RT–11/FEP and RT–11/FRP Installation and User's Guide

AA–M079B–TC

August 1982

This manual is a guide to the installation, documentation, and use of the FEP/FRP software as it is used with the RT–11 operating system, and the FORTRAN IV programming language.

This is a revision.

OPERATING SYSTEM:	RT–11, V4.0
SOFTWARE:	RT–11/FEP, V2.1 RT–11/FRP, V1.0 FORTRAN IV, V2.5

Software and manuals should be ordered by title and order number. In the United States, send orders to the nearest distribution center. Outside the United States, orders should be directed to the nearest DIGITAL Field Sales Office or representative.

Northeast/Mid–Atlantic Region

Digital Equipment Corporation
PO Box CS2008
Nashua, New Hampshire 03061
Telephone:(603)884–6660

Central Region

Digital Equipment Corporation
Accessories and Supplies Center
1050 East Remington Road
Schaumburg, Illinois 60195
Telephone:(312)640–5612

Western Region

Digital Equipment Corporation
Accessories and Supplies Center
632 Caribbean Drive
Sunnyvale, California 94086
Telephone:(408)734–4915

digital equipment corporation • marlboro, massachusetts

First Printing, October 1981
Revised, August 1982

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

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

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

The following are trademarks of Digital Equipment Corporation:

DEC	DECnet	IAS
DECUS	DECsystem–10	MASSBUS
DECSYSTEM–20	PDT	PDP
DECwriter	RSTS	UNIBUS
DIBOL	RSX	VAX
EduSystem	VMS	VT
digital	RT	

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

Contents

Chapter 3 Installing FORTRAN IV

Chapter 4 Installing the FEP/FRP Laboratory Software

Figures

Tables

Index

Preface

MANUAL OBJECTIVES

This manual:

- introduces RT–11/FEP and RT–11/FRP — the FORTRAN Enhancement Package and the FORTRAN Real–Time Package that include the RT–11 operating system, the FORTRAN IV programming language, and the FEP/FRP Laboratory Software
- guides you through the procedures for installing RT–11, FORTRAN IV, and the FEP/FRP Laboratory Software
- describes ways to manage disk storage efficiently
- introduces program-development techniques

INTENDED READER

The reader of this manual should be a FORTRAN programmer experienced in scientific applications. The reader need not be familiar with the RT–11 operating system or the FEP/FRP Laboratory Software.

MANUAL STRUCTURE

Part I of this manual, the introduction, contains Chapter 1, an overview of the hardware, software, and documentation.

Part II contains three chapters on installing software. Chapter 2 is a guide to installing the RT–11 operating system. Chapter 3 is a guide to installing the FORTRAN IV programming language. Chapter 4 gives instructions for installing the FEP/FRP Laboratory Software subprograms.

Part III contains four chapters on using the software. Chapter 5 presents program development techniques. Chapter 6 describes restrictions to be observed in the use of the REAL–11/MNC software. Chapter 7 contains information on using the RGL/11 software. Chapter 8 provides information on using the PLOT55 video graphics software.

RELATED DOCUMENTS

For information on related documents in the RT–11 documentation library, consult the *RT–11 Documentation Directory* AA–5285E–TC.

For information on the FORTRAN IV programming language, consult:

PDP–11 FORTRAN Language Reference Manual
AA–1855D–TC

RT–11,RSTS/E FORTRAN IV User's Guide
AA–5749B–TC

RT–11 FORTRAN IV Installation Guide
AA–5240D–TC

For information concerning FEP/FRP software, consult:

REAL–11/MNC FORTRAN Programmer's Reference Manual
AA–D631B–TC

Scientific Subroutines Programmer's Reference Manual
AA–1101C–TC

Instrument Bus Subroutines Programmer's Reference Manual
AA–5613B–TC

Laboratory Subroutines Programmer's Reference Manual
AA–C984B–TC

FORTRAN Debugging Technique Reference Manual
AA–H069B–TC

RGL/11 Programmer's Reference Manual
AA–K783A–TC

For information concerning hardware, consult:

DECLAB–11/MNC User's Guide
EK–MNC11–UG

MINC Thermocouple Preamplifier User's Guide
EK–MNCTP–UG

VT125 User's Guide
EK–VT125–UG–001

VT105 Graphic Terminal User's Manual
EK–VT105–UG

DOCUMENTATION CONVENTIONS

This manual conforms to the following documentation conventions:

- In examples of terminal-programmer dialog, text printed in red indicates your input to the system. Text printed in black indicates responses supplied by the system.

