Accessing TFTR Data on the VMS Cluster

M. E. Thompson
T. R. Gibney
D. C. McCune
J. A. Murphy
S. D. Scott
H. H. Towner
R. M. Wieland
M. C. Zarnstorff

July 1994

Plasma Physics Laboratory
P. O. Box 451
Princeton University
Princeton, New Jersey 08543

Accessing TFTR Data on the VMS Cluster

M. E. Thompson, T. R. Gibney, D. C. McCune, J. A. Murphy,
S. D. Scott, H. H. Towner, R. M. Wieland, M. C. Zarnstorff

July 1994

Plasma Physics Laboratory
P. O. Box 451
Princeton University
Princeton, New Jersey 08543

Abstract

This document is an overview for physicists and engineers who want to access TFTR experimental data on the VMS cluster. It describes basic cluster commands, the structure of the TFTR data, and the analysis tools most commonly used to study this data. It has been compiled by the High Level Data Analysis (HLDAS) group. This report supercedes PPPL-TM-388, "Accessing TFTR Data on the VAX Cluster".

Contents

1 Introduction **7**

1.1 TFTR Experiments 7

1.2 Data Collection 7

2 Getting Started on the VMS Cluster **7**

2.1 Logging In 8

2.2 Introduction to VMS—the Virtual Memory System 9

2.3 Login Definitions and SETUP 10

2.4 Editors 11

2.5 Terminal Emulators: VersaTerm, DECterm, Xterm, XTC 11

2.6 XPLOT—Display Graphics Files 12

2.7 Printers 12

2.8 TPLOT, PSPLOT—Print Graphics Files 14

2.9 BULLETIN—TFTR Schedules and Notices 14

2.10 FERRET—Search HELP libraries 15

2.11 FINGER—Display Information about Users 16

2.12 LOOKUP—Laboratory Phone Book 16

2.13 MOSAIC—World Wide Web Browser 16

3 Overview of the TFTR Data **17**

3.1 Raw Data 17

3.1.1 Finding and Restoring Raw Data Files 18

3.2 Waveform Files 19

3.2.1 Finding Waveform Files 20

3.3 Results Data 20

3.3.1 Diagnostic Results Files 20

3.3.2 TRANSP Input Files 21

3.3.3 Finding Results Files 21

3.4 SNAP Output 23

3.4.1 Finding SNAP Output 23
3.5 TRANSP Data 24
3.6 The INGRES TFTR Database 24
3.7 The Optical Disk System 25

4 How to Look at the Data **26**
4.1 CUPLOT—The Basic Display Program 26
4.2 TASKPRINT and OPERPRINT—Shot Commentary 28
4.3 LOCUS—Display and Analyze Database Data 28
4.4 MINGL—Database Utilities 29
4.5 UGRAF1 and UGRAF2—Display UFILE Data 30
4.6 Other UFILE Utilities 30
4.7 Display SNAP Results 31
4.8 RPLOT—Display TRANSP Results 32

5 Keeping Abreast of TFTR Operation **33**
5.1 TFTR_SHOT—Current Shot Number 33
5.2 BULLETIN TFTR_SHOTS—Summary of Recent Shots 33
5.3 SHOTCLOCK—TFTR Shot Clock Emulator 34
5.4 OPERLOOK and TASKLOOK Tied to the Shot Cycle 34
5.5 View TFTR Control Room Screens 35
5.6 Event-driven jobs 35
5.7 BQ—Analyze a Day's Shots from Batch 36
5.8 QUE_PERDAY 36

6 Analysis Codes **36**
6.1 SNAP—One Dimensional Power Balance 36
6.2 TRANSP—Time Dependent Transport Analysis 37

7 How to Write Your Own Applications **37**
7.1 User Interfaces 38

7.2 Graphics 38
7.3 Read and Write Waveforms 38
7.4 Read Raw Data 38
7.5 Read and Write Results 39
7.6 Read and Write Database Data 39
7.7 Read SNAP Data 39
7.8 Read TRANSP Data 40
7.9 Directories for Applications 40

8 Management of the User Disks **41**

A Related Documents **43**

B Diagnostic Identifiers **44**

1 Introduction

The Tokamak Fusion Test Reactor (TFTR) is the largest magnetic fusion energy experimental machine in the United States. This document gives a brief overview of the experimental organization and describes how to access the data produced by these experiments.

1.1 TFTR Experiments

TFTR experiments are organized by written proposals, coordinated by Task Groups whose meetings are announced in the TFTR Bulletin. Experimental proposals are presented to and approved at meetings of the Task Group Leaders Council, which are open to all. Accepted Experimental Proposals are available in the TFTR Control Room and in the office of Mariam Rahman, B260.

Prior to a run day, the session leader constructs a detailed shot plan. During the run, the session leader maintains a brief description of the run and a summary of each shot. These shot lists are available in an on-line data base. Booklets of copies of these lists are distributed periodically.

At the beginning of each run day there is a brief meeting, usually in the TFTR Control Room, to review the previous run day and the current run plan. Viewgraphs from the morning meetings are kept in notebooks in Kevin McGuire's office, B263.

1.2 Data Collection

Up to 1.3 Gbytes of data are collected for each TFTR plasma shot. A typical run day is two shifts of operation and produces up to 100 shots. During the shot, data from diagnostic devices are collected by CICADA computer system, and some microVAXes. Following the shot, raw and reduced data are transferred to the VMS cluster. Programs on the cluster analyze and display the data.

2 Getting Started on the VMS Cluster

The VMS Cluster is several VAX, AXP, and microVAX computers that all access the same disks and share print and batch queues and other processing resources. The PPL cluster nodes that permit more than two users at a time are listed below; the last column is the number of CPU seconds to execute a PPPL floating point benchmark test.

Node name	CPU	Primary use	benchmark CPU secs
RAX	VAX 6000-420	TFTR	.52
HAX	VAX 8700	TFTR	.60
BEANIE	VAX 4000-500	PBX	.10
EPX	VAX 8600	TFTR	.70
USC	VAX 8600	TFTR	.70
SNAX	AXP 3000/300	TFTR	.03
WILLOW	AXP 3000/300	TFTR	.03

Note that this list may have changed since this document was written. For a compute bound job AXP execution is typically 10 times faster than RAX. However, at present, all the TFTR data disks are served from VAX nodes; thus for I/O bound jobs the VAX nodes are more efficient.

2.1 Logging In

Once you get an account, you can log onto a VMS node through the terminal server. There are two supported protocols: LAT and TELNET. LAT is more efficient from a PPL office; TELNET must be used from a remote host or to log onto non-VMS systems. For a LAT connection:

```
local>        c rax        (or hax, willow, etc.)
```

For a TELNET connection:

```
local>        t rax        (or hax, willow, etc.)
```

From a remote site the TELNET connection is:

```
telnet rax.pppl.gov
```

In all cases, to log in:

```
Username:
Password:
```

The VAX cluster nodes used primarily for TFTR processing are RAX and HAX. You should get the best response on these nodes. When PBX is not running, BEANIE can be used for some purposes, however it does not have licenses for some third party products; e.g., it can not be used for any data base access.

You need certain privileges if you want to access the on-line TFTR data. These can be set up when your account is opened if you know what privileges you need or they can be added later. The most commonly needed privileges are:

Privilege	Data
TFTR_RESULTS_OWNER	Raw and Results
SNAP_USER_OWNER	SNAP Results
TRANSPDATA	TRANSP Results

You can see what privileges you have by typing:

```
$ SHOW PROCESS/PRIV
```

Note that `$` is the default VMS command prompt. You do not type the `$` when entering commands. If you are missing any needed privileges, contact your sponsor.

When you log on to the cluster, a procedure file named LOGIN.COM, which resides in your login directory is executed. If you are a new user, when you first log on, do:

```
$ COPY USR:[COM]LOGIN.COM *
```

to get the prototypical TFTR login file. Further sections in this chapter describe other commands you might want to put into your login file.

2.2 Introduction to VMS—the Virtual Memory System

The most commonly used VMS commands are:

```
$ HELP
$ DIRECTORY
$ MAIL
$ PRINT
$ TYPE
$ SEARCH
$ DIFF
$ LOGOUT
```

Most VMS commands are adequately described by the VMS HELP system. If further information is needed, a complete set of VMS manuals is in the HLDAS Room, B235. In particular, the "Introduction to VMS" provides a detailed description of VMS features. With some commands additional help is available by typing HELP while running the command. For example:

```
$ HELP MAIL
```

describes how to run MAIL from VMS, while:

```
$ MAIL
> HELP
```

documents all MAIL subcommands. You can obtain a printed copy of an entire HELP topic by telling HELP to create a file with everything about that topic and then printing that file. For example, to print the entire CUPLOT help file on the HT0 printer (and delete the file after printing):

```
$ HELP/OUTPUT=CUPLOT.HLP CUPLOT...
$ PRINT/QUEUE=HT0/DELETE CUPLOT.HLP
```

Note that there are no spaces before the "...". In addition to vendor supplied help topics, many PPL written topics are available in the help system. To list these topics:

```
$ HELP TFTR       an on-line version of this document
$ HELP @USR       lots of useful tools are described here
$ HELP @TFTR      GETSHOTS, OPREAD and all that
$ HELP @PPL       Archiver, etc.
$ HELP @HLDAS     user's helpful hints for HLDAS codes
$ HELP VMS_ALPHA  differences between AXP and VAX
```

2.3 Login Definitions and SETUP

There are two commands that **MUST** be in your LOGIN.COM file, preferably near the beginning:

```
$ SETUP TERMLOG
$ SETUP TFTRLOG
```

Note that TERMLOG *must precede* TFTRLOG! These commands are in the prototypical `USR:[COM]LOGIN.COM` file, in the correct order.

TERMLOG is a procedure that sets your terminal characteristics and terminal related logical names that are used by many PPPL libraries and tools. If you log on to different terminals at different times, use the above form which prompts you each time you log on for the terminal type. If you always log on to the same terminal (e.g., a Macintosh), you can use:

```
$ SETUP TERMLOG Mac
```

For more information, do `$ HELP TERMLOG`.

