

AD-A182 581 INTEGRATED INFORMATION SUPPORT SYSTEM (IIS) VOLUME 8

12

SCHEENECTADY NY PRODUCTION RESOURCES CONSU

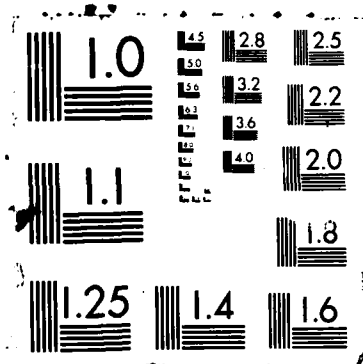
UNCLASSIFIED

C MORENC ET AL 01 NOV 85 UM-620144400B

F/G 12/5

MI

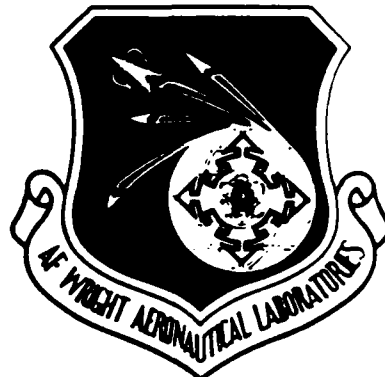
A 10x10 grid of squares. The grid is mostly black, with a few white squares forming a sparse pattern. The pattern includes a small cluster of white squares in the top-left corner and a larger, more complex shape in the bottom-right corner.



AD-A182 581

AFWAL-TR-86-4006
Volume VIII
Part 13

DTIC FILE COPY



INTEGRATED INFORMATION
SUPPORT SYSTEM (IISS)
Volume VIII - User Interface Subsystem
Part 13 - Forms Editor User Manual

General Electric Company
Production Resources Consulting
One River Road
Schenectady, New York 12345

Final Report for Period 22 September 1980 - 31 July 1985

November 1985

Approved for public release; distribution is unlimited.

MATERIALS LABORATORY
AIR FORCE WRIGHT AERONAUTICAL LABORATORIES
AIR FORCE SYSTEMS COMMAND
WRIGHT-PATTERSON AFB, OH 45433-6533

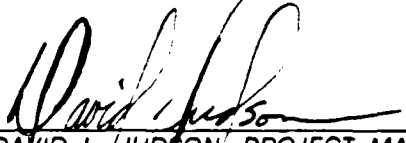
DTIC
ELECTE
JUL 16 1987
S E

NOTICE

When Government drawings, specifications, or other data are used for any purpose other than in connection with a definitely related Government procurement operation, the United States Government thereby incurs no responsibility nor any obligation whatsoever; and the fact that the government may have formulated, furnished, or in any way supplied the said drawings, specifications, or other data, is not to be regarded by implication or otherwise as in any manner licensing the holder or any other person or corporation, or conveying any rights or permission to manufacture, use, or sell any patented invention that may in any way be related thereto.

This report has been reviewed by the Office of Public Affairs (ASD/PA) and is releasable to the National Technical Information Service (NTIS). At NTIS, it will be available to the general public, including foreign nations.

This technical report has been reviewed and is approved for publication.



DAVID L. JUDSON, PROJECT MANAGER
AFWAL/MLTC
WRIGHT PATTERSON AFB OH 45433

5 Aug 1986

DATE

FOR THE COMMANDER:



GERALD C. SHUMAKER, BRANCH CHIEF
AFWAL/MLTC
WRIGHT PATTERSON AFB OH 45433

7 Aug 86

DATE

"If your address has changed, if you wish to be removed from our mailing list, or if the addressee is no longer employed by your organization please notify AFWAL/MLTC, W-PAFB, OH 45433 to help us maintain a current mailing list."

Copies of this report should not be returned unless return is required by security considerations, contractual obligations, or notice on a specific document.

REPORT DOCUMENTATION PAGE

1a REPORT SECURITY CLASSIFICATION Unclassified		1b RESTRICTIVE MARKINGS	
2a SECURITY CLASSIFICATION AUTHORITY		3 DISTRIBUTION/AVAILABILITY OF REPORT Approved for public release; distribution is unlimited.	
2b DECLASSIFICATION/DOWNGRADING SCHEDULE			
4 PERFORMING ORGANIZATION REPORT NUMBER(S)		5. MONITORING ORGANIZATION REPORT NUMBER(S) AFVAL-TR-86-4006 Vol VIII, Part 13	
6a NAME OF PERFORMING ORGANIZATION General Electric Company Production Resources Consulting	6b OFFICE SYMBOL (If applicable) AFVAL/MLTC	7a NAME OF MONITORING ORGANIZATION AFVAL/MLTC	
6c ADDRESS (City, State and ZIP Code) 1 River Road Schenectady, NY 12345		7b. ADDRESS (City, State and ZIP Code) VPAFB, OH 45433-6533	
8a NAME OF FUNDING/SPONSORING ORGANIZATION Materials Laboratory Air Force Systems Command, USAF	8b. OFFICE SYMBOL (If applicable) AFVAL/MLTC	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER F33615-80-C-5155	
8c ADDRESS (City, State and ZIP Code) Wright-Patterson AFB, Ohio 45433		10 SOURCE OF FUNDING NOS	
11 TITLE (Include Security Classification) (See Reverse)		PROGRAM ELEMENT NO. 78011F	PROJECT NO. 7500
12 PERSONAL AUTHOR(S) Morenc, Carol and Stafford, Frances		TASK NO. 62	WORK UNIT NO. 01
13a TYPE OF REPORT Final Technical Report	13b TIME COVERED 22 Sept 1980 - 31 July 1985	14. DATE OF REPORT (Yr., Mo., Day) 1985 November	15. PAGE COUNT 133
16 SUPPLEMENTARY NOTATION ICAM Project Priority 6201 The computer software contained herein are theoretical and/or references that in no way reflect Air Force-owned or -developed computer software.			
17 COSATI CODES		18 SUBJECT TERMS (Continue on reverse if necessary and identify by block number)	
FIELD	GROUP	SUB GR	
1308	0905		
19 ABSTRACT (Continue on reverse if necessary and identify by block number) This manual explains how to define and maintain electronic forms using the Form Definition Language and the Forms Driven Form Editor. General electronic form characteristics are also described.			
20 DISTRIBUTION/AVAILABILITY OF ABSTRACT UNCLASSIFIED/UNLIMITED ☒ SAME AS RPT ☐ DTIC USERS ☐		21 ABSTRACT SECURITY CLASSIFICATION Unclassified	
22a NAME OF RESPONSIBLE INDIVIDUAL David L. Judson		22b TELEPHONE NUMBER (Include Area Code) 813-255-6976	22c OFFICE SYMBOL AFVAL/MLTC

11. Title

Integrated Information Support System (IISS)
Vol VIII - User Interface Subsystem
Part 13 - Forms Editor User Manual

A S D 86 0031
9 Jan 1986

Accession For	
NTIS GRA&I	☒
DTIC TAB	☐
Unannounced	☐
Justification	
By	
Distribution/	
Availability Codes	
Dist	Avail and/or Special
A-1	



PREFACE

This user's manual covers the work performed under Air Force Contract F33615-80-C-5155 (ICAM Project 6201). This contract is sponsored by the Materials Laboratory, Air Force Systems Command, Wright-Patterson Air Force Base, Ohio. It was administered under the technical direction of Mr. Gerald C. Shumaker, ICAM Program Manager, Manufacturing Technology Division, through Project Manager, Mr. David Judson. The Prime Contractor was Production Resources Consulting of the General Electric Company, Schenectady, New York, under the direction of Mr. Alan Rubenstein. The General Electric Project Manager was Mr. Myron Hurlbut of Industrial Automation Systems Department, Albany, New York.

Certain work aimed at improving Test Bed Technology has been performed by other contracts with Project 6201 performing integrating functions. This work consisted of enhancements to Test Bed software and establishment and operation of Test Bed hardware and communications for developers and other users. Documentation relating to the Test Bed from all of these contractors and projects have been integrated under Project 6201 for publication and treatment as an integrated set of documents. The particular contributors to each document are noted on the Report Documentation Page (DD1473). A listing and description of the entire project documentation system and how they are related is contained in document FTR620100001, Project Overview.

The subcontractors and their contributing activities were as follows:

TASK 4.2

Subcontractors

Role

Boeing Military Aircraft
Company (EMAC)

Reviewer.

D. Appleton Company
(DACOM)

Responsible for IDEF support,
state-of-the-art literature
search.

General Dynamics/
Ft. Worth

Responsible for factory view
function and information
models.

UM 620144400B
1 November 1985

<u>Subcontractors</u>	<u>Role</u>
Illinois Institute of Technology	Responsible for factory view function research (IITRI) and information models of small and medium-size business.
North American Rockwell	Reviewer.
Northrop Corporation	Responsible for factory view function and information models.
Pritsker and Associates	Responsible for IDEF2 support.
SofTech	Responsible for IDEFO support.

TASKS 4.3 - 4.9 (TEST BED)

<u>Subcontractors</u>	<u>Role</u>
Boeing Military Aircraft Company (BMAC)	Responsible for consultation on applications of the technology and on IBM computer technology.
Computer Technology Associates (CTA)	Assisted in the areas of communications systems, system design and integration methodology, and design of the Network Transaction Manager.
Control Data Corporation (CDC)	Responsible for the Common Data Model (CDM) implementation and part of the CDM design (shared with DACOM).
D. Appleton Company (DACOM)	Responsible for the overall CDM Subsystem design integration and test plan, as well as part of the design of the CDM (shared with CDC). DACOM also developed the Integration Methodology and did the schema mappings for the Application Subsystems.

UM 620144400B
1 November 1985

Subcontractors

Role

Digital Equipment
Corporation (DEC)

Consulting and support of the
performance testing and on DEC
software and computer systems
operation.

McDonnell Douglas
Automation Company
(McAuto)

Responsible for the support and
enhancements to the Network
Transaction Manager Subsystem
during 1984/1985 period.

On-Line Software
International (OSI)

Responsible for programming the
Communications Subsystem on the
IBM and for consulting on the
IBM.

Rath and Strong Systems
Products (RSSP) (In 1985
became McCormack & Dodge)

Responsible for assistance in
the implementation and use of
the MRP II package (PIOS) that
they supplied.

SofTech, Inc.

Responsible for the design and
implementation of the Network
Transaction Manager (NTM) in
1981/1984 period.

Software Performance
Engineering (SPE)

Responsible for directing the
work on performance evaluation
and analysis.

Structural Dynamics
Research Corporation
(SDRC)

Responsible for the User
Interface and Virtual Terminal
Interface Subsystems.

Other prime contractors under other projects who have
contributed to Test Bed Technology, their contributing
activities and responsible projects are as follows:

<u>Contractors</u>	<u>ICAM Project</u>	<u>Contributing Activities</u>
Boeing Military Aircraft Company (BMAC)	1701, 2201, 2202	Enhancements for IBM node use. Technology Transfer to Integrated Sheet Metal Center (ISMC).

UM 620144400B
1 November 1985

<u>Contractors</u>	<u>ICAM Project</u>	<u>Contributing Activities</u>
Control Data Corporation (CDC)	1502, 1701	IISS enhancements to Common Data Model Processor (CDMP).
D. Appleton Company (DACOM)	1502	IISS enhancements to Integration Methodology.
General Electric	1502	Operation of the Test Bed and communications equipment.
Hughes Aircraft Company (HAC)	1701	Test Bed enhancements.
Structural Dynamics Research Corporation (SDRC)	1502, 1701, 1703	IISS enhancements to User Interface/Virtual Terminal Interface (UI/VTI).
Systran	1502	Test Bed enhancements. Operation of Test Bed.

TABLE OF CONTENTS

	<u>Page</u>
SECTION 1.0 INTRODUCTION	1-1
SECTION 2.0 DOCUMENTS	2-1
2.1 Reference Documents	2-1
2.2 Terms and Abbreviations	2-3
SECTION 3.0 ELECTRONIC FORM CHARACTERISTICS	3-1
3.1 Fields	3-1
3.1.1 Item Fields	3-1
3.1.2 Form Fields	3-1
3.1.3 Window Fields	3-2
3.2 Text	3-2
3.3 Interactive	3-2
3.4 Attributes	3-3
3.4.1 Background	3-3
3.4.2 Calculated Value	3-3
3.4.3 Conversion	3-3
3.4.4 Data Type	3-4
3.4.5 Entry	3-4
3.4.6 Display	3-4
3.4.7 Help	3-4
3.4.8 Identification	3-4
3.4.9 Justification	3-5
3.4.10 Location	3-5
3.4.11 Repetition	3-5
3.4.12 Size	3-5
3.4.13 Scroll	3-5
3.4.14 Value Limit	3-5
3.5 Defining Electronic Forms	3-6
3.6 Storing Forms in the IISS Environment	3-6
SECTION 4.0 FORM DEFINITION LANGUAGE	4-1
4.1 FDL Format Notation	4-1
4.2 Form Definition Language Syntax	4-2
4.3 Form Definition Syntax	4-6
4.3.1 CREATE FORM Statement	4-6
4.3.2 SIZE Clause	4-6
4.3.3 BACKGROUND Clause	4-7
4.3.4 PROMPT Clause	4-7
4.3.5 Field Definition Syntax	4-8
4.3.5.1 Item Fields	4-8
4.3.5.1.1 ITEM Statement	4-9

4.3.5.1.2	Location Clause	4-9
4.3.5.1.3	SIZE Clause	4-9
4.3.5.1.4	VALUE Clause	4-10
4.3.5.1.5	DISPLAY AS Clause	4-11
4.3.5.1.6	DOMAIN Clause	4-12
4.3.5.1.7	HELP Clause	4-13
4.3.5.1.8	PROMPT Clause	4-14
4.3.5.2	Form Fields	4-14
4.3.5.2.1	FORM Statement	4-14
4.3.5.2.2	Location Clause	4-15
4.3.5.2.3	SIZE Clause	4-15
4.3.5.2.4	PROMPT Clause	4-16
4.3.5.3	Window Fields	4-16
4.3.5.3.1	WINDOW Statement	4-17
4.3.5.3.2	Location Clause	4-17
4.3.5.3.3	SIZE Clause	4-17
4.3.5.3.4	BACKGROUND Clause	4-18
4.3.5.3.5	PROMPT Clause	4-18
4.3.6	Location Clause/Parameter	4-19
4.3.6.1	Absolute Location	4-21
4.3.6.2	Relative Location	4-23
4.3.6.3	Relative Location (Above/Below)	4-24
4.3.6.4	Relative Location (Right/Left)	4-26
4.3.6.5	Location Relative to Two Fields	4-27
4.3.6.6	Combination Location	4-29
4.3.7	Repeat Spec Parameter	4-32
4.3.7.1	Row of Fields	4-33
4.3.7.2	Array of Fields	4-33
4.3.7.3	Array of an Array of Fields	4-34
4.3.7.4	Single Dimension Scrolled Array	4-34
4.3.7.5	Two Dimensional Scrolled Array	4-35
4.4	Statement Format	4-36
4.5	Restrictions	4-36
4.6	Abbreviations	4-36
4.7	Including Comments	4-36
4.8	Reserved Words	4-37

SECTION 5.0	FORMS DRIVEN FORM EDITOR	5-1
5.1	FDFE Functional Organization	5-1
5.2	FDFE Editing Features	5-2
5.3	System Operation	5-3
5.3.1	Accessing the FDFE	5-3
5.4	Work Task Screen	5-4
5.4.1	Choosing a Work Task	5-4
5.4.2	LS (List FDL Sources)	5-6
5.4.2.1	List FDL Source Files Screen	5-8
5.4.3	IS (Insert FDL Source)	5-8

5.4.4	MS (Modify FDL Source)	5-9
5.4.5	SS (Select FDL Source)	5-11
5.4.6	CS (Copy FDL Source)	5-12
5.4.7	RS (Rename FDL Source)	5-13
5.4.8	DS (Drop FDL Source)	5-14
5.4.9	LC (List Compiled form definitions)	5-15
5.4.9.1	List of Compiled Form Definitions Screen	5-16
5.4.10	VC (View Compiled form definition)	5-16
5.4.11	DC (Drop Compiled form definition)	5-18
5.4.12	EX (EXit form driven form editor)	5-19
5.5	Edit Task Screen	5-19
5.5.1	Choosing an Edit Task	5-20
5.5.2	LF (List Forms)	5-20
5.5.2.1	List of Forms Screen	5-22
5.5.3	WF (Write Form)	5-22
5.5.4	VF (View a Form)	5-24
5.5.5	SF (Select a Form)	5-25
5.5.5.1	Using Single Field Mode to Review a Form	5-26
5.5.5.2	Using Form Mode to Review a Form	5-29
5.5.5.3	Using Layout Mode to Review a Form	5-34
5.5.6	IF (Insert a Form)	5-34
5.5.6.1	Using Single Field Edit Mode to Define a Form	5-35
5.5.6.1.1	Form Area Fields	5-36
5.5.6.1.2	Required Field Information	5-38
5.5.6.1.3	Optional Field Information	5-40
5.5.6.1.4	Item Only Information	5-43
5.5.6.2	Using Form Mode to Define a Form	5-43
5.5.6.3	Using Layout Mode to Define a Form	5-46
5.5.6.3.1	Layout Mode Symbols	5-47
5.5.7	MF (Modify a Form)	5-49
5.5.7.1	Using Single Field Mode to Modify a Form	5-50
5.5.7.2	Using Form Mode to Modify a Form	5-53
5.5.7.3	Using Layout Mode to Modify a Form	5-56
5.5.8	DF (Drop a Form)	5-57
5.5.9	EW (Exit Write)	5-58
5.5.10	EC (Exit Compile)	5-59
5.5.11	EN (EXIT NO SAVE)	5-59
5.6	Limited Edit Task Screen	5-60
SECTION 6.0	FDL COMPILER - FLAN	6-1
6.1	FLAN Error Messages	6-1
6.1.1	Warning Message	6-1
6.1.2	Error Messages	6-1

6.1.3	Fatal Messages	6-4
SECTION 7.0	FORM TOOLS	7-1
7.1	Generating Include Members	7-1
SECTION 8.0	SAMPLE FORM DEFINITION	8-1

FIGURES

FIGURE 3-1	IISS Form Storage Hierarchy	3-7
4-1	Field Reference Points	4-20
4-2	Absolute Location	4-21
4-3	Relative Location	4-23
4-4	Relative Location (Above/Below)	4-24
4-5	Relative Location (Right/Left)	4-26
4-6	Location Relative to Two Fields	4-27
4-7	Absolute Row/Relative Column	4-29
4-8	Absolute Column/Relative Row	4-30
4-9	Row of Fields	4-33
4-10	Array of Fields	4-33
4-11	Array of An Array of Fields	4-34
4-12	Single Dimension Scrolled Array	4-35
4-13	Two Dimensional Scrolled Array	4-35
5-1	FDFE Functions Mapped to the IISS Form Storage Hierarchy	5-2
5-2	VT100 Keypad Layout for the FDFE	5-3
5-3	Work Task Screen	5-4
5-4	Choosing a Work Task - Menu Selection	5-5
5-5	Choosing a Work Tas - Command Entry	5-6
5-6	Choosing LS Using Menu Selection	5-7
5-7	List FDL Source Files Screen	5-8
5-8	Choosing IS Using Manual Selection	5-9
5-9	Choosing MS Using Menu Selection	5-10
5-10	Choosing SS Using Menu Selection	5-11
5-11	Choosing CS Using Menu Selection	5-12
5-12	Choosing RS Using Menu Selection	5-13
5-13	Choosing DS Using Menu Selection	5-14
5-14	Choosing LC Using Menu Selection	5-15
5-15	List Compiled Form Definitions Screen	5-16
5-16	Choosing VC Using Menu Selection	5-17
5-17	Choosing DC Using Menu Selection	5-18
5-18	Edit Task Screen	5-19
5-19	Choosing LF Using Menu Selection	5-21
5-20	List of Forms Screen	5-22
5-21	Choosing WF Using Menu Selection	5-23

5-22	Choosing CF Using Menu Selection	5-24
5-23	Choosing SF Using Menu Selection	5-26
5-24	Reviewing a Form in Single Field Mode	5-27
5-25	Reviewing a Form in Form Mode	5-29
5-26	FIELD CHARACTERISTICS TABLE-Part One	5-30
5-27	FIELD CHARACTERISTICS TABLE-Part Two	5-31
5-28	FIELD CHARACTERISTICS TABLE-Part Three	5-32
5-29	FIELD CHARACTERISTICS TABLE-Part Four	5-33
5-30	Reviewing a Form in Layout Mode	5-34
5-31	Choosing IF Using Menu Selection	5-35
5-32	Defining a Form in Single Field Mode	5-36
5-33	Form Mode Screen	5-43
5-34	Defining a Form in Form Mode	5-45
5-35	Layout Mode Screen	5-46
5-36	Defining a Form in Layout Mode	5-48
5-37	Choosing MF Using Menu Selection	5-50
5-38	Single Field Mode Screen	5-51
5-39	Using Single Field Mode to Delete a Field	5-52
5-40	Form Mode Screen	5-54
5-41	Using Form Mode to Delete a Field	5-56
5-42	Modifying a Form in Layout Mode	5-57
5-43	Choosing DF Using Menu Selection	5-58
5-44	Limited Edit Task Screen	5-60
8-0	Sample Form	8-1

SECTION 1

INTRODUCTION

The Form Editor is a high level utility for defining and maintaining electronic forms. In the User Interface environment, these forms are used to communicate between an application program and the user through calls to the Form Processor. The Form Processor is the IISS run time package of routines that an application program uses to manipulate and display forms.