- RET indicates the RETURN key on your terminal. Press the RETURN key at the end of any line you type unless instructed otherwise.
- NO SCROLL indicates the NO SCROLL key on your video terminal.
- CTRL/x represents the CTRL key and another key (represented here by x) on your terminal. To perform some functions, you must press the CTRL key and the other key at the same time.
- SHFT/x represents the shift key and another key.
- In descriptions of command syntax, capital letters represent the command name, which you must type. Lower case letters represent a variable, for which you must supply a value.

  Square brackets [ ] enclose optional choices; you can include an item in brackets, or you can omit it, as you choose.

  The ellipsis symbol (...) represents repetition. You can repeat the item that precedes the ellipsis.
- RL01/RL02 indicates either an RL01 or RL02 disk.
- Unless otherwise noted, all programs are on drive DK:.
- References to the FB environment also apply to the XM environment, unless otherwise stated.

Part I
Introduction

Part I is an overview of the hardware, software, and documentation.

Chapter 1
Hardware and Software Overview

This chapter introduces the hardware, software and software documentation. The hardware is from the MINC family (MINC–11, MINC–23, or MINC/DECLAB23). The software consists of the RT–11 operating system, the FORTRAN IV programming language, and the components of the FEP/FRP Laboratory Software. The software documentation includes sets of manuals on RT–11, FORTRAN IV, and the FEP/FRP Laboratory Software.

1.1 Hardware

Your hardware includes the following hardware components:

- a PDP 11/03 processor (with the FIS floating-point hardware) or a PDP 11/23 processor (with the FPU floating-point hardware)
- at least 56k (k = 1,024) bytes of random-access memory
- at least one video or hard-copy terminal (See Section 5.1 for information about terminals.)
- a chassis that can contain up to eight MNC modules

Your hardware also includes either two RL01/RL02 disk drives or two RX02 diskette drives. RL01 and RL02 disks have a much greater storage capacity than RX02 diskettes. An RL01 disk can store 5.2 million bytes of data, and an RL02 disk can store over 10 million bytes. However, an RX02 can store only 512,512 bytes.

This difference in capacity determines how you can use your disk storage. One RL01 or RL02 disk is large enough to store all of your software. However, two or more RX02 diskettes together are required to store the same software. Many sections of this manual include one discussion related to RL01/RL02 disks and another discussion related to RX02 diskettes.

Software installation, generation, and program development procedures are performed one way on RL01/RL02 disks and another way on RX02 diskettes.

1.2 Software

Your FEP/FRP software includes the RT–11 operating system, the FORTRAN IV programming language, and the following groups of FORTRAN–callable subprograms, which compose the FEP/FRP Laboratory Software:

- REAL–11/MNC, which provides real-time control of all MNC–series modules and the serial–ASCII transfer controller
- IBS, the Instrument Bus Subroutines, which control IEEE bus instruments
- SSP, the Scientific Subroutine Package, which performs mathematical data manipulations such as matrix inversions
- LSP, the Laboratory Subroutine Package, which performs signal data processing such as Fast Fourier transforms
- RGL/11, the ReGIS Graphics Library for FEP Users, which draws figures and plots data on the VT125 terminal (not available with the FRP package).
- FDT, the FORTRAN Debugging Technique, which is an on-line debugging tool

1.3 Hardware Requirements

You must have particular hardware to execute some FEP/FRP components.

- To execute REAL–11/MNC subprograms, you must have the MNC–series module(s) needed by the subprogram(s) you will use.
- To execute IBS subprograms, you must have an IBV11–A IEEE bus and at least one instrument that is compatible with the IBV11–A IEEE bus.
- To execute the RGL/11 subprograms, you must have a VT125 video terminal.
- To execute the remaining FEP/FRP components — SSP, LSP, and FDT — you need only standard MINC–family hardware.

1.4 Distribution Media

If you have two RL01 or RL02 disk drives, your software has been distributed to you on three disk packs. If you have only RX02 diskette drives, your software has been distributed to you on 19 diskettes; some of these diskettes are in RX01 (single density) format while others are in RX02 (double density) format. See Table 1–1, Software Distribution Media, for the types of disks on which each software component is distributed.