TFTRLOG defines many utilities, procedures, and disk logical names that you need to access TFTR data. Most users will want the form shown above. If you need to include additional Ufile utilities and translators use:

```
$ SETUP TFTRLOG UALL
```

Do `$ HELP TRANSP JUNK` to learn about "junk" directories if you use the UALL option.

As you may have surmised, SETUP is a utility (actually a PPL written procedure) that sets up the necessary definitions so that you can use a variety of commands that are not part of the operating system. For example:

```
$ SETUP TEX          !TEX, LATEX word processors
$ SETUP TAIL         !look at the tail of a file
```

SETUP can be run interactively or be used in your LOGIN.COM file. Type `$ SETUP HELP` for a list of what SETUP can define.

2.4 Editors

Choosing an editor seems to be a very personal decision. Several editors are available on the cluster:

Editor	login.com set up	Command
EDT	$ EDIT :== EDIT/EDT	EDIT
TPU	$ EDIT :== EDIT/TPU	EDIT
LSE		LSE
EMACS	$ SETUP EMACS	EMACS
TEDI	$ SETUP TEDI	TEDI

The default DEC editor on the VAX nodes is EDIT/EDT, while on the AXP nodes it is EDIT /TPU, so although the EDIT help file says you don't need to define the symbol EDIT we recommend that you do. EMACS is presently not available on the AXP nodes.

2.5 Terminal Emulators: VersaTerm, DECterm, Xterm, XTC

The most commonly used Macintosh based terminal emulator is VersaTermPRO. VersaTermPRO provides VT100 support, Tektronix 4014 and 4105 graphics support, and built-in Kermit.

From a DEC X-window environment, you can run multiple DECterm sessions. DECterm provides VT100 functionality, but no graphics support. For on-screen graphics, use XTC, which provides Tektronix 4014 support with frame buffering.

From a UNIX X-window environment, use Xterm. Xterm provides VT100 and Tektronix 4014 support.

Graphics output created by PPL utilities can generate Tektronix 4014 or Tektronix 4105 commands. 4105 mode is recommended if you use VersaTermPRO and want to transfer the graph to MacDraw. The cluster printers, XTC, and the Control Room display screens can only handle 4014 mode files. Tektronix 4014 files have the extension .PLT; 4105 files have

.4105. Some utilities, such as CUPLOT and LOCUS, let you select the graphics mode from within the program. You can set the default mode for all programs by:

```
$ DEFINE TERMINAL_4105 YES          set 4105 mode
$ DEAS TERMINAL_4105                set 4014 mode
```

VersaTermPro graphs can be made into MacDrawPRO documents. This allows you to make presentation quality graphics. Also a MacDraw graph can be registered with the Graphics Arts Department for publication. Procedures and hints are described in the document "Transferring VersaTerm Graphics to MacDraw" by Harry Towner and Marilee Thompson.

2.6 XPLOT—Display Graphics Files

XPLOT displays selected frames of a graphics file on your terminal. It reads from the file named on the command line or, if none, from the file identified by logical name PLOT. For example:

```
$ XPLOT CUPLOT.PLT
            or
$ DEFINE PLOT CUPLOT.PLT
$ XPLOT
```

After each frame, type **<RETURN>** to go to the next frame, **S** to be prompted for the number of frames to skip forward, **B** to skip backward, **T** to go to a specific frame number, or **X** to exit. For more information, do **$ HELP SG XPLOT**.

XPLOT can convert files from Tektronix 4014 mode to 4105 and vice versa. It determines the mode of the file from its extension. XPLOT's output mode is determined by the current setting of the logical TERMINAL_4105. XPLOT is maintained by Marilee Thompson.

2.7 Printers

The cluster printers are from several different manufacturers. The QMS, Talaris, and DEC ScriptPrinter printers can output either text or Tektronix 4014 graphics files. Some models also support PostScript. In addition, many of the Apple Laserwriters that are part of the PPPL Macintosh network are accessible from the cluster.

The unadorned PRINT command:

```
$ PRINT MY_FILE.DAT
```

outputs to the printer defined by the logical name SYS$PRINT, which is set in the system tables and is different on each node: HT0 for HAX, RL0 for RAX, etc. You can override the default by redefining SYS$PRINT in your LOGIN.COM:

```
$ DEFINE SYS$PRINT HT0      set default printer to HT0
```

Also you can specify the printer when you type the PRINT command:

```
$ PRINT/QUEUE=HT0 MY_FILE.DAT
```

The cluster printers include:

Printer	Type	Location	Postscript
RL0	QMS	TFTR Control Room	no
RT0	Talaris	TFTR Control Room	yes
UT0	Talaris	PPLCC Output Room	yes
HT0	Talaris	HLDAS Room B235	yes
HL1	ScriptPrinter	HLDAS Room B235	yes

For a more up-to-date list of printers do `$ HELP PPPL_PRINT QUEUES`. For a complete list do:

```
$ SHOW QUEUE/DEVICE=PRINT
```

This list will contain several printers such as

```
Printer queue HLDAS_1LW, idle, on RAX::"B235 LW@LOB W 2nd fl"
```

which is the Apple Laserwriter in the HLDAS room, B235. For information about parameters and queue-specific switches:

```
$ HELP QMS
$ HELP TALARIS
$ HELP SCRIPTPRINTER
```

For text files (but not for plot files), the Talaris printers can print multiple pages on a single piece of paper.

The QMS and Talaris print queues figure out what kind of file is being printed from the file extension and handle the file appropriately:

```
$ PRINT/QUE=HT0 MY_PLOT.PLT             4014 graphics
$ PRINT/QUE=HT0 MY_FILE.PS              PostScript
$ PRINT/QUE=HT0/SETUP=2L MY_PROG.FOR    text 2-up, landscape
```

To print on HL1, *you* must match the queue name to the type of file:

```
$ PRINT/QUE=HL1_ANSI MY_FILE.DAT        text
$ PRINT/QUE=HL1_TEK MY_PLOT.PLT         4014 graphics
$ PRINT/QUE=HL1_PS MY_FILE.PS           PostScript
```

To print a Tektronix 4014 graphics file (an SG .PLT file) on a QMS or Talaris printer, use the normal PRINT command. The Scriptprinter can also print multiple plot frames on a single page. This example outputs 4 graphs/page in landscape orientation.

```
$ PRINT/QUE=HL1_TEK/PARAM=(NUM=4,PAGE_OR=LAND) MY_PLOT.PLT
```

A common LOGIN.COM addition is to define symbols for your favorite print commands:

```
$ HL4 :== PRINT/QUE=HL1_TEK/PARAM=(NUM=4,PAGE_OR=LAND)
```

Then you can print a file by using `$ HL4 MY_PLOT.PLT`.

To see your jobs in the print queue, type:

```
$ SHOW ENTRY/DEVICE=PRINTER
```

2.8 TPLOT, PSPLOT—Print Graphics Files

TPLOT is a utility to convert a 4105 graphics file to 4014 mode so it can be printed. If you want to print a 4105 file on any cluster printer use:

```
$ TPLOT FOO.4105 RT0        print 4105 file on RT0
```

PSPLOT is a utility to print a 4014 or 4105 graphics file as a PostScript file.

```
$ PSPLOT FOO.4105 HL1_PS    print 4105 file on HL1
$ PSPLOT FOO.PLT HL1_PS     print 4014 file on HL1
```

2.9 BULLETIN—TFTR Schedules and Notices

BULLETIN lets you read and post messages on PPPL bulletin boards. There are several bulletin boards of interest to the TFTR user:

TFTR	Machine Schedule, Access Information, and Meeting Notices
TFTR_SHOTS	Parameter Summary for Recent Shots
INGRES	INGRES Bulletins
LOCUS	LOCUS Bulletins
SNAP	SNAP and SNAPIN Bulletins
TRANSP	TRANSP Bulletins

To see a complete list of bulletin boards, do **`dir/folder`** from within BULLETIN. To receive bulletins from these bulletin boards, run BULLETIN once to set things up:

```
$ BULLETIN
>select TFTR      select the TFTR bulletin board
>set readnew      to be prompted to read new
                    TFTR bulletins when logging in  OR
>set brief        to be told of new TFTR bulletins
                    as they are posted
>set notify       to be told when there's a new message
                    in TFTR
```

If you prefer not to be prompted for TFTR messages when logging in, then **set noreadnew**. If you don't want the one line description that there are new messages in TFTR, then **set nobrief**. (Repeat the settings selecting SNAP, TRANSP or any other bulletin.) To read a new TFTR message, type:

```
$ BULLETIN TFTR
>read or <return>
>exit
```

The HELP from within BULLETIN is good (not so with **$ HELP BULLETIN**). BULLETIN is a lot like VMS MAIL.

2.10 FERRET—Search HELP libraries

FERRET searches help libraries for a particular string. You can then use the output from FERRET with HELP to read the topics where the string was found. This is useful if you want to know if something is documented or if you know that it is but can't remember under what topic. To use FERRET, first do **$ SETUP FERRET**. The command is **FERRET topic string**. For example, to find the description of RESDIR under the TFTR topic, do:

```
$ FERRET TFTR RESDIR
  @USR
  --> TFTR Data_Organization Results_Data Find_Files

$ HELP TFTR DATA_ORGANIZATION RESULTS_DATA FINDING_FILES
```

You can use wildcards in the topic name; you can also specify just a topic name to get a list of its subtopics.

```
$ FERRET *TEX  " APS"    ! Is there any Tex info on APS?
$ FERRET "SNAP"          ! List subtopics in SNAP help.
```

See **$ HELP FERRET** for more information. FERRET is maintained by Tom Gibney.

2.11 FINGER—Display Information about Users

The FINGER utility displays information about the users currently logged in to the local system or a remote system.

```
$ FINGER [user-name][@host-name]
```

When the username is given in the command, FINGER displays the text in the user's PLAN.TXT if one exists in that user's login directory. For example:

```
$ FINGER MURPHY

RAX VAX/VMS Tuesday, 28-Nov-1993 14:40

Process     Personal name   Program  Term     Login   CPU
LTA323_E    Jane Murphy     Emacs             12:56   0:13
LTA323_D    Jane Murphy     Finger            12:56   0:02
 Logged in since: Tuesday, 28-Nov-1993 12:46
Mail: 1 new message.
 Tue 28-Nov-1993 13:46  From: HAX::WIELAND
 Plan:
Office:B236  Phone:3234  Mail:MURPHY  Vacation:Dec 23-31
```

where MURPHY$:[000000]PLAN.TXT contains:

```
Office:B236  Phone:3234  Mail:MURPHY  Vacation:Dec 23-31
```

For the benefit of other users, you should create a PLAN.TXT file in your login directory. See $ HELP FINGER for more information.

