO DY HR MN	FILE RECORD	ADDRESS BYTE
1	1646500	Q	80/08/04 06:00	80/08/08 19:00	96	437	I4	NO	96/06/03 12:34	5	1
2	1646500	Q	80/11/21 01:00	80/12/10 24:00	24	480	I4	NO	96/06/03 12:34	5	1749
3	1646500	Q	81/07/20 06:00	81/08/04 06:00	24	361	I4	NO	96/06/03 12:34	5	3669
4	1646500	Q	81/08/14 24:00	81/09/13 24:00	24	721	I4	NO	96/06/03 12:34	5	5113
5	1646500	Q	81/09/23 00:30	81/09/23 24:00	96	95	I4	NO	96/06/03 12:34	6	1765
6	1646500	Z	86/06/01 00:15	86/06/03 02:00	96	200	I2	NO	96/06/03 12:34	6	2145
7	1646500	Z	86/06/01 00:15	86/06/03 02:00	96	200	R4	NO	96/06/03 12:34	6	2545
8	1647600	Z	81/08/19 00:15	81/09/21 16:15	96	3233	I2	NO	96/06/03 12:34	6	3345
9	1647600	Z	81/09/22 09:30	81/10/01 24:00	96	923	I2	YES	96/06/03 12:34	7	3579
10	1652100	Z	81/09/19 00:15	81/10/01 24:00	96	1248	I2	YES	96/06/03 12:34	7	5425
11	1652588	Z	80/08/04 06:00	80/08/08 19:00	96	437	I2	NO	96/06/03 12:34	8	1689
12	1652588	Z	81/08/19 00:15	81/10/01 24:00	96	4224	I2	NO	96/06/03 12:34	8	2563
13	1655480	Z	80/08/04 06:00	80/08/08 19:00	96	437	I2	NO	96/06/03 12:34	9	4779
14	1655480	Z	80/11/21 01:00	80/12/10 24:00	96	1917	I2	NO	96/06/03 12:34	9	5653
15	1655480	Z	81/06/04 00:15	81/07/09 24:00	96	3456	I2	NO	96/06/03 12:34	10	3255
16	1655480	Z	81/07/19 18:00	81/08/04 06:00	96	1489	I2	NO	96/06/03 12:34	11	3935
17	1655480	Z	81/08/04 12:15	81/10/03 12:00	96	5760	I2	NO	96/06/03 12:34	12	681
18	1655480	WD	81/07/21 01:00	81/07/28 10:30	48	356	I4	NO	96/06/03 12:34	13	5969
19	1655480	WS	81/07/21 01:00	81/07/28 10:30	48	356	I4	NO	96/06/03 12:34	14	1161
20	1658710	Z	81/08/19 00:15	81/10/01 24:00	96	4224	I2	NO	96/06/03 12:34	14	2585
21	1660800	Z	81/05/25 00:15	81/07/06 24:00	96	4128	I2	NO	96/06/03 12:34	15	4801
22	1661475	Z	81/08/19 00:15	81/10/01 24:00	96	4224	I2	NO	96/06/03 12:34	17	593
23	1661590	Z	81/08/19 00:15	81/10/01 24:00	96	4224	I2	NO	96/06/03 12:34	18	2809
24	Computed 1646500	A	80/11/21 01:00	80/12/10 24:00	96	1917	I4	NO	96/06/03 12:34	19	5025
25	Computed 1646500	B	80/11/21 01:00	80/12/10 24:00	96	1917	I4	NO	96/06/03 12:34	21	229
26	Computed 1646500	Q	80/11/21 01:00	80/12/10 24:00	96	1917	I4	NO	96/06/03 12:34	22	1665
27	Computed 1646500	Z	80/11/21 01:00	80/12/10 24:00	96	1917	I2	NO	96/06/03 12:34	23	3101
28	Computed 1652588	A	80/11/21 01:00	80/12/10 24:00	96	1917	I4	NO	96/06/03 12:34	24	705
29	Computed 1652588	B	80/11/21 01:00	80/12/10 24:00	96	1917	I4	NO	96/06/03 12:34	25	2141
30	Computed 1652588	Q	80/11/21 01:00	80/12/10 24:00	96	1917	I4	NO	96/06/03 12:34	26	3577
31	Computed 1652588	Z	80/11/21 01:00	80/12/10 24:00	96	1917	I2	NO	96/06/03 12:34	27	5013
32	Computed 1655480	A	80/11/21 01:00	80/12/10 24:00	96	1917	I4	NO	96/06/03 12:34	28	2617
33	Computed 1655480	B	80/11/21 01:00	80/12/10 24:00	96	1917	I4	NO	96/06/03 12:34	29	4053
34	Computed 1655480	Q	80/11/21 01:00	80/12/10 24:00	96	1917	I4	NO	96/06/03 12:34	30	5489
35	Computed 1655480	Z	80/11/21 01:00	80/12/10 24:00	96	1917	I2	NO	96/06/03 12:34	32	693
36	Observed 1660800	Q	81/05/05 19:50	81/05/06 09:20	288	163	I4	NO	96/06/03 12:34	32	4527
37	Observed 1660800	Q	84/05/07 08:00	84/05/07 19:00	288	133	I4	NO	96/06/03 12:34	32	5179
38	Observed 1660800	Q	84/06/28 07:45	84/06/28 19:30	288	142	I4	NO	96/06/03 12:34	32	5713
39	Observed 1660800	Q	84/10/16 06:40	84/10/16 18:00	288	137	I4	NO	96/06/03 12:34	33	49

DADIO RETURN CODE = 0

Figure 28. Example SUMMARY output—print file showing index summary.

CALENDAR-YEAR SUMMARY OF DATA STORED IN THE TDDB
PROCESSED BY DADIO - VERSION: 5.6 1996/06/03
PROCESSING DATE AND TIME 96/06/03 12:34

WATER-SURFACE ELEVATION (I2)

CALENDAR YEAR - 1981 STATION ID. 1655480

MONTH	DAY 1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
JAN.																															
FEB.																													-	-	-
MARCH																															
APRIL																															-
MAY																															
JUNE				U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	-
JULY	U	U	U	U	U	U	U	U	U										*	U	U	U	U	U	U	U	U	U	U	U	U
AUG.	U	U	U	*	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U
SEPT.	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	-
OCT.	U	U	*																												
NOV.																															-
DEC.																															

CODES

* = PARTIAL RECORD; % = PARTIAL RECORD,CONTAINS FLAGGED DATA; # = DUAL DENSITY RECORD; $ = COMPLETE RECORD,CONTAINS FLAGGED DATA
P = 1440 READINGS/DAY; Q = 720 READINGS/DAY; R = 288 READINGS/DAY; S = 240 READINGS/DAY; T = 144 READINGS/DAY;
U = 96 READINGS/DAY; V = 48 READINGS/DAY; W = 24 READINGS/DAY; X = 12 READINGS/DAY; Y = 8 READINGS/DAY; Z = 1 READINGS/DAY

Figure 29. Example SUMMARY output—print file showing calendar-year summary.

```
      Valid values associated with data stored in a TDDB

 Storage-types:
    Code        Description
     I2     two-byte integer
     I4     four-byte integer
     R4     four-byte floating-point
     R8     eight-byte floating-point

 Recording densities (readings per day):
     1  8  12  24  48  96  144  240  288  720 1440

Data-parameters:
   Parameter       Description           Parameter       Description
      A      CROSS-SECTIONAL AREA            B      CROSS-SECTIONAL WIDTH
      C      CONDUCTIVITY                    D      DENSITY
      P      PRECIPITATION                   Q      CROSS-SECTIONAL DISCHARGE
      S      SALINITY                        T      WATER TEMPERATURE
      V      CROSS-SECTIONAL VELOCITY        Z      WATER-SURFACE ELEVATION
     AC      CONVEYANCE AREA                AP      ATMOSPHERIC PRESSURE
     AR      ATMOSPHERIC RADIATION          AS      STORAGE AREA
     AT      AIR TEMPERATURE                BC      CONVEYANCE WIDTH
     BS      STORAGE WIDTH                  C0      DISSOLVED CONCENTRATION
     C1      DISSOLVED CONCENTRATION        C2      DISSOLVED CONCENTRATION
     C3      DISSOLVED CONCENTRATION        C4      DISSOLVED CONCENTRATION
     C5      DISSOLVED CONCENTRATION        C6      DISSOLVED CONCENTRATION
     C7      DISSOLVED CONCENTRATION        C8      DISSOLVED CONCENTRATION
     C9      DISSOLVED CONCENTRATION        DO      DISSOLVED OXYGEN
     GU      MAX. WIND-GUST VELOCITY-U      GV      MAX. WIND-GUST VELOCITY-V
     K0      SUSPENDED CONCENTRATION        K1      SUSPENDED CONCENTRATION
     K2      SUSPENDED CONCENTRATION        K3      SUSPENDED CONCENTRATION
     K4      SUSPENDED CONCENTRATION        K5      SUSPENDED CONCENTRATION
     K6      SUSPENDED CONCENTRATION        K7      SUSPENDED CONCENTRATION
     K8      SUSPENDED CONCENTRATION        K9      SUSPENDED CONCENTRATION
     L0      LATERAL CONCENTRATION          L1      LATERAL CONCENTRATION
     L2      LATERAL CONCENTRATION          L3      LATERAL CONCENTRATION
     L4      LATERAL CONCENTRATION          L5      LATERAL CONCENTRATION
     L6      LATERAL CONCENTRATION          L7      LATERAL CONCENTRATION
     L8      LATERAL CONCENTRATION          L9      LATERAL CONCENTRATION
     M0      MEASURED CONCENTRATION         M1      MEASURED CONCENTRATION
     M2      MEASURED CONCENTRATION         M3      MEASURED CONCENTRATION
     M4      MEASURED CONCENTRATION         M5      MEASURED CONCENTRATION
     M6      MEASURED CONCENTRATION         M7      MEASURED CONCENTRATION
     M8      MEASURED CONCENTRATION         M9      MEASURED CONCENTRATION
     MU      MEAN VELOCITY-U                MV      MEAN VELOCITY-V
     MW      MEAN VELOCITY-W                PH      pH
     PU      POINT VELOCITY-U               PV      POINT VELOCITY-V
     PW      POINT VELOCITY-W               QL      LATERAL DISCHARGE
     QT      TRIBUTARY DISCHARGE            R0      RADIATION
     R1      RADIATION                      R2      RADIATION
     R3      RADIATION                      R4      RADIATION
     R5      RADIATION                      R6      RADIATION
     R7      RADIATION                      R8      RADIATION
     R9      RADIATION                      RH      RELATIVE HUMIDITY
     SR      SOLAR RADIATION                T0      TRIBUTARY CONCENTRATION
     T1      TRIBUTARY CONCENTRATION        T2      TRIBUTARY CONCENTRATION
     T3      TRIBUTARY CONCENTRATION        T4      TRIBUTARY CONCENTRATION
     T5      TRIBUTARY CONCENTRATION        T6      TRIBUTARY CONCENTRATION
     T7      TRIBUTARY CONCENTRATION        T8      TRIBUTARY CONCENTRATION
     T9      TRIBUTARY CONCENTRATION        UL      LATERAL FLOW VELOCITY
     UX      TRANSVERSE VELOCITY            VY      LONGITUDINAL VELOCITY
     WD      WIND DIRECTION                 WP      WETTED PERIMETER
     WS      WIND SPEED                     WU      WIND VELOCITY-U
     WV      WIND VELOCITY-V                WZ      VERTICAL VELOCITY


 DADIO RETURN CODE =   0
```

Figure 30. Example SUMMARY output—print file showing conventions summary.