Table 1–1: Software Distribution Media (RX01/RX02 or RL01/RL02)

The numbers indicate disks required to store each software component

Type of Disk	RT–11	FORTRAN IV	REAL–11	IBS	SSP	LSP	RGL/11	FDT
RX01		3		1	2	1		1
RX02	5		2				4	
RL01/RL02	1	1	1					

1.5 Installing Software

Install your software in the following sequence:

1. RT–11 installation (and system generation if necessary)
2. FORTRAN IV compiler and OTS (Object Time System) installation and generation
3. FEP/FRP component installation

Part II of this manual guides you through both the installation procedures and the documentation for each software component.

1.6 Standard Software

Some software components are shipped in standard form and are ready to use when you receive them. However, you can also configure much of the software to fit your particular needs.

The RT–11 distribution media contain standard monitors as well as programs for creating specialized monitors. Read Chapter 2 of this manual to determine whether or not you should customize your system and/or execute the system-generation (SYSGEN) procedure.

The FORTRAN IV distribution disk contains a prebuilt FORTRAN IV compiler and object-time system (OTS) as well as programs for generating a customized compiler and OTS. You must execute the FORTRAN generation procedure to utilize your floating-point hardware. (The prebuilt FORTRAN performs multiplication, division, and floating-point arithmetic with software routines. Therefore, the prebuilt FORTRAN IV performs some arithmetic operations much more slowly than FORTRAN IV, which utilizes floating-point hardware.)

The FEP/FRP Laboratory Software distribution disks contain standard versions of all FEP/FRP Laboratory Software. You can also customize all FEP/FRP Laboratory Software except FDT. For information on installing and customizing FEP/FRP, see Chapter 4 of this manual.

1.7 Software Documentation

For a list of manuals that describe the software, see Related Documents in the Preface.

Part II
Installing Software

The software consists of RT–11, FORTRAN IV, and FEP/FRP and must be installed in that order. (Figure 2–1, Installing Software, provides an overview of the procedure for installing software.)

Part II contains detailed instructions for installing RT–11 and the FEP/FRP Laboratory Software but is only a guide to FORTRAN IV installation. This part of the manual also describes the order in which you must read RT–11 and FORTRAN IV installation manuals and contains information that supplements those manuals.

Chapter 2
Installing RT–11

For FEP/FRP users, the major steps in the RT–11 installation procedure are:

1. Preparing for installation
2. Using the FEP or FRP Installation Disk to create a development system disk
3. Creating other system disks (optional)
4. Customizing the system (optional)
5. Testing the development system disk
6. Performing the system-generation process (optional)

Figure 2–1 is a diagram of the installation overview, and Figure 2–2 is a flowchart of the steps in the procedure for installing RT–11. Note that the procedure includes three optional steps, which are creating other system disks, customizing the system, and performing the system-generation process. This chapter explains the procedure for installing RT–11 and helps you decide which optional steps to execute. (Note that while this chapter fully describes some steps in the procedure — those not found in other manuals — the chapter also directs you to RT–11 manuals for more complete descriptions of other steps.)

The installation procedures discussed in this chapter are the same for FEP and FRP. Therefore, only FEP examples are presented.