This manual is intended for application programmers who write application programs that use the Form Processor and for anyone who creates electronic forms with the Form Definition Language or the Forms Driven Form Editor. Knowledge of the Integrated Information Support System (IISS) environment is assumed.

In this manual the term "user" refers to the person running an application program with an electronic forms interface.

SECTION 2

DOCUMENTS

2.1 Reference Documents

- [1] Structural Dynamics Research Corporation, IISS Forms Language Compiler Development Specification, DS 620144401B, 1 November 1985.
- [2] Structural Dynamics Research Corporation, IISS Forms Driven Form Editor Development Specification, DS 620144402B, 1 November 1985.
- [3] Systran, ICAM Documentation Standards, 15 September 1983, IDS150120000C.

This manual is one of a set of user manuals that together describe how to operate in the IISS environment. The complete set consists of the following manuals listed here for reference:

- [1] Structural Dynamics Research Corporation, IISS Form Editor User Manual, UM 620144400B, 1 November 1985.
- [2] Structural Dynamics Research Corporation, IISS Form Processor User Manual, UM 620144200B, 1 November 1985.

Explains how to define and maintain electronic forms. It is intended to be used by programmers writing application programs that use the Form Processor.

Describes the set of callable execution time routines available to an application program to process electronic forms. It is intended to be used by programmers writing application programs for the IISS environment.

- [3] Structural Dynamics Research Corporation, IISS Terminal Operator Guide, OM 620144000 , 1 November 1985.

Explains how to operate the generic IISS terminal when running an IISS application program. The IISS end user environment, function selection and some predefined applications are also described.

- [4] Structural Dynamics Research Corporation, IISS Text Editor User Manual, UM 620144600B, 1 November 1985.

Explains how to use the file editing functions including: inserting, deleting, moving and replacing text.

- [5] Structural Dynamics Research Corporation, IISS Rapid Application Generator User Manual, UM 620144502 , 1 November 1985.

Describes the Application Definition Language and the process used for translating textual definitions of interactive database applications into programs that access selected data base information resident in the Common Data Model. This information is accessible through the IISS Neutral Data Manipulation Language.

- [6] Structural Dynamics Research Corporation, IISS Report Writer User Manual, UM 620144501 , 1 November 1985.

Describes the Report Definition Language and the process of creating a hard copy report of selected data base information resident in the Common Data Model. This information is accessible through the IISS Neutral Data Manipulation Language.

- [7] Structural Dynamics Research Corporation, IISS Virtual Terminal User Manual, UM 620144300B, 1 November 1985.

Explains the program callable interface to the IISS Virtual Terminal. The callable routines, Virtual Terminal commands and the implementation of additional terminals are described. It is intended for application and system programmers working in the IISS environment.

2.2 Terms and Abbreviations

American Standard Code for Information Interchange: (ASCII), the character set defined by ANSI X3.4 and used by most computer vendors.

Application Interface: (AI), subset of the IISS User Interface that consists of the callable routines that are linked with applications that use the Form Processor or Virtual Terminal. The AI enables applications to be hosted on computers other than the host of the User Interface.

Application Process: (AP), a cohesive unit of software that can be initiated as a unit to perform some function or functions.

Attribute: field characteristic such as blinking, highlighted, black, etc. and various other combinations. Background attributes are defined for forms or windows only. Foreground attributes are defined for items. Attributes may be permanent, i.e., they remain the same unless changed by the application program, or they may be temporary, i.e., they remain in effect until the window is redisplayed.

Device Drivers: (DD), software modules written to handle I/O for a specific kind of terminal. The modules map terminal specific commands and data to a neutral format. Device Drivers are part of the UI Virtual Terminal.

Display List: is similar to the open list, except that it contains only those forms that have been added to the screen and are currently displayed on the screen.

Extended Binary Coded Decimal Interchange Code: (EBCDIC), the character set used by a few computer vendors (notably IBM) instead of ASCII.

Field: two-dimensional space on a terminal screen.

Form: structured view which may be imposed on windows or other forms. A form is composed of fields. These fields may be defined as forms, items, and windows.

Form Definition: (FD), forms definition language after compilation. It is read at runtime by the Form Processor.

Forms Definition Language: (FDL), the language in which electronic forms are defined.

Forms Driven Form Editor: (FD FE), subset of the FE which consists of a forms driven application used to create Form Definition files interactively.

Form Editor: (FE), subset of the IISS User Interface that is used to create definitions of forms. The FE consists of the Forms Driven Form Editor and the Forms Language Compiler.

Form Hierarchy: a graphic representation of the way in which forms, items and windows are related to their parent form.

Forms Language Compiler: (FLAN), subset of the FE that consists of a batch process that accepts a series of forms definition language statements and produces form definition files as output.

Form Processor: (FP), subset of the IISS User Interface that consists of a set of callable execution time routines available to an application program for form processing.

Form Processor Text Editor: (FPTE), subset of the Form Processor that consists of software modules that provide text editing capabilities to all users of applications that use the Form Processor.

IISS Function Screen: the first screen that is displayed after logon. It allows the user to specify the function he wants to access and the device type and device name on which he is working.

Integrated Information Support System: (IISS), a test computing environment used to investigate, demonstrate and test the concepts of information management and information integration in the context of Aerospace Manufacturing. The IISS addresses the problems of integration of data resident on heterogeneous data bases supported by heterogeneous computers interconnected via a Local Area Network.

Item: non-decomposable area of a form in which hard-coded descriptive text may be placed and the only defined areas where user data may be input/output.

Message: descriptive text which may be returned in the standard message line on the terminal screen. They are used to warn of errors or provide other user information.

Message Line: a line on the terminal screen that is used to display messages.

Network Transaction Manager: (NTM), IISS subsystem that performs the coordination, communication and housekeeping functions required to integrate the Application Processes and System Services resident on the various hosts into a cohesive system.

Open List: a list of all the forms that have been and are currently open for an application process.

Operating System: (OS), software supplied with a computer which allows it to supervise its own operations and manage access to hardware facilities such as memory and peripherals.

Page: instance of forms in windows that are created whenever a form is added to a window.

Paging and Scrolling: a method which allows a form to contain more data than can be displayed with provisions for viewing any portion of the data buffer.

Physical Device: a hardware terminal.

Qualified Name: the name of a form, item or window preceded by the hierarchy path so that it is uniquely identified.

Subform: a form that is used within another form.

User Data: data which is either input by the user or output by the application programs to items.

User Interface: (UI), IISS subsystem that controls the user's terminal and interfaces with the rest of the system. The UI consists of two major subsystems: the User Interface Development System (UIDS) and the User Interface Management System (UIMS).

User Interface Development System: (UIDS), collection of IISS User Interface subsystems that are used by applications programmers as they develop IISS applications. The UIDS includes the Form Editor and the Application Generator.

User Interface Management System: (UIMS), the runtime UI. It consists of the Form Processor, Virtual Terminal, Application Interface, the User Interface Services and the Text Editor.

User Interface Monitor: (UIM), part of the Form Processor that handles messaging between the NTM and the UI. It also provides authorization checks and initiates applications.

User Interface Services: (UIS), subset of the IISS User Interface that consists of a package of routines that aid users in controlling their environment. It includes message management, change password, and application definition services.

User Interface/Virtual Terminal Interface: (UI/VTI), another name for the User Interface.

Virtual Terminal: (VT), subset of the IISS User Interface that performs the interfacing between different terminals and the UI. This is done by defining a specific set of terminal features and protocols which must be supported by the UI software which constitutes the virtual terminal definition. Specific terminals are then mapped against the virtual terminal software by specific software modules written for each type of real terminal supported.

Window: dynamic area of a terminal screen on which predefined forms may be placed at run time.

Window Manager: a facility which allows the following to be manipulated: size and location of windows, the device on which an application is running, the position of a form within a window. It is part of the Form Processor.

SECTION 3

ELECTRONIC FORM CHARACTERISTICS

Electronic forms are very similar to the paper forms we encounter in our everyday lives. They both contain two types of information. First, the forms have information already printed on them - titles, headings, descriptions of items to be filled in, and instructions. Second, the forms have blank spaces for information, such as the date, to be filled in by the users.

3.1 Fields

Electronic forms consist of areas called fields which allow the user to enter and view variable data. Fields on electronic forms can be defined three ways:

- As item fields
- As form fields
- As window fields

3.1.1 Item Fields

A field on a form is usually defined as an item. An item is a field that holds data. For example, if you enter the date on a form, the area containing the date is an item field.

3.1.2 Form Fields

In many applications, there is a large volume of information to be communicated. This can result in many forms and also forms with many items. As the number of forms and item fields per form increases, there may be groups of items that are shared by more than one form or that make sense by themselves. For example, all IRS forms require identification information such as the name, address, and social security number of the taxpayer. These logical groups of items can be made into separate forms and incorporated into other forms as subforms. To incorporate a subform within a form, the area on the host form where the group of items should be located is defined as a form field. A subform within a form is similar to using macros or procedures in a programming language. Forms can be nested to any level that makes sense to keep the form definition for an application as simple and straightforward as possible.

3.1.3 Window Fields

Although the form within a form concept provides a great deal of flexibility in defining forms, these form arrangements are static and cannot be changed. Often times, the information a user needs to enter for an application is dependent on information that was previously entered. For example, if you have a part inventory application, the information needed for the part may depend on the part number or the storage location entered. This results in the need for dynamic forms. A form can be made dynamic by defining a field as a window. A window is used as a place holder on the form for any number of subforms. At run time, the application then determines what subform(s) the window will contain. A window field can contain more than one subform at a time. Each subform added to a window creates a page. The Form Processor User Manual explains how to create and use window pages for an application program.

3.2 Text

The information already printed on paper forms and the displayed part of an electronic form is called text. This text is either prompt text or background text. Prompt text is usually associated with a field and positioned relative to it. It is used to tell the user what to enter in a particular field. For example, if "DATE: _____" appears on a form, the word "DATE:" would be prompt text. Background text is not related to any given field and is used primarily for titles and formatting. If the words "Please print" appear at the top of a form, this would be background text. A line of asterisks separating a form into sections or a box around a set of fields would also be background text.

3.3 Interactive

The ability to interact with a machine in a user-friendly environment is a very important factor to consider when developing a high quality user interface. This user-friendly environment can be provided in part by giving the user a comfortable, non-intimidating means of interaction. Electronic forms have many advantages over paper forms, one of which is this ability to interact with the user in a helpful manner. This interaction can provide the user with help in filling out forms by prompting for information and providing feedback about the "correctness" of that information once it has been entered.

3.4 Attributes

To a user running an IISS application, a form is a collection of information displayed on the terminal screen. To the User Interface, a form is a data structure that specifies how information is to be displayed on a terminal and how the user can interact with that display. Attributes are the characteristics of electronic forms that specify this information. The complete list of electronic form attributes is described next.

3.4.1 Background

The background attribute applies to forms and windows. It is analogous to printing paper forms on different colors of paper. You have the option of displaying white characters on a black background or black characters on a white background (sometimes known as reverse video).

3.4.2 Calculated Value

The calculated value attribute applies to item fields. It allows the value for an item field to be specified as an expression. A constant expression acts as a default value for the field when it is initially displayed to the user. This attribute also provides a way to display the current date and time, and array indices. An array index field will contain the name of the array and the subscripts of the first displayed element. The subscripts will change dynamically as the array is paged or scrolled.

Constant expressions which act as default values and array index fields are currently supported. Future releases will support expressions which combine strings, integers, fields and functions with operators such as addition and concatenation to provide field values. This type of expression will only be evaluated when one of its components changes.

3.4.3 Conversion

The conversion attribute causes case conversion of a data value that has been entered in an item field. The value can be converted to all lowercase or uppercase alphabetic characters. Specifying this attribute is optional. If this attribute is not specified, the data value remains as it was entered by the user.

3.4.4 Data Type

The data type attribute is used to determine the type of data that can be entered in an item field. If the numeric option is specified, the value entered for the item must contain all numbers. If numeric is not specified, the default data type is alphanumeric. If a value limit attribute is specified, numeric is automatically assumed.

3.4.5 Entry

The entry attribute specifies entry requirements for item fields. You can specify that a value must be entered for the item before the form can be processed by the application program. You can also specify that the value must fill every position of the item.

3.4.6 Display

The foreground attribute controls access to an item field and how it appears to the user. If an item is input, the user may enter a value for the item. You can specify whether the user's input will or will not be echoed on the screen. An output item means that only the application program may enter a value for it. One of these options must be specified but it can be changed or temporarily overridden using the FP routine PUTATT as explained in the Form Processor User's Manual.

3.4.7 Help

The help attribute provides user assistance for entering data values for item fields. Help can be either a string or a form that is displayed when the cursor is in the item field and the <HELP> key is pressed or you can specify that the application program decides what to do when the <HELP> key is pressed. How to use the <HELP> function key is explained in the IISS Terminal Operator Guide.

3.4.8 Identification

The identification attribute is a unique name that identifies a form or field. These names are used by the application program to access form data as explained in the Form Processor User's Manual.

3.4.9 Justification

The justification attribute positions an entered data value in an item field. This attribute affects the internal representation of the value as well as how it appears to the user when displayed. A value can begin in the leftmost position of the field or end in the rightmost position of the field. Specifying this attribute is optional. If this attribute is not specified, the position of the data value remains as it was entered by the user.

3.4.10 Location

The location attribute specifies where text and fields will be positioned on the form. Location may be absolute with respect to the origin of the form or relative to other fields on the form.

3.4.11 Repetition

The repetition attribute specifies that the field appears on the form more than once to create arrays of fields.

3.4.12 Size

The size attribute determines the area a field occupies on the form or the area a subform occupies in a form or window field.

3.4.13 Scroll

The scroll attribute allows a form to contain more data than can be displayed at one time. The user can go into the scroll/page mode of the keyboard as explained in the IISS Terminal Operator Guide and press function keys to view different portions of the data in the display area. Scrolling is especially useful for those applications which must display a list of objects from a database or file and the list is larger than the area available for display.

3.4.14 Value Limit

The value limit attribute sets limits on the values that can be entered in a numeric item field. A maximum or minimum value can be set as well as a range of allowable values. Specifying this attribute is optional. If this attribute is specified, the data type is assumed to be numeric.

3.5 Defining Electronic Forms

The Form Editor provides two methods for defining electronic forms. They are the Forms Driven Form Editor (FDFE) and the Form Definition Language (FDL). Use the FDFE if you want an interactive utility that allows you to define a form by graphically positioning fields and prompts on the terminal screen, by entering values in response to prompts for the necessary information, or by retrieving and copying the necessary information from existing form definitions. Use the FDL if you prefer a programming-language style of working or you do not have a video terminal.

3.6 Storing Forms in the IISS Environment

All form definitions are stored in FDL source files. An FDL source file can contain more than one form definition. The FDL compiler (FLAN) must then be used to convert the form definitions into a format understood by the Form Processor. FLAN reads an FDL source file and creates a separate compiled form for each form definition contained in the source file. A compiled form is referred to as an FD file. A hierarchy is implied by this as shown in Figure 3-1.

Note that the FDL source files that result from forms defined using the FDFE can be edited using any text editor and syntactically correct form definitions created by writing FDL entries directly into an FDL source file can be operated on by the FDFE.

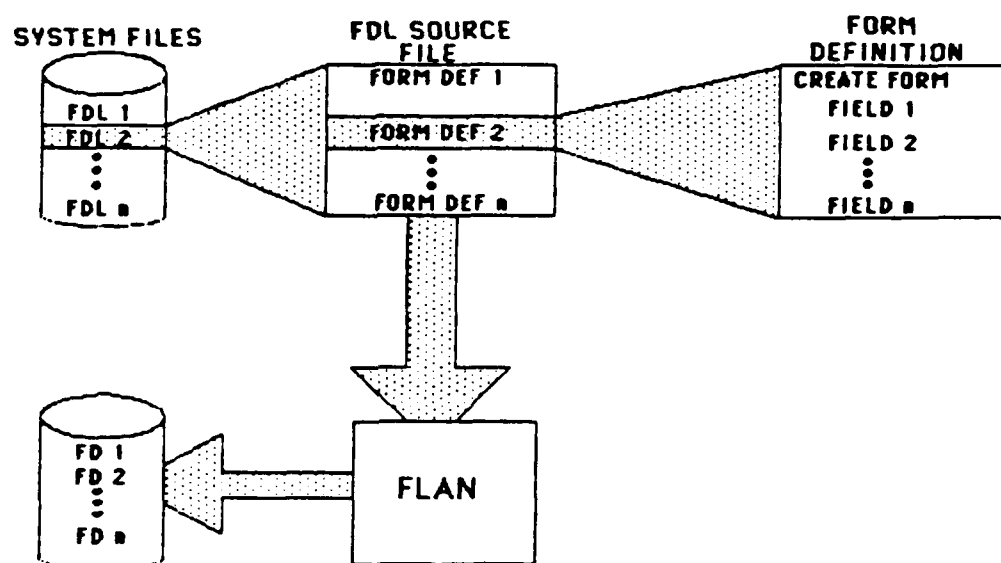


Figure 3-1 IISS Form Storage Hierarchy

SECTION 4

FORM DEFINITION LANGUAGE

The Form Definition Language (FDL) provides a very precise and flexible method for defining electronic forms. Form definitions are created by writing FDL entries directly to an FDL source file with any text editor you might use to prepare a program source file. More than one form may be defined in the same FDL source file.

4.1 FDL Format Notation

This manual uses the following notation to describe the syntax of the FDL entries:

UPPER-CASE	identifies reserved words that have specific meanings in the FDL. These words are generally required unless the portion of the statement containing them is itself optional.
lower-case	identifies names, numbers, or character strings that the user must supply.
Initial upper-case	identifies a statement or clause that is defined later on.
_ Underscores	identify reserved words or portions of reserved words that are optional.
{ } Braces	enclosing vertically stacked options indicate that one of the enclosed options is required.
[] Brackets	indicate that the enclosed clause or option is optional. When two or more options are vertically stacked within the brackets, one or none of them may be specified.
... Ellipsis	indicates that the preceding statement or clause may be repeated any number of times.

4.2 Form Definition Language Syntax

The complete FDL Syntax is listed in this section. The following sections contain a detailed explanation of the syntax by statement and clause.

Form Definition

CREATE FORM form_name

[SIZE int-1 [BY int-2]]

[BACKGROUND {BLACK}]
 {WHITE}

[PROMPT Location prompt_string ...]

[Field_Definition ...]