PROGRAMMING INSTRUCTIONS FOR DATA STORAGE AND RETRIEVAL

Four storage and retrieval routines, the Direct-Access Data Input/Output routines (DADIO, DADI, DADO, and DADSUM), provide the interface between the TDDB and the TDDS, hydrodynamic/transport simulation models, and other application programs. No other means of direct access to the TDDB is provided because data are maintained in the TDDB in a binary format unique to these Fortran routines. Models or other application programs can easily gain access to the TDDB through proper use of these four routines.

DADIO

To access a TDDB, programs must initially call the DADIO routine. After that, calls can be made to any of the four routines as necessary. DADIO initializes important variables associated with the TDDB and DSR files and reads and sorts the TDDB index and DSR file. Arguments to DADIO define Fortran file unit numbers for the print file and the unit numbers and file names for the TDDB and DSR files. Another argument controls the production of tabular summaries of data in the TDDB. Three types of summaries are available: a calendar-year summary, formatted to show the status of data on a daily basis for each station; an index summary of the TDDB data sets; and a data conventions summary. Table 13 outlines the argument list required to invoke DADIO. Figures 28, 29, and 30, in the SUMMARY section above, show examples of these summaries.

Table 13. Argument list for the DADIO routine

Argument	Type	Status	Description
LUPRT[1]	Integer	Input	Fortran file-unit number for print file (normally 66).
LUDSR[1]	Integer	Input	Fortran file-unit number for DSR file (normally 8).
LUTDDB[1]	Integer	Input	Fortran file-unit number for TDDB (normally 98).
LISTN	Integer	Input	Option to list the TDDB summaries (-1 = print both calendar-year summary and index summary; 0 = do not print summaries; 1 = print only index summary; 2 = list valid parameter and storage-type codes with definitions and readings per day).
DSRNAM	Character	Input	File name of DSR file to open.
TDDBNM	Character	Input	File name of TDDB to open.
IRETCD[2]	Integer	Output	Data-status return code (ranges from -14 to 16).

[1]Must be specified as positive, nonzero integer values.

[2]The value for IRETCD is assigned by subroutine DADIO. Return codes are defined in table 16.

EXAMPLE: CALL DADIO (LUPRT, LUDSR, LUTDDB, LISTN, IRETCD)

DADI and DADO

DADI (Direct-Access Data Input) and DADO (Direct-Access Data Output) are called by the applicable program to store and retrieve data in a TDDB, respectively. Concurrent data storage and retrieval are possible within the same program. DADI restricts the size of a single data set stored to a time span of 90 days. DADO does not limit the length of the time sequence of data retrieved, as it can retrieve multiple data sets if they are contiguous (no missing values) and contain data at the same frequency. Attempts to store data sets coincident in time, in whole or in part, with data already in a TDDB are not processed unless at least one of the following data attributes is different: the station identifier, data-parameter code, storage-type code, or recording density. For example, the same data can be stored, each in its own data set, as four-byte integer (I4) values and as four-byte floating-point (R4) values.

DADI and DADO have identical argument lists (defined in table 14). The first 14 arguments define the data-set attributes (station identifier, data-parameter code, storage-type code, beginning and ending date and time, and readings per day). The 15th argument (IDATA) passes the address of the first data value of the time sequence. The 16th argument (ISTRIP) is a flag used by DADO only, which specifies whether to strip status flags previously assigned to individual two-byte, integer water-surface elevation values by DAFIX. Messages, defining warning conditions encountered by DADI or DADO, are optionally written to the print file according to the 17th argument (PRTMSG). (Messages are always written for critical errors, such as those assigned a negative return code; these critical errors require corrective action before attempting to rerun the applicable program.) The last argument (IRETCD) is the return code assigned by DADI or DADO. A return code value of zero indicates no problems were encountered during the execution. See table 16 for a complete list of return codes and their meaning.

Time sequences of data are passed to and from a TDDB in a two-byte integer (Integer*2) array passed as the 15th argument to DADI or DADO. If four-byte integer (Integer*4) or four- or eight-byte real (Real*4 or Double Precision) data are to be stored or retrieved, arrays of these storage types also can be passed to DADI or DADO because Fortran arguments are passed by address only. Some Fortran compilers issue subroutine argument mismatch messages if other than an Integer*2 array is passed to DADI or DADO. A remedy for this problem is to equivalence an Integer*2 array to the Integer*4, Real*4, or Double Precision array (dimensioned appropriately) that contains or will contain the data values using a Fortran equivalence statement. Then, pass the Integer*2 array as the 15th argument and set the storage-type code to the storage type of the original data, not I2. Equivalence means that the two arrays overlap (normally at the first position of the arrays) at the same location in memory (address). For consistency, it is recommended to always pass an Integer*2 array.

DADSUM

Use the DADSUM (Direct-Access Data SUMmary) routine to generate summaries of data in a TDDB. Three types of summaries are available: a calendar-year summary, formatted to show the status of data on a daily basis for each station, an index summary of the TDDB data sets, and a data conventions summary. These summaries are written to the file associated with the unit LUPRT as passed to the DADIO routine. Table 15 outlines the argument list required to invoke DADSUM. Figures 28, 29, and 30, in the SUMMARY section above, show examples of the summaries.

Table 14. Argument list for the DADI and DADO routines

Argument	Type	Status	Description
STAID[1]	Character*16	Input	Station identifier associated with the data set.
DTYPE[2]	Character*2	Input	Data-parameter code (for example, Z = water-surface elevation).
STYPE	Character*2	Input	Storage-type code of data (I2, I4, R4, or R8).
IBYR	Integer	Input	Beginning year of time span (0 to 99).
IBMO	Integer	Input	Beginning month of time span (1 to 12).
IBDY	Integer	Input	Beginning day of time span (1 to 31).
IBHR	Integer	Input	Beginning hour of time span (0 to 24).
IBMN	Integer	Input	Beginning minute of time span (0 to 59).
IEYR	Integer	Input	Ending year of time span (0 to 99).
IEMO	Integer	Input	Ending month of time span (1 to 12).
IEDY	Integer	Input	Ending day of time span (1 to 31).
IEHR	Integer	Input	Ending hour of time span (0 to 24).
IEMN	Integer	Input	Ending minute of time span (0 to 59).
RDPDY	Integer	Input	Number of readings per day (1, 8, 12, 24, 48, 96, 144, 240, 288, 720, or 1440).
IDATA[3]	Integer*2[4]	Input or Output	Array of time-dependent data (input for DADI and output for DADO).
ISTRIP[5]	Integer	Input	Option to indicate how to handle data flags assigned to two-byte, integer water-surface elevation values (-1 = strip flags, 0 = do not strip flags, 1 = strip update flags (22000)—do not strip other flags (11000, -11000, -22000).
PRTMSG	Logical	Input	Option to print warning messages (True = print; False = do not print).
IRETCD[6]	Integer	Output	Data-status return code (ranges from -14 to 14).

[1]Station identifiers are interpreted as right justified on the first nonblank character in the 16-digit field and left filled with blanks. Other embedded blanks are significant.

[2]Data-parameter codes are defined in table 2.

[3]Values must be assigned before calling DADI. Values will be returned upon calling DADO. DATA must be dimensioned large enough to hold the complete data set (number of values equals ending date-time minus beginning date-time, in minutes, divided by the recording interval in minutes, plus one).

[4]If four-byte integer data are to be stored or retrieved, an Integer*4 array is passed to DADI or DADO. On some computers an Integer*4 array that is equivalenced to an Integer*2 array (dimensioned appropriately) must be passed to avoid subroutine argument interface checks.

[5]Data-flags are defined in table 4.

[6]The value for IRETCD is assigned by DADI or DADO. Return codes are defined in table 16.

EXAMPLE: CALL DADI (STAID, DTYPE, STYPE, IBYR, IBMO, IBDY, IBHR, IBMN, IEYR, IEMO, IEDY, IEHR, IEMN, RDPDY, IDATA, ISTRIP, PRTMSG, IRETCD)

Table 15. Argument list for the DADSUM routine

Argument	Type	Status	Description
LISTN	Integer	Input	Option to list the TDDB summaries (-1 = print both calendar-year summary and index summary; 0 = do not print summaries; 1 = print only index summary; 2 = list valid parameter and storage-type codes with definitions and readings per day).
IRETCD	Integer	Output	Data-status return code (ranges from -14 to 14). See table 16 for definition of all DADIO return codes.

EXAMPLE: CALL DADSUM (LISTN, IRETCD)

DADIO Return Codes

A value indicating the success or failure of the desired operation is returned to the calling program in the last argument (IRETCD) by the four DADIO routines. Table 16 defines the error or warning conditions that correspond to each DADIO return code.