Figure 2–1: Installing Software

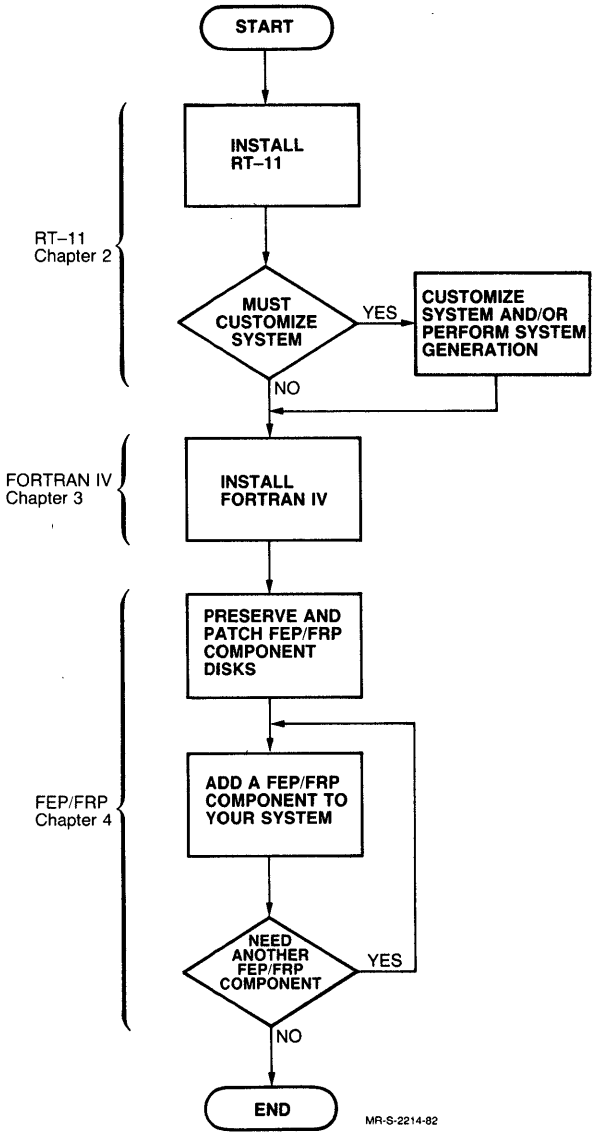
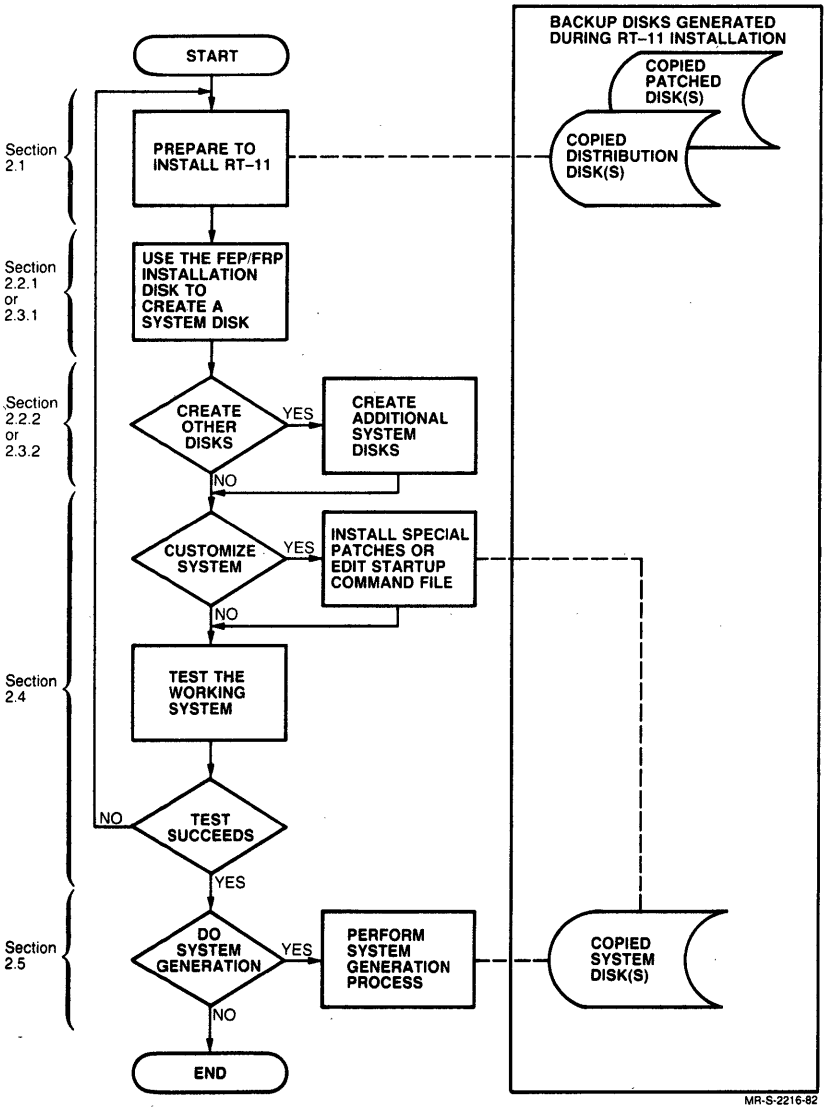


Figure 2–2: Installing RT–11



2.1 Preparing to Install RT–11

Prior to installing RT–11:

- Gather the following:

  RT–11 Installation and System Generation Guide

  RT–11 Autopatch Kit

  RT–11 distribution media

  Three blank RL01/RL02 disks or eleven blank RX02 diskettes

- If you have two RL01/RL02 disk drives, power on the MINC computer system by following the instructions in Section 2.2, Operating Procedures, in the *DECLAB–11/MNC User's Guide* and mount the FEP distribution disk in drive 0.

