Update Notice #1

February 1987

VAX TDMS
Forms Manual

AD-GS13B-T1

NEW AND CHANGED INFORMATION

This update contains changes and additions made to the *VAX TDMS Forms Manual* for Version 1.7.

INSTRUCTIONS

Place the enclosed pages in the *VAX TDMS Forms Manual* Version 1.7 as replacements for or additions to current pages. Change bars on replacement pages indicate changed text. For new pages and pages where most of the text has been substantially revised, no change bars are used. Instead, only the Version 1.7 release date is shown on the bottom corner of the page.

Old Page	New Page
Title Page/Copyright	Title Page/Copyright
iii to viii	iii to viii
xiii/xiv	xiii
2-1/2-2	2-1/2-2
2-27 to 2-30	2-27 to 2-30
3-1/3-2	3-1/3-2
4-5/4-6	4-5/4-6
5-11/5-12	5-11/5-12
Reader's Comments/Mailer	Reader's Comments/Mailer

VAX TDMS
Forms Manual

Order No. AA-GS13B-TE
Including AD-GS13B-T1

February 1987

This manual describes the use of the TDMS Form Definition Utility (FDU), including the use of the form editor.

OPERATING SYSTEM: **VMS**
MicroVMS

SOFTWARE VERSION: **VAX TDMS V1.7**

digital equipment corporation, maynard, massachusetts

The information in this document is subject to change without notice and should not be construed as a commitment by Digital Equipment Corporation. Digital Equipment Corporation assumes no responsibility for any errors that may appear in this document.

The software described in this document is furnished under a license and may be used or copied only in accordance with the terms of such license.

No responsibility is assumed for the use or reliability of software on equipment that is not supplied by DIGITAL or its affiliated companies.

The postage-paid READER'S COMMENTS form on the last page of this document requests your critical evaluation to assist us in preparing future documentation.

The following are trademarks of Digital Equipment Corporation:

ACMS	MicroVMS	VAX
CDD	PDP	VAXcluster
DATATRIEVE	Rdb/ELN	VAXinfo
DEC	Rdb/VMS	VAX Information Architecture
DECnet	ReGIS	VIDA
DECUS	TDMS	VMS
MicroVAX	UNIBUS	VT

digital™

Contents

Tables

Technical Changes and New Features

VAX TDMS V1.7 has added two new commands to the Form Definition Utility (FDU). They are:

- ATTACH
- SPAWN

See the *VAX TDMS Reference Manual* for information about these new features.

Walking Through a Simple TDMS Application 2

A major function in any business enterprise is to collect information. Generally, the best way to do this is to use forms. Forms vary in clarity and complexity, as anyone who has filled out a government form knows. TDMS allows you not only to design the best form for your needs, but to customize how you want to organize, display, and collect that information.

A common business task is to keep track of employee data. Very simply, you need to know the employee's name, birth date, and identification number. In addition, if the employee is married, you may want to collect information about the spouse.

This chapter gives you step-by-step instructions on creating a simple TDMS application to collect employee information. In addition, the application lets you update records and display information. Feel free to adapt the application to your particular needs. The employee number, for example, can easily be changed to a social security number. This helps you to create a database that has meaning to you.

You should allow approximately two hours to complete this walkthrough, which is organized in the following main parts:

- Creating a form definition
- Creating records
- Creating requests
- Creating a request library definition
- Building a request library file
- Writing the application program

2.1 Getting Started

Before you can create a form definition, you must have an area in the CDD where you want to store your form definition (and any other TDMS definitions you create or use). If you do not have a CDD directory, (or are unsure of what a CDD directory is), refer to the *VAX Common Data Dictionary Utilities Reference Manual* before continuing with this walkthrough.

For example, you might want to store the definitions that you create or modify in a CDD subdirectory using your last name. In that case, type the following logical assignment (using your own last name) at DCL level:

```
$ DEFINE CDD$DEFAULT CDD$TOP.(your_last_name)
```

When you have defined CDD$DEFAULT in this manner, CDD precedes any partial references that you make to a CDD directory or object with CDD$TOP.(your_last_name).

The name you assign the form is the CDD given name as it is stored in the CDD.

In this walkthrough, the name you assign the form is FAMILY_FORM. Because you defined CDD$DEFAULT to point to your personal CDD directory earlier, FDU will store the form in that directory:

CDD$TOP.(name_of_your_directory).FAMILY_FORM

If you get an error when you enter the CREATE FORM command, then probably:

- You did not define CDD$DEFAULT to point to your personal CDD directory. Change your default CDD directory and try to create the request again.
- You already have a request named FAMILY_FORM in your personal CDD directory. Delete the existing form.