Field Definition - Items

ITEM item_name [Repeat_Spec]

Location

[SIZE int-1 [BY int-2]]

[VALUE { string constant }]
 { INDEX(field_name) }
 { '._TIME' }
 { '._DATE' }

{ INPUT }

{ OUTPUT }

DISPLAY AS { HIDDEN }

{ TEXT }

[DOMAIN ([LEFT] [UPPER]
 [RIGHT] [LOWER] [MUST ENTER]

[MUST FILL] [NUMERIC] [MAXIMUM int-1]

[MINIMUM int-2])]

{ help_string }
[HELP { help_form_name }]
 { APPLICATION }

[PROMPT Location prompt_string ...]

Field Definition - Forms

FORM form_name [Repeat_Spec]

Location

SIZE int-1 [BY int-2]

[PROMPT Location prompt_string ...]

Field Definition - Windows

WINDOW windows_name [Repeat_Spec]

Location

SIZE int-1 [BY int-2]

{ BLACK }
[BACKGROUND { WHITE }]

[PROMPT Location prompt_string ...]

Location

```

/
{ [ int ] { LEFT } OF [ field_name ] } +-
{ { RIGHT } } | AND
{ COLUMN int } +-
{ [ int ] { BELOW } [ field_name ] } -+
{ { ABOVE } } |
{ ROW int } -+
{ [ int ] { ABOVE } [field_name ] } +-
{ { BELOW } } | AND
{ ROW int } +-
[Rpt] AT { [ int ] { RIGHT } OF [ field_name ] } -+
{ { LEFT } } |
{ COLUMN int } -+
{ [ int ] { LEFT } OF [ Rpt OF ] [ field_name ] }
{ { RIGHT } }
{ [ int ] { ABOVE } [ Rpt OF ] [ field_name ] }
{ { BELOW } }
int-1 int-2 [RELATIVE TO [Rpt OF] [field_name] ]
\

```

UM 620144400B
1 November 1985

Rpt

```

/
| TOP LEFT |
| TOP      |
| TOP RIGHT|
| LEFT     |
| CENTER   |
| RIGHT    |
| BOTTOM LEFT|
| BOTTOM    |
| BOTTOM RIGHT|
\

```

Repeat Spec

```

+-
| ( { int-1      } {HORIZONTAL} [ WITH int-3 SPACES ] [ . ... ] ) |
| { int-1/int-2 } {VERTICAL   } |
| { int-1/int-1 } |
| { *           } |
| { int-1/ *     } |
+-

```

4.3 Form Definition Syntax

The collection of FDL statements that define a form is a form definition. A form definition is written to an FDL source file with any text editor you might use to prepare a program source file. An FDL source file may contain more than one form definition. The syntax for a form definition is:

```
CREATE FORM form_name  
  
[ SIZE int-1 [ BY int-2 ] ]  
  
[ BACKGROUND { BLACK }  
  { WHITE } ]  
  
[ PROMPT Location prompt_string ... ]  
  
[ Field_Definition ... ]
```

4.3.1 CREATE FORM Statement

Every form definition must begin with the CREATE FORM statement. This statement tells the compiler that what follows is a form definition. It also signals the end of the previous form definition if this is not the first one in the source file.

CREATE FORM is the statement keyword.

form_name is a unique name of up to 10 letters and/or numbers associated with each form. It is used by the application program to identify the form. A form_name is required and cannot begin with a number. The length of this name may be further restricted by the system you are running on.

4.3.2 SIZE Clause

This clause determines the number of columns and rows the form will occupy when it is displayed. This is specified by the width and height. This clause is optional and if specified, when the form is displayed in a form or window field, the size of the form or window takes precedence. This means that the form will be "clipped" if the form size is larger than the size

of the form or window it is displayed in. When the form size is smaller than the field size, the form size "grows" to fill the form or window field. If you do not specify the size of the form, it defaults to the size of the form or window it is displayed in.

SIZE is the clause keyword.

int-1 is the width of the form. It is expressed as the number of columns it will occupy when displayed.

BY is a reserved word that must be included when int-2 is specified. There must be a space before and after this word when used.

int-2 is the height of the field. It is expressed as the number of rows it will occupy when displayed. This parameter is optional and defaults to one if not entered.

4.3.3 BACKGROUND Clause

This clause allows you to define the background of the form. This is analogous to specifying what color of paper a paper form is printed on. This clause is optional. If it is omitted, the background of the form defaults to black.

BACKGROUND is the clause keyword.

WHITE displays black characters on an opaque white background (sometimes known as reverse video).

BLACK displays white characters on an opaque black background.

4.3.4 PROMPT Clause

This clause allows you to specify the background text that will appear on the form. Prompts are optional and may be repeated as many times as needed to specify the background text for the form.

PROMPT is the clause keyword.

Location describes where the background text will be positioned on the form. The syntax for this parameter is described in section 4.4.5 of this manual.

prompt_string is the background text to be displayed on the form and must be enclosed in double quotes ("text").

4.3.5 Field Definition Syntax

Field definitions specify the fields a form is composed of. The information required to define a field is different for each of the three field types.

4.3.5.1 Item Fields

An item field is one that holds data. The application program may read from and write to these fields. The syntax for an item field definition is:

```
ITEM item_name [ Repeat_Spec ]

Location

[ SIZE int-1 [ BY int-2 ] ]

[ VALUE { string constant   } ]
      { INDEX(field_name) }
      {   _TIME             }
      {   _PAGE              }

      { INPUT   }
      { OUTPUT  }
DISPLAY AS { HIDDEN }
           { TEXT  }

[ DOMAIN ( [ MUST ENTER ] [ MUST FILL ] [ LEFT ] [ UPPER ]
           [ RIGHT ] [ LOWER ]

           [ NUMERIC ] [ MAXIMUM int-1 ] [ MINIMUM int-2 ] ) ]
```

```
      { help string      }  
[ HELP { help_form_name } ]  
      { APPLICATION      }
```

```
[ PROMPT Location prompt_string ... ]
```

4.3.5.1.1 ITEM Statement

This statement specifies that the form contains a data field.

ITEM is the clause keyword.

item_name is a unique name of up to 10 letters, numbers, and/or underscores. An application program can read from or write to an item field by providing the Form Processor with the name of the item to be accessed. An item_name is required and cannot begin with a number.

Repeat Spec specifies that the field appears on the form more than once and whether or not the area can be scrolled. A field may repeat, either horizontally or vertically, n times with m spaces between repetitions to form rows or columns. That repeat specification may then be repeated to form two dimensional arrays of fields. When an array needs to be scrolled, the number of actual data occurrences and how many occurrences should be displayed at one time are both specified. The syntax for this parameter is described in section 4.3.7 of this manual.

4.3.5.1.2 Location Clause

The Location clause specifies where the item field will be positioned on the containing form. The syntax for Location is described in section 4.3.6 of this manual.

4.3.5.1.3 SIZE Clause

The SIZE clause determines the area on the form that each occurrence of the item occupies. This is specified by the width and height. An item may not overlap other fields or text and must be within the boundaries of its containing form. If a VALUE clause is included in the field definition, SIZE is

optional for item fields. In this case, the width of the item defaults to the length of the VALUE string and the height is one.

SIZE is the clause keyword.

int-1 is the width of the field. It is expressed as the number of columns it occupies on the form.

BY is a reserved word that must be included when int-2 is specified.

int-2 is the height of the field. It is expressed as the number of rows it occupies on the form. This parameter is optional and defaults to one if not entered.

4.3.5.1.4 VALUE Clause

The VALUE clause allows you to specify the value for the item field as an expression. The expression is only evaluated when one of its components changes. A constant expression acts as a default value that will be shown in the field when the form is first displayed to the user. If the DISPLAY AS parameter for the field is "INPUT", the user can change this value. If the user does not enter another value for the item, the string constant is returned to the application program as the default value for the item. This clause also provides a way to display the current data and time and array indices. An array index field contains the name of the array and the subscripts of the first displayed element. The subscripts will change dynamically as the array is paged or scrolled. Future releases will support expressions which combine strings, integers, fields and functions with operators such as addition and concatenation. This clause is optional. If omitted the item is blank filled. If a constant expression is used to specify a default value and SIZE is omitted from the field definition, the item is assumed to be one dimensional and its length is the number of characters in the string constant.

VALUE is the clause keyword.

string constant is a character string enclosed in double quotes ("default value"). If the SIZE clause is included in the field definition, the length of the string constant can be no more than the total number of characters specified

by the size. For example, if size is 4 by 3, then the string constant should be no more than 12 characters long. When entering default values for multi-dimensional fields concatenate the values and they will be split apart appropriately to fill the field. For example, if the size is 4 by 3 and the value_string is "*****+++==", it will be displayed as

```
****  
++++  
=====
```

INDEX(field_name) specifies that the value for this field is the name of the first displayed element of the array "field_name". The array must be on the same form as this item field and be enclosed in single quotes (i.e., INDEX('myfield')). If "myfield" is a two dimensional array, an example index would be "myfield(1,1)".

'._TIME specifies the current time. The format is HH:MM:SS and the length of the field must be at least 9.

'._DATE' specifies the current date. The format is MM/DD/YY and the length of the field must be at least 9.

4.3.5.1.5 DISPLAY AS Clause

The DISPLAY AS clause controls access to the field. This clause is required for every item field definition.

DISPLAY AS is the clause keyword.

INPUT means that the user may enter a value for this item and the value is echoed on the screen. The area where the user may type is highlighted on the form.

OUTPUT means that only the application program may enter a value for this item. The value is displayed in bold type on terminals that support it.

TEXT is the same as OUTPUT except the value is not displayed in bold type.

HIDDEN means that the user may enter a value for this item but the value is not echoed on the screen. This option is typically used for items of privileged information such as passwords. The area where the user may type is highlighted on the form.

4.3.5.1.6 DOMAIN Clause

At this time, the DOMAIN clause allows you to reformat the value a user enters and specify entry requirements and restrictions for the item field. This clause is optional. When used, the options may be entered in any order.

DOMAIN is the clause keyword.

LEFT positions an entered value in the leftmost position of the field. Any leading blanks are removed. The result affects both the internal representation of the value and how it appears when displayed.

RIGHT positions an entered value so that it ends in the rightmost position of the field. The value is padded with leading blanks. The result affects both the internal representation of the value and how it appears when displayed.

If neither LEFT nor RIGHT is specified, the position of the data value will be as it was entered in the field.

UPPER converts all lower-case characters of a data value to upper-case.

LOWER converts all upper-case characters of a data value to lower-case.

If neither UPPER nor LOWER is specified, the data value remains as it was entered in the field.

- NUMERIC** specifies that only numbers will be accepted for the field value. If this option is not specified, all alphanumeric characters are accepted.
- MAXIMUM int-1** automatically specifies that only numeric values will be accepted for the field and that the highest value is the number "int". The abbreviation "MAX" may be used for this option.
- MINIMUM int-2** automatically specifies that only numeric values will be accepted for the field and that the lowest value is the number "int". The abbreviation "MIN" may be used for this option.
- If either MAXIMUM or MINIMUM is specified, the field becomes NUMERIC. Both the MAXIMUM and MINIMUM options may be specified to define a range of acceptable field values.
- MUST ENTER** means that the user is required to enter a value for the item before the form can be processed by the application program.
- MUST FILL** means that the user is required to enter a value for the item and it must fill every position of the item. This option is typically used for items such as phone numbers or social security numbers.

4.3.5.1.7 HELP Clause

The HELP clause specifies a string or another form that is displayed when the cursor is in this field and the <HELP> key is pressed or that the application program decides what to do next. The <HELP> key and other function keys are explained in the IISS Terminal Operator Guide.

- HELP** is the clause keyword.
- help_string** is a string of up to 60 characters enclosed in double quotes ("help_string") that is displayed in the message line of the screen when the <HELP> key is pressed.

`help_form_name` identifies another form that is displayed when the `<HELP>` key is pressed. Display of the form containing this field is suppressed until the `<ENTER>` key is pressed. The help form can be defined in the current FDL source file or a different one.

`APPLICATION` specifies that the application program will decide what to do when the `<HELP>` key is pressed. The abbreviation `AP` may be used for this option.

4.3.5.1.8 PROMPT Clause

The `PROMPT` clause allows you to specify information associated with the field such as labels and instructions.

`PROMPT` is the clause keyword.

`Location` specifies where the information appears when the form is displayed. The syntax for this parameter is described in section 4.4.5.

`prompt_string` is the information to be displayed enclosed in double quotes ("`prompt_string`").

4.3.5.2 Form Fields

To incorporate a subform within a form, form fields are used. The syntax for a form field definition is:

```
FORM form_name [ Repeat_Spec ]
```

```
    Location
```

```
    SIZE int-1 [ BY int-2 ]
```

```
    [ PROMPT Location prompt_string ]
```

4.3.5.2.1 FORM Statement

This statement specifies that you are incorporating a subform into the form being defined. The subform must be defined with its own form definition. The form definition for the subform can be in the current FDL source file or a different one.

FORM is the statement keyword.

form_name is a unique name of up to 10 letters and/or numbers. This **form_name** corresponds to the **form_name** on the **CREATE FORM** statement in the definition of the subform.

Repeat Spec specifies that the subform appears on the form more than once and whether or not the area can be scrolled. A subform may repeat, either horizontally or vertically, *n* times with *m* spaces between repetitions to form rows or columns. That repeat specification may then be repeated to form two dimensional arrays of subforms. When an array needs to be scrolled, the number of actual data occurrences and how many occurrences should be displayed at one time are both specified. The syntax for this parameter is described in section 4.3.7 of this manual.

4.3.5.2.2 Location Clause

This clause specifies the position of the first occurrence of the subform. Repeating occurrences are then positioned relative to this occurrence based on the repeat specification. The syntax for this clause is described in section 4.3.6 of this manual.

4.3.5.2.3 SIZE Clause

The **SIZE** clause determines the area on the form that each occurrence of the subform may occupy. The top left character of the form field becomes the origin of the subform it contains.

SIZE is the clause keyword.

int-1 is the width of the field. It is expressed as the number of columns it occupies on the form.

BY is a reserved word that must be included when **int-2** is specified.

int-2 is the height of the field. It is expressed as the number of rows it occupies on the form. This parameter is optional and defaults to one if not entered.

For form fields, the SIZE clause specifies the maximum space on the host form that a subform may occupy. If the size of a subform is not the same as the form field, the size of the form field takes precedence.

4.3.5.2.4 PROMPT Clause

The PROMPT clause allows you to display information associated with the field such as labels and instructions.

PROMPT is the clause keyword.

Location specifies where to position the information on the form containing the field. The syntax for this parameter is described in section 4.4.5.

prompt_string is the information to be displayed. It must be enclosed in double quotes ("prompt_string").

4.3.5.3 Window Fields

Window fields are used as place holders on a form for subforms to be supplied by the application program at run time. The syntax for a window field definition is:

```
WINDOW window_name [ Repeat_Spec ]  
  
    Location  
  
    SIZE int-1 [ BY int-2 ]  
  
        { BLACK }  
    BACKGROUND { WHITE }  
  
    [ PROMPT Location prompt_string ... ]
```

4.3.5.3.1 WINDOW Statement

This statement specifies that you are defining a field on the form as a window. Its contents will be determined by the application at run time.

WINDOW is the statement keyword.

window_name is a unique name of up to 10 letters, numbers and/or underscores. This name cannot begin with a number and is used by the application to tell the Form Processor where to put subforms.

Repeat Spec specifies that the window appears on the form more than once and whether or not the area can be scrolled. A window may repeat, either horizontally or vertically, *n* times with *m* spaces between repetitions to form rows or columns. That repeat specification may then be repeated to form two dimensional arrays of fields. When an array needs to be scrolled, the number of actual data occurrences and how many occurrences should be displayed at on time are both specified. The syntax for this parameter is described in section 4.3.7 of this manual.

4.3.5.3.2 Location Clause

This clause specifies the position of the first occurrence of the window on the form. Repeating occurrences are then positioned relative to this occurrence based on the repeat specification. The syntax for this clause is described in section 4.3.6 of this manual.

4.3.5.3.3 SIZE Clause

The **SIZE** clause determines the area on the form that each occurrence of the window may occupy. The top left character of the window field becomes the origin of any subforms it contains.

SIZE is the clause keyword.

int-1 is the width of the field. It is expressed as the number of columns it occupies on the form.

BY is a reserved word that must be included if **int-2** is specified.

int-2 is the height of the field. It is expressed as the number of rows it occupies on the form. This parameter is optional and defaults to one if not entered.

For window fields, the **SIZE** clause reserves the space on the host form for subforms to be supplied at run time. If a subform is bigger than the reserved window, it will be "clipped". If it is smaller, it will "grow" to fill the window.

4.3.5.3.4 BACKGROUND Clause

This clause allows you to define the background of the window. This background will be seen when there are no pages in the window as well as around forms which are smaller than the window. If it is omitted, the background of the window defaults to black.

WHITE displays black characters on an opaque white background (sometimes known as reverse video).

BLACK displays white characters on an opaque black background.

4.3.5.3.5 PROMPT Clause

The **PROMPT** clause allows you to specify information associated with the field such as labels and instructions.

PROMPT is the clause keyword.

Location specifies where to position the information on the form containing the field. The syntax for this parameter is described in section 4.4.5 in this manual.

prompt_string is the information to be displayed. It must be enclosed in double quotes ("prompt_string").

4.3.6 Location Clause/Parameter

Location specifies where text and fields will be positioned on a form. Location may be absolute with respect to the origin of the form or relative to fields on the form. The origin of a form is at the top left corner with rows being positive down and columns positive to the right. Relative locations are especially useful for positioning text that is associated with fields. The syntax for location is:

```

/
{ [ int ] { LEFT } OF [ field_name ] } +-
{           { RIGHT }           } | AND
{ COLUMN int                       } +-
|
|           { [ int ] { BELOW } [ field_name ] } -+
|           {           { ABOVE }           } |
|           { ROW int                       } -+
|
| { [ int ] { ABOVE } [field_name ] } +-
| {           { BELOW }           } | AND
| { ROW int                       } +-
[Rpt] AT <
|           { [ int ] { RIGHT } OF [ field_name ] } -+
|           {           { LEFT }           } |
|           { COLUMN int                       } -+
|
| { [ int ] { LEFT } OF [ Rpt OF ] [ field_name ] }
| {           { RIGHT }           }
|
| { [ int ] { ABOVE } [ Rpt OF ] [ field_name ] }
| {           { BELOW }           }
|
| int-1 int-2 [RELATIVE TO [Rpt OF] [field_name] ]
\

```

When defining relative locations, field reference points are used. Figure 4-1 shows the nine possible reference points that a field can have.

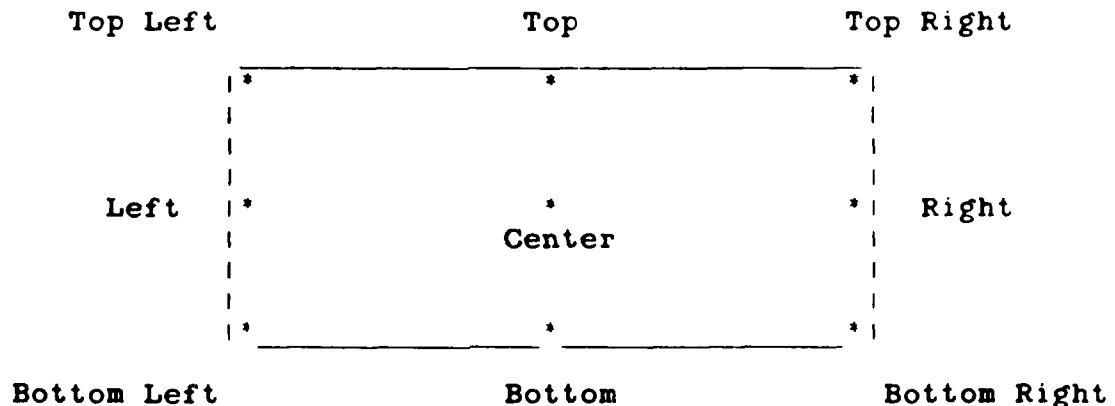


Figure 4-1 Field Reference Points

Each of the reference points represents a character. This means that if you have a one character field, all nine points are the same.