  If you have only RX02 diskette drives, power on the computer system by executing the following procedures:

  1. Mount the FEP Installation Diskette in drive 0.
  2. Set the terminal's ON/OFF switch to the ON position.
  3. Power on the processor by pressing the red switch above drive 1, and be sure the light in the switch is lit.

  Regardless of which type of disks you have, once you have powered on the processor, the system automatically bootstraps RT–11 and displays on your terminal:

```
RT-11/FEP V2.1
;
;
;  Welcome to RT-11/FEP version 2.1.
;
;
```

  (followed by startup file commands and additional messages. If you have two RL01/RL02 disk drives, see Section 2.2.1 for a list of these additional messages; if you have only RX02 diskette drives, see Section 2.3.1.)

- If you have a video terminal, set auto XOFF/XON in SET–UP mode before beginning the RT–11 installation procedure. Do so by executing the following instructions:

  1. Press the SET–UP key to put the terminal in SET–UP A mode.
  2. Press the 5 key on the main keyboard to put the terminal in SET–UP B mode.
  3. The terminal displays four groups of binary values. Look at the fourth entry in the second group of values. If that value is one, then AUTO XOFF/XON is correctly set: press the SET–UP key again to return the terminal to normal operation, and ignore steps 4, 5 and 6 below. If that value is zero, then AUTO XOFF/XON is not set: perform steps 4, 5, and 6.

4. Press the right-arrow (→) key, which is located at the upper right corner of the main keyboard, until the cursor is positioned over the fourth entry in the second group of displayed values.

5. When the cursor is correctly positioned, press the 6 key on the main keyboard to change the value from zero to one.

6. Save the current SET–UP features by typing SHFT/S. The terminal clears the screen, displays WAIT, and returns to SET–UP A mode.

7. Press the SET–UP key again to return the terminal to normal operation.

   (See the *VT125 User's Guide* or the *VT105 Graphic Terminal User's Manual* for further information.)

- Sometimes you must halt and bootstrap the processor. For example, you must do so to recover from a system failure or to bootstrap from a new disk on your system drive. Any time you must halt and bootstrap the processor, type the commands in the following dialog:

  BREAK (Press the BREAK key, which is located near the upper right corner of your keyboard. This halts the processor.)

  ```
  nnnnnn
  @773000G
  ```
  (This bootstraps the processor from drive 0. Also, any time the terminal displays @ and the processor has halted, you can bootstrap the processor by typing this number.)

  ```
  RT-11xx V04.00
  ```
  (followed by any startup file commands)

  ```
  .DATE dd-mmm-yy RET
  .TIME hh:mm:ss RET
  ```

  where `nnnnnn` is the address plus 2 at which the processor stopped

  `xx` is the name of the monitor in operation

  `dd-mmm-yy` is the day, month, and year in the form 10–JAN–80

  `hh:mm:ss` is the current time in hours, minutes, and seconds

  NOTE

  If the preceding procedure does not bootstrap the processor, see the *DECLAB–11/MNC User's Guide* for additional information.

- If you are unfamiliar with RT–11, read the *Introduction to RT–11*, but do not attempt to perform the exercises in that book. (You can perform the exercises after you have installed RT–11 and FORTRAN IV.)

- Begin the RT–11 installation procedure. If you have two RL01/RL02 disks, follow instructions in Section 2.2; if you have only RX02 diskette drives, follow instructions in Section 2.3.

NOTE

Never type SET TT NOPAGE. Doing so could cause unpredictable results.

2.2 Installing RT–11 with Two RL01/RL02 Disk Drives

This section contains information on installing RT–11 on one or more RL01/RL02 disks. Section 2.2.1 presents procedures for using the FEP distribution disk to create a development system disk. Section 2.2.2 discusses procedures for creating additional system disks.

Sometimes these procedures instruct you to write-protect or write-enable a disk drive. To write-protect a drive, press the drive's write-protect switch, and be sure the light in the switch is lit. To write-enable a disk, press the drive's write-protect switch, and be sure the light in the switch is not lit.

2.2.1 Installing RT–11 by Using the FEP Distribution Disk

This section explains procedures for using the FEP distribution disk to:

- copy itself
- copy any other disk
- install an SJ or FB monitor

Note that Section 2.5.1 explains another use for the FEP distribution disk; that is, how to use that disk for installing the XM monitor.