Table 16. DADIO return codes

Code	Description
-14	No data sets selected based on stations identified in the DSR file, TDDB not empty, perhaps wrong DSR file specified.
-13	Error opening the DSR file; a valid file must exist before accessing the TDDB.
-12	Error opening the TDDB; check for correct file name.
-11	Error writing the TDDB index.
-10	Error reading the TDDB index; possible record length mismatch.
-9	Invalid number of readings per day encountered for a data set; possibly invalid program version.
-8	The DSR file is empty or a null file.
-7	Fortran file unit number for TDDB equals unit number for print or DSR file; check values of LUTDDB, LUDSR, and LUPRT arguments to DADIO.
-6	TDDB index has not been read; be sure the DADIO routine is called before DADI or DADO.
-5	TDDB index is empty; be sure the DADIO routine is called before DADI or DADO or check assigned file names.
-4	Maximum number of station identifiers in DSR file exceeded, check MAXFLD parameter in dimpar.cmn include file.
-3	System supplied date and (or) time is errant.
-2	Maximum number of years (25) of record allowed in the TDDB has been exceeded; create a new TDDB or increase parameter MAXYRS in dimpar.cmn include file and compile and load a new version of the TDDS.
-1	TDDB index records are full, delete unwanted data sets or reconstruct the TDDB using the BACKUP utility. If a sufficient number of data sets were not combined or deleted to allow continued use of the TDDB, you should create a backup copy of the TDDB, modify the value of parameter MAXENT of dimpar.cmn include file, compile and load a new version of the TDDS, and then restore the data base using the TDDS built using an increased MAXENT value.
0	No error encountered.
1	Number of readings per day is invalid, valid values are: 1, 8, 12, 24, 48, 96, 144, 240, 288, 720, or 1440.
2	Invalid station identifier, not found in DSR file.
3	Invalid date and (or) time specified.
4	Requested data contain previously flagged values (data flags 11000, -11000, or -22000).
5	Invalid data-parameter code (see table 2 for list of defined parameters).
6	Number of readings per day specified does not match the value for the data stored in the TDDB.
7	No data available for time span of request.
8	Data previously stored for specified time span.
9	Maximum number of days (90) of record allowed per data set exceeded by time span; split storage of data into multiple storage requests.
10	Requested data contains previously flagged values (update flag 22000 assigned by DAFIX).
11	Insufficient data available for time span specified.
12	Return codes 10 and 11.
13	Return codes 4 and 11.
14	Specified time is not an even increment of recording interval.
15	Invalid storage-type code specified, valid values are: I2, I4, R4, or R8.
16	Return codes 5 and 15.
17	Return codes 4 and 10.
18	Return codes 4, 10, and 11.

UTILIZATION OF A TDDB

Several operational simulation models store and retrieve time sequences of data directly or indirectly from a TDDB. One-dimensional models typically store and retrieve data directly through calls to the DADIO routines. Two- and three-dimensional simulation models frequently require that time sequences be retrieved and preprocessed by a separate input data processor. In these cases, the preprocessor program calls the DADIO routines or reads a data file created by the TDDS. Simulated results are analyzed using the TDDS or separate postprocessor programs. Whichever method is used, access to data in a TDDB is easily incorporated into models or application programs using the DADIO, DADI, DADO, and DADSUM routines.

Two operational simulation models have been linked to the TDDS directly. The branch-network, unsteady-flow model (BRANCH) (Schaffranek and others, 1981) is a one-dimensional, numerical model for simulating unsteady, singular riverine and estuarine reaches, and networks of reaches composed of interconnected channels. The TDDS has been used in support of several modeling studies together with the BRANCH model (see for example: Holtschlag, 1981; Stedfast, 1982; Lipscomb, 1989; Schaffranek, 1989; Goodwin, 1991; Bower and others, 1993). The BRANCH model can also store computed water-surface elevation, discharge, cross-sectional area, and conveyance width at selected cross sections in a TDDB.The Branched Lagrangian Transport Model (BLTM) (Schoellhamer and Jobson, 1986a,b; Jobson and Schoellhamer, 1987) is a one-dimensional model for simulating the fate of water-quality constituents in open-channel networks. The BLTM model also has an imbedded TDDB interface for direct storage and retrieval of time-sequences of data.

The DAGET utility allows indirect transfer of time sequences of data from a TDDB to models or other data-analysis programs. DAGET creates a text file of time sequences of data in several predefined or user-specified formats. Therefore, any one-, two-, or three-dimensional flow and (or) transport model, or data-analysis program, that accepts formatted input, can retrieve data indirectly from the TDDB. DAGET produces data files formatted for input to the BRANCH model and to the input data processor of the SWIFT2D model (Leendertse, 1987). SWIFT2D is a two-dimensional model to simulate the hydrodynamics, transport, and water-quality conditions in well-mixed water bodies, such as estuaries, coastal seas, harbors, lakes, rivers, and inland waterways. DAGET also formats time sequences of data in the USGS WATSTORE daily- and unit-values formats.

REFERENCES

Bower, D.E., Sanders, C.L., and Conrads, P.A., 1993, Retention time simulation for Bushy Park Reservoir near Charleston, South Carolina: U.S. Geological Survey Water-Resources Investigations Report 93-4079, 47 p.

Dempster, G.R., 1990, National Water Information System user's manual, Volume 2, Chapter 3. Automated data processing system: U.S. Geological Survey Open-File Report 90-116, 330 p.

Goodwin, C.R., 1991, Simulation of the effects of proposed tide gates on circulation, flushing, and water quality in residential canals, Cape Coral, Florida: U.S. Geological Survey Open-File Report 91-237, 43 p.

Holtschlag, D.J., 1981, Flow model of Saginaw River near Saginaw, Michigan: U.S. Geological Survey Open-File Report 81-1061, 21 p.

Hutchison, N.E., compiler, 1975, WATSTORE--National Water Data Storage and Retrieval System of the U.S. Geological Survey: U.S. Geological Survey Open-File Report 75-426, 532 p.

Hutchison, N.E., Stuthmann, N.G., Merk, C.F., and Isherwood, W.L., Jr., 1977, WATSTORE--National Water Data Storage and Retrieval System--user's guide--digital data systems, v. 5, chap. I-IV: U.S. Geological Survey Open-File Report 75-729-I.

Jobson, H.E., and Schoellhamer, D.H., 1987, Revised 1993, Users manual for a branched Lagrangian transport model: U.S. Geological Survey Water-Resources Investigations Report 87-4163, 80 p.

Lai, C., Schaffranek, R.W., and Baltzer, R.A., 1978, An operational system for implementing simulation models, a case study: American Society of Civil Engineers Seminar on Computational Hydraulics at the 26th Annual Specialty Conference of the Hydraulics Division, University of Maryland, College Park, Md., p. 415-454.

Leendertse, J.J., 1987, Aspects of SIMSYS2D: A system for two-dimensional flow simulation, Report No. R-3572-USGS: Santa Monica, Calif., The RAND Corporation, 80 p.

Lipscomb, S.W., 1989, Flow and hydraulic characteristics of the Knik-Matanuska River Estuary, Cook Inlet, Southcentral Alaska: U.S. Geological Survey Water-Resources Investigations Report 89-4064, 52 p.

Schaffranek, R.W., 1989, Proceedings of the advanced seminar on one-dimensional, open-channel flow and transport modeling: U.S. Geological Survey Water-Resources Investigations Report 89-4061, 99 p.

Schaffranek, R.W., and Baltzer, R.A., 1978, Fulfilling model time-dependent data requirements: American Society of Civil Engineers Symposium on Technical, Environmental, Socioeconomic, and Regulatory Aspects of Coastal Zone Management, "Coastal Zone '78", v. III, p. 2069-2084.

Schaffranek, R.W., Baltzer, R.A., and Goldberg, D.E., 1981, A model for simulation of flow in singular and interconnected channels: U.S. Geological Survey Techniques of Water-Resources Investigations, book 7, chap. C3, 110 p.

Schoellhamer, D.H., and Jobson, H.E., 1986a, Programmers manual for a one-dimensional Lagrangian transport model: U.S. Geological Survey Water-Resources Investigations Report 86-4144, 101 p.

Schoellhamer, D.H., and Jobson, H.E., 1986b, Users manual for a one-dimensional Lagrangian transport model: U.S. Geological Survey Water-Resources Investigations Report 86-4145, 95 p.

Stedfast, D.A., 1982, Flow model of the Hudson River estuary from Albany to New Hamburg, New York: U.S. Geological Survey Water-Resources Investigations Report 81-55, 69 p.

APPENDIXES

APPENDIX A—GLOSSARY OF TERMINOLOGY AND DEFINITIONS

Technical terms common to the field of computational hydraulics and computer applications, as well as terms specific to the TDDS, are used throughout this report. It is the intent of the glossary to define these terms, to clarify their usage.

Address

The designation by name, number, or some other reference of an exact location in storage. The address of a TDDB data set is the physical location of its initial value as defined by record number and character number (byte offset) within the record.

ADR (Analog-Digital Recorder)

Equipment to convert analog signals to digital values and record these values on a paper or magnetic tape. Data are recorded at a fixed-time interval that is determined by an external timer.

ALLOCATE

TDDS utility to create and initialize a computer file for use as a TDDB and to create and modify the DSR file.

Alphanumeric

The letters, numbers, and symbols that form a character set and can be interpreted by a computer.

ASCII (American National Standard Code for Information Interchange)

A standardized coding scheme that uses numeric values to represent the letters, numbers, and symbols of a character set. The predominant resident character set on computer systems.

BACKUP

TDDS utility to create a copy of the TDDB for archival purposes and to restore the TDDB from a backup file.

Binary

Binary is used synonymously with machine code to mean the computer's internal representation of data, which is normally a sequence of binary digits.

Bit

A binary digit.

Boundary Conditions

A requirement that the dependent variable of a differential equation must satisfy along a boundary of the solution domain.

Boundary-Value Data

Specified values for a dependent variable at given values of the independent variable(s).

BRANCH

Branch-network, unsteady-flow model as documented in Schaffranek and others (1981).

Break Key

Keyboard sequence used to halt execution of a process and return control to the operating system of the computer. This sequence may be one or more of the following: press the Break key on the keyboard, or hold the control key down and press the letter "p" (designated as ^p), ^c, or ^d.

Byte

A group of bits (usually eight) representing a unit of storage.