The syntax for the Rpt parameter in the location syntax is:

```
/
TOP LEFT
TOP
TOP RIGHT
LEFT
CENTER
RIGHT
BOTTOM LEFT
BOTTOM
BOTTOM RIGHT
\
```

When positioning text and fields, they must be contained within the boundaries of the containing form and cannot overlap other fields or text. Column one of a form must always be blank and there must be one blank space between an item field and any prompt text. For example:

prompt: _____

is legal and

prompt: _____

is illegal.

The following sections describe and show how to use the location syntax to position fields. The syntax is basically the same for positioning text. * represents the form origin and ^ represents the default field reference point.

4.3.6.1 Absolute Location

An absolute location positions the first character of a text string or a reference point of a field at the intersection of row (n) and column (m) with respect to the form origin. Both coordinates must be given. When positioning a field, the default reference point is the top left if the reference point is not given.

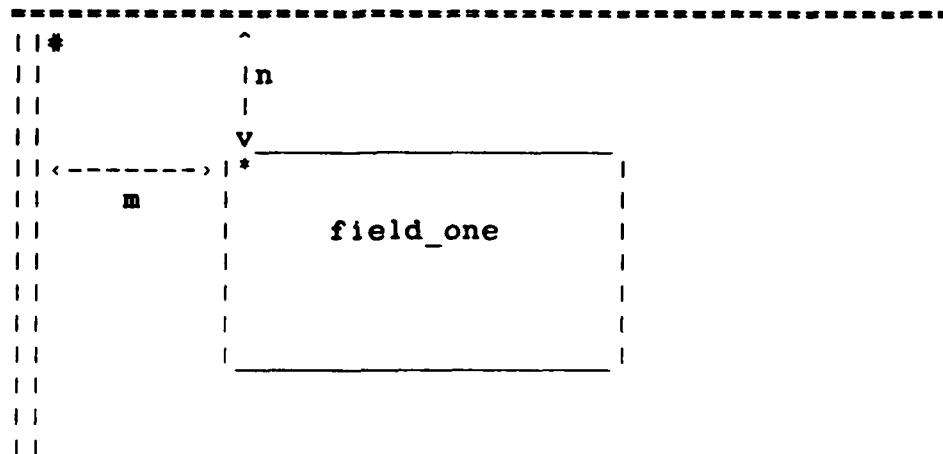


Figure 4-2 Absolute Location

The reference point of field_one (*) is positioned at the intersection of row n and column m. To position field_one as shown, you would use the location syntax:

UM 620144400B
1 November 1985

[Rpt] AT COLUMN int AND ROW int

or

[Rpt] AT ROW int AND COLUMN int

or

[Rpt] AT int-1 int-2 [RELATIVE TO [Rpt OF]
field_name]]

where int-1 is the row and int-2 is the column.
Some valid Locations are:

TOP LEFT AT COL m AND ROW n

TOP LEFT AT n m

AT n m

4.3.6.2 Relative Location

The reference point of a field can be positioned at a point that is relative to the reference point of another field on the form. If a specific reference point for either of the fields is not given, the default is the top left.

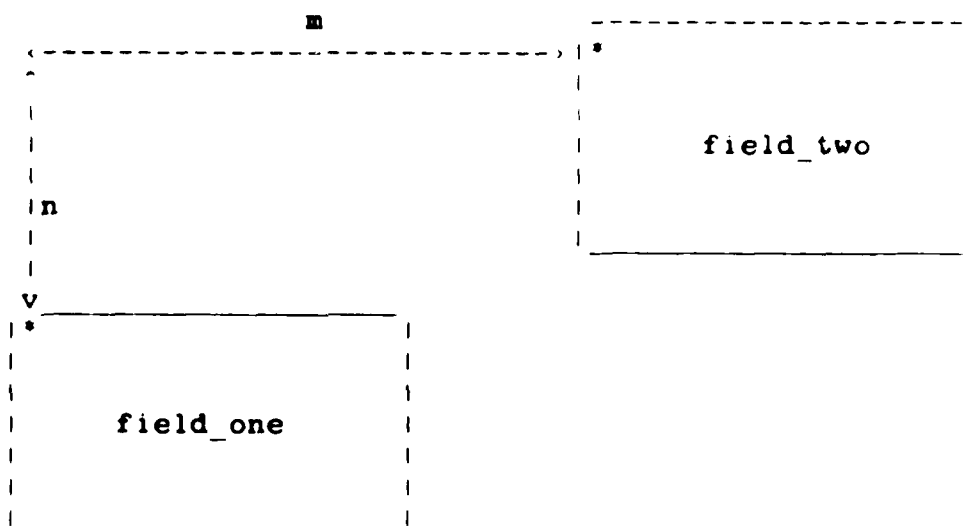


Figure 4-3 Relative Location

The reference point of field_two is positioned at row n and column m relative to the reference point of field_one. To position field_two as shown, you would use the location syntax:

[Rpt] AT int-1 int-2

where int-1 is the row and int-2 is the column. Some valid Locations are:

TOP LEFT AT -n m RELATIVE TO TOP LEFT OF field_one

AT -n m RELATIVE TO field_one

4.3.6.3 Relative Location (Above/Below)

The reference point of a field can be positioned n rows above or below the reference point of another field on the form. If reference points are not given for either of the fields, the default reference points are as shown by the '*'s.

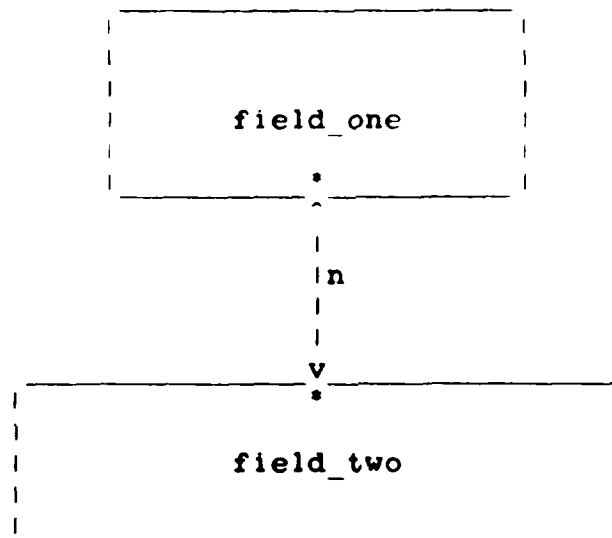


Figure 4-4 Relative Location (Above/Below)

The reference point of field_two is positioned n rows below the reference point of field_one. To position field_two as shown, you would use the location syntax:

```
[ Rpt ] AT [int] { ABOVE } [ Rpt OF ] [field_name]
                { BELOW }
```

Some valid Location clauses are:

TOP AT n BELOW BOTTOM OF field_one

AT n BELOW field_one

UM 620144400B
1 November 1985

Using the ABOVE keyword, you can position the reference point of field_one n rows above the reference point of field_two. Some valid Locations are:

BOTTOM AT n ABOVE TOP OF field_two

AT n ABOVE field_two

4.3.6.4 Relative Location (Right/Left)

The reference point of a field can be positioned *m* columns right or left of the reference point of another field on the form. If reference points are not given for either of the fields, the default points are as shown by the '*'s.

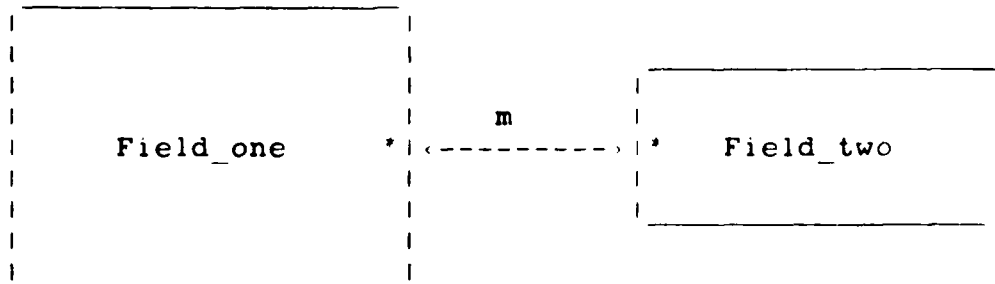


Figure 4-5 Relative Location (Right/Left)

The reference point (Rpt) of field_one is *m* columns to the left of the field_two reference point or the field_two reference point is *m* columns to the right of the field_one reference point. To position the fields as shown, you would use the location syntax:

```
[ Rpt ] AT [int] { LEFT } OF [Rpt OF] [field_name]
                        { RIGHT }
```

Some valid Locations are:

RIGHT AT *m* LEFT OF LEFT OF field_two

AT *m* LEFT OF LEFT OF field_two

LEFT AT *m* RIGHT OF RIGHT OF field_one

AT *m* RIGHT OF RIGHT OF field_one

AT *m* RIGHT OF field_one

NOTE: The Rpt is optional so locations 1 and 2 shown above are identical as are locations 3 and 4.

4.3.6.5 Location Relative to Two Fields

The reference point of a field can be positioned *n* rows above or below the default reference point of one field and *m* columns right or left of the default reference point of another field. Any reference point can be specified for the field being positioned. The reference points for the other fields default to the edge of the field closest to the field being positioned as shown by the *'s.

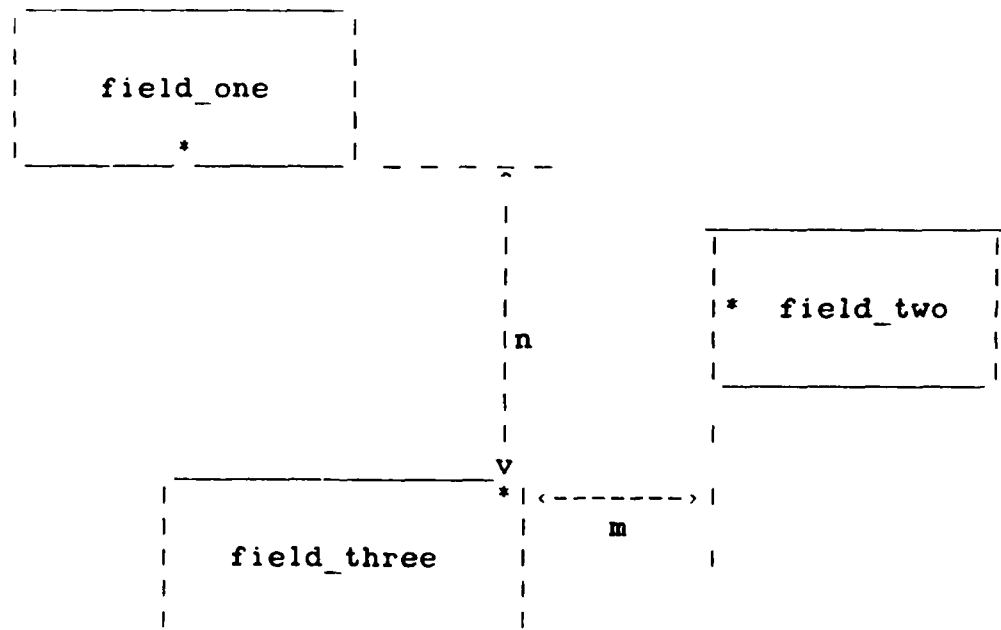


Figure 4-6 Location Relative to Two Fields

The reference point of field_three is positioned *n* rows below the bottom edge of field_one and *m* columns left of the left edge of field_two. To position field_three as shown, you would use the location syntax:

```
[ Rpt ] AT [int] { ABOVE } [field_name]
                { BELOW }
```

```
AND [int] { LEFT } OF [field_name]
        { RIGHT }
```

UM 620144400B
1 November 1985

It does not matter which direction is specified first. Some valid Locations are:

TOP RIGHT AT n BELOW field_one AND m LEFT OF field_two

AT m LEFT field_two AND n BELOW field_one

AT n BELOW field_one AND m LEFT field_two

4.3.6.6 Combination Location

Field positions can be a combination of absolute and relative locations. The reference point of a field can be positioned at row *n* and *m* columns right or left of the default reference point of another field or at column *m* and *n* rows above or below the default reference point of another field. Any reference point can be specified for the field being positioned. The reference point for the other field defaults to the edge of the field closest to the field or text being positioned as shown by the *'s.

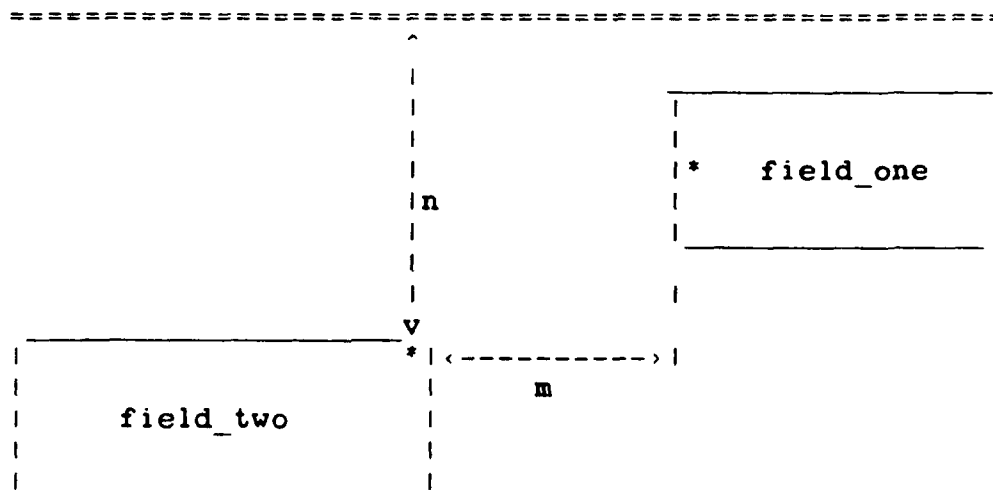


Figure 4-7 Absolute Row/Relative Column

The field_two reference point is in row *n*, *m* columns left of the left edge of field_one. To position field_two as shown, you would use the location syntax:

```
[ Rpt ] AT ROW int AND [int] { LEFT } OF [field_name]
                                   { RIGHT}
```

or

```
[ Rpt ] AT [int] { LEFT } OF [field_name] AND ROW int
                                   { RIGHT}
```

Some valid Locations are:

TOP RIGHT AT ROW n AND m LEFT OF field_one

AT m LEFT field_one AND ROW n

AT ROW n AND m LEFT field_one

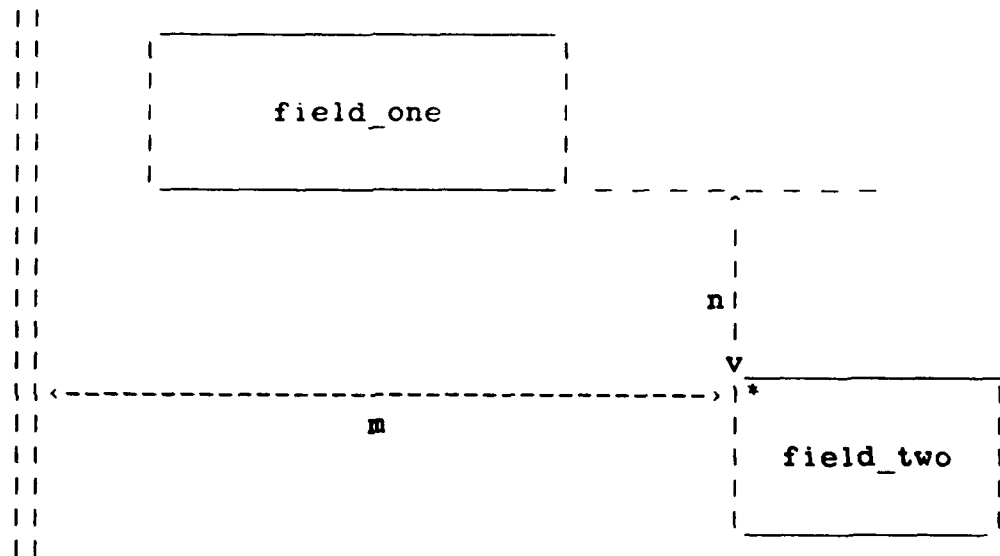


Figure 4-8 Absolute Column/Relative Row

The field_two reference point is in column m, n rows below the bottom edge of field_one. To position field_two as shown, you would use the location syntax:

[Rpt] AT [int] {ABOVE} [field_name] AND COLUMN int
{BELOW}

or

[RPT] AT COLUMN int AND [int { ABOVE } [field_name]
{ BELOW }

UM 620144400B
1 November 1985

Some valid locations are:

TOP LEFT AT n BELOW field_one AND COL m

AT TOP LEFT n BELOW field_one AND COL m

AT COL m AND n BELOW field_one

4.3.7 Repeat Spec Parameter

The repeat specification specifies that the field appears on the form more than once and whether or not the area can be scrolled. A field may repeat, either horizontally or vertically, n times with m spaces between repetitions to form rows or columns. The repeat specification may then be repeated to form two dimensional arrays of fields. When an array needs to be scrolled, the number of actual data occurrences and how many occurrences should be displayed at one time are both specified. The syntax for the Repeat Specification is:

```

+-                                     +-
| ( { int-1          } {HORIZONTAL} [ WITH int-3 SPACES ] [ . ... ] ) |
| { int-1/int-2    } {VERTICAL   } |
| { int-1/int-1    } |
| { *              } |
| { int-1/ *       } |
+-                                     +-

```

int-1	is how many times to repeat the field on the form (i.e., display size).
int-1/int-2	indicates that the field is scrollable using the function keys in the "scrll/page" mode of the keyboard. Int-1 is how many times to repeat the field on the form and int-2 is the actual number of times the field occurs.
int-1/int-1	indicates that the field is scrollable using the function keys in the "scrll/page" mode of the keyboard but the keys are passed back to the application program. Int-1 is how many times to repeat the field on the form.
*	specifies that the field repeats indefinitely on the form.
int-1/ *	indicates a scrollable open-ended array which is not yet supported.
HORIZONTAL	states that the field repeats in a horizontal direction. The abbreviation H may be used for this option.

VERTICAL states that the field repeats in a vertical direction. The abbreviation V may be used for this option.

WITH is an optional reserved word that may be included for readability.

int-3 is how many spaces to leave between field occurrences. It defaults to one if omitted.

SPACES is an optional reserved word that may be included for readability.

4.3.7.1 Row of Fields

The Repeat Spec (3 H 1) repeats a field three times horizontally with one space between repetitions. This would appear as shown in Figure 4-9 when displayed to the user.

```
+---+ +---+ +---+
|///| |///| |///|
|///| |///| |///|
+---+ +---+ +---+
```

Figure 4-9 Row of Fields

4.3.7.2 Array of Fields

The Repeat Spec (2 V 1,3 H 1) repeats the row of fields defined by '3 H 1' two times vertically with one blank row between repetitions. This would appear as shown in Figure 4-10 when displayed to the user.

```
+---+ +---+ +---+
|///| |///| |///|
|///| |///| |///|
+---+ +---+ +---+

+---+ +---+ +---+
|///| |///| |///|
|///| |///| |///|
+---+ +---+ +---+
```

Figure 4-10 Array of Fields

4.3.7.3 Array of An Array of Fields

The Repeat Spec (2 V 5,2 V 1,3 H 1) repeats the array of fields defined by '2 V 1,3 H 1' two times vertically with five blank rows between repetitions. This would appear as shown in Figure 4-11 when displayed to the user.

```

+----+ +----+ +----+
|    | |    | |    |
|    | |    | |    |
+----+ +----+ +----+

+----+ +----+ +----+
|    | |    | |    |
|    | |    | |    |
+----+ +----+ +----+

+----+ +----+ +----+
|    | |    | |    |
|    | |    | |    |
+----+ +----+ +----+

+----+ +----+ +----+
|    | |    | |    |
|    | |    | |    |
+----+ +----+ +----+

```

Figure 4-11 Array of An Array of Fields

4.3.7.4 Single Dimension Scrolled Array

The Repeat Spec (3/5 HORIZONTAL WITH 2 SPACES) or (3/5 H 2) displays five occurrences of a field three at a time horizontally with two spaces between each occurrence. This would appear as shown in Figure 4-12 when displayed to the user.

```

+----+ +----+ +----+ +   +   +   +
| 1 | | 2 | | 3 |   4   5
+----+ +----+ +----+ +   +   +

```

Figure 4-12 Single Dimension Scrolled Array

The fields which are actually displayed are enclosed in boxes. The scrolling and paging function keys used to display the other fields are explained in the IISS Terminal Operator Guide.