Throughout the dialogs listed in this section and in Section 2.5.1, note that:

- The terminal displays an asterisk (*) before each line that contains a question.
- The terminal displays a question mark (?) before each line that contains a completion message or error message.
- The terminal displays a semicolon (;) before each line that does not contain a question, completion message, or error message.
- Brackets ([]) enclose information about possible answers to a question. For example, if a question is followed by [Y/N], you can answer Y(RET) or N(RET); if a question is followed by [S], your answer can be any string (consisting of one or more characters).
- Whenever a line in the dialog instructs you to insert a disk into a drive, wait until the white light on the drive is lit before you insert the disk.

To install the SJ or FB monitor:

1. Since you have powered on the MINC computer system according to the procedure listed in Section 2.1, the following dialog is displayed on your terminal. Copy the FEP distribution disk by typing the commands in this dialog:

```
RT-11/FEP V2.1
;
;
; Welcome to RT-11/FEP version 2.1.
;
;
; The disk in DL0: is a distribution disk.
; The first thing you must do is to copy the files on that disk
; to an unused disk in DL1:. The following procedure initializes
; an unused disk and copies the FEP distribution disk.
;
; Insert an unused disk in DL1:
* Are you ready to continue? [Y/N]: Y(RET)
;
; Initializing DL1:
;
; Checking DL1: for bad blocks
? DUP-I-No bad blocks detected DL1:
;
; Copying files from DL0: to DL1:
;
; The disk in DL1: now contains a bootable kit
; which will help you to install FEP/RT.
;
; Remove the disk from DL0: and store it in a safe place.
; Remove the disk from DL1: and boot it in DL0:
```

2. Make a copy of the RT–11 distribution disk by typing the commands in the following dialog:

```
RT-11/FEP V2.1
;
;
; Welcome to RT-11/FEP version 2.1.
;
; This disk provides automatic procedures for
; software installation.
;
; You must use the RT-11/FEP and RT-11/FRP Installation and User's
Guide
; in conjunction with these procedures.

Please enter
Today's date (dd-mmm-yy):
```

   Enter a date in which `dd` represents the day (a decimal number from 1 to 31), `mmm` represents the first three characters of the name of the month, and `yy` represents the year (a decimal number from 73 to 99): `dd-mmm-yy(RET)`

```
;
; Select the procedure you want to perform by typing one
; of the following codes:
;
; RT      Build an RT-11 development system disk
; SYSGEN  Sysgen an XM monitor
; COPY    Copy (preserve) any disk
;
* Select a procedure [S]: COPY RET
;
; This procedure copies a disk and begins by
; initializing an output disk.
;
; Insert an unused disk in DL1:
* Are you ready to continue? [Y/N]: Y RET
;
* Any data on DL1: will be lost. Continue? [Y/N]: Y RET
;
;
; Initializing DL1:
;
; Checking DL1: for bad blocks
? DUP-I-No bad blocks detected DL1:
;
; For the COPY procedure:
;
; The output volume is already in DL1:.
; The input volume is the disk you want to copy.
; The system volume is the FEP distribution disk.
;
Mount output volume in DL1:; Continue? Y RET
Mount input volume in DL0:; Continue? Y RET
Mount system volume in DL0:; Continue? Y RET
;
; Finished copying.
;
* Do you want to copy bootstrap information? [Y/N]:
```

If the disk in drive 1 is the copied RT–11 Distribution Disk or the RT–11 Autopatch disk, type Y RET, otherwise, type N RET. If you type Y RET, the following question also appears.

```
;
* Enter filename of monitor [S]:
```

If the disk in drive 1 is the copied RT–11 Distribution Disk, type RT11BL.SYS RET. If the disk in drive 1 is the RT–11 Autopatch disk, type AUTOP.SYS RET.

```
;
; Bootstrap copied.
;
; Select the procedure you want to perform by typing the
; appropriate code.
;
; RT      Build an RT-11 development system disk
; SYSGEN  Sysgen an XM monitor
; COPY    Copy (preserve) any disk
;
* Select a procedure [S]:
```

NOTE

You can use the preceding COPY procedure to copy any disk (including the RT–11, FORTRAN IV, and FEP/FRP laboratory-software distribution disks as well as the RT–11 Autopatch Kit).

3. Use the following procedure to patch the copies of the RT–11 distribution disk:

   - Bootstrap the copy of the RT–11 distribution diskette in drive 0. Remove protection from files on the copied disk by typing the following command. (Use the NOLOG option to prevent a list of renamed files from appearing on the terminal.)