If neither of these corrects the problem, see your system manager.

2.2 Creating a Form Definition

Figure 2-1 shows the form that you create in this walkthrough as it might appear after the operator enters information for a new employee. The form definition that you create includes:

- **Background text** (characters that TDMS always displays when the form is on the screen)
- **Fields** (locations on the form in which TDMS displays data or where the operator can enter data)

Note

RDU usually remains at the RDUDFN> prompt and continues to take additional request instructions when you make an error. If RDU encounters an error in your request text from which it cannot recover, it may return you to the RDU> prompt before you enter the END DEFINITION instruction. If this happens, you must reenter the request starting with the CREATE REQUEST command.

2.4.1 Entering the Request Definition Utility (RDU)

To enter RDU, type the following command at DCL level:

```
$ RUN SYS$SYSTEM:RDU.EXE
```

The system responds with:

```
RDU>
```

Once you are in RDU, you can issue commands:

- To create requests and request library definitions
- To manipulate (modify, delete, copy, list, replace, and so on) requests and request library definitions

2.4.2 Creating a Simple Request

You can create a request using one of two methods: interactive or file. In this walkthrough, you use the interactive method. The file method is described in the *VAX TDMS Request and Programming Manual.*

To create the requests interactively, type the CREATE REQUEST command and a request name at the RDU> prompt. To start creating your first request, enter the following:

```
RDU> CREATE REQUEST FAMILY_DISPLAY_REQUEST
```

After you enter the CREATE REQUEST command and name the request, RDU displays the RDUDFN> prompt. This prompt indicates that RDU is ready to receive request instructions.

```
RDUDFN>
```

In the next sections, you enter the instructions that make up the FAMILY_DISPLAY_REQUEST. This request clears the terminal screen and displays the form FAMILY_FORM. It also takes data from the database record FAMILY_RECORD and displays it in the form fields. Figure 2-5 shows the two parts of a request:

- The header, which identifies the forms and records used by the request
- The base, which contains instructions that TDMS performs every time an application program calls this request

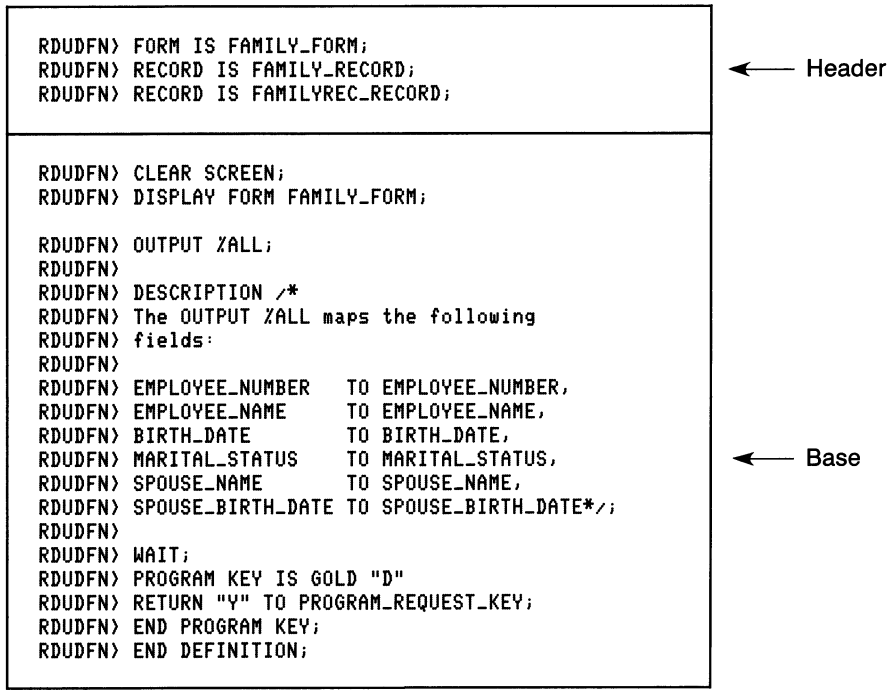

Figure 2-5: Two Parts of FAMILY_DISPLAY_REQUEST

The request header contains the FORM IS and RECORD IS instructions. You must enter these instructions before any mapping instructions.

The request base contains form usage and mapping instructions that TDMS reads and executes each time an application program calls that request.

2.4.2.1 FORM IS Instruction — Usually the first instruction in a request is the FORM IS instruction. This instruction identifies the form or forms you refer to in later instructions.