2.12 LOOKUP—Laboratory Phone Book

LOOKUP is an on-line telephone book. You can define it by $ SETUP LOOKUP. To use, enter the person's last name, for example.

```
$ LOOKUP TOWNER
```

See $ HELP LOOKUP for more information.

2.13 MOSAIC—World Wide Web Browser

MOSAIC is the NCSA (National Center for Supercomputing Applications) World Wide Web browser and networked information discovery, retrieval, and collaboration tool developed

at the University of Illinois supercomputer center. The VMS version requires that you have an X-windows workstation or X-terminal. A Macintosh version is also available from the User Support Group, extension 2275. To use it, do `$ SETUP MOSAIC`. This and other PPPL documents are available under the URL "`http://lyman.pppl.gov`".

3 Overview of the TFTR Data

There are several types of TFTR data—raw data, waveform, results files, analysis results, and databases. Raw data are written in *keyword* files. Results from diagnostic programs are usually stored in Ufiles or waveform files; some are stored in keyword files. SNAP, and TRANSP analysis results are in their own formats. The databases are usually stored with INGRES, but some are in an ISAM format. (See LOCUS section 4.3.)

The raw data are archived to tape. The archiving is done automatically and you run GETSHOTS which is described in section 3.1.1 to restore files.

The other types of data are written to magnetic disks and archived on the *optical disk system*. These files are dynamically restored to the magnetic disks as needed. For example, if you specify a results file that is on the optical disk to a program like CUPLOT, the program pauses while the file is retrieved. If you are running interactively, a message is output on your terminal. When the file has been restored, program execution resumes. If this wait is excessive you can abort the application with `^Y` and the request will still be made. (The magnetic disks used in the optical disk system are called the *read cache*.)

To access raw data, waveforms, and results files you need to know the diagnostic identifier. Identifiers are 2 characters long and are called *digraphs*. This document uses *di* to denote an arbitrary digraph, as in `[TFTR.di]`. The list of digraphs is available in the file `TFTR_INF:DIGRAF.INF`. The present list is given in Appendix B.

3.1 Raw Data

Raw data files include the keyword raw data files (RDT), parameter files (PFT), calibration files, etc. For most digraphs there are 2 raw data files per shot, an RDT file and a PFT file.

Raw data files are transferred to the VMS *spooling disks* during the shot cycle. They are archived to tape as soon as possible and skimmed within one day to make room for incoming files. These raw data files can be restored with GETSHOTS to the *raw data disks*. They can also be copied from the spooling disks to the raw data disks with COPYSHOTS. Each diagnostic has a directory, `[TFTR.di]`, on one of the spooling disks and on one of the raw data disks. These disks are accessed by logical names DRW0, DRW1, and DSP0 which are described in section 3.1.1. Users cannot create, modify or delete files on the spooling disks. Users *can* create files on the raw data disks, but they *will not* be archived. It's assumed that files on these disks have already been archived. Users with the TFTR_RESULTS_OWNER privilege can delete files from the raw data disk. (Be careful, since you could delete other peoples' files.) This lets you manage your own space when you need to restore lots of data.

Raw data files are *skimmed* or deleted by expiration date which is updated when a file is accessed. Each diagnostic is given a quota on the spooling disk and a quota on the raw data disk. When a diagnostic exceeds its quota and space is needed on that disk, then that diagnostic's files are skimmed. See Lew Randerson to get your quota changed if your files are being skimmed too quickly. For additional information do `$ HELP SKIMMING` or contact Lew Randerson.

3.1.1 Finding and Restoring Raw Data Files

$ DIRECTORY (The VMS command) can be used to see what raw data files are on disk by using the logical disk names DSP0, DRW1, and DRW0 and the directory `[TFTR.di]`.

```
$ DIR DRW0:[TFTR.EH]PFT12345.EH   Raw and Spooling
$ DIR DRW1:[TFTR.EH]PFT12345.EH   Raw only
$ DIR DSP0:[TFTR.EH]PFT12345.EH   Spooling only
```

DRW0 also includes the results disks.

$ GETSHOTS retrieves raw data from the tape archive system to the raw data disks. GETSHOTS lets you specify a digraph, prefixes, and a series of shot or sequence numbers or full filenames. GETSHOTS searches the TFTR archive indices for the requested files. You have the option to be told what files exist before the retrieve request is made, however, it can take 2–5 minutes to generate this list. A batch job is submitted to the TFTR_RETRIEVE queue to request retrieval of the files. At the next half-hourly scheduled retrieval these files will be restored provided there is operator coverage (usually 8 a.m. to 9 p.m. weekdays). After the files have been restored, RETRIEVE.LOG in your login directory is updated with the names of the files and the result of the request. The command is `$ GETSHOTS` which is defined in TFTRLOG. See `$ HELP GETSHOTS` for more information.

You must have an account on the *data management* computer (this changes from time to time) for GETSHOTS to work. Contact Lisa Hancock (mail to LHANCOCK) to find out if you have an account on the data management computer. Lew Randerson maintains GETSHOTS.

$ COPYSHOTS copies files from the digraph's spooling disk to the raw data disk. The input requested by COPYSHOTS is similar to that for GETSHOTS, however the action (copying the file) is immediate. You can use this command to "save" files for interesting shots before they are skimmed from the spooling disks. Remember that as long as you stay within your quota, files should not be skimmed from the raw data disks—and *you* control which files are on the raw data disks.

3.2 Waveform Files

There are two classes of waveforms: *file waveforms* and *derived waveforms*. *File waveforms* are created by analysis programs on CICADA or on the cluster; they can have multiple versions. Examples are: `MB-IP-SL`, `FM-TOTNE`. *Derived waveforms* are algorithms which calculate the data dynamically from file waveforms, e.g., `M-PTOT`.

There is a single waveform file for each shot. Every shot starting with 8486 with minimal plasma current has a waveform file. Most waveforms are written to the file within 5 minutes after the shot by analysis tasks that run automatically for various diagnostics. However the cluster analysis tasks can be run at any time after the shot. Also new versions of a given waveform can be written months or years later to correct for bad calibration data or other problems.

The standard utility to look at waveform data is CUPLOT which is described in section 4.1. Waveforms have names that are up to 12 characters and begin with dd-, where dd is the diagnostic identifier, or d-, where d is a class of derived waveform. For instance, MB-IP-SL is generated by the MB (magnetics) diagnostic; it is the plasma current and is the slow sweep waveform (covering the duration of the plasma shot). When a waveform is written with the same name as an existing waveform, a new version of the waveform is created. The default for CUPLOT is to return the last entered and presumably best version. Each waveform can have a maximum of 2048 points and typically has 800 points. This is a table of some basic waveforms:

Waveform	Description
FM-LD-06	MIRI Line Density, R=2.68 M
MB-AP-SL	Minor Radius
MB-IP-SL	Plasma Current
MB-RP-SL	Major Radius
MD-BP-SL	Beta Poloidal (diamagnetic)
NB-BP-SL	Neutral Beam Power Summary
YS-TE240	Te at R=240 cm
YS-TE255	Te at R=277 cm
M-ETOT	Total Energy
M-TAUE	Confinement Time

`TFTR_INF:WAVEFORMS.INF` has a more complete list with descriptions of the waveforms.

Waveform titles are text strings up to 72 characters long that provide more information about the waveform. To get a complete list of waveforms and their titles for a *particular shot* (remember not all waveforms are available for every shot), use CUPLOT. For example, to get a list for shot 76778:

```
$ CUPLOT SH 76778 TT ,   Type all waveform names & titles
$ CUPLOT SH 76778 TT MB  Type MB waveform names & titles
```

If you have questions about the data, talk to the diagnositician responsible for that data. Marilee Thompson is responsible for the waveform files.

3.2.1 Finding Waveform Files

$ WFDIR displays the waveform files stored on magnetic disk or on the optical disks. You provide a shot number or shot range.

```
$ WFDIR 55800
$ WFDIR 55800-55802
 55800 on disk
 55801 on optical
 55802 on optical
```

$ WFGET explicitly retrieves waveforms from optical storage to the TFTR read cache. Although most programs will automatically access the optical disks, there may be times when you want to copy several files in advance. WFGET is defined in

```
$ WFGET 55800-55802
$ WFGET @SHOTS.LIS
```

3.3 Results Data

Results files are organized in two main directory trees. One tree has subdirectories for a particular diagnostic's results files. The other has subdirectories for users who prepare files for TRANSP runs. Results files are usually in the Ufiles format. A Ufile contains either a one dimensional function $f(x)$ or a two dimensional function $f(x, y)$, with only one function per Ufile. (The file name is made from a prefix of up to 16 characters, the shot number, and up to a 16 character filename extension. For example, `AV12345.CUR` might contain plasma current for shot 12345.) Thus there can be many Ufiles for a given shot and digraph. `TFTR_INF:UFILES.INF` contains a list of the commonly used diagnostic Ufiles. See Doug McCune for Ufiles documentation.

3.3.1 Diagnostic Results Files

Each diagnostic has two results directories: `[TFTR.di]` and `[TFTR.di.USER]` . A search list of these directories is defined by the logical name `di_ALL`. The conventions for using these directories are:

1. The `[TFTR.di]` area is for results files made by a cognizant responsible physicist.
2. The `[TFTR.di.USER]` area is for results files possibly needing more caution. In particular, these areas were created for results files not made by the responsible physicist for the diagnostic, for example when MCZ translates YS data.
3. Files that are truly exploratory, temporary, or just for debugging should not go in either `[TFTR.di]` or `[TFTR.di.USER]`, but should go in one's own directory or on the scratch disk.

No other directories should be created on the results disks!