CalComp Graphics Library

The set of Fortran-callable routines (PLOTS, PLOT, LINE, AXIS, SCALE, SYMBOL, NUMBER, FACTOR, WHERE, NEWPEN) that provide the basic functions of graphics generation, graphics device initialization, such as line drawing, axis generation, data scaling, and text annotation. This functional library was developed by California Computer Products, Inc.

Card-Image File

A computer file containing values of variables or data stored as a sequence of "card-images" or records, each having a maximum of 80 characters (columns). Access to these records is achieved using sequential read and write commands. The term card-image file has its origin in the early era of computer development when binary and character data were punched on 80-column paper cards.

Carriage Return

Keyboard sequence used to send to the CPU the data entered using the keyboard since the previous carriage return. This sequence refers to the key labeled with the word Enter or Return and (or) with a bent left arrow (↵); and is designated as **<cr>** in this report.

Character

A sequence of one or more bytes representing a single symbol, such as a member of the set of alphanumeric symbols derived from the character set of the host computer.

Compile

The process of translating software from its programming language to machine language instructions.

CPU (Central Processing Unit)

The portion of a computer that performs the instructions requested in the software.

DADI (Direct-Access Data Input)

Fortran routine called to store data into a TDDB.

DADIO (Direct-Access Data Input/Output)

Fortran routine that must first be called by any program that needs access to data in a TDDB. This routine reads the TDDB index, assigns important system parameters, and produces summary listings.

DADO (Direct-Access Data Output)

Fortran routine called to retrieve data from a TDDB.

DAFILE (DAta FILEing)

TDDS utility to store data into the TDDB. Data may be entered in a variety of formats, including WATSTORE unit-values, user-specified, and interactive session.

DAFIX (DAta FIXing)

TDDS utility to modify or delete data already stored in the TDDB. DAFIX can apply shift corrections and produce concise printed listings of selected data in the TDDB.

DAGET (DAta GET)

TDDS utility to retrieve data from the TDDB and output text files of those data. Data files can be produced in a variety of formats, including WATSTORE daily- and unit-values, BRANCH and SWIFT2d boundary-condition, and user-specified formats.

DAPLOT (DAta PLOTting)

TDDS utility to plot data stored in the TDDB on either a line printer or digital plotter.

Data Base

Information (data) stored in a common format and location that can be read directly by a computer program.

Data File

A collection of information systematically stored on computer storage media in a form that can be read directly by a computer. The typical media are magnetic disks or tapes; but other storage media, including optical disks and CD-ROM, are available.

Data Flag

Numeric offset added to two-byte, integer water-surface elevation values to indicate a particular condition about a data value. Four data flags (11000 = rate of change exceeded, 22000 = updated value as assigned by DAFIX, -11000 = repeat count exceeded, and -22000 = value is out of range) are recognized by the TDDS.

Data Set

A collection of logically associated data values, related by a fixed set of criterion and dimensions, possibly with one dimension allowed to vary. An example of a data set would be a time series of water-level data recorded at a particular geographic location at a fixed-time interval by an automatic data recorder.

Data-Station Reference (DSR) File

A card-image file that controls access to the TDDB. This file uniquely identifies those stations for which data have been stored in the TDDB. No data for a given station location will be written to or read from the TDDB unless the station identifier is correctly identified in the DSR file.

Datum

Any numerical value or geometric quantity (such as a point, line, surface, or horizontal plane) that serves as a base reference for other quantities or values (such as bathymetric soundings, ground elevations, and water-surface elevations).

Default Value

The value assigned by a computer program for a variable not otherwise specified by the user (left blank or only a carriage return is entered in response to a prompt).

Direct Access

A type of storage device, such as a file, which enables information at various locations to be retrieved without processing intermediate information.

Directory

A type of file used to group other files, which may be of multiple file types. No two entries in the same directory can have the same name.

Discharge (Q)

The volumetric rate of water passing through a cross section of a channel.

DOS (Disk Operating System)

A single user, single task operating system predominantly used for personal computers.

File

A named collection of information treated as a unit that can be written to and (or) read from by a computer program. A file has certain attributes, including access permissions (read, write, execute), type (random, sequential, terminal, directory), and format (binary, text, raster, vector).

File Name

A name, consisting of between one byte and the maximum number of bytes allowed by the operating system, associated with a file. For maximum portability, file names should consist only of the characters A-Z, a-z, 0-9, ., _, and -, and begin with an alphabetic letter.

Fortran (FORmula TRANslation)

A programming language designed to be like algebra and employed in the efficient management of scientific applications. It was developed in the mid-1950's by a group at IBM headed by John Backus. Its name is derived from for(mula) tran(slation). Current and past versions of Fortran include IV, 66, 77, and 90.

GKS (Graphical Kernel System)

The American National Standards Institute (ANSI) and International Standards Organization (ISO) standard for two-dimensional graphics functionality. It is described in Computer Graphics—Graphical Kernel System (GKS) Function Description, ANSI X3.124, 1985.

Hardware

Those devices, including electronic, magnetic, and mechanical, which are components of a computer.

Hydrograph

A graphical representation of flow characteristics, such as stage, discharge, and velocity, at a given point as a function of time.

Index

The first few records of the TDDB; used to uniquely identify each data set within the TDDB.

Index Entry

A unit of the TDDB index uniquely identifying a particular data set. Each entry identifies the station identifier, beginning and ending date and time, number of values per day, parameter code, data status, processing date and time, and file address of one data set in the TDDB.

Initial Condition

A set of values that prescribes the state of a dynamic system at some specified time; for all subsequent times, the mathematical equations describing the system and boundary conditions determine its state.

Initial-Value Data

In initial-value problems, if numerical data are given for certain values of the independent variable as the initial conditions, such data are referred to as initial-value data.

Input/Output (I/O)

Those devices and procedures used to communicate with the computer.

Machine Code

The computer's internal representation of data.

Machine Language

The set of instructions a computer can execute.

Master File

See TDDB_DSR.MTR.

Mathematical Model

A model comprised of an abstract mathematical system of equations that describes the interrelation of components of the process to be simulated. There are three major types of mathematical models: (1) Analytical model—relies on analytical solution methods; (2) Numerical model—relies on numerical solution methods; and (3) Amalgamative model—combines the above two types.

Model

A mathematical (equations, data, and assumptions) or physical entity, obeying certain specified conditions, physical laws, or mathematical principles, whose behavior is used to understand a physical, biological, or social system to which it is analogous in some fashion.

Operating System

Software residing on a particular computer hardware platform that controls all parts of a computer system, such as the execution of computer programs, scheduling of events, control of devices, data management, and related services.

Program

A set of instructions in a computer programming language that performs a certain function.

Program-Control File

A text file that contains the input values of variables required to direct the data processing task of a software product, coded to meet the input data specifications of the software.

Random Access

A data storage and retrieval technique in which information can be accessed directly at any location in the memory or file regardless of its sequential or physical position.

Software

Programmer-written instructions and directives required to achieve solutions using a computer.

Source Code

A computer program written in a programming language for input to a compiler or interpreter.

Stage (Z)

The height (elevation) of the water surface relative to a fixed datum or plane of reference.

SUMMARY

TDDS utility used to determine the availability and status of data in the TDDB for all stations defined in the DSR file. It provides two complementary means for reviewing the contents of a TDDB, calendar-year summary and directory summary.

SWIFT2D (Surface-Water Integrated Flow and Transport 2-Dimensional model)

A two-dimensional, finite-difference, vertically-integrated model for simulating the hydrodynamics, transport, and water-quality conditions in well-mixed water bodies.

System Prompt

The characters displayed by the operating system to indicate that the computer is ready to accept information.

TDDB

See Time-Dependent Data Base.

TDDB_DSR.MTR

A text file that identifies by name the TDDB and DSR files used during the most recent or current execution of the TDDS program in the current directory. Also referred to as the 'master file'. A master file is automatically generated if it does not exist.

TDDS

See Time-Dependent Data System.

Text File

A computer file in which data (text and (or) values) are stored as a sequence of records in the default character set of the host computer, normally ASCII or EBCDIC. Because most computers use the ASCII character set, text files are often referred to as ASCII files. Text files can be listed or modified using any text editor.

Time-Dependent Data (Time-Series Data)

A contiguous, time-sequence of individual values of a particular parameter recorded at a preselected time interval and at a fixed geographic location.

Time-Dependent Data Base (TDDB)

An indexed, direct-access data base that contains time-dependent data constituting field-recorded values or model-generated results. The storage format of the TDDB enables any data set stored in it to be readily available to numerical simulation models or other application programs for further analyses and processing. This report documents the process of storing data into, and retrieving data from, a TDDB.

Time-Dependent Data System (TDDS)

An efficient, easy-to-use system of computer programs that performs all necessary processing functions for the compilation and storage of time-dependent data.

Water Level

See stage.

Water-Surface Elevation (Z)

See stage.

WATSTORE (National **Wat**er Data **Stor**age and **Re**trieval System)

A computer system developed by the U.S. Geological Survey to store and disseminate water data acquired through its many activities. It was first implemented in November 1971.

Word

A group of characters or bits (often 32 bits) treated as a unit and can be stored in one location.

UNIX

A multiuser, multitasking operating system used on a wide variety of computers.