4.3.7.5 Two Dimensional Scrolled Array

The Repeat Spec (1/2 V 1,3/6 H 4) defines a display area that is three elements horizontally with 4 spaces between. The actual size of the array is six columns wide and 2 rows deep. This would appear as shown in Figure 4-13 when displayed to the user.

```

+-----+ +-----+ +-----+ +   +   +   +   +
| 1,1 | | 1,2 | | 1,3 |   1,4   1,5   1,6
+-----+ +-----+ +-----+ +   +   +   +   +

+   +   +   +   +   +   +   +
  2,1   2,2   2,3   2,4   2,5   2,6
+   +   +   +   +   +   +   +

```

Figure 4-13 Two Dimensional Scrolled Array

The fields which are actually displayed are enclosed in boxes. The scrolling and paging function keys used to display the other fields are explained in the IISS Terminal Operator Guide.

4.4 Statement Format

FDL statements can be entered in free format. Free format means that keywords and numbers can be separated by any number of spaces. The FDL compiler treats tabs, comments, and carriage returns as spaces.

Form and Field definition clauses may be in any order.

The DOMAIN clause options may be entered in any order.

4.5 Restrictions

Every form definition must begin with the CREATE FORM statement.

There must be at least one space before and after every keyword in the syntax.

A field_name may only be omitted from the Location syntax in a PROMPT clause.

4.6 Abbreviations

Underscores in the FDL syntax indicate reserved words or portions of reserved words that are optional.

4.7 Including Comments

You can include comments in a form definition by enclosing the comment text in (/*) and (*//). Comments are treated as spaces by the FDL compiler. For example:

```
CREATE FORM form1 /* main form */
```

4.8 Reserved Words

This is an alphabetized list of the reserved words in the Form Definition Language.

ABOVE	COLUMN	ITEM	RELATIVE	VERTICAL
AND	CREATE	LEFT	RIGHT	WHITE
APPLICATION	DISPLAY AS	LOWER	ROW	WINDOW
AT	DOMAIN	MAXIMUM	SIZE	WITH
BACKGROUND	FORM	MINIMUM	SPACES	
BELOW	HELP	MUST ENTER	TEXT	
BLACK	HIDDEN	MUST FILL	TO	
BOTTOM	HORIZONTAL	OF	TOP	
BY	INDEX	OUTPUT	UPPER	
CENTER	INPUT	PROMPT	VALUE	

SECTION 5

FORMS DRIVEN FORM EDITOR

The Forms Driven Form Editor (FDFE) is a software tool for interactively defining and maintaining form definitions used in the IISS environment.

5.1 FDFE Functional Organization

The FDFE functions are organized into three levels of tasks. They are called Work Tasks, Edit Tasks, and Field Tasks. You begin an edit session at the work task level. At this level you can perform maintenance functions on FDL source or FD files or move to the edit task level for a specific FDL source file. At the edit task level, you are editing a specific form definition. Again, you can perform maintenance functions on the form definition or move to the field task level for the specified form definition. At the field task level, you can maintain and edit the individual field definitions that define the specified form.

Figure 5-1 maps the FDFE functional organization to the IISS Form Storage Hierarchy.

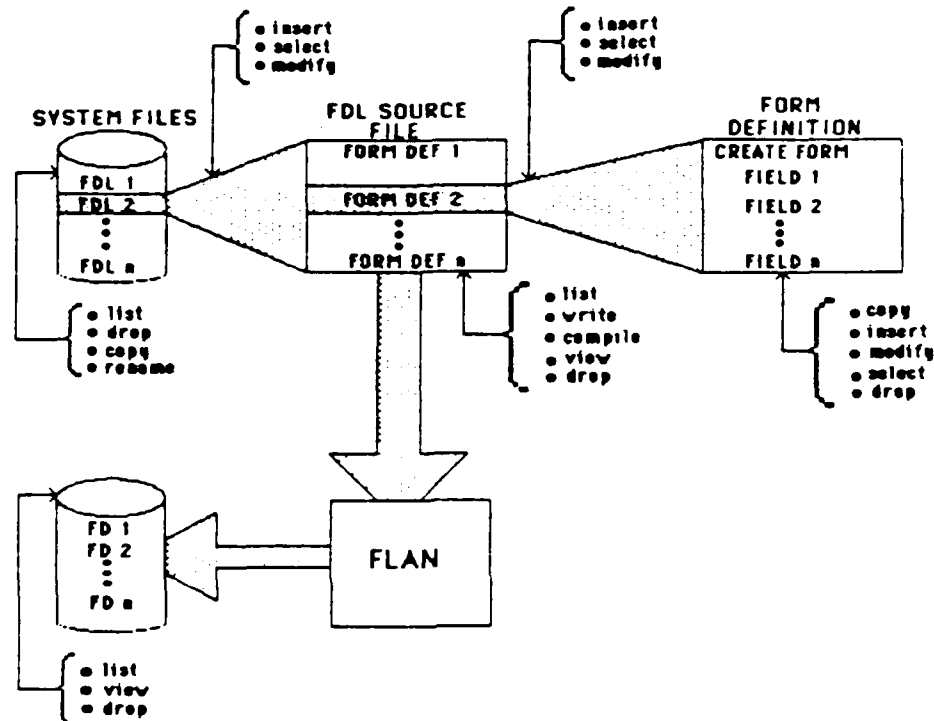


Figure 5-1 FDFE Functions Mapped to the IISS Forms Storage Hierarchy

5.2 FDFE Editing Features

The FDFE provides three modes of operation for editing form definitions. These are Single Field mode, Form mode, and Layout mode. In Single Field mode you edit a form one field at a time. A screen is displayed that provides a template for displaying and defining the characteristics of an individual field. In form mode you edit a form by entering values in a field characteristic table. In layout mode you can graphically position fields and text on the screen and evaluate the overall appearance of a form. You are allowed to switch back and forth between modes to evaluate the results and make changes. When saved, the FDFE stores the form definitions as an FDL source file that can be retrieved and modified.

5.3 System Operation

The FDFE is an application program that you run in the IISS environment. You perform FDFE tasks by entering the appropriate responses to prompts displayed on the screen of your video display terminal. Responses are entered in input fields represented by shaded or underlined areas on the screen. Each time a display appears on the screen, it contains a cursor. The cursor is an underscore or a small flashing rectangle to show where information will appear when entered through the keyboard.

How you interact with screens is described in the IISS Terminal Operator Guide. Figure 5-2 shows the layout of the VT100 keypad for the FDFE in the "application" mode of the keyboard.

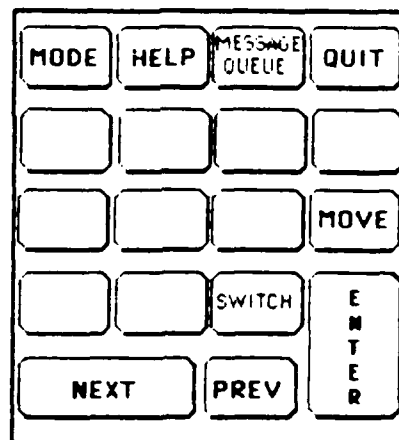


Figure 5-2 VT100 Keypad Layout for the FDFE Application

5.3.1 Accessing the FDFE

The FDFE is available as an application in the IISS environment as explained in the IISS Terminal Operator Guide. To access the FDFE, enter SDFDFEZZZZ as the FUNCTION on the IISS Function Screen. When you have successfully accessed the FDFE, the following display appears on your terminal screen:

FORMS DRIVEN FORM EDITOR - VERSION 2.0 JUNE 1, 1985

Command Entry

WORK TASKS	Command	Pic	For/From Name	To/New Name	Help
List FDL Sources	(LS)				
Insert FDL Source	(IS)				
Modify FDL Source	(MS)				
Select FDL Source	(SS)				
Copy FDL Source	(CS)				
Rename FDL Source	(RS)				
Drop FDL Source	(DS)				
List Compiled form definitions	(LC)				
View Compiled form definition	(VC)				
Drop Compiled form definition	(DC)				
Exit form driven form editor	(EX)				

Msg: ☐ Enter command on command entry line or use menu selection application

Figure 5-3 Work Task Screen

5.4 Work Task Screen

This screen is a list of the FDFE functions that allow you to maintain your FDL source and FD files and edit a specific FDL source file.

5.4.1 Choosing A Work Task

A work task can be chosen with menu selection or command entry. With menu selection, you enter any nonblank character in the "Pic" field for the desired task and the necessary parameters for the task in the input fields that appear in the same horizontal line of the screen. Note that the "Help" field is not a required task parameter. When you press the <HELP> key from this field, an explanation of the task is displayed on the screen. An example of choosing a Work Task is shown in the next screen:

FORMS DRIVEN FORM EDITOR - VERSION 2.0 JUNE 1, 1985

Command Entry

WORK TASKS	Command	Pic	For/From Name	To/New Name	Help
List FDL Sources	(LS)	x	newfrms		
Insert FDL Source	(IS)				
Modify FDL Source	(MS)				
Select FDL Source	(SS)				
Copy FDL Source	(CS)				
Rename FDL Source	(RS)				
Drop FDL Source	(DS)				
List Compiled form definitions (LC)					
View Compiled form definition (VC)					
Drop Compiled form definition (DC)					
Exit form driven form editor (EX)	(EX)				

Msg: ☐ Enter command on command entry line or use menu selection application

Figure 5-4 Choosing a Work Task-Menu Selection

Note that command entry overrides menu selection and an error message is displayed if more than one menu selection is made.

With command entry, you enter the appropriate two letter task command followed by values for the necessary parameters in the "Command Entry" field and press the <ENTER> key. The parameters must be entered in the order specified by menu selection and separated by blanks. For example:

FORMS DRIVEN FORM EDITOR - VERSION 2.0 JUNE 1, 1985

Command Entry

WORK TASKS	Command	Pic	For/From Name	To/New Name	Help
List FDL Sources	(LS)				
Insert FDL Source	(IS)				
Modify FDL Source	(MS)				
Select FDL Source	(SS)				
Copy FDL Source	(CS)				
Rename FDL Source	(FS)				
Drop FDL Source	(DS)				
List Compiled form definitions (LC)					
View Compiled form definition (VC)					
Drop Compiled form definition (DC)					
Exit form driven form editor (EX)					

Msg: Enter command on command entry line or use menu selection application

Figure 5-5 Choosing a Work Task-Command Entry

Note that command entry overrides menu selection.

The following sections summarize the functions performed by each of the Work Tasks.

5.4.2 LS (List FDL Sources)

Use this task to display the names of all your FDL source files.

The Command Entry format is: LS<ENTER>

To execute this task using menu selection, enter any nonblank character in the "Pic" field for List form language Sources and press the <ENTER> key. For example:

FORMS DRIVEN FORM EDITOR - VERSION 2.0 JUNE 1, 1985

Command Entry

WORK TASKS	Command	Pic	For/From Name	To/New Name	Help
List FDL Sources	(LS)				
Insert FDL Source	(IS)				
Modify FDL Source	(MS)				
Select FDL Source	(SS)				
Copy FDL Source	(CS)				
Rename FDL Source	(RS)				
Drop FDL Source	(DS)				
List Compiled form definitions	(LC)				
View Compiled form definition	(VC)				
Drop Compiled form definition	(DC)				
Exit form driven form editor	(EX)				

Msg: Enter command on command entry line or use menu selection application

Figure 5-6 Choosing LS Using Menu Selection

When this task is executed, the next screen displayed is:

```
FORMS DRIVEN FORM EDITOR - VERSION 2.0 JUNE 1, 1985

List FDL Source Files

APFRONT      RMFRONT
ARTEST       TESTAP
BAKGRD       UIS
CHPF LD      UISERV
EDITOR
FDFE
FLFRONT
HEADERD
HRWFRM
HRWTST
ITMEDIT
MM
OLDFRMS
ORDAND
PSCREEN
PTEST
RPT45

Msg: ☐ Press (QUIT) to return                                application
```

Figure 5-7 List FDL Source Files Screen

5.4.2.1 List FDL Source Files Screen

This screen lists the names of all your FDL source files. When there are more names than can be displayed on the screen, the message "Press (ENTER) for more names or (QUIT) to return" is displayed in the message line. This will continue until all the names have been displayed and then the message "Press (QUIT) to return" is displayed. You will return to the Work Task screen when you press the (QUIT) key.

5.4.3 IS (Insert FDL Source)

Use this task to define new form(s) and store them in a new FDL source file. You must enter the name of the new FDL source file you want to create without an extension or directory name. If the name you enter already exists as an FDL source file, an

error message will be displayed in the message line.

The Command Entry format is: IS filename<ENTER>

To execute this task using menu selection, enter any nonblank character in the "Pic" field for Insert form language Source, the name of the new FDL source file as the "For/From Name", and press the <ENTER> key. For example:

FORMS DRIVEN FORM EDITOR - VERSION 2.0 JUNE 1, 1985

Command Entry

WORK TASKS	Command	Pic	For/From Name	To/New Name	Help
List FDL Sources	(LS)		newfms		
Insert FDL Source	(IS)				
Modify FDL Source	(MS)				
Select FDL Source	(SS)				
Copy FDL Source	(CS)				
Rename FDL Source	(RS)				
Drop FDL Source	(DS)				
List Compiled form definitions (LC)					
View Compiled form definition (VC)					
Drop Compiled form definition (DC)					
Exit form driven form editor (EX)					

Msg: Enter command on command entry line or use menu selection application

Figure 5-8 Choosing IS Using Manual Selection

When this task is executed, the next screen displayed is the Edit Task screen as shown in section 5.5.

5.4.4 MS (Modify FDL Source)

Use this task to modify an existing FDL source file. This includes inserting new form definitions in the file and

modifying any that the file already contains. You must specify the name of the FDL source file.

The Command Entry format is: MS filename<ENTER>

To execute this task using menu selection, enter any nonblank character in the "Pic" field for Modify form language Source, the existing FDL source file name as the "For/From Name", and press the <ENTER> key. For example:

FORMS DRIVEN FORM EDITOR - VERSION 2.0 JUNE 1, 1985

Command Entry

WORK TASKS	Command	Pic	For/From Name	To/New Name	Help
List FDL Sources	(LS)	x	oldfrms		
Insert FDL Source	(IS)				
Modify FDL Source	(MS)				
Select FDL Source	(SS)				
Copy FDL Source	(CS)				
Rename FDL Source	(RS)				
Drop FDL Source	(DS)				
List Compiled form definitions	(LC)				
View Compiled form definition	(VC)				
Drop Compiled form definition	(DC)				
Exit form driven form editor	(EX)				

Msg:

application

Figure 5-9 Choosing MS Using Menu Selection

When this task is executed, the next screen displayed is the Edit Task Screen as shown in section 5.5.

5.4.5 SS (Select FDL Source)

Use this task to review the form definitions contained in an FDL source file. You must enter the name of the FDL source file that you want to review. Options in this task include listing the names of all the forms defined in this source file, displaying a form as it will appear when used in an application program, and reviewing the characteristics of an individual form in a read only mode.

The Command Entry format is: SS filename<ENTER>

To execute this task using menu selection, enter any nonblank character in the "Pic" filed for Select form language Source, the existing FDL source file name as the "For/From Name", and press the <ENTER> key. For example:

FORMS DRIVEN FORM EDITOR - VERSION 2.0 JUNE 1, 1985

Command Entry

WORK TASKS	Command	Pic	For/From Name	To/New Name	Help
List FDL Sources	(LS)		oldfrms		
Insert FDL Source	(IS)				
Modify FDL Source	(MS)				
Select FDL Source	(SS)				
Copy FDL Source	(CS)				
Rename FDL Source	(RS)				
Drop FDL Source	(DS)				
List Compiled form definitions	(LC)				
View Compiled form definition	(VC)				
Drop Compiled form definition	(DC)				
Exit form driven form editor	(EX)				

Msg: ☐ Enter command on command entry line or use menu selection application

Figure 5-10 Choosing SS Using Menu Selection

When this task is executed, the next screen displayed is a limited Edit Task screen as shown in section 5.6.

5.4.6 CS (Copy FDL Source)

Use this task to make a copy of an existing FDL source file. You must enter the name of the existing file and a new name for the copy.

The Command Entry format is: CS fromname toname<ENTER>

To execute this task using menu selection, enter any nonblank character in the "Pic" field for Copy form language Source, the existing FDL file name as the "For/From Name", the copy file name as the "To/New Name", and press the <ENTER> key. For example:

FORMS DRIVEN FORM EDITOR - VERSION 2.0 JUNE 1, 1985

Command Entry

WORK TASKS	Command	Pic	For/From Name	To/New Name	Help
List FDL Sources	(LS)	x	oldfrms	copyfrms	
Insert FDL Source	(IS)				
Modify FDL Source	(MS)				
Select FDL Source	(SS)				
Copy FDL Source	(CS)				
Rename FDL Source	(RS)				
Drop FDL Source	(DS)				
List Compiled form definitions (LC)					
View Compiled form definition (VC)					
Drop Compiled form definition (DC)					
Exit form driven form editor (EX)					

Msg: ☐ Enter command on command entry line or use menu selection application

Figure 5-11 Choosing CS Using Menu Selection

When this task is executed, the message "Copy was successful" is displayed in the message line and you remain at the Work Task level.

5.4.7 RS (Rename FDL Source)

Use this task to rename an existing FDL source file. You must enter both the old and the new names for the file.

The Command Entry format is: RS oldname newname<ENTER>

To execute this task using menu selection, enter any nonblank character in the "Pic" field for Rename form language Source, the old file name as the "For/From Name", the new file name as the "To/New Name", and press the <ENTER> key. For example:

FORMS DRIVEN FORM EDITOR - VERSION 2.0 JUNE 1, 1985

Command Entry

WORK TASKS	Command	Pic	For/From Name	To/New Name	Help
List FDL Sources	(LS)		oldfrms	newname	
Insert FDL Source	(IS)				
Modify FDL Source	(MS)				
Select FDL Source	(SS)				
Copy FDL Source	(CS)				
Rename FDL Source	(RS)				
Drop FDL Source	(DS)				
List Compiled form definitions (LC)					
View Compiled form definition (VC)					
Drop Compiled form definition (DL)					
Exit form driven form editor (EX)					

Msg: ☐ Enter command on command entry line or use menu selection application

Figure 5-12 Choosing RS Using Menu Selection

When this task is executed, the message "Rename was successful" is displayed in the message line and you remain at the work task level.

5.4.8 DS (Drop FDL Source)

Use this task to drop (delete) an entire FDL source file from your Form Definition Language source files. You must enter the name of the FDL file you want to delete. If this file has been compiled, the DC task must be used to delete the compiled forms that were created from this FDL file.

The Command Entry format is: DS filename ENTER

To execute this task using menu selection, enter any nonblank character in the "Pic" field for Drop form language Source, the name of the FDL source file you want to delete as the "For From Name", and press the ENTER key. For example:

FORMS DRIVEN FORM EDITOR - VERSION 2.0 JUNE 1, 1985

Command Entry

WORK TASKS	Command	Pic	For From Name	To New Name	Help
List FDL Sources	(LS)				
Insert FDL Source	(IS)				
Modify FDL Source	(MS)				
Select FDL Source	(SS)				
Copy FDL Source	(CS)				
Rename FDL Source	(RS)				
Drop FDL Source	(DS)		oldfms		
List Compiled form definitions (LC)					
View Compiled form definition (VC)					
Drop Compiled form definition (DC)					
Exit form driven form editor (EX)					

Msg: ☐ Enter command on command entry line or use menu selection application

Figure 5-13 Choosing DS Using Menu Selection

When this task is executed, the message "Drop was successful" is displayed in the message line and you remain at the work task level.

5.4.9 LC (List Compiled form definitions)

Use this task to display a list of all your compiled forms.