By using the `di_ALL` search list or the `RES$:[TFTR.di]` directory specification you write files to RES$W, the *results write cache*. When you read, RES$R, the *results read cache* is also searched. The read cache is write and delete protected; it only contains copies of files that exist on the optical system.

If a file is in `RES$W`, it has not been copied to the optical disks. You can delete a file from the write cache. If it is on `RES$R`, it can be superseded by writing a new version. Multiple versions of files on `RES$W` are purged before they are archived.

3.3.2 TRANSP Input Files

In addition to diagnostic results directories, special user directories exist for storage of modified Ufile data. These directories are used mainly for the preparation of input data for TRANSP runs. The Ufile data in these directories have generally been smoothed or otherwise modified from the originals in the diagnostic results directories. The names of these directories are `TR_DISK:[username]` where "username" gives the name of the physicist creating the modified Ufile data.

These modified results directories are archived on the optical disk system. Such directories are available to new users by request only. For more information contact Doug McCune.

3.3.3 Finding Results Files

$ DIRECTORY (The VMS command) can be used to display results files in the cache only. The first example shows how to get a directory of `.ECE` files in `[TFTR.YM]` and `[TFTR.YM.USER]`. The second example shows how to get a directory of all files in `[TFTR.YM.USER]` only.

```
$ DIR YM_ALL:*.ECM
$ DIR RES$:[TFTR.YM.USER]
```

The following logical disk names can be used: `RES$` refers to the results disk cache. `RES$W` is just the write cache for results files, and `RES$R` is just the read cache for results files. `di_RES` is the logical name for a digraph's results disk; it is defined to be `RES$` for all digraphs. The preferred disk specification is now `RES$`.

$ RESDIR displays a directory of all results files: files on the magnetic cache and on the optical disks. The optical disk directory also includes the platter and the date and time the file was archived. You must specify the directory and the file name. Wildcards are allowed in the file name only. The command is `$ RESDIR filespec`. Examples are:

```
$ RESDIR [TFTR.BR]S42811.BE2,S42797.BE2
  Platter   D  Date      Time        Filename
```

```
P05A      F  20-OCT-89 23:26:16    S42811.BE2
1 file found in [TFTR.BR].
Platter   D  Date      Time        Filename
P07A      F  19-OCT-89 23:37:12    S42797.BE2
1 file found in [TFTR.BR].
```

```
$ RESDIR YS_ALL:T42%%%.*,[TFTR.YG]*.ECE,*.ABC
```

If you do not find a file with RESDIR or the above directory commands, then the file does not exist on the results disks.

$ RESGET copies files from the optical disks to the magnetic disks. Wildcards are allowed in the file name, but not in the directory name. The command is `$ RESGET filespec y/n`. The second parameter tells RESGET whether you want to wait until the file is retrieved or not. Although most programs will automatically access the optical disks, there may be times when you want to copy several files in advance. Examples are:

```
$ RESGET  [TFTR.BR]S42811.BE2,S42797.BE2  Y
$ RESGET YS_ALL:T42%%%.*,[TFTR.YG]*.ECE,*.ABC  N
$ RESGET  TR_DISK:[MCCUNE]*.T1T Y
```

$ OPREAD also copies data from the optical disks. The command is `$ OPREAD filespec [platter] /[NO]WAIT`. The platter, which can be gotten with RESDIR, is required if wildcards are used in the file specification. OPREAD does not support the device logical names, like YS_ALL, as RESGET does, however, it may be faster if you are retrieving lots of files with a wildcard specification. For example to retrieve files with a wildcard specification:

```
$ OPREAD [TFTR.BR]S42797.* P07A
```

or to retrieve a single file:

```
$ OPREAD [TFTR.BR]S42797.BE2 /NOWAIT
```

Note /WAIT, which is the default, causes OPREAD to wait until the request has been completed before exiting. /NOWAIT causes OPREAD to issue the request and then return control back to the user. Sometime later the request will completed. It is not necessary to use OPREAD if the program you are using to access the data knows how to retrieve from the optical system (e.g. all programs accessing waveforms and Ufiles).

$ OPSTATUS lists files that are queued for restoring from or archiving to the optical disks.

```
$ OPSTATUS/LIST         list files queued for restore
$ OPSTATUS/SAVE/LIST    list files queued for archive
```

Other qualifiers are:

```
/USER=username /USER=* /SAVE /CANCEL /CONFIRM
```

OPREAD and OPSTATUS are maintained by Tina Ludescher. More information is available in `$ HELP JUKEBOX`.

You may not want to use the `di_ALL` search list directory specification when deleting files. The following command:

```
$ DELETE RES$W:[TFTR.BR.USER]P12345.ABC;1
```

deletes the file `P12345.ABC;1` in `RES$W:[TFTR.BR.USER]`. If a file of the same name also exists in `RES$W:[TFTR.BR]`, it is *not* be deleted. If you had used `BR_ALL` instead of `RES$W:[TFTR.BR.USER]`, *both* files would be deleted.

3.4 SNAP Output

The SNAP analysis program is now typically run between shots. Each SNAP analysis of a given shot generates an output file identified by the shot-try number. Both the input file (`SNAP.DAT`) and output file are archived to the optical disk for each SNAP shot-try. There can be several hundred SNAP shot-tries for the most popular TFTR shots.

3.4.1 Finding SNAP Output

$ SURVEY lists SNAP shot-tries with the comments for each shot-try.

```
$ SURVEY
```

When you are asked for the "shot entry", enter either a single shot or a range, optionally followed by the try numbers or try range in parentheses:

```
73268              All tries of shot 73268
73253-73273        All tries for shots 73253 to 73273
73268(1, 3, 5-7)   Tries 1, 3, 5, 6, 7 for shot 73268
```

See `$ HELP SURVEY` for additional information. This utility is maintained by Marilee Thompson.

$ RECENT_SNAPS gives a list, one item per line, of shot-tries restricted by date range and/or user id.

$ SNAPSHOTS can also be used to find out what shots have been analyzed by SNAP and their related comments. SNAPSHOTS uses the INGRES forms interface.

```
$ SNAPSHOTS
```

When the form comes up fill in the fields. You can enter a shot range or a date range. (TAB goes to the next field, ^P to the previous field, PF2 gets help, etc.) SNAPSHOTS is also available from the MINGL UTILITIES menu and is maintained by Dick Wieland.

$ SRERUNT copies the SNAP.DAT input file for a single shot-try from the optical or from the read cache to your current default directory:

```
$ SRERUNT 73268 252          try 252 for shot 73268
```

$ REQUEST_SNAP restores either the input or output file for a shot-try:

```
$ REQUEST_SNAP 73268 252 DATA     restore data file
$ REQUEST_SNAP 73268 252 NAME     restore input file
```

The utilities that access SNAP data automatically restore the files from the optical as needed. This command is useful if you need several files at once, for example if you are building a database.

3.5 TRANSP Data

TRANSP data is identified by the tokamak (e.g., TFTR), the shot year, and a TRANSP run identifier, either a four digit run number or an eight character "shot-try" string. Run results are archived on the optical disk system. Runs prior to September, 1989, that have never been restored to disk (i.e., runs done before the optical disk implemention) can be restored from tape with GETRUN. GETRUN can also provide a list of TRANSP runs.

Read **$ HELP TRANSP** if you want to use GETRUN or the RPLOT utility that displays TRANSP results. Doug McCune maintains the TRANSP data.

3.6 The INGRES TFTR Database

INGRES is a relational database management system. An INGRES database is a system of files, not in typical VMS format. The files are accessible only through INGRES and are most often accessed from the more user-friendly programs MINGL and LOCUS. For more information about INGRES do **$ HELP INGRES** or see the INGRES manuals in the HLDAS room, B235. Dick Wieland is our INGRES administrator.

INGRES is a licensed product and not all nodes have a license; at login you will see a message if you can not run any INGRES applications on that node. Also, at present, MINGL and LOCUS have not been converted to run on the AXP nodes. You will also get a message at login if these utilities are not available.

TFTR is the name of the INGRES database that contains hundreds of tables of TFTR data. The following data tables are maintained for use by all the TFTR physicists. (Many "privately" owned tables and databases also exist.) All tables can be looked at with the INGRES version of LOCUS except for text oriented tables such as TASKLOG and OPERLOG.

TASKLOG has the session leader summary for each shot, which includes selected parameter, a grade for the shot, the "star" field (whether the shot is significant to the XP or is of other interest), and a comment. The commands `$ TASKLOOK` and `$ TASKPRINT` allow you to peruse and/or print shots from this table.

OPERLOG has the Physics Opertor comments for each shot. These comments tend to be technical details about the setup for the shot. The commands `$ OPERLOOK` and `$ OPERPRINT` allow you to access this table.

ALLSHOTS is a table that contains a select set of plasma parameters and event times for every TFTR shot that meets minimal plasma criteria, starting with the first plasma shot in December 1983.

WAVEFTFTR is a table updated on a nightly basis that contains an extensive set of waveform parameters extracted at several times of interest for shots going back to 1986.

E_TIMES contains an extensive set of event times for each shot catalogued in WAVEFTFTR..

SNAPS and SNAPP contain scalar and derived values from selected SNAP runs going back to 1986. The parameters in these tables are defined in the SNAPDAB Template Document available in room LOB B235, as well as in the DBASE_TMPL on-line documentation file available by running MINGL, selecting DBASE, then DBASE_TMPL, then HELP, then WhatToDo.

CONTENTS is a table of contents with information on who created the table, when, what program was used, and in the originator's own words, what the table contains. Two procedures allow access to this table:

```
$ CONTENTS          browse mode
$ UCON              detail mode
```

With either program, you can use the ENTER key to leaf through every entry, or can tab to a particular field and enter a constraint (e.g., type ZARNSTORFF in the User column) before proceeding. Users unfamiliar with INGRES forms should consult the "Users' Guide to the MINGL Database System for TFTR".

3.7 The Optical Disk System

The optical disk system is a hierarchical system with primary storage, called the *cache*, on magnetic disk and secondary storage on the optical disks. New files are written to the *write cache*. Periodically the optical disk archiver copies the files from the write cache to the optical disks. Once archived on the optical disks the files are copied to the *read cache* and deleted from the write cache. The data on the read cache is skimmed by access date. Most of the software that reads data from this system first checks the magnetic cache for the existence of the requested file. If the file is not found, the optical disks are searched. If the file is found there, it is copied to the read cache where it stays until it is skimmed.