APPENDIX B—EXAMPLE INTERACTIVE SESSION OF THE TDDS

The following is a listing of an interactive terminal session to assign the names of the Time-Dependent Data Base (TDDB) and DSR file, create a DSR file with a USGS 8-digit station number and two 16-digit latitude and longitude identifiers, and store two data sets in a new TDDB. The files (dataset.1 and dataset.2) containing the data to be stored in the TDDB were created using a text editor. Each file contains 4 days of hourly water-level data, with each record in the files consisting of eight values coded such that there is one value every eight positions across the record. The following conventions are used in the example:

- User responses are in bold-face type.
- The symbol **<cr>** after a user response stands for a carriage return (enter key).
- A **<cr>** by itself denotes that the default value for a prompt was accepted.
- Commentaries (lines that begin with ###) are also included in the examples for information purposes only; these lines would not appear on the display.

```
 Welcome to the Time-Dependent Data System - Version: 5.2 1996/06/03

 To accept the displayed or default value given in a prompt, press
 enter.  The default response to a "yes/no" query is yes.  In
 response to TDDS prompts, enter "x" to exit the TDDS or "q" to
 return to the main menu.

 ENTER THE FILE NAME OF THE MASTER FILE
 (NEW) TDDB_DSR.MTR:
appendix.mtr<cr>
```

Because the master file did not exist before this execution of TDDS, DAOPEN is automatically

executed to assign the file names for the TDDB and DSR file.

```
                    ** DAOPEN MENU **

 The file names assigned in the master file are:

   Time-Dependent Data Base    : TDDB
   Data-Station Reference file : DSRFILE

    Code               Activity
      0  --  ACCEPT file names as displayed (default)
      1  --  CHANGE the file name for the TDDB
      2  --  CHANGE the file name for the DSR file
 Enter 0, 1, or 2:
1<cr>

 ENTER THE file name OF THE TDDB
 (NEW) TDDB:
appendix.tdd<cr>

 SPECIFIED FILE DOES NOT EXIST YET, IS THAT OK [Y,n]?
y<cr>
```

```
                    ** DAOPEN MENU **

 The file names assigned in the master file are:

   Time-Dependent Data Base    : potomac.tdd
   Data-Station Reference file : DSRFILE

    Code               Activity
      0  --  ACCEPT file names as displayed (default)
      1  --  CHANGE the file name for the TDDB
      2  --  CHANGE the file name for the DSR file
 Enter 0, 1, or 2:
2<cr>

 ENTER THE file name OF THE DSR FILE
 (NEW) DSRFILE:
appendix.dsr<cr>

 SPECIFIED FILE DOES NOT EXIST YET, IS THAT OK [Y,n]?
y<cr>

                    ** DAOPEN MENU **

 The file names assigned in the master file are:

   Time-Dependent Data Base    : potomac.tdd
   Data-Station Reference file : potomac.dsr

    Code               Activity
      0  --  ACCEPT file names as displayed (default)
      1  --  CHANGE the file name for the TDDB
      2  --  CHANGE the file name for the DSR file
 Enter 0, 1, or 2:
<cr>

 These files are used with subsequent executions of the TDDS unless
 you update the master file.
```

Because the DSR file did not exist prior to this execution, ALLOCATE executes automatically to
create and add station identifiers to the named DSR file.

```
 The DSR file named in the master file does not exist or is empty.
 It must exist to continue.  Enter station identifiers to be stored
 in the DSR file or quit.

                    ** DSR FILE UPDATE MENU **
 Prompts are issued to update the values stored in the DSR file.
 These values define the valid station identifiers and reference
 adjustments for data sets stored in the TDDB.  Use the following
 options to update the DSR file.

    Code               Activity
      S  --  SHOW a table of the DSR file contents.
      C  --  CHANGE the values of a table entry, enter its number
              (as shown in the table) and then the new values.
      D  --  DELETE an entry from the DSR file.
      A  --  ADD a new entry to the DSR file.
      F  --  FINISHED (saves changes).
      ?  --  DISPLAY this message.

 ENTER VALUE FOR SELECTION CODE (S, C, D, A, F, OR ?)
 CURRENT VALUE = "A":
<cr>

 When finished adding stations, enter "stop" to the query.
```

```
 Station identifiers can be any, up to 16-digit sequence of characters.
 TDDS converts leading zeros to blanks and removes trailing blanks from
 identifiers (i.e. right-justifies). A recommended form for identifiers is
 a Latitude and Longitude with sequence number. For identifiers of this type,
 enter a "$" when queried, prompts will then be issued for the three components.

 ENTER VALUE FOR THE STATION IDENTIFIER, $, OR ?
 CURRENT VALUE = "        11447500":
```
1646500<cr>

Note that the reference adjustment (datum correction as applied to water-level data) is entered as a
floating-point number but is stored in the DSR file as the integer value of the adjustment value
multiplied by one hundred. For example, if -8.23 is entered, it is stored in the DSR file as -823.

```
 ENTER FLOATING-POINT VALUE FOR THE REFERENCE ADJUSTMENT TO BE APPLIED TO OUTPUT
 CURRENT VALUE =        0.000:
```
<cr>

```
 When finished adding stations, enter "stop" to the query.

 ENTER VALUE FOR THE STATION IDENTIFIER, $, OR ?
 CURRENT VALUE = "         1646500":
```
$<cr>

```
 ENTER INTEGER VALUE FOR LATITUDE (UP TO 7 DIGITS, DDMMSS, I.E. 0380730)
 CURRENT VALUE = 380730:
```
384940<cr>

```
 ENTER INTEGER VALUE FOR LONGITUDE (UP TO 7 DIGITS, DDMMSS, I.E. 0771530)
 CURRENT VALUE = 771530:
```
770149<cr>

```
 ENTER INTEGER VALUE FOR TWO-DIGIT STATION SEQUENCE NUMBER
 CURRENT VALUE =      0:
```
1<cr>

```
 ENTER FLOATING-POINT VALUE FOR THE REFERENCE ADJUSTMENT TO BE APPLIED TO OUTPUT
 CURRENT VALUE =        0.000:
```
<cr>

```
 When finished adding stations, enter "stop" to the query.

 ENTER VALUE FOR THE STATION IDENTIFIER, $, OR ?
 CURRENT VALUE = " 384940077014901":
```
384320077020100<cr>

```
 ENTER FLOATING-POINT VALUE FOR THE REFERENCE ADJUSTMENT TO BE APPLIED TO OUTPUT
 CURRENT VALUE =        0.000:
```
<cr>

```
 When finished adding stations, enter "stop" to the query.

 ENTER VALUE FOR THE STATION IDENTIFIER, $, OR ?
 CURRENT VALUE = " 384320077020100":
```
stop<cr>

```
 ENTER VALUE FOR SELECTION CODE (S, C, D, A, F, OR ?)
 CURRENT VALUE = "F":
```
s<cr>

```
 Num     Identifier       Datum   Num     Identifier       Datum
   1          1646500     0.00     2   384940077014901     0.00
   3   384320077020100     0.00

 ENTER VALUE FOR SELECTION CODE (S, C, D, A, F, OR ?)
 CURRENT VALUE = "F":
```
<cr>

```
                              TDDS MAIN MENU
 ======================================================================
 | Code   Utility                  Function                            |
 | ----   -------                  --------                            |
 |  1     DAOPEN     Assign Time-Dependent Data Base (TDDB) and        |
 |                     Data-Station Reference (DSR) file names         |
 |  2     ALLOCATE   Initialize a TDDB or Create or Edit DSR file      |
 |  3     DAFILE     Store data in a TDDB                              |
 |  4     DAGET      Retrieve data and output in WATSTORE daily- or    |
 |                     unit-values, BRANCH instream-input, SWIFT_IDP   |
 |                     input, or user-specified format                 |
 |  5     DAPLOT     Plot data on digital or line-printer devices      |
 |  6     DAFIX      Modify, delete, and print data                    |
 |  7     BACKUP     Create backup file of TDDB or re-create a TDDB    |
 |                     from a backup file                              |
 |  8     SUMMARY    Print directory summaries of a TDDB               |
 ======================================================================

 ENTER INTEGER VALUE FOR SELECTION CODE (1-8)
 CURRENT VALUE =      1:
3<cr>
```