As you enter instructions, RDU checks that the form you specify exists in the CDD. If the form does not exist, RDU gives you an error message and does not create the request in the CDD. RDU does, however, continue to accept further request instructions and check them for errors.

The form name must be a legal CDD path name. You can select either a *given*, a *full*, or a *relative* path name.

Enter this text:

```
RDUDFN> FORM IS FAMILY_FORM;
```

Be sure to complete each instruction with a semicolon.

A single call to a request can display no more than one form. However, you can identify more than one form definition in a request containing conditional instructions. You will create a conditional request later in this walkthrough.

2.4.2.2 RECORD IS Instruction — You must also name the CDD record definitions you will use later in mapping instructions within the request.

Because you created two records, you need two RECORD IS instructions. Enter this text:

```
RDUDFN> RECORD IS FAMILY_RECORD;
RDUDFN> RECORD IS FAMILYREC_RECORD;
```

2.4.2.3 CLEAR SCREEN Instruction — You want to clear the screen before displaying a form. To do this, use the CLEAR SCREEN instruction. This ensures that there is nothing on the screen before TDMS displays a form. Enter this text:

```
RDUDFN> CLEAR SCREEN;
```

2.4.2.4 DISPLAY FORM Instruction — To view the form on the screen, use the DISPLAY FORM instruction. Enter this text:

```
RDUDFN> DISPLAY FORM FAMILY_FORM;
```

2.4.2.5 OUTPUT TO Instruction — In your application, you want TDMS to move data from the records and display it on the form. To do this, you use the OUTPUT TO instruction with the %ALL parameter. To continue creating your request, enter the following:

```
RDUDFN) OUTPUT %ALL;
```

If you use %ALL, TDMS displays data to all those form fields that have identically named record fields. Note that you do not need to specify any record fields within FAMILY_RECORD.

2.4.2.6 DESCRIPTION Instruction — You can use the DESCRIPTION instruction any place in a request or request library definition where you want to include descriptive text except embedded in a request instruction or a request library definition instruction. The text you enter following the keyword DESCRIPTION and the slash and asterisk symbols (/*) is stored with the request or request library definition in the CDD. You end the descriptive text with the asterisk and slash symbols and a semicolon (*/;).

As in this example, you might want to list which fields are being mapped for output. This is done simply for clarity; the OUTPUT %ALL instruction does the actual mapping. Enter this text:

```
RDUDFN) DESCRIPTION /*
RDUDFN) The OUTPUT %ALL maps the following
RDUDFN) fields:
RDUDFN)
RDUDFN)    EMPLOYEE_NUMBER   TO EMPLOYEE_NUMBER,
RDUDFN)    EMPLOYEE_NAME     TO EMPLOYEE_NAME,
RDUDFN)    BIRTH_DATE        TO BIRTH_DATE,
RDUDFN)    MARITAL_STATUS    TO MARITAL_STATUS,
RDUDFN)    SPOUSE_NAME       TO SPOUSE_NAME,
RDUDFN)    SPOUSE_BIRTH_DATE TO SPOUSE_BIRTH_DATE */;
```

2.4.2.7 WAIT Instruction — You must use the WAIT instruction to ensure that the form and the information you mapped to it stay on the screen until you press the RETURN key, PRK, or other termination key at run time. Enter this text:

```
RDUDFN) WAIT;
```

2.4.2.8 PROGRAM KEY IS Instruction — You might want to leave the application while it is running. A convenient way to do this is to use a program request key (PRK).

Using the Form Definition Utility (FDU) and the Form Editor 3

This chapter describes some important FDU commands, including those that let you use the form editor, and provides an overview of the form editor. The rest of this manual provides a more detailed explanation. Complete information about the syntax of all FDU commands can be found in the *VAX TDMS Reference Manual.*

FDU allows you to create, modify, and store customized form definitions. The product of your work with the form editor is a form definition, which FDU stores in the Common Data Dictionary (CDD). The form definition contains the information that identifies:

- Screen image of the form. The screen image includes the location of background text and fields as well as video highlighting. (**Background text** is text that is always displayed when the form is displayed; **fields** are locations on the form where data can be collected or displayed.)
- Length and data type of each field.
- Set of attributes for each field on the form (including means for validating data).
- Location of scrolled regions on the form.
- Name of a Help form, which the operator can display at run time.

3.1 Entering FDU

To enter FDU from DCL level, type the command:

```
$ RUN SYS$SYSTEM:FDU.EXE
```

You can set up a global symbol in your login command file to allow you to enter FDU commands at DCL level. For example:

```
$ FDU :== $FDU
```

When you enter FDU, the FDU> prompt is displayed on your terminal. When the FDU> prompt is displayed, you can give only valid FDU commands.