The optical disks are stored in a jukebox. When a file is requested the *platter* which contains that file is mounted. This can take up to a minute, but may take longer if usage is heavy. The file is then copied to the read cache. Two platters can be mounted simultaneously and they remain mounted until a new platter is needed. Once the platter is mounted, it typically takes about 5 seconds to restore the file. Thus the time that it takes to retrieve a file is variable. Local directories of all files stored on the opticals are maintained for the various types of data. These directories are searched first to see if the file exists on the opticals.

When you access a file that is on the optical disk, your process sends a request to a separate JUKE_SERVER process. Interactive jobs normally display a message such as:

```
Notified JUKE_SERVER - you are first in queue.
                  or
Request queued - JUKE_SERVER is busy - Waiting for copy.
```

(There is no such message for batch jobs.) If you get a message indicating some sort of error, call Computer Operations at 3444.

4 How to Look at the Data

Several general tools are available for looking at and analyzing the TFTR experimental data.

4.1 CUPLOT—The Basic Display Program

CUPLOT is a program for manipulating and displaying vector data from TFTR waveform files, raw data (RDT) files, 1D and 2D Ufiles, SNAP results files, and ASCII (text) files. You can also write data as a 1D Ufile, an ASCII file, or an HDF file.

Data are identified by *data element name* and a *shot number*. You can read up to 64 data elements at a time and display these with multiple frames per page and multiple curves in each frame. The data elements entered by the DE command and the shot number entered by the SH command define the primary curves. The number of frames is determined by the number of data elements in the DE list. For example, to display one frame of the plasma current waveform vs time for shot 76767:

```
$ CUPLOT
Command[  ]? SH 76767              !select the shot number
Command[SH]? DE MB-IP-SL ;         !select Ip waveform
Command[DE]? G                     !go make the plot
```

The D2 command sets the overlay data element(s). To display 4 frames, each with 2 curves, including the Thomson Scattering waveforms vs their radius waveform: radius waveform:

```
Command[  ]? SH 76767
Command[SH]? DE
     D-PTOT NE-DD-01 YM-T1-SL:YM-TM-SL TS-TE-PR:TS-RA-PS ;
Command[DE]? D2
     BR-PR-SL FM-LD-06 NB-BP-SL TS-NE-PR:TS-RA-PS ;
Command[YM]? G
```

The S2 command sets the shot number for overlay curve(s). To display 2 frames overlaying beam power and neutron strength for 2 shots:

```
Command[  ]? SH 76770              !base curve shot
Command[SH]? S2 76769              !overlay shot
Command[S2]? DE NB-BP-SL NE-CU-S3 ;
Command[DE]? XT 3 5                !x-axis window
Command[XT]? G
```

You can look at the same data elements for a different shot by simply entering the new shot number and typing GO; or you can just change the data elements and GO. If you are missing some data, the Menu command lists the status of all the data element reads with text descriptions of many of the error conditions.

The form of the data element name tells CUPLOT what kind of data to read:

Waveform names are up to 12 characters with a dash in either the second or third position, e.g., MB-IP-SL or M-LMODE-G

Ufiles have a dot separating the prefix and the extension and can have a preceding directory, e.g., B4050_.SVB, [YG]T.G1S where [YG] denotes YG_ALL.

SNAP channels can be entered as the channel number or name, with digraph "S". To look at T_e vs r for shot 73268 try 252:

```
$ CUPLOT SH 73268_252  DI S  DE TE ; GO
```

Raw data names have a dash somewhere after the third character, e.g. DFM1-FIR-FFC-6.

The digraph command, DI, sets the default directory for raw data and Ufiles (to di_ALL).

The GO command is a shorthand for the READ and PLOT commands. You can set scaling parameters which apply when the data is read. Once the data has been read, you can set or change various options that affect the appearance of the plot, such as axis windows, log scale, and descriptive text, and then PLOT the data. You can also manipulate the data using the smooth, Fourier Transform, least squares curve fit, and vector calculator functions.

The H (single letter) command gives a one-page list of the commands. The HELP command from within CUPLOT calls up the same help file as the VMS command $ HELP CUPLOT. There is also a document "Guide to CUPLOT on the VMS Cluster". CUPLOT is maintained by Marilee Thompson.

4.2 TASKPRINT and OPERPRINT—Shot Commentary

$ TASKPRINT displays comments from the TASKLOG table to your terminal, a print queue, or an output file.

```
$ TASKPRINT
Year, single Date, single XP, Shot, Help, eXit[ ]? D
Date (dd-mmm-yyyy, dd-- for this month)[ ]? 27-MAY-1994
 getting possible XP's:
   608
output to tty(,), print queue(e.g.,rl0), or file(name.ext)[ ]? ,
```

$ OPERPRINT displays comments from the OPERLOG table to your terminal, a print queue, or an output file.

```
$ OPERPRINT
Shot range, Date range, search STring, Help, eXit[ ]?
                    S 76764 76778
output to tty(,), print queue(e.g.,rl0), or file(name.ext)[ ]? ,
```

These tools are maintained by Marilee Thompson.

4.3 LOCUS—Display and Analyze Database Data

LOCUS is the PPL standard utility for manipulating and displaying databases. An example of its use is the determination of scaling laws. To do this you can enter and modify data, define subsidiary variables as arbitrary functions of original data, plot such variables, and perform linear and multiple linear regression analysis. A cursor option in LOCUS lets you see parameters that correspond to points on a graph. This is useful for identifying the shot number of selected points, for example.

There are two implementations of LOCUS—one uses INGRES databases and the other VMS ISAM files. You can use either implementation. If you want to share your data with other people at PPL, the INGRES version is recommended. If you want to transport your database to another VMS system that doesn't have INGRES, then use the ISAM version. There is also a program, INGRATE, that converts a LOCUS ISAM file to an INGRES database table.

TFTRLOG sets up the INGRES version of LOCUS. If you want the ISAM version, do **$ SETUP LOCUS ISAM RESET** *after* calling TFTRLOG. (RESET is necessary to force the new definition.) An extensive users' manual is available. **$ HELP LOCUS** tells how to get it.

The following session shows how to graph the plasma current vs the average density for shots since 73000 with beam power greater than 10MW from the table ALLSHOTS in the TFTR database.

```
$ LOCUS
```

```
<0=SD;> Select database and tables <DB>:
            DB TFTR              !TFTR Database
            PT ALLSHOTS ;;       !ALLSHOTS Parameter Table
<1=MO;> Plot,Regress,Surface,PRint,Edit,SQL? <PL>:
            PL                   !Select Plot
<2=N;> Set: Y,X,C,S,G,G+,K,SO,J,Mu,An,Ou,Ex,Re? <G>:
            Y IP                 !IP is Y axis
<2=N;> Set: Y,X,C,S,G,G+,K,SO,J,Mu,An,Ou,Ex,Re? <Y>:
            X NEBAR              !NEBAR is X axis
<2=N;> Set: Y,X,C,S,G,G+,K,SO,J,Mu,An,Ou,Ex,Re? <X>:
            C SHOT>73000 BP>10;  !set Constraints
<2=N;> Set: Y,X,C,S,G,G+,K,SO,J,Mu,An,Ou,Ex,Re? <C>:
            ;                    !end setup, generate plot
```

At any prompt, typing **<RETURN>** gives an explanatory list of the possible responses; e.g., at the **<2=N;>** level prompt:

```
\* Options are:
 1) Y < >
 2) X < >
 3) Constraints < >
 4) Symbols < >
      ...
```

You can enter either the option number or enough of the keyword to uniquely identify the option (the part displayed in capital letters). LOCUS remembers the previous response to each option and shows this in the prompt; to retain this value, type a comma.

The information in the angle brackets indicates the current level. In general, **** backs up a level. In particular, **\MO** backs up to the Main Option menu. At any level, you can type **\X** to exit, **?** to see the TK user interface commands, or **\H** for help. LOCUS ISAM is maintained by Jane Murphy and LOCUS INGRES is maintained by Jane Murphy and Dick Wieland.

4.4 MINGL—Database Utilities

MINGL is a collection of INGRES database utilities which includes database management tools, table construction from SNAP, TRANSP, waveform, Ufile, and raw data, and the LOCUS and INGRES programs. MINGL allows you to switch between the various programs while staying connected to your database. MINGL is defined with **$ SETUP LOCUS INGRES**. See the "Users' Guide to the MINGL Database System for TFTR" by Dick Wieland who maintains MINGL. When in MINGL, use the **PF2** keypad key for help. To execute:

```
$ MINGL
```

4.5 UGRAF1 and UGRAF2—Display UFILE Data

UGRAF1 and UGRAF2 plot the contents of a 1D and 2D Ufiles respectively. You must know the dimensionality of the Ufile you are going to plot and the full name of that Ufile, including the disk and directory information.

When you start either program, you first encounter the UFNTER menu which lets you set up the name form; i.e., the file name less the shot number.

```
$ UGRAF1
...
UFNTER:   N TR_DISK:[MCCUNE]S.T1T Q   !set file name form
...
UGRAF1:   55851 P                     !set shot and plot
```

plots the file `TR_DISK:[MCCUNE]S55851.T1T`. `Q` exits the file form menu; generally `Q` is used to exit menus in Ufiles utilities and `?` redisplays the menu.

```
$ UGRAF1 N XI_ALL:S.T1T Q
UGRAF1:   65684 P Q
UGRAF1:   S 65685 P Q
```

plots `XI_ALL:S65684.T1T`, then `XI_ALL:S65685.T1T`, leaving control at the main UGRAF1 menu. From the main menu, or after drawing any plot, you can set scaling defaults, line style, grid style, plot size, type of axes, etc.