Enter a carriage return to accept the displayed default name. Enter another name if desired;
for example, enter a new name so as not to overwrite an existing file. (Note that output files are
automatically overwritten without warning if a file exists of the same name as generated by the
TDDS based on the control file name.)

```
 A control file is required to define the TDDS task.

 ENTER THE file name OF THE CONTROL FILE
 (NEW) dafile.rdr:
<cr>

 It will be created interactively.

 Printed output can be found in dafile.prt

 Option to list TDDB index summaries BEFORE the TDDS task:

    Code               Activity
      0  --  Do not print summaries (default)
      1  --  Print only the directory summary
      2  --  Print directory and calendar-year summaries
 Enter 0, 1, or 2:
<cr>

 Option to list TDDB index summaries AFTER the TDDS task:

    Code               Activity
      0  --  Do not print summaries (default)
      1  --  Print only the directory summary
      2  --  Print directory and calendar-year summaries
 Enter 0, 1, or 2:
1<cr>

 Before proceeding, you must have your data prepared to enter.  This can be
 either manually or from a disk file.  If not please quit.

                   ** DAFILE MENU **

 Code            Data Set Characteristics
   1  --  Data values to be entered interactively
   2  --  Free-formatted (values separated by space or comma)
   3  --  8 values/record with 1 value every 10 characters
   4  --  Created in a user-specified format
   5  --  Created previously by the DAGET utility
   6  --  WATSTORE unit-values format (Z,Q,DO,T,WS,WD,C data only)
   7  --  Finished creating control file (default)
```

```
 ENTER INTEGER VALUE FOR SELECTION CODE (1-7)
 CURRENT VALUE =      7:
4<cr>

 You specified that the data set is contained in a file in the
 specified format, if not please quit.

 ENTER VALUE FOR THE STATION IDENTIFIER, $, OR ?
 CURRENT VALUE = " 384320077020100":
384940077014901<cr>

 ENTER THE PARAMETER CODE (Q, Z, A, B, WS, WD, ETC)
 CURRENT VALUE: " Z"
<cr>

 ENTER VALUE FOR THE STORAGE-TYPE CODE (I2, I4, R4, or R8)
 CURRENT VALUE = "I4":
r4<cr>

 ENTER VALUE FOR THE BEGINNING DATE AND TIME OF REQUEST [YR/MO/DY HR:MN]
 CURRENT VALUE = "87/11/28 18:00":
79/09/11 09:00<cr>

 ENTER VALUE FOR THE ENDING DATE AND TIME OF REQUEST [YR/MO/DY HR:MN]
 CURRENT VALUE = "87/11/28 19:00":
79/09/15 08:00<cr>

 ENTER INTEGER VALUE FOR NUMBER OF READINGS PER DAY
 CURRENT VALUE =     96:
24<cr>

 ENTER FLOATING-POINT VALUE FOR THE REFERENCE ADJUSTMENT TO BE APPLIED TO OUTPUT
 CURRENT VALUE =        0.000:
<cr>

 SELECT a data format using the code numbers from the table below
 or ENTER a valid Fortran format specification (including
 parenthesis).  Interpret the formats below using the following
 examples: (10I6) means 10 integer values per record with each
 value right justified every 6 record positions (columns);
 (10F8.2) means 10 decimal values per record with a value
 specified every 8 columns (on input, if the decimal point is not
 specified for a value, it is assumed to be right justified with
 two implied decimal places, that is, "    12345 " is read as
 1234.50); (I6,4X,I6) means two integer values per record, the
 first right justified in column 6 and the second in column 16;
 (38X,6F7.2) means 6 floating-point values per record specified
 every 7 columns beginning at column 39 (positions 1-38 are
 ignored); (5D12.5) means 5 double precision values specified
 every 12 columns.

 PRESS ENTER TO CONTINUE:
<cr>

 Code     Format
  ----     ------
   0  --  (I5) (default)
   1  --  (10I6)
   2  --  (8I10)
   3  --  (10F8.2)
   4  --  (F10.2)
   5  --  (38X,6F7.2)
   6  --  (5D12.5)
   7  --  (4D20.5)

 ENTER VALUE FOR A SELECTION CODE OR FORTRAN FORMAT SPECIFICATION
 CURRENT VALUE = "(8F10.2)                                          ":
(8F8.2)<cr>
```

```
 The data set may contain records preceding the time-series data.  DAFILE
 reads these records and uses them for annotation purposes.  Each time-series
 data record must be identically formatted and be contiguous.  Any records
 input following the data records are ignored.

 ENTER INTEGER VALUE FOR NUMBER OF RECORDS THAT PRECEDE THE FIRST DATA RECORD
 CURRENT VALUE =      0:
2<cr>

 ENTER THE file name OF THE FILE CONTAINING THE DATA TO BE STORED
 (NEW) DAFILE.DAT:
dataset.1<cr>

                   ** DAFILE MENU **

 Code            Data Set Characteristics
   1  --  Data values to be entered interactively
   2  --  Free-formatted (values separated by space or comma)
   3  --  8 values/record with 1 value every 10 characters
   4  --  Created in a user-specified format
   5  --  Created previously by the DAGET program
   6  --  Finished creating control file (default)

 ENTER INTEGER VALUE FOR SELECTION CODE (1-6)
 CURRENT VALUE =      6:
4<cr>

 You specified that the data set is contained in a file in the
 specified format, if not please quit.

 ENTER VALUE FOR THE STATION IDENTIFIER, $, OR ?
 CURRENT VALUE = " 384940077014901":
384320077020100<cr>

 ENTER THE PARAMETER CODE (Q, Z, A, B, WS, WD, ETC)
 CURRENT VALUE: " Z"
<cr>

 ENTER VALUE FOR THE STORAGE-TYPE CODE (I2, I4, R4, or R8)
 CURRENT VALUE = "R4":
<cr>

 ENTER VALUE FOR THE BEGINNING DATE AND TIME OF REQUEST [YR/MO/DY HR:MN]
 CURRENT VALUE = "79/09/11 09:00":
<cr>

 ENTER VALUE FOR THE ENDING DATE AND TIME OF REQUEST [YR/MO/DY HR:MN]
 CURRENT VALUE = "79/09/15 08:00":
<cr>

 ENTER INTEGER VALUE FOR NUMBER OF READINGS PER DAY
 CURRENT VALUE =     24:
<cr>

 ENTER FLOATING-POINT VALUE FOR THE REFERENCE ADJUSTMENT TO BE APPLIED TO OUTPUT
 CURRENT VALUE =        0.000:
<cr>

 Code     Format
  ----     ------
   0  --  (I5) (default)
   1  --  (10I6)
   2  --  (8I10)
   3  --  (10F8.2)
   4  --  (F10.2)
   5  --  (38X,6F7.2)
   6  --  (5D12.5)
   7  --  (4D20.5)

 ENTER VALUE FOR A SELECTION CODE OR FORTRAN FORMAT SPECIFICATION
 CURRENT VALUE = "(8F8.2)                                          ":
(8F10.2)<cr>
```

```
 The data set may contain records preceding the time-series data.  DAFILE
 reads these records and uses them for annotation purposes.  Each time-series
 data record must be identically formatted and be contiguous.  Any records
 input following the data records are ignored.

 ENTER INTEGER VALUE FOR NUMBER OF RECORDS THAT PRECEDE THE FIRST DATA RECORD
 CURRENT VALUE =      2:
<cr>

 ENTER THE file name OF THE FILE CONTAINING THE DATA TO BE STORED
 (NEW) DAFILE.DAT:
dataset.2<cr>

                    ** DAFILE MENU **

 Code            Data Set Characteristics
   1  --  Data values to be entered interactively
   2  --  Free-formatted (values separated by space or comma)
   3  --  8 values/record with 1 value every 10 characters
   4  --  Created in a user-specified format
   5  --  Created previously by the DAGET program
   6  --  Finished creating control file (default)

 ENTER INTEGER VALUE FOR SELECTION CODE (1-6)
 CURRENT VALUE =      6:
<cr>

                    ** EXECUTE MENU **

    Code                Activity
      0  --  BEGIN execution (default)
      1  --  LIST the contents of the control file
      2  --  CREATE the control file
 Enter 0, 1, or 2:
1<cr>

 This is the contents of the control file.

                                      0       1
  384940077014901 FROM  79/09/11 09:00  TO  79/09/15 08:00RD=  24 TP= Z ST=R4
          REAL REFADJ=   0.000 NCOM= 2 CKTIM= 1 FMT= (8F8.2)
 STATION = Rte. 1 Bridge  BEGIN DATE 79/09/11 09:00  END DATE 79/09/15 08:00
 DATA TYPE = STAGE  24 VALUES RECORDED PER DAY
   13.50   13.45   13.40   13.35   13.32   13.32   13.32   13.30
   13.30   13.30   13.32   13.32   13.30   13.32   13.30   13.30
   13.30   13.30   13.30   13.30   13.20   13.15   13.10   13.05
   13.00   12.93   12.90   12.85   12.83   12.81   12.78   12.75
   12.71   12.68   12.68   12.66   12.62   12.58   12.57   12.55
   12.53   12.50   12.50   12.50   12.50   12.49   12.49   12.49
   12.50   12.50   12.50   12.50   12.50   12.50   12.45   12.45
   12.40   12.45   12.45   12.45   12.45   12.50   12.50   12.50
   12.60   12.60   12.60   12.60   12.65   12.65   12.64   12.62
   12.60   12.72   12.83   13.05   13.41   13.97   14.32   14.52
   14.68   14.46   14.55   14.52   14.55   14.54   14.50   14.52
   14.53   14.76   14.07   14.16   14.42   14.62   14.76   15.88
  384320077020100 FROM  79/09/11 09:00  TO  79/09/15 08:00RD=  24 TP= Z ST=R4
          REAL REFADJ=   0.000 NCOM= 2 CKTIM= 1 FMT= (8F10.2)
 STATION # = 03216600  BEGIN DATE 79/09/11 09:00  END DATE 79/09/15 08:00
 DATA TYPE = STAGE  24 VALUES RECORDED PER DAY
     15.40     15.24     15.15     15.08     15.04     15.00     14.99     14.90
     14.98     14.98     14.90     14.99     14.90     15.01     14.98     14.98
     14.98     14.97     14.97     14.91     14.71     14.60     14.56     14.30
     14.30     14.20     14.15     14.09     14.06     14.05     14.04     14.03
     13.99     13.96     13.96     13.95     13.91     13.85     13.82     13.80
     13.80     13.80     13.78     13.82     13.75     13.75     13.75     13.74
     13.76     13.70     13.77     13.77     13.85     13.76     13.65     13.40
     13.56     13.56     13.56     13.56     13.55     13.54     13.54     13.60
     13.40     13.61     13.61     13.66     13.66     13.66     13.71     13.66
     13.66     13.96     14.10     14.27     14.43     14.64     14.90     15.22
     15.54     15.78     16.01     16.03     16.09     16.13     16.13     16.13
     16.30     16.58     16.77     17.04     17.38     17.63     17.79     17.92
```

```
                  ** EXECUTE MENU **

   Code              Activity
     0  --  BEGIN execution (default)
     1  --  LIST the contents of the control file
     2  --  CREATE the control file
 Enter 0, 1, or 2:
<cr>

 Please wait while your request is processed.


 ****   STORAGE operation completed ****

 DO YOU WANT TO LIST THE OUTPUT PRINT FILE [Y,n]?
y<cr>

 ENTER INTEGER VALUE FOR DISPLAY WIDTH (UP TO 132 CHARACTERS)
 CURRENT VALUE =     80:
<cr>


 A TDDB was created or initialized by DADIO - VERSION: 5.6 1996/06/03
 with the following attributes as defined in include file dimpar.cmn:

 MAXENT =   476 Maximum number of index entries
 MAXFLD =   150 Maximum number of station in DSR file
 MAXDYS =    90 Maximum number of days per data set
 MAXYRS =    25 Maximum number of years per TDDB
 LENREC =  6232 Length of each record in the TDDB, in bytes
 MAXREC =  6232 TDDB record length based on Fortran I/O
 LENIND =    52 Length of each index entry, in bytes


            NEW TIME-DEPENDENT DATA SETS STORED IN THE DIRECT-ACCESS FILE
                  PROCESSED BY DAFILE, VERSION: 5.1 1996/02/15
                                ********
                           PROCESSING DATE : 96/06/03
                                ********

          STATION = Rte. 1 Bridge  BEGIN DATE 79/09/11 09:00  END DATE 79/09/15
          DATA TYPE = STAGE  24 VALUES RECORDED PER DAY


 More:
<cr>


                ==========================================
                           DATA RECORD DESCRIPTION
                ==========================================
                DATA FROM REQUEST NO.                    1
                NO. OF READINGS PER DAY                 24
                STATION ID NUMBER           384940077014901
                DATA-PARAMETER CODE                      Z
                STORAGE-TYPE CODE                       R4
                BEGIN DATE AND TIME          79/09/11 09:00
                END DATE AND TIME            79/09/15 08:00
                REFERENCE ADJUSTMENT APPLIED          0.00
                DATA SET SIZE                           96
                ==========================================

 ***DATA SET STORED IN TDDB***
          STATION # = 03216600  BEGIN DATE 79/09/11 09:00  END DATE 79/09/15 08:
          DATA TYPE = STAGE  24 VALUES RECORDED PER DAY


 More:
<cr>
```

```
                  ========================================
                              DATA RECORD DESCRIPTION
                  ========================================
                  DATA FROM REQUEST NO.                    2
                  NO. OF READINGS PER DAY                 24
                  STATION ID NUMBER           384320077020100
                  DATA-PARAMETER CODE                      Z
                  STORAGE-TYPE CODE                       R4
                  BEGIN DATE AND TIME          79/09/11 09:00
                  END DATE AND TIME            79/09/15 08:00
                  REFERENCE ADJUSTMENT APPLIED           0.00
                  DATA SET SIZE                           96
                  ========================================

 ***DATA SET STORED IN TDDB***
1
                                  INDEX SUMMARY OF DATA SETS STORED IN THE TDDB
                                  PROCESSED BY DADIO - VERSION: 5.6 1996/06/03
                                     PROCESSING DATE AND TIME 96/06/03 12:34

 ================================================================================

 More:
  DATA|     STATION    |PARM| START OF ENTRY |  END OF ENTRY  |RDS/|NO OF|STOR|D
  SET |   IDENTIFIER   |CODE| YR MO DY HR MN | YR MO DY HR MN | DAY| RDS |TYPE|F
 ================================================================================
    1 | 384320077020100|  Z | 79/09/11 09:00 | 79/09/15 08:00 |  24|   96| R4 |
    2 | 384940077014901|  Z | 79/09/11 09:00 | 79/09/15 08:00 |  24|   96| R4 |
 ================================================================================


 PRESS ENTER TO CONTINUE:
x<cr>
 eXit to operating system requested
```