The Command Entry format is: LC<ENTER>

To execute this task using menu selection, enter any nonblank character in the "Pic" field for List Compiled form definitions and press the <ENTER> key. For example:

FORMS DRIVEN FORM EDITOR - VERSION 2.0 JUNE 1, 1985

Command Entry

WORK TASKS	Command	Pic	For/From Name	To/New Name	Help
List FDL Sources	(LS)				
Insert FDL Source	(IS)				
Modify FDL Source	(MS)				
Select FDL Source	(SS)				
Copy FDL Source	(CS)				
Rename FDL Source	(RS)				
Drop FDL Source	(DS)	X			
List Compiled form definitions	(LC)				
View Compiled form definition	(VC)				
Drop Compiled form definition	(DC)				
Exit form driven form editor	(EX)				

Msg: Enter command on command entry line or use menu selection application

Figure 5-14 Choosing LC Using Menu Selection

When this task is executed, the next screen displayed is:

```

FORMS DRIVEN FORM EDITOR - VERSION 2.0 JUNE 1, 1985

List Compiled Form Definitions

ITHINF2    MODFORM    ORJB05    RSHELP    TYPHELP
ITHINF3    MRECORDS   ORJB05A   RMFRONT   UPDCOM
ITHINF4    MRGNFRM    ORJB0P    SAVEFRM   VALHELP
ITHINF5    MSGLIN     ORJBOPM   SEAFRM    VCHHELP
ITHONLY    MSGLINE    PAGE45    SEMHELP   WFHELP
ITHVAL     MSGSCRN    PATHCOM   SFHELP    WHLCOM1
KEYPAD     MSHELP     PMHELP    SPEC       WHLCOM2
LAYOUT     NAMEFRM    POSHELP   SSHELP    WHLCINF
LCHHELP    NAMEHELP   PPAGE     STYPE     WHLEDIT
LENHELP    ORJB01     PSCREEN   SZFHELP   WHDPG1NP
LFHELP     ORJB01A    PSWORD    TASKINF   WHDPGR
LOADFRM    ORJB02     REFS      TASKS     WRKLIST
LOCHHELP   ORJB02A    REFLD     TEST      WRKMENU
LSHELP     ORJB03     REPHELP   TESTAP    WRKTASK
MENU       ORJB03A    REPLFRM   THHELP
MFHELP     ORJB04     REPTFRM   TOPFORM
MM         ORJB04A    RINGHELP  TOPHELP

Msg. ☐ Press (QUIT) to return
                                           application

```

Figure 5-15 List Compiled Form Definitions Screen

5.4.9.1 List Compiled Form Definitions Screen

This screen lists the names of all your FD files. When there are more names than can be displayed on the screen, the message "Press ENTER for more names or QUIT to return" is displayed in the message line. This will continue until all the names have been displayed and then the message "Press QUIT to return" is displayed. You will return to the work task screen when you press the QUIT key.

5.4.10 VC (View Compiled form definition)

Use this task to display a compiled form as it would appear when used in an application program. You must enter the name of the form you want to display.

The Command Entry format is: VC formname<ENTER>

To execute this task using menu selection, enter any nonblank character in the "Pic" field for View Compiled form definition, the name of the form as the "For/From Name", and press the <ENTER> key. For example:

FORMS DRIVEN FORM EDITOR - VERSION 2.0 JUNE 1, 1985

Command Entry

WORK TASKS	Command	Pic	For/From Name	To/New Name	Help		
List FDL Sources	(LS)						
Insert FDL Source	(IS)						
Modify FDL Source	(MS)						
Select FDL Source	(SS)						
Copy FDL Source	(CS)						
Rename FDL Source	(RS)						
Drop FDL Source	(DS)						
List Compiled form definitions (LC)							
View Compiled form definition (VC)	x						
Drop Compiled form definition (DC)							
Exit form driven form editor (EX)	(EX)						

Msg: Enter command on command entry line or use menu selection application

Figure 5-16 Choosing VC Using Menu Selection

When you execute this task, the form is displayed on the screen and the message "User view of the form, press the <QUIT> key to return" is displayed in the message line. You will return to the Work Task screen when you press the <QUIT> key

5.4.11 DC (Drop Compiled form definition)

Use this task to drop (delete) a compiled form from your compiled forms. You must enter the name of the form you want to delete.

The Command Entry format is: DC formname<ENTER>

To execute this task using menu selection, enter any nonblank character in the "Pic" field for Drop Compiled form definition, the name of the form as the "For/From Name", and press the <ENTER> key. When this task is executed, the message "Drop was successful" is displayed in the message line and you remain at the Work Task level. For example:

FORMS DRIVEN FORM EDITOR - VERSION 2.0 JUNE 1, 1985

Command Entry

WORK TASKS	Command	Pic	For/From Name	To/New Name	Help
List FDL Sources	(LS)				
Insert FDL Source	(IS)				
Modify FDL Source	(MS)				
Select FDL Source	(SS)				
Copy FDL Source	(CS)				
Rename FDL Source	(RS)				
Drop FDL Source	(DS)				
List Compiled form definitions (LC)					
View Compiled form definition (VC)					
Drop Compiled form definition (DC)					
Exit form driven form editor (EX)					

Msg: ☐ Enter command on command entry line or use menu selection application

Figure 5-17 Choosing DC Using Menu Selection

5.4.12 EX (EXit form driven form editor)

Use this task to terminate your current FDFE edit session.

The Command Entry format is: EX<ENTER>

To execute this task using menu selection, enter any nonblank character in the "Pic" field for EXit form driven form editor and press the <ENTER> key. Pressing the <QUIT> key from anywhere on the work task screen is another way to terminate your current FDFE edit session.

5.5 Edit Task Screen

This screen is a list of the FDFE functions that allow you to maintain the form definitions contained in the specified FDL source file and edit a specific form definition. It is the next screen displayed when you execute the IS or MS work task.

FORMS DRIVEN FORM EDITOR - VERSION 2.0 JUNE 1, 1985

Command Entry

EDIT TASKS	Command	Pic	Form Name	Edit Mode	Help
List	Forms(LF)				
Write	Forms(WF)				
Compile	Forms(CF)				
Select a Form (SF)					
Insert a Form (IF)					
Modify a Form (MF)					
Delete a Form (DF)					
Exit Write (EW)					
Exit Compile (EC)					
Exit No save (EN)					

Work Task: Insert FDL Source
Form Source: NEWFRMS

Msg: Enter command on command entry line or use menu selection application

Figure 5-18 Edit Task Screen

Note that the order of the edit tasks on the screen does not imply an execution order. For example, if you choose IS, you are creating a new FDL source file and there won't be any form definitions to list or modify until you have executed the INSERT task.

The values displayed in the "Work Task" and "Form Source" fields tell you which work task you chose and the FDL source file you are working with, respectively.

5.5.1 Choosing an Edit Task

Edit tasks can be chosen using menu selection or command entry as explained in the "Choosing a Work Task" section of this manual.

The following sections explain each of the edit tasks.

5.5.2 LF (List Forms)

Use this task to display the names of all the forms defined in the specified FDL source file. These names correspond to the form_name on the CREATE FORM entries of the FDL syntax as explained in the Form Definition Language Section of this manual.

The Command Entry format is: LF<ENTER>

To execute this task using menu selection, enter any nonblank character in the "Pic" field for List Forms and press the <ENTER> key. For example:

UM 620144400B
1 November 1985

FORMS DRIVEN FORM EDITOR - VERSION 2.0 JUNE 1, 1985

Command Entry

EDIT TASKS	Command	Pic	Form Name	Edit Mode	Help
List	Forms(LF)	x			
Write	Forms(WF)				
Compile	Forms(CF)				
Select a Form	(SF)				
Insert a Form	(IF)				
Modify a Form	(MF)				
Drop a Form	(DF)				
Exit Write	(EW)				
Exit Compile	(EC)				
Exit No save	(EN)				

Work Task: Modify FDL Source
Form Source: OLDFRMS

Msg: Enter command on command entry line or use menu selection application

Figure 5-19 Choosing LF Using Menu Selection

When this task is executed, the next screen displayed is:

```

FORMS DRIVEN FORM EDITOR - VERSION 2.0 JUNE 1, 1985

List of Forms for FDL File OLDFRMS

EDTTASK  ITMINF1  CDM2      RSHELP    LEMHELP    POSHELP
EDTALL   ITMFRM2  ITHCHMS   DSHELP    FTHelp     REPHELP
EDTLINe  ITMINF2  FRMINFO   LCHELP    FTHelp2    HLPHELP
EDTSEL   ITMFRM3  FRMCINF   VCHELP    FTHelp     VALHELP
CURINFO  ITMINF3  FRMPRMT   DCHELP    SZFHELP    EFHELP
WRKTASK  WHLCDM1  CDMINFO   LFHELP    BKHELP     RRGHELP
FDFEVSX  ITMFRM4  FLDARY    SFHELP    PTHelp     FTHelp
WRKMENU  ITMINF4  ITHHELP   SEMHELP   DIRHELP
WHLEDY   WHLCDM2  ITHVAL    WFHELP    FDMHELP
WHLCINF  ITMFRM5  LAYOUT    DFHELP    TYPHELP
DELFRM   ITMINF5  EDTLFLS   ECHELP    NAMHELP
DELWHL   ITHCHCK  WRKLIST   ENHELP    FTHelp
FLDTOP   FLDEDY   LSHelp    IFHELP    LOChelp
COMTITL  FLDCINF  ISHELP    WFHELP    FDSHELP
FLDINFO  ITHONLY  NSHELP    DFHELP    DABHELP
REPFLD   CDMINFO  SSHELP    ENHELP    DABHLP2
ITMFRM1  CDM1     CSHELP    ENHELP2   FDPHELP

Msg: ☐ Press (QUIT) to return
application

```

Figure 5-20 List of Forms Screen

5.5.2.1 List of Forms Screen

This screen lists the names of all the forms defined in the current FDL source file. When there are more names than can be displayed on the screen, the message "Press (ENTER) for more names or (QUIT) to return" is displayed in the message line. This will continue until all the names have been displayed and then the message "Press (QUIT) to return" is displayed. You will return to the Edit Task screen when you press the (QUIT) key.

5.5.3 WF (Write Forms)

Use this task to save the form definitions(s) you are currently working on in the FDL source file shown in the "Form Source" field.

The Command Entry format is: WF<ENTER>

To execute this task using menu selection, enter any nonblank character in the "Pic" field for Write Forms and press the <ENTER> key. For example:

FORMS DRIVEN FORM EDITOR - VERSION 2.0 JUNE 1, 1985

Command Entry

EDIT TASKS	Command	Pic	Form Name	Edit Mode	Help
List	Forms(LF)				
Write	Forms(WF)				
Compile	Forms(CF)				
Select a Form (SF)					
Insert a Form (IF)					
Modify a Form (MF)					
Drop a Form (DF)					
Exit Write (EW)					
Exit Compile (EC)					
Exit No save (EN)					

Work Task: Modify FDL Source
Form Source: OLDFRMS

Msg: Enter command on command entry line or use menu selection application

Figure 5-21 Choosing WF Using Menu Selection

When this task is executed, the form definitions are compiled since only syntactically correct form definitions can be edited in the FDFE. Any syntax errors are stored in the message queue and can be displayed using the <MESSAGE QUEUE> key. When there are no syntax errors and the write is complete, you remain at the Edit Task level. Note that no FD files are created, only the FDL file is created.

5.5.4 CF (Compile Forms)

Use this task to compile and save the form definition(s) contained in the FDL source file shown in the "Form Source" field without exiting the FDFE.

The Command Entry format is: CF<ENTER>

To execute this task using menu selection, enter any nonblank character in the "Pic" field for Compile Forms and press the <ENTER> key. For example:

FORMS DRIVEN FORM EDITOR - VERSION 2.0 JUNE 1, 1985

Command Entry

EDIT TASKS	Command	Pic	Form Name	Edit Mode	Help
List	Forms(LF)				
Write	Forms(WF)				
Compile	Forms(CF)				
Select a Form	(SF)				
Insert a Form	(IF)				
Modify a Form	(MF)				
Drop a Form	(DF)				
Exit Write	(EW)				
Exit Compile	(EC)				
Exit No save	(EN)				

Work Task: Modify FDL Source
Form Source: OLDFRMS

Msg: Enter command on command entry line or use menu selection application

Figure 5-22 Choosing CF Using Menu Selection

If there are syntax errors, the message "Form compile errors, review and correct" is displayed in the message line. The errors are stored in the message queue which can be displayed using the <MESSAGE QUEUE> key. When there are syntax errors, you must correct them and reexecute the CF task in order

to obtain both an FDL file and FD file(s) using FDFE tasks. If you exit without correcting the syntax errors, the FDL source file is saved as a TMP file that may be edited using EDT or some other editor and then compiled using FLAN. The FDFE can only modify FDL source files free of error.

When the forms are compiled with no syntax errors, the form definitions are saved in the FDL source file shown in the "Form Source" field and the compiled forms are saved as FD files. At this time you can press the (QUIT) key to return to the Work Task screen where you can display a form definition using the VC task or choose another FDL source file to work with.

5.5.5 SF (Select a Form)

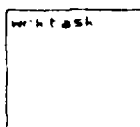
Use this task to review the characteristics and attributes of an individual form contained in the FDL source file shown in the "Form Source" field. You must enter the name of the form you want to review and an edit mode to determine the format the form definition will be presented in. Your format choices are S for single field mode, F for form mode, or L for layout mode.

The Command Entry format is: SF formname editmode(ENTER)

To execute this task using menu selection, enter any nonblank character in the "Pic" field for Select a Form, the name of the form definition you want to review as the "Form Name", S, F, or L as the "Edit Mode", and press the (ENTER) key. For example:

FORMS DRIVEN FORM EDITOR - VERSION 2.0 JUNE 1, 1985

Command Entry

EDIT TASKS	Command	Pic	Form Name	Edit Mode	Help
List	Forms (LF)				
Write	Forms (WF)				
Compile	Forms (CF)				
Select a Form (SF)					
Insert a Form (IF)					
Modify a Form (MF)					
Drop a Form (DF)					
Exit Write	(EW)				
Exit Compile	(EC)				
Exit No save	(EN)				

Work Task: Modify FDL Source
Form Source: OLDFRMS

Msg: ☐ Enter command on command entry line or use menu selection application

Figure 5-23 Choosing SF Using Menu Selection

When this task is executed, the next screen displayed depends on the Edit Mode you chose. The next three sections describe how to review a form using each of the modes. After reviewing a form, press the <QUIT> key to return to the Edit Task screen.

5.5.5.1 Using Single Field Mode to Review a Form

When you execute the SF edit task with the "S" edit mode, the next screen displayed is:

FIELD EDIT MODE		For FDL File OLDFRMS	
TASK S		FORM WRK TASK	
Field		Size	78 by 17
Type		Background	BLACK
Direct		Prompt	WORK TASKS
Get FDL		Row	6 Column 2
-- Field Information --			
REQUIRED:		OPTIONAL:	
FIELD		Display	
Row		Actual	Field
Size		Direction	Repetition
Display/		Spacing	(-)
Background		Prompt	
ITEM ONLY:		Pos	
Help			
Value			
Justify	Data Type	MIN Value	
Case	Enter/Fill	MAX Value	
Msg: Enter field to be viewed			
application			

Figure 5-24 Reviewing a Form in Single Field Mode

The SF task only allows you to read the form definition. This is shown by the "S" value in the "TASK" field which cannot be changed. The characteristics that apply to the entire form are displayed in the FORM area of the screen. The characteristics of the individual fields are displayed in the field information template. An explanation of each of the fields in the template is given in sections 5.5.6.1.2, 5.5.6.1.3, and 5.5.6.1.4.

There are several ways to specify which field you want to display. The <NEXT> and <PREV> function keys display the next or previous field with respect to the current field being displayed. Note that when this screen is displayed after choosing the SF edit task, the field information template is blank so you would use the <NEXT> key first.

You can use the "Type" and "Direct" fields in the TASK area of the screen to specify the field to be displayed. Use the "Type" field to say whether you want to display only items, forms, windows, or all the field types. The values are I, F, W, and A. Use the "Direct" field to display the next (N), previous (P), beginning (B), or ending (E) field in the form definition. Note that the "Field" field must be blank when using these fields and the "Type" field is ignored if you use the <NEXT> and <PREV> function keys. After entering the appropriate values, press the <ENTER> key to display the desired field.

You can also display a specific field by entering the field name in the "Field" field in the TASK area of the screen and pressing the <ENTER> key. Note that a value in this field overrides the "Type" and "Direct" fields.

When you are through reviewing the form, press the <QUIT> key to return to the Edit Task screen.

5.5.5.2 Using Form Mode to Review a Form

When you execute the SF edit task with the "F" edit mode, the next screen displayed is:

FORM EDIT MODE For FDL File OLDFRMS

TASK S

Type Form

Get FDL

FORM WRT TASK

Size 78 by 17

Background BLACK

Prompt WORK TASKS

Row 6 Column 2

-- Field Characteristics Table --

msg: Enter type of fields to be viewed

application

Figure 5-25 Reviewing a Form in Form Mode

The characteristics that apply to the entire form are displayed in the FORM area of the screen. As in Single Field mode, the "TASK" field value is "S" and cannot be changed. The characteristics for the fields defined on the form can be displayed in the FIELD CHARACTERISTICS TABLE. The field characteristics contained in this table correspond to the field information template as described in sections 5.5.6.1.2, 5.5.6.1.3, and 5.5.6.1.4. Use the "Type" field in the TASK area of the screen to specify which fields you want to review. Your choices for this field are:

A to display all the fields on the form.
I to display only the item fields on the form.
F to display only the form fields on the form.
W to display only the window fields on the form.

After entering the "Type" field value, press the (ENTER) key to display the FIELD CHARACTERISTICS TABLE on the screen. For example:

FORM EDIT MODE
For FDL File OLDFRMS

TASK S

Type

Get FDL Form

FORM WRTTASK

Size by

Background

Prompt

Row Column

-- Field Characteristics Table --

Field Name	T	Row	Col	Size	Display	Dsp	Act	D	Sp	Prompt	Fos
FFFEVSN	F	1	1	78	1	BLACK					
CMD	I	4	19	6	1	INFUT				Command Entry	LT
WRKMENU	F	7	39	35	1	BLACK	11	11	V		
LSHELF	I	7	76	1	1	INFUT				List FDL Sources	
ISHELF	I	8	76	1	1	INFUT				Insert FDL Source	
MSHELF	I	9	76	1	1	INFUT				Modify FDL Source	
SSHELF	I	10	76	1	1	INFUT				Select FDL Source	
CSHELF	I	11	76	1	1	INFUT				Copy FDL Source	
ASHELF	I	12	76	1	1	INFUT				Rename FDL Source	
DSHELF	I	13	76	1	1	INFUT				Drop FDL Source	
LCHELF	I	14	76	1	1	INFUT				List Compiled form	
VCHELF	I	15	76	1	1	INFUT				View Compiled form	

Msg:
application

Figure 5-26 FIELD CHARACTERISTICS TABLE - Part One

For item fields, the FIELD CHARACTERISTICS TABLE contains more information which you can display by entering any nonblank character in the "More" field below the table on this screen and the next screens displayed and pressing the (ENTER) key. The remainder of the table is shown in the following three screens:

UM 620144400B
1 November 1985

FORM EDIT MODE For FDL File OLDFRMS

TASK S

Type A

Get FDL Form

FORM WRTASK

Size 78 by 17

Background BLACK

Prompt WORK TASKS

Row 6 Column 2

-- Field Characteristics Table --

Item name	Help Message
FDREVSN	
CND	
WRK MENU	
LSHELP	FORM=LSHELP
ISHLP	FORM=ISHLP
MSHELP	FORM=MSHELP
SSHELP	FORM=SSHELP
CSHELP	FORM=CSHELP
RSHELP	FORM=RSHELP
DSHELP	FORM=DSHELP
LSHELP	FORM=LSHELP
VDHELP	FORM=VDHELP

☐ More

Msg: application

Figure 5-27 FIELD CHARACTERISTICS TABLE - Part Two

UM 6201444-3
1 November 1988

FORM EDIT MODE For FDL File OLDFRMS

TASK S FORM WRTASK

Type A Size 78 by 17

Get FDL Form Background BLACK

Prompt WORK TASKS

Row 6 Column 2

-- Field Characteristics Table --

Item Name	Item Value
FDPEVSN	
CMD	
WRKMENU	
LSHELP	
ISHLP	
MSHELP	
SSHELP	
CSHELP	
RSHELP	
DSHELP	
LDHELP	
VDHELP	

☐ More--

Msg: ☐

Figure 5-28 FIELD CHARACTERISTICS TABLE - Part 2

AD-A182 581

INTEGRATED INFORMATION SUPPORT SYSTEM (IISS) VOLUME 8 2/2

USER INTERFACE SUBS (U) GENERAL ELECTRIC CO

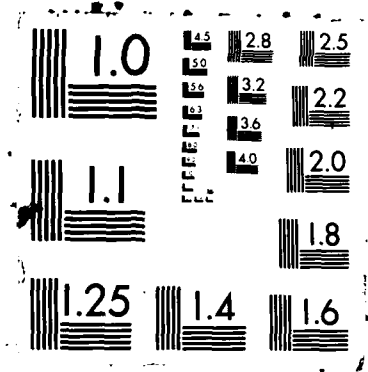
SCHENECTADY NY PRODUCTION RESOURCES CONSU

UNCLASSIFIED C MORENC ET AL 81 NOV 85 UN-6281444888

F/G 12/5

NL

										END		
										8-8/		
										DTIC		



FORM EDIT MODE
For FDL File OLDFRMS

TASK 8

Type A

Get FDL Form

FORM WRTASK

Size 78 by 17

Background BLACK

Prompt WORK TASKS

Row 6 Column 2

-- Field Characteristics Table --

Item Name	Just	Case	Type	E/F	Min	Max
FDFEVSX						
CMD	L	U				
WRKMENU						
LSHELP						
ISHLP						
WSHELP						
SSHELP						
CSHELP						
RSHELP						
DSHELP						
LDHELP						
VDHELP						

Msg: 0
application

Figure 5-29 FIELD CHARACTERISTICS TABLE - Part Four

Note that all field names are displayed on these table parts, but values for the characteristics appear for item fields only.