UGRAF2 provides slice plots, slice multi-plots, contour plots, and isometric projections. For a demonstration of UGRAF2 capabilities, do:

```
$ UGRAF2 @TRANSP$:[CTL]UGRAF2_DEMO
```

which invokes a UREAD script to make several displays from a 2D Ufile of smoothed ECE radiometer temperature vs time and ECE frequency. To have UGRAF2 menus, prompts, and script file responses echoed during execution, modify the above command to:

```
$ UGRAF2 *E @TRANSP$:[CTL]UGRAF2_DEMO *D
```

It might also be of interest simply to type out the script file that is executed by this example, `TRANSP$:[CTL]UGRAF2_DEMO.IFI`. For information about the UREAD user interface system, do `$ HELP UREAD` or type `*H` at any prompt in the utilities. For further information obtain a copy of the Ufiles manual from Doug McCune.

4.6 Other UFILE Utilities

Several specialized tools exist for displaying and manipulating Ufiles. These are defined with `$SETUP TFTRLOG UALL` or `@USR:[UFILES]UFUTIL`. Follow the instructions in `$ HELP`

`TRANSP JUNK` to set up scratch areas for error and information messages from these programs.

$ CONCAT combines multiple 1D Ufiles to make a 2D Ufile. It can also add extra columns or rows to a 2D Ufile by interpolation or extrapolation.

$ ECPROG is a special tool for ECE temperature 2D Ufiles.

$ EXTRAC extracts 1D slices in either x or y from a 2D Ufile or can be used to create a new 2D Ufile by deleting rows and/or columns.

$ GAVER1 averages multiple 1D Ufiles. It can also interpolate and sum or difference 1D Ufile data.

$ GAVER2 averages multiple 2D Ufiles. It can also interpolate and sum or difference 2D Ufile data.

$ GSMOO1 smooths and/or rescales 1D Ufiles.

$ GSMOO2 smooths and/or rescales 2D Ufiles; also swaps axes order.

$ PROPANE creates Ufiles from hand entered data.

$ SPLICE splices pieces from multiple 1D Ufiles to form a single 1D Ufile.

$ SPLICE2 splices for 2D Ufiles.

$ THIN1 thins 1D Ufiles, e.g. reduce the number of time points.

$ THIN2 thins 2D Ufiles.

$ ULOOK0 displays Ufiles scalar data.

The interface to these tools is similar to that for UGRAF1 and UGRAF2. You can also write Ufiles from CUPLOT — any *data element* can be written out as a Ufile.

4.7 Display SNAP Results

The SNAP analysis program generates scalars, 1D arrays, and 2D arrays, all identified by channel number or name. To see what quantities can be displayed, search or print the file `HLDAS$:[CURRENT]CHANNEL.MEM`. Channel numbers are 4 digit numbers, where the first digit indicates the type of data:

chan number	type of data
1xxx	1D vector vs Rmajor
2xxx	1D vector vs Rminor
3xxx	integer scalers
4xxx	real scalers

As mentioned is section 4.1, CUPLOT can be used to plot SNAP 1D arrays. It can also type SNAP scalers, and generate lists of channel names, in alphabetical order, for vectors. The following programs display SNAP output as a combination of graphics text pages and plot frames.

$ SNAPSUM plots 1 page of text summary, 1 or 2 pages of plots, and optionally an extra page of comments. The following example outputs the 3-page SNAPSUM plot file to printer HL1, in 4 plots/page mode:

```
$ SNAPSUM 68522 15
printer (rl0,ht0,...,hl1,hl2,hl4,tty,keep)  -->  HL4
Number of output pages (1,2,3,4)  3
```

$ RPLOTS displays about 4 text pages and several plot pages from the SNAP run.

```
$ RPLOTS 68522 15
```

SLOOKT, which formerly accessed an archived version of the RPLOTS file, now is a synonym for RPLOTS.

4.8 RPLOT—Display TRANSP Results

The RPLOT program allows graphical display of TRANSP output. *Multigraph packages* are supplied by RPLOT to generate standard displays such as plots of the ion power balance. RPLOT is also used to create standard hardcopy and output on successful completion of a run. In order to run RPLOT, your LOGIN.COM must set up some definitions as described in `$ HELP TRANSP`, and you need to know the identification of the TRANSP run you want to plot. To access the TRANSP data you need the TRANSPDATA privilege.

TRANSP runs are stored on disks referenced by the logical name RUNDATA. The directory for a run is determined by the tokamak name and the shot year, for example `[TRANSP.TFTR.88]` or `[TRANSP.PBXM.89]`. Runs are identified by run ids which are either four digit numbers or the newer eight character "shot-try" identifiers. For example, the TFTR.89 area contains a TRANSP run 1121 which is an analysis of a shot from 1989 with beam heating and carbon pellet injection; the TFTR.90 area contains a TRANSP run 46470A03 which is an analysis of a low current high poloidal beta shot.

For a sample of plots from run 1121, execute:

```
$ RPLOT @TRANSP$:[CTL]RPLOT_DEMO
```

This demo shows a fraction of the data available in the run. There are hundreds of scalar and profile functions and several dozen multigraph packages. It might be of interest to type out the script file `TRANSP$:[CTL]RPLOT_DEMO.TMI`.

Successful use of RPLOT requires gaining familiarity with the *names* of functions and multigraph packages. Some examples of standard names are:

TE	electron temperature vs time and spatial coordinate
XNEUT	scalar multigraph of neutron emission vs time
PDENS	profile multigraph of plasma specie densities vs time and radius

The precise set of names available depends on model options and the version of TRANSP used to create a particular result. Use:

```
$ SPAWN @TRANSP$:[COM]RPLOT_NAMES TFTR.89 1121 HT0
```

to have a listing of names from TRANSP run TFTR.89 1121 printed on printer HT0.

In addition to basic plotting capability, RPLOT offers many advanced capabilities. User defined profile functions can be defined and also saved in a Ufile. Various options for smoothing, time averaging, and integral-differential transforms are scattered throughout RPLOT.

The latest RPLOT documentation "Accessing TRANSP Output: A User's Guide" is available from Doug McCune.

5 Keeping Abreast of TFTR Operation

In the TFTR Control Room, several people are involved in coordinating the machine operation: Session Leader, Physics Operator, Chief Operating Engineer, Tritium Engineer, HLDAS operator.

5.1 TFTR_SHOT—Current Shot Number

TFTR_SHOT and TFTR_SHOT_NO are logical names that contain the current shot number.

TFTR_SHOT is usually set when the "early" waveforms arrive from CICADA, about 2 minutes after the start of the shot. It is not set for test shots (shot numbers greater than 99000). It is defined on all the primary cluster nodes and most workstations. TFTR_SHOT_NO is set by the CICADA-VAX transfer process and is is updated when the SOS (Start of Sequence) event occurs on the TFTR shot clock. To see both values do:

```
$ SHOW LOGICAL TFTR_SHOT*
```

5.2 BULLETIN TFTR_SHOTS—Summary of Recent Shots

A useful bulletin to look at is TFTR_SHOTS. An entry is made for each good shot when it is available for analysis on the cluster. If you look before noon, this bulletin has all today's and yesterday's shots; after noon, it has today's shots. The important information is in the header of each bulletin, so a "directory" list gives a one-line summary of the shots:

```
$ BULLETIN TFTR_SHOTS     select board on the command line

> 1                       to see parameter descriptions

TIME:  time of peak ETOT when PBEAM or PICH is non-zero;
       otherwise the end of the plasma current flattop.
IP:    the MB-IP-SL waveform in MW.
FM#6:  Nel from FM-LD-06.
PBEAM: from NB-BP-SL in MW.
PICH:  from IC-PI-SL in MW.
ETOT:  from M-ETOT in MJ.
TAUE:  from M-TAUE.

> dir                     to see shot summary

# Description                                          Date
 shot time Ip   fm#6 Pbeam  Pich Etot taue   time  8-SEP
42388 3.47 1.09 3.13  8.76  0.00 1.14 0.13  14:06 10-OCT
42389 3.38 1.09 3.41 10.58  0.00 1.26 0.12  14:14 10-OCT
42390 3.40 1.09 3.41 10.62  0.00 1.28 0.12  14:21 10-OCT
```

To create a file with one line for each shot, do:

```
bull>  file/all/nohead[/new] shots.lis
```

With the **/new** switch a new file is created; otherwise it appends to the file if it exists.

To keep abreast of shots while in your office, use "**set notify**" for TFTR_SHOTS.

5.3 SHOTCLOCK—TFTR Shot Clock Emulator

$ SHOTCLOCK provides a simulation of the TFTR shot clock. To use it do **$ SETUP SHOTCLOCK**. It is most useful if it can run in its own Versaterm or DECterm window.

```
$ SHOTCLOCK_EXE
```

For more information, do **$ HELP SHOTCLOCK**.

5.4 OPERLOOK and TASKLOOK Tied to the Shot Cycle

$ TASKLOOK displays the TASKLOG data base, which contains the Session Leader's comments, grade, and parameters of interest for each shot, using an INGRES forms interface. It displays all the shot information for the selected date and XP. The "Wait" command "slaves" your display to the typed updates— whenever a new comment is entered, your screen is updated.

$ OPERLOOK displays the Physics Operator's comments using an INGRES forms interface. It can display the previous 20 shots from the selected shot number. It also has a "Wait" mode, which includes a simplified shot clock emulation.

5.5 View TFTR Control Room Screens

The display screens in the TFTR Control Room are filled by plots assigned by the HLDAS Operator, and by "per-shot" jobs that automatically output to their assigned screen(s) as each shot is processed. All the screens have a "physical" name, `HS01` – `HS24`. In addition, many screens have a "logical" name, which indicates what is being displayed on the screen. To see the logical names assigned to each screen:

```
$ SHOW LOGICAL/LNM$TABLE=TFTR_SCREENS
```

To view the plot currently displayed on one of these screens you can use either the physical or logical name:

```
$ XPLOT MAG_SCREEN_1
```

You can also get a printed copy of any screen. To print the graph currently on HS08 to the RT0 printer:

```
$ VPRINT/DEVICE=HS08/QUEUE=RT0
```

The `$ VSTATUS` command shows the status and who put up the most recent plot on all screens. Do `$ HELP VGDS` for more information.

5.6 Event-driven jobs

To have a VMS job run automatically for each TFTR shot as it happens, you need to set up the job to whatever processing is needed, and select the event to trigger on.