     ```
     .RENAME/SYSTEM/NOPROTECTION/NOLOG  DL0:*.*  DL0:*.*RET
     ```

   - Install the patches for RT–11 by following instructions for doing so in the *RT–11 Autopatch User's Guide.*

   - Bootstrap the copy of the RT–11 distribution disk, which you have now patched, in drive 0. Protect files on this disk by typing the following command:

     ```
     .RENAME/SYSTEM/PROTECTION/NOLOG  DL0:*.*  DL0:*.*RET
     ```

   - The chapter titled Procedures Following an Autopatch Session in the *RT–11 Autopatch User's Guide* instructs you to make copies of the patched disk. Use the FEP distribution disk copy procedure, as described in step 2 of this section to make the copy.

4. Use the following procedure to create a development system disk:

   - Bootstrap the FEP distribution disk in drive 0.

   - Type the commands in the following dialog:

     ```
     RT-11/FEP V2.1
     ;
     ;
     ; Welcome to RT-11/FEP version 2.1.
     ;
     ; This disk provides automatic procedures for
     ; software installation.
     ;
     ; You must use the FEP/RT and RT-11/FRP Installation and User's
     Guide ; in conjunction with these procedures.

     Please enter
     Today's date (dd-mmm-yy):
     ```

     Enter a date in which `dd` represents the day (a decimal number from 1 to 31), `mmm` represents the first three characters of the name of the month, and `yy` represents the year (a decimal number from 73 to 99): `dd-mmm-yy`RET

     ```
     ;
     ; Select the procedure you want to perform by typing one
     ; of the following codes:
     ;
     ; RT      Build an RT-11 development system disk
     ; SYSGEN  Sysgen an XM monitor
     ; COPY    Copy (preserve) any disk
     ;
     * Select a procedure [S]: RTRET
     ;
     ; RT-11 INSTALLATION
     ;
     ```

```
; This procedure creates an RT-11 system that includes
; the standard monitor contained on the RT-11
; distribution kit. You will need a blank disk that the
; procedure will initialize and onto which it will copy
; the RT-11 system.
;
; Insert an unused disk in DL1:
* Are you ready to continue? [Y/N]: Y(RET)
;
* Any data on DL1: will be lost. Continue? [Y/N]: Y(RET)
;
; Initializing DL1:
;
; Checking DL1: for bad blocks
? DUP-I-No bad blocks detected DL1:
;
; For the next operation:
;
; The output volume is already in DL1:.
; The input volume is the RT-11 distribution disk.
; The system volume is the FEP installation disk.
;
Mount output volume in DL1:; Continue? Y(RET)
Mount input volume in DL0:; Continue? Y(RET)
Mount system volume in DL0:; Continue? Y(RET)
;
* Do you have a serial printer? [Y/N]:
```

Type Y(RET) or N(RET). If you type Y(RET), the following question also appears.

```
;
* Which serial line unit is it connected to (0, 1, or 2) ?
```

Normally, the line printer is connected to serial line unit number 2 (labelled SLU2) at the rear of the MINC chassis. However, the line-printer cable may instead be connected to SLU0 or SLU1. Type 0(RET), 1(RET), or 2(RET).

```
;
* Do you have a parallel line printer? [Y/N]:
```

Type Y(RET) or N(RET).

```
;
* Do you have a video terminal? [Y/N]:
```

Type Y(RET) or N(RET).

```
;
; Now deleting unneeded files on DL1:
;
; Now squeezing DL1:
;
; Your disk now has an installed RT-11 development system
; which is equivalent to that recommended in Table 2-1
; in the RT-11/FEP and RT-11/FRP Installation and
; User's Guide.
;
; To proceed with the installation of FORTRAN IV, and
; the remaining FEP/RT components, remove the disk
; from DL1:, boot it in DL0:, and follow the procedures in
; Chapter 3 and Chapter 4 of the RT-11/FEP and RT-11/FRP
; Installation and User's Guide.
;
```

2.2.2 Using the RT–11 Installation and System Generation Guide

This section contains information on creating one or more system disks that contain files not found in Table 2–1. Skip this section if you do not want to create such additional disks.

To create specialized system disks, follow procedures in the *RT–11 Installation and System Generation Guide*. Begin by reading the following chapters in that manual:

Introduction

Preparing for Installation

Installing a System Distributed on Hard Disk to Run on a Hard Disk

As you read those chapters, keep the following in mind:

- Many commands in the instructions for installing RT–11 contain the expressions XXn:, XX0:, and XX1:. For XX, type `DL`. For n, type `0` or `1` for disk drive 0 or 1. (Such instructions include INITIALIZE, RENAME, SQUEEZE, and COPY.)
- The *RT–11 Installation and System Generation Guide* presents several ways to initialize a hard disk. To initialize an RL01/RL02 disk, type:

  ```
  .INITIALIZE/BADBLOCKS DLn:RET
  ```

  where `n` is `0` or `1`.