APPENDIX C—USE OF TDDS ON A UNIX-BASED COMPUTER

Instructions for obtaining, extracting, compiling, installing, running, and testing the TDDS software on a UNIX-based computer follow.

Getting the Latest Version of TDDS

The TDDS program and documentation are accessible on the World Wide Web (WWW) using Mosaic or similar WWW reader, from the USGS Water Resources Information home page (http://h2o.usgs.gov/), and by anonymous FTP from h2o.usgs.gov (130.11.50.175) in the directory /pub/software/general/tdds/UNIX. Compressed tar files in this directory, named tdds_5.2.OS.tar.gz, contain all the files needed to install and test TDDS on a computer with a particular operating system, where OS is a string indicating the operating system the distribution is intended for. If a version is not available for your operating system, the file tdds_5.2.source.tar.gz contains the TDDS source code and other files needed to compile, install, and test the software on a UNIX-based computer. To build a TDDS executable you also need the DADIO, interactive prompting, and device independent graphics libraries. These are contained in the LIBUTL software distribution that is accessible similar to the TDDS software. The LIBUTL distribution is available in the directory /pub/software/general/libutl/UNIX. The following list gives the commands entered to retrieve the TDDS software compiled for a USGS Data General AViiON workstation using anonymous FTP:

```
ftp h2o.er.usgs.gov                  ; connect to file server
anonymous                            ; enter “anonymous” to userid prompt
your_userid                          ; enter your userid to password prompt
cd pub/software/general/tdds/UNIX    ; change to directory containing files
type binary                          ; set binary transfer mode
get tdds_5.0.DGUX.tar.gz             ; retrieve compressed tar file of the software
bye                                  ; logoff
```

Extracting Files

For either type of TDDS distribution that you have acquired, tdds_5.2.OS.tar.gz or tdds_5.2.source.tar.gz, the directory tdds_5.2 is created (or overwritten) when the files are extracted from the tar file. If the tdds_5.2 directory already exists, you may want to delete or rename it before extracting the files. The following are the steps to extract the files from a distribution tar file.

1. If the tar file is not already in the directory under which you want the distribution installed, move it there. For example:

   mv tdds_5.2.____.tar.gz /usr/opt/wrdapp

2. If you are not in the directory where the tar file is located, go there. For example:

   cd /usr/opt/wrdapp

3. Uncompress the distribution file. For example:

   gunzip tdds_5.2.____.tar.gz

4. Extract the distribution files from the tar file. For example:

   tar -xof tdds_5.2.___.tar

This creates the following directory structure (the contents of each directory are shown to the right):

```
tdds_5.2            ; copy of this README file
   `-----bin        ; compiled executable
   `-----doc        ; documentation files
   `-----src        ; Makefile and source code
   `-----test       ; scripts to run verification tests
   `-----data       ; standard data sets used in verification tests
   ‘-----examples   ; data sets used in figures of the TDDS
                      documentation
```

Notes:

- The bin directory is not included in the tdds_5.2.source.tar.gz distribution (it is created during compilation).
- Source code is included only with the tdds_5.2.source.tar.gz distribution.
- It is recommended that no user files be kept in the tdds_5.2 directory structure. If you do plan to put files in the tdds_5.2 directory structure, do so only by creating subdirectories of tdds_5.2.

Compiling

If a compiled version of the software is not available for your computer, or if you want to build the executable yourself, follow the instructions in this section. If you have retrieved a precompiled distribution of the software, skip to the Installing section below.

The source code is provided in the tdds_5.2.source.tar.gz distribution so that users can generate the executable themselves. No support can be provided for users generating their own versions of the software. In general, the requirements are a Fortran compiler, utility library libutl_5.6 (which includes screen prompting, device independent graphics, and Time-Dependent Data Base (TDDB) access routines), GKS or CalComp graphics software, and a minimal level of knowledge of the compiler and the UNIX operating system. As provided, the Makefile and source code are set up for use on Data General AViiON workstations running the DG/UX operating system.

The TDDS program graphics are written using CalComp calls, the LIBUTL library includes routines that translate CalComp calls to GKS calls. If a CalComp library is available, TDDS could easily be modified to use it, rather than the GKS library.

To generate a new executable, do the following:

1. Change directory to the source directory:

   cd tdds_5.2/src

2. Modify the beginning of the file named Makefile to correctly specify system-dependent variables:

LibDir	Top-level directory for LIBUTL software
GraphLib	Graphics libraries
F77	Fortran compiler name
FFLAGS	Fortran compiler flags

3. Use the make program to initiate compilation of the source code and installation of the software:

   make [BINDIR=directory_for_links]

See the Installing instructions below for an explanation of BINDIR.

The make will:

- ❑ create the directories tdds_5.2/bin and BINDIR if they do not already exist,
- ❑ compile the source code,
- ❑ place the executable (tdds) in tdds_5.2/bin, and
- ❑ place a link to the executable in BINDIR if specified.

Installing

To make the executable (tdds) easy to use, it should be installed in a directory included in the user's search path. The Makefile (input instructions to the UNIX make program—located in tdds_5.2/src) contains instructions to optionally place a link in a specified directory to the executable contained in tdds_5.2/bin. Use the following two commands to do this:

```
cd tdds_5.2/src

make install [BINDIR=directory_for_links]
```

If BINDIR is specified, a link to the executable is placed in the specified directory. For example, if your search path consists of:

```
/usr/bin:/usr/opt/bin:/usr/local/bin
```

use the command:

```
make install BINDIR=/usr/local/bin
```

to make the executable accessible from any directory without requiring the full pathname of the software's location.

Notes:

- ❑ Brackets "[xxx]" are used to indicate optional arguments to commands.
- ❑ To create and delete a link to the TDDS executable file, the installer must have sufficient rights in the directory that BINDIR is set to.

Running the Software

After TDDS is properly installed in a directory that is included in your PATH, the program is initiated using the command: tdds. TDDS is an interactive prompting program. The data base and DSR files that were last used with TDDS in the current directory are retained in a "master" file for subsequent use. Input files may be precoded or created interactively during a TDDS session.

NOTE: Versions of the software other than Data General have been linked to XGKS (a public domain Graphical Kernel System (GKS) software library—XGKS (C) Copyright 1993 UCAR/Unidata) to resolve graphics calls. XGKS is maintained at unidata.ucar.edu. Fonts used by XGKS are dynamically loaded when first referenced. Thus, the fonts must be installed on the machine where the XGKS application executes. The font path is compiled into the XGKS library and is set to /usr/local/unidata/lib/xgksfonts. However, it is possible to have XGKS fonts installed in a different location and still execute the software without having to recompile. To do this, set the environment variable "XGKSFontDir" to the new path before executing this application. This variable setting will tell XGKS at execution time where to find its fonts; for example (using C shell syntax):

```
setenv XGKSFontDir /usr/xgks-2.5.5/lib/xgksfonts
```

Testing

Test data sets are provided to verify that the program is correctly installed and running on the system. The tests may also be looked at as examples of how to use the program. The directory tdds_5.2/test contains the scripts to run the tests. The directory tdds_5.2/data contains the input data and expected results for each test. Tests must be run in the directory tdds_5.2/test. Run the tests using any of the commands in the table below.

To test the installation, change to the tdds_5.2/test directory and type the command:

./test.sh [m [n]]

where:

m = the number of the first test to perform, default=1
n = the number of the last test to perform, default=2

For example:

command	what happens
./test.sh	runs all of the tests
./test.sh n	runs test ‘n’ through the last test
./test.sh n m	runs test ‘n’ through ‘m’

After the tests are completed, the results are compared to the expected results. If all goes well, the only differences will be due to different processing times or pathnames. To clean up after the tests, type the command:

./clean.sh

NOTE: The standard data sets were created on a Data General AViiON workstation. You may notice slight numeric differences in the results on other computers. These are generally due to different round-off algorithms and the different architecture of the central processing unit chip.