Some utilties, such as CUPLOT and LOCUS can be set up to wait for an event declared during the automatic shot processing and execute the procedure of your choice: Commonly used triggers are:

ANT_MOST is declared about 5 minutes after the shot when the second set of waveforms is shipped from CICADA to the cluster. This usually means that all the CICADA waveforms are now available.

ICRF_DONE is declared when the ICRF waveforms are ready.

Or you can set up a procedure file, which can be run interactively or as a batch job, to execute the command WAIT_FOR_SHOT which sleeps until the event is declared and sets the symbol SHOTNUMBER to the shot number:

```
$ WAIT_FOR_SHOT ANT_MOST          !wait for next shot
$ SHOW SYMBOL SHOTNUMBER
$ CUPLOT SH 'SHOTNUMBER @MYFILE
```

5.7 BQ—Analyze a Day's Shots from Batch

BQ (Batch Queue) is a system that generates shot lists and can return one shot at a time to your procedure. This lets you process a day's shots at any time.

The command GETSHOT_NO_BQ also defines the symbol SHOTNUMBER. However, instead of getting the shot numbers as the shot happens, GETSHOT_NO_BQ gets them from either HPPSHOTS.date in the directory TFTR_BQ. These files are lists of shot numbers generated from the ALLSHOTS table of the TFTR database. The files are generated after midnight by QUE_PERDAY. For yesterday's HPP shots use:

```
$ GETSHOT_NO_BQ "HPP"
$ GETSHOT_NO_BQ "HPP" 27-MAY-1994
```

where yesterday is the default date. TFTR_EXAMPLES:BQ_EXAMPLE.COM is an example of how to use BQ. Do `$ SETUP BQ` before using GETSHOT_NO_BQ. BQ is maintained by Jane Murphy.

5.8 QUE_PERDAY

QUE_PERDAY is a procedure that runs automatically each night. It can run BQ jobs or do file cleanup for you. See Tom Gibney if you want to execute a daily procedure from QUE_PERDAY.

6 Analysis Codes

The major analysis codes are TRANSP and SNAP. This document describes the utilities available to look at the results of these codes, but does not document how to run the codes.

6.1 SNAP—One Dimensional Power Balance

SNAP is a one dimensional equilibrium kinetic analysis code; it provides a snapshot transport analysis and diagnostic simulation at a given time. SNAP is run between shots for most good shots. Additional SNAP "tries" using different assumptions and models, and other profile sources, are frequently run later for interesting shots. SNAP is maintained by Marilee Thompson.

SNAPIN is the program that prepares input files for SNAP. Given a shot number and analysis time, SNAPIN reads diagnostic data from various Ufiles and waveforms and outputs it to

a namelist file that SNAP can read. If needed SNAPIN automatically restores the namelist information from a stored shot-try. To restore the namelist before running SNAPIN, do `$ SRERUNT 73268 252`. SNAPIN is maintained by Tom Gibney. The manual, "The SNAP User's Guide", is both an overview of SNAP functionality and a guide to executing SNAPIN. It is available on-line as `$ HELP SNAP`.

AUTOSNAP is a combination of a SNAPIN preprocessor and special features built into SNAPIN, that greatly facilitates using SNAPIN. For information, do `$ HELP @HLDAS AUTOSNAP`. AUTOSNAP is maintained by Darin Ernst.

6.2 TRANSP—Time Dependent Transport Analysis

TRANSP is a system for time dependent $1\frac{1}{2}$ dimensional transport and confinement analysis and diagnostic simulation of tokamak data. A TRANSP analysis requires extensive diagnostic input data which it uses to model the evolution of the tokamak plasma, computing solutions to equations describing poloidal field diffusion, particle balance, power balance, and momentum balance. Auxiliary models are provided for neutral beams, ICRF, sawteeth, and pellets. A wide range of options are provided for modeling of phenomena such as beam heating which cannot be directly measured. A variety of diagnostic simulations are computed. For a list of references describing the various physics models used in TRANSP, see the file `TRANSP$:[CMSREF.DOC]REFS.DOC`. TRANSP and related codes are maintained by Doug McCune, Dick Wieland, Ted Terpstra, Greg Hammett.

A typical input data set consists of the one and two dimensional Ufiles which contain: the evolution of the profiles of plasma electron density, electron and ion temperature, radiative power emission, the evolution of the plasma boundary location and shape, the evolution of toroidal field, plasma current, surface voltage, and magnetic estimate of plasma energy content, and impurity content, ion temperature and/or neutron emission. In addition, imposed experimental conditions are described, e.g., in the presence of neutral injection the beam powers, voltages, and geometries are specified. TRANSP output consists of a set of data files which may be accessed by the RPLOT interactive plotting program.

Because the amount of work involved in preparing UFILES data sets for TRANSP analysis is substantial, only a small number of shots are actually analyzed in this manner. Notes from a "Seminar on How to Prepare Input for TRANSP", are available from Doug McCune.

7 How to Write Your Own Applications

This section describes libraries and tools that will help you write your own applications. These libraries include graphics, reading and writing standard files, and and PPPL user interface routines. For an up-to-date list of where the libraries live and differences between VAX and AXP versions, see `$ HELP VMS_ALPHA`.

7.1 User Interfaces

Several user interface packages exist including: UREAD, TIC, TK/UREAD, INGRES forms, and SMG. UREAD, TIC, and TK/UREAD were written at PPL and are described in HELP files. UREAD is the most powerful and is recommended for diagnostic applications programs.

7.2 Graphics

Most TFTR software tools and analysis programs that generate graphs use the SG (Scientific Graphics) library, which is an enhanced version of the Textronix PLOT-10 library. SGLIB supports the 4014 and 4105 terminal classes. Output can go to a terminal, to a file, or to a file and a terminal simultaneously with only selected frames saved in the file. Files can later be displayed using XPLOT, printed, or displayed in the TFTR Control Room using VGDS. Do `$ HELP SG` for details. The SG library is maintained by Marilee Thompson.

The DISSPLA library is also available for writing graphics applicationns, along with `TK41` (Tektronix 4105) and `V550` (Visual 550) post-processors. Output to a terminal uses the SG routines to switch between the text and graphics screen for different terminal types. Do `$ HELP DISSPLA` for more information.

7.3 Read and Write Waveforms

Waveforms are a standard way to store physics data. A single waveform file exists for each shot and a *waveform* is an array of data in that file which is accessed by a *waveform name*. Subroutine DW_GET returns the waveform data, description, units, quality flag, and time base for file or derived waveforms. DW_GETH also returns the history comment and allows you to select the version number. The status code is 0 for success; non-zero values mean an error. (See `$ HELP CMLIB WF_ERRORS`.)

WF_STOH writes a waveform to the file. Anyone writing a waveform should first check with Michael Bell that the waveform name, description, units, etc., are consistent with PPL accepted waveform practices. The history comment should start with the date and the initials of the person creating the waveform. The rest of the comment is free-form, but should indicate why and/or what was used to generate the waveform.

Link your program with `USR:[LIB]CMLIB/OPT`. Sample code for reading and writing waveforms can be found in `TFTR_EXAMPLES`. Also see `$ HELP CMLIB DW_GET` and `USR:[DOC] WFUSE.DOC`.

7.4 Read Raw Data

The raw data (RDT) files, written in PPL *keyword* format, contain data from digitizers, timing modules, scalers, and other hardware devices. Basically there is one record for each device which has a 38 word header and the device data for the shot. If you need to decode

the header, see section S3470 (CD:DREAD) and section S1402 (Special Camac Modules) in Volume 3 of the CICADA handbook. (This handbook is available in the TFTR Control Room and from the Computer Division.) The CMLIB library has one set of routines to read any file in keyword format, and specialized routines to read RDT files. The routines return a status code of 0 for success, or one of the codes defined for the CICADA versions of the routines. Sample code to read raw data files can be found in `TFTR_EXAMPLES`. Also see `$ HELP KEYFILES`.

7.5 Read and Write Results

Ufiles are an alternative to waveform files for storing physics results data. Each Ufile contains a numerical representation of either a one dimensional or a two dimensional function. Ufiles can be accessed from Fortran applications programs by using the Ufiles subroutine library. The routines to read Ufiles automatically retrieve files from the optical disk when needed. See "The Ufiles Data File System" manual, available from Doug McCune. For "quick" help on writing Ufiles applications, see `$ HELP UFILES`.

Although Ufiles are the most common format for results files, you can read and write to the results disks in any format. However, you will then need to add code to your application to retrieve files from the optical disks. Suppose, for example, that you write a file other than a Ufile from a Fortran program. Later this file is archived on the opticals and skimmed. Now you want to read this file from your Fortran program. You will have to *copy* it from the opticals first. A sample program to do this is in `TFTR_EXAMPLES:OPEN_RESULTS_TO_READ.FOR` with a corresponding .COM file which shows how the sample program is linked.

7.6 Read and Write Database Data

The library INGLIB contains routines that permit a Fortran program to access data INGRES tables. On-line help is available by doing `$ HELP INGLIB`.

The library LOCUSLIB exists for accessing LOCUS ISAM data. See either the on-line file `TFTR_EXAMPLES:LOCUS_ISAM_RW*` or print `USR:[LOCUS] LOCUS.DOC`, the LOCUS ISAM document, both of which contain a sample program and the subroutine descriptions.

7.7 Read SNAP Data

To read the data use the routines SV_GET to read a vector and SS_GET to read a scaler. Sample programs are in `TFTR_EXAMPLES`. Link your program with`HLDAS$:[CURRENT]GLOBELIB/LIB` and `USR:[LIB]CMLIB/OPT`. For SNAP developers, routine MASTER_READ restores all SNAP for a given shot-try. See Marilee Thompson for additional help.

7.8 Read TRANSP Data

To read TRANSP data into a program, use the routines TRSCALAR and TRPROFIL. Full information on how to use these routines is given in the newly revised manual "Accessing TRANSP Output: A User's Guide", which is available from Doug McCune.

7.9 Directories for Applications

This describes where to put files associated with your application or diagnostic.