This chapter discusses the use of several FDU commands that you will use most frequently, including those that allow you to use the form editor and create a listing that provides information about the form definition and objects in the CDD.

3.2 Leaving FDU

To leave FDU, type EXIT or press CTRL/Z. Both of these commands will return you to DCL level. For example:

```
FDU> EXIT
$
```

or

```
FDU> <CTRL/Z>
$
```

3.3 Using the Form Editor

You can use the form editor only when you issue one of the following commands at FDU level:

- CREATE FORM
- MODIFY FORM
- REPLACE FORM

All FDU commands can be abbreviated to their shortest unambiguous form.

3.3.1 Creating a New Form Definition

To create a new form definition, enter FDU. At the FDU> prompt, issue the command CREATE FORM followed by the form name that you want to assign to the form definition. The form name is generally the CDD path name of the form definition; it is stored in your default CDD directory unless you explicitly state a different location.

In the Form phase, you can specify field attributes that are assigned by default to any field you subsequently create in the form while in the current FDU session. Any field attribute that has been assigned by default in the Form phase can be deassigned in the Assign phase. Similarly, you can assign any attribute in the Assign phase, regardless of the default specified in the Form phase.

To assign default field attributes, type Y at the question on the Form Attributes form:

```
"Do you wish to assign default field attributes?"
```

When you complete the Form phase by pressing RETURN, the form editor displays a list of field attributes. To return to the Phase Selection menu, press GOLD-KP7. Figure 4-2 shows the Default Attributes for New Fields form.

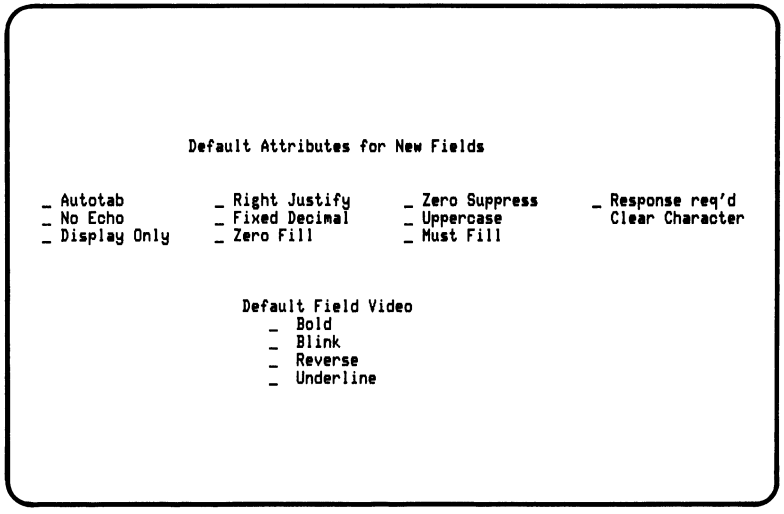

Figure 4-2: Default Attributes for New Fields Form

Enter X next to those attributes that you want to assign by default; press the space bar when the cursor is next to those attributes that you want to deassign. For Default Value and Help Text selections, type in the text you wish to be displayed by default for each field.

Remember that the default field attributes that you specify apply *only* to fields that you create later in the Layout phase. If you leave FDU, save the form, and then later modify it, any new fields will not have the defaults. You do not affect the attributes of any fields that have already been created when you specify default field attributes.

4.3 Help Forms

Help forms are forms that provide information to the operator at run time. They are created in FDU and stored in the CDD. Help forms should include only background text since you cannot map any fields on a Help form for input or output.

At run time, if the cursor is located at a field that has a help message, the help message is displayed the first time that the operator presses the HELP key (PF2 or F15). This message is displayed on the last line of the screen and is created in the Assign phase. See chapter 6, Assigning Field Attributes and Validators, for more information. When the operator presses the HELP key a second time, the entire screen is cleared (regardless of the screen area used by the Help form) and a Help form is displayed.

Each Help form can have one Help form assigned to it in the Form phase. These additional forms are displayed each time you press the HELP key.

You can have a single Help form for any form and a single help message for each field on a form.

If your form has an associated Help form, enter the name of the Help form. It is good practice to use the full CDD path name (CDD$TOP.PAYROLL.HELPFORM, for example) when identifying a Help form.

If the name of your Help form continues beyond one line, you can continue typing and the text automatically wraps to the next line. Whenever the name of a Help form is on more than one line, FDU concatenates all of the text.