All four parts of the FIELD CHARACTERISTICS TABLE scroll vertically so that you can review all the fields contained in a form if there are more than 12. To activate the paging and scrolling keys, press the **<MODE>** key until "scrl/page" appears in place of "application" in the mode field as explained in the IISS Terminal Operator Guide. You can also use these keys to scroll the "Prompt" field on the first part of the table horizontally so that an entire prompt can be seen if it contains more than 15 characters. The Field Repetition fields (Dsp, Act, D, Sp) also scroll horizontally. Return the FP mode field to "application" before pressing the **<QUIT>** key to return to the Edit Task screen.

5.5.5.3 Using Layout Mode to Review a Form

When you execute the SF edit task with the "L" edit mode, the form is displayed using the layout symbols described in section 5.5.6.3. For example:

```

)Command Entry)(-----)

WORK TASKS                                Command Pic For/From Name To/New Name@
List   FDL Sources                        (LS) [-----] _
Insert FDL Source                         (IS) ~~~~~ _
Modify FDL Source                         (MS) ~~~~~ _
Select FDL Source                         (SS) ~~~~~ _
Copy   FDL Source                         (CS) ~~~~~ _
Rename FDL Source                         (RS) ~~~~~ _
Drop   FDL Source                         (DS) ~~~~~ _
List   Compiled form definitions(LC) ~~~~~ _
View   Compiled form definition (VC) ~~~~~ _
Drop   Compiled form definition (DC) ~~~~~ _
Exit   form driven form editor (EX) ~~~~~ _

Msg: 1 Field at row 1 begins in column #1                                application
```

Figure 5-30 Reviewing a Form in Layout Mode

Press the <QUIT> key to return to the Edit Task screen.

5.5.6 IF (Insert a Form)

Use this task to define a new form. When saved, this form definition will be inserted into the FDL source file displayed in the "Form Source" field. You must enter the name of the form that you are going to define and an edit mode to determine the editing style you want to use to define the form.

The Command Entry format is: IF formname editmode<ENTER>

To execute this task using menu selection, enter any nonblank character in the "Pic" field for Insert a Form, the name of the form as the "Form Name", S, F, or L as the "Edit Mode", and press the <ENTER> key. For example:

FORMS DRIVEN FORM EDITOR - VERSION 2.0 JUNE 1, 1985

Command Entry

EDIT TASKS	Command	Pic	Form Name	Edit Mode	Help
List	Forms(LF)		newform	I	
Write	Forms(WF)				
Compile	Forms(CF)				
Select a Form (SF)					
Insert a Form (IF)					
Modify a Form (MF)					
Drop a Form (DF)					
Exit Write (EW)					
Exit Compile (EC)					
Exit No save (EN)					

Work Task: Modify FDL Source
Form Source: OLDFRMS

Msg: Enter command on command entry line or use menu selection application

Figure 5-31 Choosing IF Using Menu Selection

The next screen displayed depends on the Edit Mode you choose. The next three sections explain how to define a form using the single field, form, and layout edit modes.

5.5.6.1 Using Single Field Edit Mode to Define a Form

When you execute the IF edit task with the "S" edit mode, the next screen displayed is:

FIELD EDIT MODE
For FDL File OLDFRMS

TASK

Field

Type

Direct

Get FDL Form

FORM NEWFORM

Size by

Background

Prompt

Row Column

-- Field Information --

REQUIRED:

FIELD Type

Row Column

Size By

Display/
Background

OPTIONAL:

Display Field
Actual Repetition
Direction (-)

Spacing

Prompt

Pos Row Col

ITEM ONLY:

Help

Value

Justify Data Type MIN Value

Case Enter/Fill MAX Value

Msg: Enter task and field to be acted on application

Figure 5-32 Defining a Form in Single Field Mode

When defining a new form in single field edit mode, the first thing you need to do is define the attributes that apply to the entire form in the "FORM" area fields. The name of the form that you are defining is displayed as the "FORM" field value. This is the value you entered in the "Form Name" field on the previous screen and it cannot be changed.

5.5.6.1.1 Form Area Fields

Size is used to specify the number of columns and rows the form will occupy when it is displayed. These values are optional and if specified, the size of the form or window this form is displayed in takes precedence. This means that the form will be "clipped" if the form size is larger than the size

of the form or window this form is displayed in. When the form size is smaller than the form or window size, the form "grows" to fill the window. If you do not specify the size of the form, it defaults to the size of the form or window it is displayed in. The format is "n by m" where n is the number of columns wide and m is the number of rows deep.

- Background** allows you to define the background of the form. This is analogous to printing paper forms on different colors of paper. You have the option of displaying white characters on a black background or black characters on a white background (sometimes known as reverse video). The valid values for this field are WHITE and BLACK. This field defaults to BLACK.
- Prompt** is used to display text strings on the form that are not related to any given field. This type of prompt is used primarily for titles and formatting. This field scrolls horizontally to allow the text strings to be up to 75 characters long and vertically so that up to 30 form prompts can be defined. To activate the paging and scrolling keys, press the <MODE> key until "scrl/page" appears in place of "application" in the mode field. When you are through defining the form prompts, press the <MODE> key again until "application" appears in the mode field.
- Row and Col** are used to specify where the first character of a form prompt will be positioned on the form. The prompt is positioned with respect to the upper left corner of the form. When the "Prompt" field is scrolled vertically, these fields are also scrolled so that the position of each form prompt can be specified.

You define the fields for the form one at a time using a field information template. The INSERT task allows you to enter the desired values in the template. The COPY task retrieves the values from fields defined on an existing form. You can then modify these values and execute the INSERT task. The next three

sections explain the information that defines a field.

5.5.6.1.2 Required Field Information

FIELD	is a unique name of up to 10 letters, numbers, and/or underscores that identifies the current field you are defining. Note that if the field is a form, underscores may not be used.
Type (T)	is used to identify the field you are defining as a form (F), window (W), or item (I).
Row and Col	are used to specify where the field will be positioned on the form. The field is positioned with respect to the upper left corner of the form.
Size	is used to determine the area on the form that each occurrence of the field will occupy. The format is "n by m" where n is the number of columns wide the field is and m is the number of rows deep the field is. Note that when a form is displayed in a form or window field, the size of the form or window takes precedence over the defined size of the form being displayed. This means that the form will be "clipped" if the form definition size is larger than the field size. The converse of this (i.e., when the form size is smaller than the field size) is useful when the field is a window. It allows you to display a "pop-up" menu as the next page in the window without overwriting the entire form on the previous page.
Display/ Background	controls access to an item field and determines how it appears to the user for item fields and is analogous to printing paper forms on different colors of paper for form and window fields. Valid values are: INPUT which means that the user may enter a value for this item and the value is echoed on the screen. The area where the user may type is highlighted on the form. The IISS Terminal Operator Guide explains what "highlight" means for each terminal type. OUTPUT which means that only the application program may enter a value for this item. The value is displayed in bold type on terminals that support it.

TEXT which is the same as OUTPUT except the value is not displayed in bold type.

HIDDEN which means that the user may enter a value for this item but the value is not echoed on the screen. This option is typically used for items of privileged information such as passwords. The area where the user may type is highlighted on the form. The IISS Terminal Operator Guide explains what "highlight" means for each terminal type.

BLACK which displays white characters on a black background.

WHITE which displays black characters on a white background (sometimes known as reverse video).

5.5.6.1.3 Optional Field Information

Field Repetition Fields

are used to specify that the field appears on the form more than once to create arrays of fields and whether or not the array is scrollable. The "<->" symbol indicates that these fields all scroll horizontally to allow for multiple array dimensions as explained in the Repeat Spec Entry Item section of this manual. To activate the paging and scrolling keys, press the <MODE> key until "scrl/page" appears in place of "application" in the mode field. When you are through defining the field repetition, press the <MODE> key again until "application" appears in the mode field. "Display" is how many times to repeat the field on the form.

"Actual" is the actual number of times the field occurs.

"Direction" tells whether the field repeats in a horizontal (H) or vertical (V) direction.

"Spacing" tells how many blank spaces to leave between repetitions.

Prompt

is used to specify information such as labels and instructions that will be associated with the field. This field scrolls horizontally so that the prompt text can be up to 75 characters long. The prompt is positioned relative to the field based on the value you enter in the "Pos" field. The paging scrolling keys are activated as explained for the Field Repetition fields.

Pos is used to position a field prompt in one of 12 locations relative to the field. The absolute row and column position is then calculated for you and displayed in the "Row" and "Col" fields after the "Pos" field. The 12 relative locations are:

Top Left (TL)				
Top Center (TC)		TL	TC	TR
Top Right (TR)		x	x	x
Bottom Left (BL)		-----		
Bottom Center (BC)	LT	x		x RT
Bottom Right (BR)				
Left Top (LT)	LC	x	field	x RC
Left Center (LC)				
Left Bottom (LB)	LB	x		x RB
Right Top (RT)		-----		
Right Center (RC)		x	x	x
Right Bottom (RB)		BL	BC	BR

For prompts to the left of the field, the last character of the prompt is positioned one space left of the field. For prompts to the right of the field, the beginning character is one space to the right of the field. For prompts above and below the field, care is taken to have the prompt for left begin at the left margin of the field, for right to end at the right margin, and for center to be centered on the width of the field.

5.5.6.1.4 Item Only Information

Help is used to specify a string or another form to provide information that will be displayed when the cursor is in this field and the <HELP> key is pressed. The help string can be up to 60 characters and is displayed in the message line of the screen when the <HELP> key is pressed. For a help form, enter "FORM=formname" where "formname" is the name of the form that you want displayed when the <HELP> key is pressed. No double quotes should be included when entering this information in the "Help" field. The help form can be defined in the current FDL source file or a different one. You can also enter the word "APPLICATION" in the "Help" field to specify that the application program will decide what to do when the user presses the <HELP> key. Quotes are currently not

used to enclose a help message but in the next release, standard FDL syntax for the help information will be used. It will no longer be necessary to proceed help form names with "FORM=".

Value	is used to specify a default value for an item field. This value will be shown in the field when the form is first displayed to the user. If the user does not enter another value for the item, this value is returned to the application program as the value for the item. The default value must be less than or equal to the total number of characters specified by the field size and must be enclosed in double quotes ("value_string").
Justify	is used to right (R) or left (L) justify the value in the field.
Data Type	is used to specify that the value entered for the field must be numeric (N) or character (C or blank)
MIN/MAX Value	is used to set limits for the values that can be entered for numeric item fields. You can define a range of acceptable values by specifying both a MIN and a MAX value. The field is automatically numeric when a MIN/MAX value is entered.
Case	is used to specify that the value entered for the field be converted to upper (U) or lower (L) case.
Enter/Fill	is used to specify whether the user must enter a value for the item and whether the value must fill the field. Valid values are: blank - entering a value is optional. E - a value must be entered in order to process the form. F - A value filling every position of the field must be entered to process the form.

To execute the COPY task, enter "C" in the "TASK" field, use the "Field" or "Type" and "Direct" fields to identify the first field to copy from the existing form, enter values in the "Get FDL" and "Form" fields to identify the existing form, and press the **ENTER** key. You can then use the **NEXT** and **PREV** function keys or the "Field" or "Type" and "Direct" fields as described in section 4.5.5.1 to copy the other fields on the existing form. When you have identified a field you want to

insert into the current form definition, execute the INSERT task. Note that the COPY task does not change the existing form definition in any way.

To execute the INSERT task, enter "I" in the "TASK" field and press the <ENTER> key. The message "Make your insertions now" is displayed in the message line. At this time you can enter the desired values for the field information or make any modifications to the information copied from an existing form. When the field is defined exactly as you want, press the <ENTER> key again. The message "Field successfully inserted" will be displayed in the message line. If you decide that you do not want to insert the field currently defined in the template, press the <QUIT> key instead of the <ENTER> key to cancel the current INSERT task.

After you have inserted fields in the form, you can use the SELECT, MODIFY, FORM, and DROP field tasks as described in section 5.5.7.1 to review and modify the current form definition.

5.5.6.2 Using Form Mode to Define a Form

When you execute the IF edit task with the "F" edit mode, the next screen displayed is:

FORM EDIT MODE For FDL File OLDFRMS

TASK
Type
Get FDL

FORM NEWFORM
Size 0 by 0
Background BLACK
Prompt Example Form
Row 1 Column 10

-- Field Characteristics Table --

Msg: 1 Enter task and type of fields to be acted on application

Figure 5-33 Form Mode Screen

When defining a new form in form edit mode, the first thing you need to do is define the attributes that apply to the entire form in the FORM area fields. The name of the form that you are defining is displayed as the "FORM" field value. This is the value you entered in the "Form Name" field on the previous screen and it cannot be changed on this screen. The remaining fields in the FORM area of the screen are described in section 5.5.6.1.1.

Fields for the form are defined by making entries in the FIELD CHARACTERISTICS TABLE. This table is displayed on the screen by executing the INSERT or COPY field task.

The INSERT task allows you to define fields by entering values for the characteristics directly into the table. The COPY task retrieves field characteristics from an existing form definition and displays them in the table. You can then modify the table entries and execute the INSERT task when the fields are defined as you want them.

To execute the COPY task, enter "C" in the "TASK" field, "I", "F", "W", or "A" in the "Type" field to copy item, form, window, or all field types, the name of the existing form and the FDL it is contained in, and press the <ENTER> key. The fields identified will then be displayed as entries in the FIELD CHARACTERISTICS TABLE. After modifying the entries so that the fields are defined as you want and blanking out the names of the fields you do not want to insert, press the <ENTER> key and then execute the INSERT task. The blanks will be ignored and all fields in the table without blank field names will be inserted.

To execute the INSERT task, enter "I" in the "TASK" field, "A" in the "Type" field, and press the <ENTER> key. The message "Make your insertions now" is displayed in the message line. At this time you can enter the desired values for the field information or make any further modifications to the field information copied from the existing form. For example:

FORM EDIT MODE For FDL File OLDFRMS

TASK 1

Type a

Get FDL Form

FORM NEWFORM

Size 0 by 0

Background BLACK

Prompt EXAMPLE FORM

Row 1 Column 10

-- Field Characteristics Table --

Field Name	T	Row	Col	Size	Display	Dsp	Act	D	Sp	Prompt	Pos
FIELD1	I	3	15	15	1	INFUT				Item 1:	LT
FIELD2	I	4	15	15	1	INFUT				Item 2:	LT
FIELD3	W	7	15	15	5	BLACK				Window 1	TL

More--

Msg: 1 Make Your Insertions now' application

Figure 5-34 Defining a Form in Form Mode

To define item fields there are three more parts to the table which you display by entering any nonblank character in the "More" field below the first three parts of the table as explained in section 5.5.5.2. All four parts of the table scroll vertically so that up to 30 fields can be defined. To activate the paging and scrolling keys, press the (MODE) key until "scrl1/page" appears in place of "application" in the mode field.

The information that defines a field in this table corresponds to the field information template used to define a field in Single Field edit mode as explained in sections 5.5.6.1.2, 5.5.6.1.3, and 5.5.6.1.4.

When the fields are defined exactly as you want, press the `<ENTER>` key again. A message saying the last field defined in the table was successfully inserted is displayed in the message line. The number preceding the message line tells how many fields were inserted. You can use the `<MESSAGE QUEUE>` key to view the message queue and verify that the other fields in the table were successfully inserted.

After you have inserted fields on the current form, you can use the `SELECT`, `MODIFY`, `FORM`, and `DROP` field tasks as explained in section 5.5.7.2 to review and modify them.

5.5.6.3 Using Layout Mode to Define a Form

When you execute the IF edit task with the "L" edit mode, the next screen displayed is:

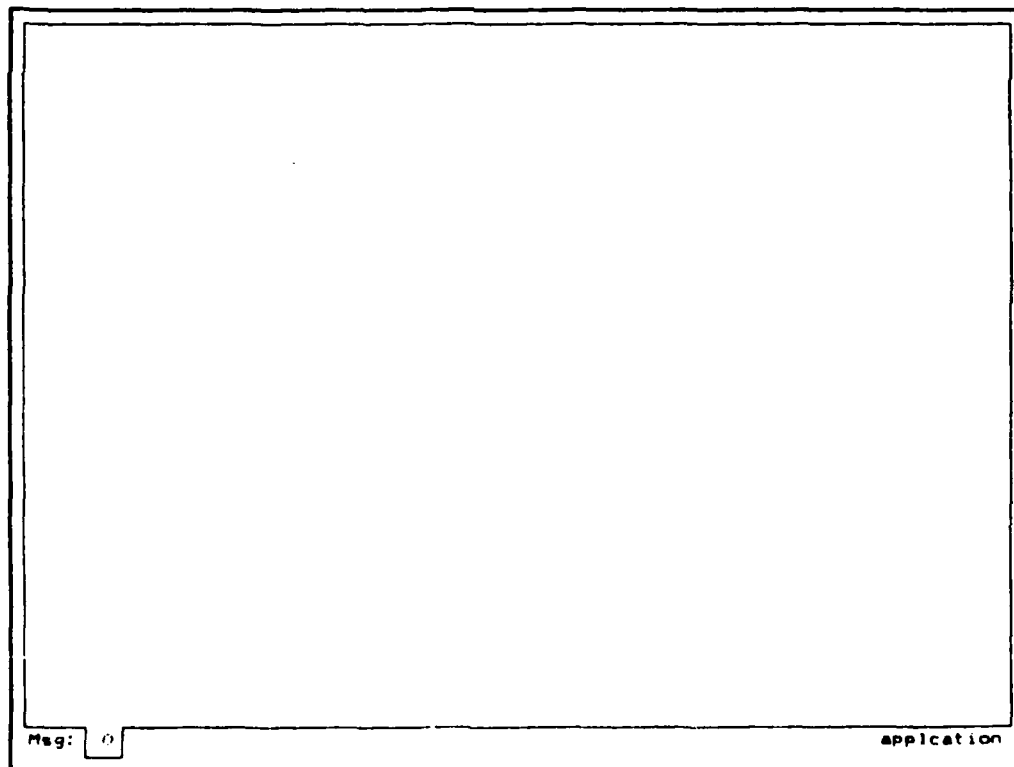


Figure 5-35 Layout Mode Screen

This screen allows you to graphically position fields and prompts on the form using special symbols. This saves time since location parameters and field sizes are calculated for you. However, all fields defined in Layout mode are assumed to be input items and you cannot define arrays of fields. To override these defaults, you can press the <SWITCH> key to go into Single Field mode and modify the information for the individual fields. You can specify which field to work on first by positioning the cursor on the first position of the desired field before pressing the <SWITCH> key. You can remain in Single Field mode to modify any of the fields as explained in section 5.5.7.1 or insert new fields as explained in section 5.5.6.1. Pressing the <SWITCH> key again will return you to Layout mode. Whether or not you use the <SWITCH> key, when the form is defined as you want it, press the <ENTER> key to record the form definition. Then press the <QUIT> key to return to the Edit Task screen and execute the WF or EW task to save the form definitions. NOTE that if you press the <QUIT> key to return to the Edit Task screen without first pressing the <ENTER> key, you will lose the current form definition data entered.