The tests are described in the table below. Test is the test number, program is the program used to run the test, and the usage column indicates how a file is used, with i for input, o for output, and i/o for both input and output.

test	program	description of test and files	file name and usage	
1	tdds	This test initiates a TDDB (tdds1.tdd), builds a DSR file (tdds1.dsr), loads two data sets, produces a directory and summary listing, creates a backup of an existing TDDB, adds some stations to an existing DSR file, restores a backed up TDDB, corrects a data value, retrieves data and formats it in four different output formats, and produces plots of two data sets.		
		program-control file	allocate1.rdr	i
		allocate output file	allocate1.prt	o
		program-control file	summary1.rdr	i
		summary output file	summary1.prt	o
		program-control file	summary2.rdr	i
		summary output file	summary2.prt	o

	program-control file	backup1.rdr	i
	backup output file	backup1.prt	o
	program-control file	backup2.rdr	i
	backup output file	backup2.prt	o
	program-control file	backup3.rdr	i
	backup output file	backup3.prt	o
	program-control file	dafix1.rdr	i
	dafix output file	dafix1.prt	o
	program-control file	daget1.rdr	i
	daget output file	daget1.prt	o
	program-control file	daget2.rdr	i
	daget output file	daget2.prt	o
	program-control file	daget3.rdr	i
	daget output file	daget3.prt	o
	program-control file	daget4.rdr	i
	daget output file	daget4.prt	o
	program-control file	daplot1.rdr	i
	daplot output file	daplot1.prt	o
	program-control file	daplot2.rdr	i
	daplot output file	daplot2.prt	o
	data set to store in TDDB	dataset.1	i
	data set to store in TDDB	dataset.2	i
	Binary backup copy of TDDB	BACKUP.TDD	i/o
	DSR file for Potomac River data	potomac.dsr	i
	TDDB for Potomac River data	potomac.tdd	i
	DIGS plot configuration file	tdds1.cfg	i
	TDDS response file to run tests	tdds1.echo	i
	TDDS log of TDDS messages of test	tdds1.log	o
	Graphical output of test	tdds1.plt	o
	GKS error file	GKS.errors	o
	Master file of TDDB and DSR names	TDDB_DSR.MTR	i/o
	TDDS log of responses during test	TDDSLOG.DAT	o
	TDDB created during tests	tdds1.tdd	i/o
	DSR file created during tests	tdds1.dsr	i/o
	DIGS plot configuration file	DIGSPLT1.CFG	o
2 tdds	This test initiates a TDDB (example.tdd), builds a DSR file (example.dsr), loads two data sets, and produces a directory and summary listing.		
	program-control file	dafile2.rdr	i
	dafile output print file	dafile2.prt	o
	DSR file for test	example.dsr	i/o
	TDDB for test	example.tdd	o
	TDDS response file to run tests	tdds2.echo	i
	TDDS log of TDDS messages of test	tdds2.log	o

APPENDIX D—USE OF TDDS ON A DOS-BASED COMPUTER

The TDDS software requires a DOS-based computer with the following minimum specifications: Intel 80386 CPU with math coprocessor; 2MB of RAM, 3MB of space on storage device (hard drive). Instructions for obtaining, extracting, compiling, installing, running, and testing the TDDS software on a DOS-based computer follow.

Getting the Latest Version of TDDS

The TDDS program and documentation are accessible on the World Wide Web (WWW) using Mosaic or similar WWW reader, from the USGS Water Resources Information home page (http://h2o.usgs.gov/), and by anonymous FTP from h2o.usgs.gov (130.11.50.175) in the directory /pub/software/general/tdds/DOS. Self-extracting archive files in this directory, named tdds5_2.exe (compiled) and tdds5_2s.exe (source), contain all the files needed to install and test TDDS on a DOS-based computer and compile a new version of TDDS, respectively. The following list gives the commands entered to retrieve the TDDS software compiled for a DOS-based computer using anonymous FTP:

```
ftp h2o.er.usgs.gov                  ; connect to file server
anonymous                            ; enter "anonymous" to userid prompt
your_userid                          ; enter your userid to password prompt
cd pub/software/general/tdds/DOS     ; change to directory containing files
type binary                          ; set binary transfer mode
get tdds5_2.exe                      ; retrieve self-extracting archive file of the
                                       software
bye                                  ; logoff
```

Extracting Files

For either type of TDDS distribution file that you have acquired, tdds5_2.exe or tdds5_2s.exe, the directory tdds_5.2 is created (or overwritten) when the files are extracted. If the tdds_5.2 directory already exists, you may want to delete or rename it before extracting the files. The following are the steps to extract the files from a distribution file. Note, replace <hard disk drive> with the drive letter where you want to install TDDS and optionally replace [directory] with the name of a directory on that drive.

1. If you are not in the directory where the distribution file is located, go there. For example (if already on "c" disk drive):

   cd c:\wrdapp

2. Extract the files using the command: tdds5_2 -d <hard disk drive>:\[directory] Substitute "tdds5_2s" for "tdds5_2" if you are installing the source code distribution, tdds5_2s.exe. Note, be sure to include the -d option and ":\" in the command. Examples are:

   tdds5_2 -d c:\

   tdds5_2 -d c:\wrdapp

The following directory structure will be created (the contents of each directory are shown to the right):

```
tdds_5.2            ; copy of this README file
   `-----bin        ; compiled executable and Lahey error file
   `-----doc        ; documentation files
   `-----src        ; Makefile and source code
   `-----test       ; batch file to run verification tests
   `-----data       ; standard data sets used in verification tests
```

Notes:

- ❑ The bin directory is not included in the tdds5_2s.exe distribution file (it is created during compilation).
- ❑ Source code is included only with the tdds5_2s.exe distribution file.
- ❑ It is recommended that no user files are kept in the tdds_5.2 directory structure. If you do plan to put files in the tdds_5.2 directory structure, do so only by creating subdirectories of tdds_5.2.

Compiling

The source code is provided in the tdds5_2s.exe distribution file so that users can generate the executable themselves. No support can be provided for users generating their own versions of the software. In general, the requirements are a Fortran compiler, utility library libutl_5.6 (which includes screen prompting, device independent graphics, and Time-Dependent Data Base (TDDB) access routines), partial CalComp graphics software, and a minimal level of knowledge of the compiler and the DOS operating system. As provided, the Makefile and source code are optimized for use on a Pentium personal computer using the Lahey Fortran 90 compiler.

The TDDS program graphics are written using CalComp calls; the LIBUTL library includes routines to resolve some of these CalComp calls. Other CalComp calls (PLOTS, PLOT, NEWPEN, and FACTOR) must be provided by the user. The Lahey graph90 library contains these calls.

To generate a new executable, do the following:

1. Change directory to the source directory:

   cd tdds_5.2\src

2. Modify the beginning of the file named Makefile to correctly specify system-dependent variables:

GRAPHLIB	Graphics libraries
LibDir	Top-level directory for LIBUTL software
FFLAGS	Fortran compiler flags
FC	Fortran compiler name
LINKER	Fortran linker name
LNKFLGS	Fortran linker flags

3. Use the make program to initiate compilation of the source code and installation of the software:

   make [BINDIR=directory_for_executable]

See the Installing instructions below for an explanation of BINDIR.

The make will:

- create the directories tdds_5.2\bin and BINDIR if they do not already exist,
- compile the source code,
- place the executable (tdds) in tdds_5.2\bin, and
- place a copy of the executable in BINDIR if specified.

Installing

To make the TDDS program accessible from any directory, the tdds_5.2\bin directory should be included in the PATH environment variable. Add a line similar to the following to AUTOEXEC.BAT:

```
PATH=%PATH%;C:\tdds_5.2\bin
```

Substitute the appropriate drive letter and pathname if not C:\ as shown above.

As an alternative, the tdds executable can be installed in a directory already included in the PATH environment variable. The makefile (input instructions to the Lahey make program—located in tdds_5.2\src) contains instructions to optionally place a copy of the executable contained in tdds_5.2\bin in a specified directory. Use the following two commands to do this:

```
cd tdds_5.2\src
make install [BINDIR=directory_for_executable]
```

If BINDIR is specified, the executable is copied to the specified directory.

Note brackets "[xxx]" are used to indicate optional arguments to commands.

Running the Software

After TDDS is properly installed in a directory that is included in your PATH, the program is initiated using the command: tdds. TDDS is an interactive prompting program. The data base and DSR files that were last used with TDDS in the current directory are retained in a "master" file for subsequent use. Input files may be precoded or created interactively during a TDDS session.

Testing

Test data sets are provided to verify that the program is correctly installed and running on the system. The tests may also be looked at as examples of how to use the program. The directory tdds_5.2\test contains batch files to run the tests. The directory tdds_5.2\data contains the input data and expected results for each test. Run the tests in the directory tdds_5.2\test using the command:

```
test
```

After the tests are completed, the results can be compared to the expected results. If all has gone well, the only differences will be due to different processing times or pathnames. To clean up after the tests, type the command:

```
clean
```

The tests are described in the table below. Test is the test number, program is the program used to run the test, and the usage column indicates how a file is used, with i for input, o for output, and i/o for both input and output.

test	program	description of test and files	file name and usage	
1	tdds	This test initiates a TDDB (tdds1.tdd), builds a DSR file (tdds1.dsr), loads two data sets, produces a directory and summary listing, creates a backup of an existing TDDB, adds some stations to an existing DSR file, restores a backed up TDDB, corrects a data value, retrieves data and formats it in four different output formats, and produces plots of two data sets.		
		program-control file	allocate.rdr	i
		allocate output file	allocate.prt	o
		program-control file	summary1.rdr	i
		summary output file	summary1.prt	o
		program-control file	summary2.rdr	i
		summary output file	summary2.prt	o
		program-control file	backup1.rdr	i
		backup output file	backup1.prt	o
		program-control file	backup2.rdr	i
		backup output file	backup2.prt	o
		program-control file	backup3.rdr	i
		backup output file	backup3.prt	o
		program-control file	dafix1.rdr	i
		dafix output file	dafix1.prt	o
		program-control file	daget1.rdr	i
		daget output file	daget1.prt	o
		program-control file	daget2.rdr	i
		daget output file	daget2.prt	o
		program-control file	daget3.rdr	i
		daget output file	daget3.prt	o
		program-control file	daget4.rdr	i
		daget output file	daget4.prt	o
		program-control file	daplot1.rdr	i
		daplot output file	daplot1.prt	o
		program-control file	daplot2.rdr	i
		daplot output file	daplot2.prt	o
		data set to store in TDDB	dataset.1	i
		data set to store in TDDB	dataset.2	i
		Binary backup copy of TDDB	BACKUP.TDD	i/o
		DSR file for Potomac River data	potomac.dsr	i
		TDDB for Potomac River data	potomac.tdd	i
		Master file of TDDB and DSR names	TDDB_DSR.MTR	i/o
		TDDB created during tests	tdds1.tdd	i/o
		DSR file created during tests	tdds1.dsr	i/o
2	tdds	This test initiates a TDDB (example.tdd), builds a DSR file (example.dsr), loads two data sets, and produces a directory and summary listing.		
		program-control file	dafile2.rdr	i
		dafile output print file	dafile2.prt	o
		DSR file for test	example.dsr	i/o
		TDDB for test	example.tdd	o