4.4 Assigning Input Field Highlighting

VAX TDMS allows you to specify video highlighting for fields when they are available (open) for input at run time. The video attributes that you assign for input fields during the Form phase *replace* other video attributes that you might assign to the field during the Layout phase but *only* when the field is the current input field. If the cursor is not in the current input field, the video attributes assigned in the Layout phase override those set in the Form phase.

Video characteristics set by a request override the input field highlighting video attribute.

Table 5-4: Form Field Data Types (Cont.)

Picture Characters	Field Data Type
Any combination of identifiers	TEXT
All 9s or all Ns with size field validator	Named size field validator (UNSIGNED BYTE, SIGNED WORD, and so on)
Date (inserted using DATE key)	DATE
Time (inserted using TIME key)	TIME

For example, a form field defined as AAAAA is a five-character TEXT field, and a field defined as NNN is a three-character SIGNED NUMERIC field. You can also use field validators to define numeric fields to have data types of SIGNED or UNSIGNED BYTE, WORD, or LONGWORD or SIGNED QUADWORD. You learn how to use and assign field validators in Chapter 6.

You can use combinations of picture characters to identify a field. For example, if your invoice numbers are always two letters followed by five numbers, you would define a field as:

```
AA99999
```

At run time, if that field is open for operator input, the operator can enter only alphabetic characters in the first two positions and only numeric characters in the last five positions. (You can impose additional validation of operator input for each field using field validators, as described in Chapter 6.)

Any field that includes at least one Text picture character has a TEXT data type. The following field pictures show fields with TEXT data types:

```
AAAAAA
(999)CC9-9999
XX-9999
A.X.B.9
```

In a TDMS application, the data type of a form field must be compatible with the data type of any record field to which the form field is mapped in a request. If the form and record fields are incompatible, TDMS request library files cannot be built.

5.6.5 Creating Special Fields: DATE and TIME Keys

DATE-FIELD key (GOLD-D)

The form editor provides a special function key that automatically inserts a Date field in the form definition. To use the DATE-FIELD key, press the sequence GOLD-D. When you press DATE-FIELD, the cursor status line clears and five date formats are displayed on the bottom line of the screen:

```
1 Month Day, Year (AAAAAAAAAB99,B9999)
2 Day-Month-Year (99-AAA-99)
3 Month/Day/Year (99/99/99)
4 Day-Month-Year (99-99-99)
5 Day-Month-Year (99-AAA-9999)
```

To select a date format, type the appropriate number for the format and press RETURN. The form editor inserts a Date field in the format that you chose on the form. If there is insufficient space on the line for the Date field, the form editor signals an error. Once the Date field is inserted, you can delete only the entire field and not a portion of it.

When you create a field using the DATE-FIELD key, the field has a DATE data type. If a form contains a date field, TDMS displays the current date in the Date field unless a default value on the form or an output mapping in the request specifically overrides it.

TIME-FIELD key (GOLD-T)

The form editor provides a special function key that automatically inserts a Time field on the form definition. To use the TIME-FIELD key, press the sequence GOLD-T.

When you press TIME-FIELD, the cursor status line clears and two time formats are displayed on the bottom line of the screen. Choose one of the two formats by typing the appropriate number. The two time field formats are:

```
1 hour:minute:second (99:99:99)
2 hour:minute:AM/PM (99:99BAA)
```

VAX TDMS
Forms Manual
AA-GS13B-TE
Including: AD-GS13B-T1

Reader's Comments

Note: This form is for document comments only. DIGITAL will use comments submitted on this form at the company's discretion. If you require a written reply and are eligible to receive one under Software Performance Report (SPR) service, submit your comments on an SPR form.

Did you find this manual understandable, usable, and well-organized? Please make suggestions for improvement. _____

_____

_____

_____

Did you find errors in this manual? If so, specify the error and the page number.

_____

_____

_____

Please indicate the type of user/reader that you most nearly represent.

- ☐ Assembly language programmer
- ☐ Higher-level language programmer
- ☐ Occasional programmer (experienced)
- ☐ User with little programming experience
- ☐ Student programmer
- ☐ Other (please specify) _____

Name _____ Date _____

Organization _____

Street _____

City _____ State _____ Zip Code or Country _____

Do Not Tear - Fold Here and Tape

digital

No Postage Necessary if Mailed in the United States

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO.33 MAYNARD MASS

POSTAGE WILL BE PAID BY ADDRESSEE

ATTN: DISG Documentation
ZK02-2/N53
Digital Equipment Corporation
110 Spit Brook Road
Nashua, NH 03062-2698

Do Not Tear - Fold Here and Tape

Cut Along Dotted Line