5.5.6.3.1 Layout Mode Symbols

- Indicates a field position. For example, defines a five position field. Only one dimension horizontal fields may be defined this way. Note that when fields defined this way are reviewed in Layout mode, they are converted to the bracket format described in the next paragraph.
- [] Indicates the first and last positions of a field, inclusively. For example, [] also defines a five position field. These symbols may be used to define two dimension as well as one dimension fields. When displayed in layout mode, the area between the [] is filled with dashes. For example, [-----] or

[-----]
[-----]
[-----].
- @ Defines enclosed text as background text on the form.
- , Defines enclosed text as a prompt associated with field to immediate right.

- ' Defines enclosed text as a prompt associated with field to immediate left.
- " Defines enclosed text as a prompt associated with field directly below.
- ^ Defines enclosed text as a prompt associated with field directly above.

Note that the starting or ending symbol for a prompt is not necessary if the first or last character of the prompt is at the edge of the form.

- Indicates occurrences of fields that have been defined as arrays using Single Field or Form mode.
- @ Indicates a field to be moved when entered in first position of the field. Then position the cursor at new field location and press the <MOVE> key. Note that only one field at a time may be moved.
- ? Indicates an error when in the first position of a field.

The following screen shows how the form "NEWFORM" defined as shown in section 5.5.6.2 appears in Layout mode.

```
SAMPLE FORM

Item 1: [-----]
Item 2: [-----]

Window 1
[-----]
[-----]
[-----]
[-----]
[-----]

Msg: 0 Application
```

Figure 5-36 Defining a Form in Layout Mode

5.5.7 MF (Modify a Form)

Use this task to modify an existing form definition in the FDL source file shown in the "Form Source" field. This includes inserting new field definitions in the form and modifying any that the form already contains. You must enter the name of the form that you are going to modify and an edit mode to determine the editing style you want to use to modify the form.

The Command Entry format is: MF formname editmode<ENTER>

To execute this task using menu selection, enter any nonblank character in the "Pic" field for Modify a Form, the name of the form as the "Form Name", S, F, or L as the "Edit Mode", and press the <ENTER> key. For example:

FORMS DRIVEN FORM EDITOR - VERSION 2.0 JUNE 1, 1985

Command Entry

EDIT TASKS	Command	Pic	Form Name	Edit Mode	Help
List	Forms (LF)	wrktask		S	
Write	Forms (WF)				
Compile	Forms (CF)				
Select a Form	(SF)				
Insert a Form	(IF)				
Modify a Form	(MF)				
Drop a Form	(DF)				
Exit Write	(EW)				
Exit Compile	(EC)				
Exit No save	(EN)				

Work Task: Modify FDL Source
Form Source: OLDFRMS

Msg: ☐ Enter command on command entry line or use menu selection application

Figure 5-37 Choosing MF Using Menu Selection

The next screen displayed depends on the Edit Mode you choose. The next three sections explain how to modify a form using the Single Field, Form, and Layout edit modes.

5.5.7.1 Using Single Field Mode to Modify a Form

When you execute the MF edit task with the "S" edit mode, the next screen displayed is:

```

FIELD EDIT MODE                               For FDL File OLDFRMS

TASK  Field  Type  Direct  Get FDL  Form
FORM NEWFORM
Size  0 by 0
Background BLACK
Prompt EXAMPLE FORM
Row  1 Column 10

-- Field Information --
REQUIRED:
FIELD  Row  Size  Display/Background
OPTIONAL:
Display Actual Direction Spacing
Field Repetition (-)
Prompt Pos Row Col

ITEM ONLY:
Help Value Justify Case Data Type Enter/Fill MIN value MAX value

Msg: 1 Enter task and field to be acted on.
application
  
```

Figure 5-38 Single Field Mode Screen

If you want to modify information in the FORM area of the screen, make the desired changes, enter "F" in the "TASK" field, and press the **(ENTER)** key.

You can insert new fields in the form by executing the INSERT or COPY field task as described in section 5.5.6.1.

You can delete fields from the form using the DROP field task. To execute the DROP task, enter "D" in the "TASK" field, use the "Field" or "Type" and "Direct" fields to identify the field you want to delete, and press the **(ENTER)** key or press the

(NEXT) or (PREV) key after entering "D" in the "TASK" field to identify the field to be dropped. The field is displayed in the field information template and the message "Field on screen dropped!" is displayed in the message line. For example:

FIELD EDIT MODE		For FDL File OLDFRMS	
TASK	D	FORM NEWFORM	
Field	FIELD	Size	0 by 0
Type		Background	BLACK
Direct		Prompt	EXAMPLE FORM
Get FDL	Form	Row	1 Column 10
-- Field Information --			
REQUIRED:		OPTIONAL:	
FIELD FIELD	Type 1	Display	Field
Row 3	Column 10	Actual	Repetition
Size 15	By 1	Direction	(-)
Display INPUT		Spacing	
Background		Prompt	Item 1:
		Pos	LT Row 3 Col 7
ITEM ONLY:			
Help			
Value			
Justify	Data Type	MIN Value	
Case	Enter Field	MAX Value	
Msg: 1 Field on screen dropped!			
application			

Figure 5-39 Using Single Field Mode to Delete a Field

If you decide that you do not want this field deleted, execute the INSERT task to cancel the DROP task by immediately entering "I" in the task field and pressing the (ENTER) key.

You can modify any of the fields that the form currently contains using the MODIFY field task. To execute the MODIFY task, enter "M" in the "TASK" field, use the "Field" or "Type" and "Direct" fields to identify the field to be modified and press the (ENTER) key. You can also press the (NEXT) or (PREV) key after entering "M" in the "TASK" field to identify the field to be modified. The field will be displayed in the field

information template and the message "Make you modifications now" is displayed in the message line. At this time you make any changes to the field information and press the `<ENTER>` key when you have the field modified as you want. The message "Field successfully modified" is displayed in the message line. If you decide that you do not want to modify this field after all, press the `<QUIT>` key to cancel the task and restore values if you have already made changes. If you have not made any changes in the field information, you can press the `<ENTER>` or the `<QUIT>` key to cancel the task.

You can review any of the fields that the form currently contains using the SELECT field task. To execute the SELECT field task, enter "S" in the "TASK" field, use the "Field" or "Type" and "Direct" fields to identify the field to be reviewed and press the `<ENTER>` key. You can also press the `<NEXT>` or `<PREV>` key after entering "S" in the "TASK" field to identify the field to be reviewed. The field will be displayed in the field information template but you cannot change any of the information. You can continue reviewing fields on the form as described in section 5.5.5.1. If you find that a field needs modifying or that you want to use a field to help you define a new one, you can then execute the MODIFY or INSERT field task.

5.5.7.2 Using Form Mode to Modify a Form

When you execute the MF edit task with the "F" edit mode, the next screen displayed is:

FORM EDIT MODE		For FDL File OLDFRMS	
TASK		FORM NEWFORM	
Type		Size	0 by 0
Get FDL		Background	BLACK
	Form	Prompt	
		Row	Column
-- Field Characteristics Table --			
Msg: Enter task and type of fields to be acted on			
application			

Figure 5-40 Form Mode Screen

If you want to modify information in the FORM area of the screen, make the desired changes in these fields, enter "F" in the "TASK" field, and press the <ENTER> key.

You can define new fields for the current form using the INSERT or COPY field task as explained in section 5.5.6.2.

You can modify existing fields on the form using the MODIFY field task. To execute the MODIFY task, enter "M" in the "TASK" field, "I", "F", "W", or "A" in the "Type" field to identify the type of fields you want to modify, and press the <ENTER> key. The FIELD CHARACTERISTICS TABLE will be displayed with the identified fields and the message "Make your modifications" now is displayed in the message line. At this time you can modify any of the entries in the table and press the <ENTER> key to

enter the changes. If you decide not to make any modifications or you want to restore the values you changed, press the <QUIT> key instead of the <ENTER> key to cancel the MODIFY task.

You can review any of the fields that the form currently contains using the SELECT field task. To execute the SELECT task, enter "S" in the "TASK" field, "I", "F", "W", or "A" in the "Type" field to identify the type of fields you want to review, and press the <ENTER> key. The fields will be displayed in the FIELD CHARACTERISTICS TABLE but you cannot change any of the information. If you decide that the field information needs modifying or that you want to use a field to help you define a new one, you can then execute the MODIFY or INSERT field task.

You can delete fields from the form using the DROP field task. To execute the DROP task, enter "D" in the "TASK" field, "I", "F", "W", or "A" in the "Type" field to identify the type of fields you want to delete, and press the <ENTER> key. The message "Mark fields to be dropped with an "*"!" is displayed in the message line and the FIELD CHARACTERISTICS TABLE is displayed with a "DELETE" column. For example:

FORM EDIT MODE
For FDL File OLDFRMS

TASK

Type

Get FDL

FORM NEWFORM

Size by

Background

Prompt

Row Column

-- Field Characteristics Table --

Field Name	T	Row	Col	Size	Display	Disp	Act	D	Sp	Prompt	Pos
FIELD1	I	3	15	15	1	INPUT				Item 1:	LT
FIELD2	I	4	15	15	1	INPUT				Item 2:	LT
FIELD3	W	7	15	15	5	BLACK				Window 1	TL

Msg: Mark fields to be dropped with an '*'

application

Figure 5-41 Using Form Mode to Delete a Field

Mark the field(s) to be dropped by entering an "*" in the appropriate delete column position(s) and press the <ENTER> key. The number preceding the message line tells you how many fields were dropped and a message verifying the last field deleted is displayed in the message line. You can use the <MESSAGE QUEUE> key to verify that any other marked fields were successfully deleted.

5.5.7.3 Using Layout Mode to Modify a Form

When you execute the MF edit task with the "L" edit mode, the form is displayed using the special layout symbols described in section 5.5.6.3. For example:

```
EXAMPLE FORM2

)Item 1:)[-----]
)Item 2:)[-----]

"Window 1"
[-----]
[-----]
[-----]
[-----]

Page: 0 application
```

Figure 5-42 Modifying a Form in Layout Mode

This screen allows you to modify the form by graphically positioning fields and prompts as described in section 5.5.6.3.

5.5.8 DF (Drop a Form)

Use this task to drop (delete) a form definition from the FDL source file displayed in the "Form Source" field. You must enter the name of the form that you want to delete.

The command entry format is: DF formname<ENTER>

To execute this task using menu selection, enter any nonblank character in the "Pic" field for Drop a Form, the name of the form as the "Form Name", and press the <ENTER> key. For example:

FORMS DRIVEN FORM EDITOR - VERSION 2.0 JUNE 1, 1985

Command Entry

EDIT TASKS	Command	Pic	Form Name	Edit Mode	Help
List	Forms(LF)				
Write	Forms(WF)				
Compile	Forms(CF)				
Select a Form (SF)					
Insert a Form (IF)					
Modify a Form (MF)					
Drop a Form (DF)					
Exit Write (EW)					
Exit Compile (EC)					
Exit No save (EN)					

Work Task: Modify FDL Source
Form Source: OLDFRMS

Msg: Enter command on command entry line or use menu selection application

Figure 5-43 Choosing DF Using Menu Selection

When this task is executed, the message "Drop was successful" is displayed in the message line and you remain at the Edit Task level. Note that if the FDL source file has been compiled, the FD file for the form is not deleted. You delete this file using the DC Work Task.

5.5.9 EW (Exit Write)

Use this task to execute the WF (Write Form) Edit Task and exit the FDFE. This task saves the current FDL source file without compiling it and generating the FD files.

The Command Entry format is: EW<ENTER>

To execute this task using menu selection, enter any nonblank character in the "Pic" field for Exit Write and press the <ENTER> key.

5.5.10 EC (Exit Compile)

Use this task to compile the form definition(s) you are currently working on and then exit the FDFE.

The Command Entry format is: EC<ENTER>

To execute this task using menu selection, enter any nonblank character in the "Pic" field for Exit Compile and press the <ENTER> key.

If there are syntax errors, the message "Form compile errors, review errors and correct" is displayed in the message line. The errors are stored in the message queue which can be displayed using the <MESSAGE QUEUE> key. When there are no syntax errors, the form definition(s) will be saved in the FDL source file shown in the "Form Source" field, the compiled form(s) will be saved as FD file(s), and you will exit the FDFE.

5.5.11 EN (Exit No save)

Use this task to terminate your current FDFE edit session without saving or compiling anything.

The Command Entry format is: EN<ENTER>

To execute this task using menu selection, enter any nonblank character in the "Pic" field for Exit No save and press the <ENTER> key.

5.6 Limited Edit Task Screen

FORMS DRIVEN FORM EDITOR - VERSION 2.0 JUNE 1, 1985

Command Entry

EDIT TASKS	Command	Fic	Form Name	Edit Modes	Help
List	Forms(LF)				
Select a Form	(SF)				
Exit No save	(EN)				

Work Task: Select FDL Source for - READING ONLY
Form Source: OLDFRMS

Msg: Enter command on command entry line or use menu selection application

Figure 5-44 Limited Edit Task Screen

This screen is the next screen displayed when you execute the SS work task. It is a list of the READ ONLY FDFE functions available for form definitions contained in the specified FDL source file. How to execute these tasks is explained in section 5.5.

November 1985

SECTION 6

FDL COMPILER - FLAN

The compiler for the Form Definition Language is FLAN. FLAN must be used to convert form definitions into a format understood by the Form Processor. FLAN reads an FDL source file and creates a separate compiled form for each form definition contained in the source file. A compiled form is referred to as an FD file.

6.1 Executing FLAN

FLAN is available as an application in the IISS environment or as a batch program on the host system.

6.1.1 FLAN in the IISS Environment

To execute FLAN in the IISS environment, enter SDFLANZZZZ as the FUNCTION on the IISS function screen. When you have successfully accessed FLAN, the following display appears on your terminal screen:

[illegible]

Enter the name of the FDL source file you want to compile.
".FDL" is the default extension. The compiled forms (.fd files)
are written to the logical IISSULIB.

6.1.2 FLAN as a Batch Program

FLAN is also available as a batch program outside the IISS and UIS environment. When you use this version of FLAN, you are prompted for the FDL source file name. Contact your system manager for the name of the executable.

6.2 FLAN Error Messages

Error messages in FLAN consist of the line number of the error, the level of the error, and an error message which briefly describes the error. There are three levels of errors:

- Warnings: The error does not prevent the creation of a .FD file although the form may not display as expected.
- Errors: The error is sufficiently serious to prevent the creation of a .FD file but syntax checking will continue from this point.
- Fatais: The error is very serious and prevents the creation of a .FD file and further compilation.

6.2.1 Warning Messages

1. form <form name> too wide for standard screen.
2. form <form name> too long for standard screen.
3. help message too long, truncated.
4. string too long.

6.2.2 Error Messages

1. <number of> errors detected.
2. unable to open file <filename> for form <form name>.
3. value too big for field.
4. size not specified or invalid.
5. no display attribute specified.
6. field <field name> referenced in <field type>
 <field name> [prompt] not defined.

7. circular reference in location of <field type>
 <field name> [prompt].
8. overlap between <field type> <field name> [prompt] and
 <field type> <field name> [prompt].
9. <field type> <field name> [prompt] off top of screen.
10. <field type> <field name> [prompt] off left of screen.
11. unknown background attribute: <attribute name>.
12. must specify relative field name.
13. duplicate field name: <field name>.
14. duplicate display attribute specified.
15. domain only legal for items.
16. duplicate justification specified.
17. duplicate case specified.
18. duplicate minimum specified.
19. duplicate maximum specified.
20. help only legal for items.
21. duplicate help specified.
22. duplicate value specified.
23. duplicate size specified.
24. unterminated string.
25. value only legal for items.
26. unknown function <function name>.
27. invalid argument for INDEX.
28. form <form name> too narrow: fields extend to column
 <integer>.

UM 620144400B
1 November 1985

29. form (form name) too short: fields extend to row
(integer).

30. yacc stack overflow.

31. syntax error.

6.2.3 Fatal Messages

1. out of memory.

2. unable to open input file (file name).

3. unterminated comment.

4. internal error: corrupt poslst.

SECTION 7

FORM TOOLS

7.1 Generating Include Members

After compiling a form definition, you can use MAKINC to generate an include file containing the data structure which corresponds to the form. This data structure can then be used in application programs which get data from, or put data to, the form using the FP routines GDATA and PDATA. The following table describes the series of prompts and your responses.

System Prompt	Your Response
Your system prompt	Log on to the system
Your system prompt	Invoke MAKINC
AVAILABLE LANGUAGES:	
0 - C	
1 - COBOL	
2 - PL/I	
ENTER LANGUAGE:	1
	For this example we will use COBOL
FORM NAME:	MM
	You can enter the name of any form which has been compiled. For this example we will use the Message Management form whose definition is shown in Section 8.
FORM NAME:	⋄END OF FILE⋄
	This sequence is usually CONTROL/Z. You can make include files for more than one form, so the prompt for form name is repeated. When you are finished making include files, enter the ⋄END OF FILE⋄ sequence for your system.

UM 620144400B
1 November 1985

Your include files are created on your current directory with the same name as the form and a type (or extension) of INC on a VAX host. If you want to view the files, you can use any available editor or display command. For this example, we use the type command as follows:

```
$ TYPE MM.INC
  01  MM-FORM-DATA.
      02  MBASE PIC X(5).
      02  MSGLIN-FORM-DATA OCCURS 10 TIMES.
          03  NUMBER PIC X(5).
          03  MSGNAM PIC X(8).
          03  MSGDES PIC X(60).
```

This is your COBOL include file.

NOTE that the logical IISILIB points to the directory containing the source for COBOL include files, and you may want to move any COBOL include files generated by MAKINC to this directory.

SECTION 8

SAMPLE FORM DEFINITION

This is the FDL source for the form used in the Message Management program. Figure 8-0 shows the form that is defined.

```

      ERROR MESSAGE DEFINITION SCREEN
      -----

Message Base Number: _____

NUMBER     NAME        DESCRIPTION
-----
          _____
          _____
          _____
          _____
          _____
          _____
          _____
          _____
          _____
          _____

Msg: 0                                           application

```

Figure 8-0 Sample Form

```
CREATE FORM mm
  PROMPT CENTER AT 1 20
    "ERROR MESSAGE DEFINITION SCREEN"
  PROMPT CENTER AT 2 20
    "-----"
  PROMPT AT 6 2
    "NUMBER"
  PROMPT AT 6 10
    "NAME"
  PROMPT AT 6 20
    "DESCRIPTION"
```

UM 620144400B
1 November 1985

ITEM mbase
DISPLAY AS input
HELP "Enter the error message's base number"
PROMPT AT 4 2
"Message Base Number:"
AT 4 25
SIZE 5

FORM msglin (10 V WITH 0 SPACES)
DISPLAY AS output
AT 7 2
SIZE 78

CREATE FORM msglin

ITEM number
DISPLAY AS output
AT 1 2
SIZE 5

ITEM msgnam
DISPLAY AS input
HELP "This value is the name of the error"
AT 1 9
SIZE 8

ITEM msgdes
DISPLAY AS input
HELP "Description of the error"
AT 1 19
SIZE 60

END

8-87

DTIC