TFTR_INF is a directory that contains much useful information, in files with a .INF extension. This includes:

```
DIGRAPH.INF       list of digraphs
di.INF            information about digraph 'di'
WAVEFORMS.INF     descriptions of all waveforms
UFILES.INF        descriptions of Ufiles
XPnnn.INF         information on XP 'nnn'
PHONES.INF        phones of TFTR people and places
```

TFTR_PLT is a directory for .PLT files that are not needed on disk once they have been hardcopied or displayed during the shot. To use it, write to TFTR_PLT. For example:

```
$ DEFINE PLOT TFTR_PLT:MY_PLOT.PLT
```

will send your SG plot files to TFTR_PLT. Files in this area will be deleted after about a week.

TFTR_LOG is a directory for log files. Files in TFTR_LOG are automatically purged after about 4 days.

```
$ SUBMIT/LOG=TFTR_LOG: MY_JOB
$ DIR TFTR_LOG:MY_JOB.LOG
```

TFTR_EXAMPLES is a directory which contains example procedures and subroutine calls. Examples may be complete programs in .FOR files with corresponding .LNK files showing how to link the routines being described. Other examples may be program fragments or procedures.

TFTR_COM:[di] are directories for digraph-related executables and procedure files.

SCRATCH is a disk that is sometimes available. When available, it is intended to be used by anyone who needs lots of space for a short time. To see if it is available:

```
$ SHOW DEV SCRATCH
```

To use it:

```
$ CREATE/DIRECTORY SCRATCH:[MYNAME]
$ SET DEF SCRATCH:[MYNAME]
```

The scratch disk is *not* backed up.

8 Management of the User Disks

User disks are for source, procedures, text, executables, etc. Each user is given a quota on one of these disks. Most files are archived. Occasionally multiple versions that are more than two months old are purged.

A logical name is defined that allows access to that user's top level directory. All users should reference directories by these logical names. For example, use `REAL$:[CUPLOT]` instead of `RX1:[REAL.CUPLOT]` or `SCOTT$:[000000]` instead of `RX1:[SCOTT]` for the top level directory. Note the $ in the logical name. Since users are occasionally moved from one user disk to another, procedures will continue to work after the move if the logical names are used.

At regular intervals (currently once a month), operations runs a procedure that uses the ARCHIVER program to archive your files. The files that get archived include the highest version of each file in your entire directory structure, excluding directory files and "temporary" files, that meet either of the following conditions:

1. The file has not been previously archived by this system, or
2. The file has been modified since it was last archived by this system.

(A "temporary" file is defined to be a file with any of the following extensions: `.TMP`, `.TEM`, `.OBJ`, `.MAP`, `.LIS`, or `.LOG`). Once a file has been archived by this system, it is given an access control list (ACL) entry with an identifier of SYS$ARCHIVED. You can use this access control entry (ACE) to tell if a given file has been archived for you.

In addition to the regular archiving that is done for you by operations, you can force a file to be archived and given the SYS$ARCHIVED ACE by using the command ASAVE. To retrieve files archived by this system, you should use the command ARECOVER. The ARECOVER command can also be used to get a directory of what files have been archived and when they were archived. The ASAVE and ARECOVER commands run procedures that were written at PPPL to provide you with an easy interface to ARCHIVER. On startup ARECOVER accesses the most recent ARCHIVER *index*, which has information about all files saved since the last *index split*. If the file was saved before the start date for the present index, ARECOVER will fail to find it in the default index but will give you the option to search an older index.

To list files that have been archived, with the option to restore a file:

```
$ ARECOVER CBAY.FOR
ARC_RECOVER - Procedure To Recover Archived User Files
-- Archive V1.8 Brief listing of archived files for THOMPSON
71774.ARQ on 11-JUL-1989 20:21  HX2:[THOMPSON.RGA]CBAY.FOR;158
88546.ARQ on 15-AUG-1989 00:59  HX2:[THOMPSON.RGA]CBAY.FOR;161

SEQUENCE NUMBER(S) of file(s) to be restored (* for all): 88546
Enter directory for file restore [[THOMPSON.RGA]]:   <RETURN>
%ARC_RECOVER-I-FILPRO HX2:[THOMPSON.RGA]CBAY.FOR;161 processed
```

To cause a file to be archived:

```
$ ASAVE CBAY.FOR
```

To see if files on the disk have been archived:

```
$ DIR/ACL/DATE CBAY.FOR

CBAY.FOR;161          21-JUL-1989 12:50:24.60
          (IDENTIFIER=SYS$ARCHIVED,...)
CBAY.FOR;160          21-JUL-1989 02:04:36.62
CBAY.FOR;158          21-JUN-1989 09:48:56.56
          (IDENTIFIER=SYS$ARCHIVED,...)
```

For more information on the archiving and retrieving of user files, type `$ HELP @PPL ARCHIVE`. Report any problems using this system by sending MAIL to Jack Abraitis via GRIPE.

A Related Documents

Guide to CUPLOT on the VMS Cluster. Marilee Thompson. This is available from Marilee.

The LOCUS User's Guide, A Front-End to INGRES Databases Jane Murphy, Dick Wieland. This is available from Jane or Dick.

User's Guide to the MINGL Database System for TFTR. Dick Wieland. This is available from Dick.

The SNAP User's Guide. Jane Murphy, Steve Scott, Harry Towner.

Accessing TRANSP Output: A User's Guide. Ted Terpstra, Doug McCune. This is available from Ted.

Transferring VersaTermPro Graphics to MacDraw. Harry Towner, Marilee Thompson. This is available from Harry.

B Diagnostic Identifiers

ID	Diagnostic
BR	Bolometer Arrays
EF	Charge Exchange Fluctuations
EG	Fusion Gamma
EH	Horizontal Charge Exchange
EM	Charge Exchange Mini-Analyzer
EN	Diagnostic Neutral Beam
EV	Vertical Charge Exchange
FF	Faraday Rotation
FM	Multichannel InfraRed Interferometer (MIRI)
GS	Gas System
GT	Strain Gauges
IC	ICRF Heating
LI	Disruption Monitor
MA	Global Magnetics
MB	B-Theta Loops
MD	Diamagnetic Loops
MM	Mirnov B-Theta Loops
MR	Rogowski Loops
MV	Voltage Loops
N1-N5	Neutral Beams
NA	Neutron Activation System
NB	Neutral Beams
NE	Epithermal Neutrons
NF	Fusion Products
NL	Lost Alpha System
NM	Multichannel Neutron Collimator
NR	Lost Alpha Midplane Probe
NS	Neutron Fluctuations
PC	Power Conversion
PP	Plasma Position and Current Control
PX	Physics Configuration
RI	Pellet Injection
RP	RF Probe
SB	Beam Emission Spectrometer (BES)
SH	H-alpha Interference Filter (HAIFA)
SP	Survey Spectroscometer (SPRED)
SV	Visible Impurity Photometric Spect. (VIPS)
SW	Charge Exchange Recomb. Spect (CHERS)
SX	Soft X-ray Spectrometer (SOXMOS)
TE	Thomson Scattering Edge System
TS	Thomson Scattering I
TT	Second Thomson Scattering Pulse
TV	Thomson Scattering II
VP	Periscope Camera (Plasma TV - IR TV)
VG	Residual Gas Analyzer - C bay
VR	Residual Gas Analyzer - R bay
WE	Edge Probes

XC	X-ray Crystal Spectrometer
XH	Hard X-ray
XI	X-ray Imaging
XP	X-ray Pulse Height Analysis
XV	X-ray Crystal Vertical System
YF	Microwave Scattering Correlator
YG	Grating Polychrometer
YI	1mm Microwave Interferometer
YM	Michelson Interferometer
YR	Reflectometer
YS	Scanning Heterodyne Radiometer
YU	Microwave Scattering

Index

archive, 17, 21, 24, 25, 39, 41

bulletin boards, 14

Codes
  SNAP, 36
  TRANSP, 37
Commands
  ARECOVER, 41
  ASAVE, 41
  BULLETIN, 14
  COPYSHOTS, 18
  CUPLOT, 19, 26
  FERRET, 15
  GETSHOT_NO_BQ, 36
  GETSHOTS, 18
  LOCUS, 28
  LOOKUP, 16
  MINGL, 29
  MOSAIC, 16
  OPERPRINT, 28
  OPREAD, 22
  OPSTATUS, 22
  PSPLOT, 14
  RECENT_SNAPS, 23
  RESDIR, 21
  RESGET, 22
  RPLOT, 32
  RPLOTS, 32
  SNAPIN, 36
  SNAPSHOTS, 23
  SNAPSUM, 32
  SRERUNT, 24
  SURVEY, 23
  TASKPRINT, 28
  TPLOT, 14
  UGRAF1, 30
  UGRAF2, 30
  VSTATUS, 35
  WFDIR, 20
  WFGET, 20
  XPLOT, 12
  XTC, 11

Database, 17, 24, 29, 39
  display, 28
  tables
    ALLSHOTS, 25
    CONTENTS, 25
    E_TIMES, 25
    OPERLOG, 25
    SNAPP, 25
    SNAPS, 25
    TASKLOG, 25
    WAVEFTFTR, 25
diagnostic identifiers, 17, 44
display
  curve fit, 27
  FFT, 27
  raw data, 26
  SNAP data, 26, 31
  TRANSP data, 32
  UFILES, 26, 30
  waveforms, 26

Examples, 40
experiments, 7

Files
  log files, 40
  plot files, 40
  raw data, 17
  raw data files, 17, 18
  results files, 17, 21, 39
  scratch files, 40
  UFILES, 20, 39
  waveform files, 17, 19, 20

Information
  DIGRAF.INF, 17
  UFILES.INF, 20
  WAVEFORMS.INF, 19

Libraries
  CMLIB, 39
  raw data files, 39
  SGLIB, 38

SNAP routines, 39
TRANSP routines, 40
UFILES, 39
waveform routines, 38

optical disk system, 17
optical disks, 21, 22

phone numbers, 16
printers, 12
privileges, 8, 17

quota
raw data, 18

raw data, 38
retrieve
raw data, 18
results files, 22
waveforms, 20

shot comments, 28
shot number, 33
shot summary, 33
skimming
optical disk cache, 25
raw data, 18

terminal emulators, 11
VersaTerm, 11
XTC, 11
terminal settings, 10
TFTR, 7

user interfaces, 38