Table 2–1: Recommended Contents of an RL01/RL02 Development System Disk

File	Comment
RT11SJ.SYS	If you want to use the SJ monitor.
RT11FB.SYS	If you want to use the FB monitor.
DL.SYS	
LP.SYS or LS.SYS	If you have a line printer.
NL.SYS	
TT.SYS	
BA.SYS	
SWAP.SYS	
CREF.SAV	
DIR.SAV	
DUP.SAV	
KED.SAV	If you have a video terminal.
EDIT.SAV	If you have a hard-copy terminal.

(continued on next page)

Table 2–1: Recommended Contents of an RL01/RL02 Development System Disk (Cont.)

File	Comment
FORMAT.SAV	
HELP.SAV	
LIBR.SAV	
LINK.SAV	
MACRO.SAV	
PATCH.SAV	
PAT.SAV	
PIP.SAV	
RESORC.SAV	
SIPP.SAV	
SLP.SAV	
SRCCOM.SAV	
SYSMAC.SML	
BATCH.SAV	
HELP.MLB	
HELP.EXE	
STARTF.COM	If you have included RT11FB.SYS.
STARTS.COM	If you have included RT11SJ.SYS.
ODT.OBJ	
PLOT55.OBJ	
TEST55.FOR	
SYSLIB.OBJ	
DEMOED.TXT	
DEMOX1.MAC	
DEMOBG.MAC	
DEMOFG.MAC	
DEMOF1.FOR	

- The chapter entitled Installing a System Distributed on Hard Disk to Run on a Hard Disk in the *RT–11 Installation and System Generation Guide* contains discussions of the following procedures; read these discussions and perform the procedures described:

  1. Bootstrapping the distribution disk
  2. Preserving the distribution disk

3. Installing mandatory patches
4. Creating the system from chosen components
5. Installing the bootstrap on the disk

2.3 Installing RT–11 with Two RX02 Diskette Drives

This section contains information on creating RT–11 system diskettes. Section 2.3.1 presents procedures for using the FEP Installation Diskette to create a development system diskette. Section 2.3.2 discusses procedures for creating an additional system diskette, one which contains RT–11 utilities.

As you perform procedures described in this section, keep in mind the following:

- Drive 0 (also called DY0:) is the drive on the left. Drive 1 (DY1:) is the drive on the right.
- The RT–11 distribution diskettes are labelled 1/4, 2/4, 3/4, and 4/4. The diskette labelled 1/4 is also known as diskette 1; the diskette labelled 2/4 is diskette 2...the diskette labelled 4/4 is diskette 4.
- You cannot write-protect an RX02 diskette. (However, individual files may have been protected by software. For information on protecting files by software, see the discussion of the RENAME command in the *RT–11 System User's Guide*.) Therefore, be careful to avoid inadvertently destroying an important diskette during a FORMAT, INITIALIZE, COPY, or SQUEEZE operation.
- During execution of a COPY command, your terminal may display a message such as the following, which indicates that you have used all available space on a diskette:

```
?PIP-F-Device full DY1:filnam.ext
```

where `filnam.ext` is the name of the first file not copied to the output device (normally DY1: is the output device).

If this happens, create free space on your diskette by performing the following procedures.

1. Delete unnecessary files from the diskette by typing:

```
.DELETE  DY1:aaaaaa.ttt,DY1:bbbbbb.ttt,...,DY1:zzzzzz.ttt↵
DY1:aaaaaa.ttt  ? Y↵
DY1:bbbbbb.ttt  ? Y↵
DY1:zzzzzz.ttt  ? Y↵
```

where `aaaaaa.ttt` through `zzzzzz.ttt` are the names of a maximum of six files to be deleted.

NOTE

If you are deleting a system (.SYS) file, use the /SYSTEM option with the DELETE command. If a file you want to delete is protected, use the RENAME command to remove protection before you delete the file. (For further information on the DELETE and RENAME commands see the *RT–11 System User's Guide.*)

2. Collect free space on the diskette into one contiguous area by typing:

```
.SQUEEZE DY1:RET
DY1:/Squeeze; Are you sure? YRET
```

3. After the terminal displays the system dot, again type the COPY command that failed.

2.3.1 Installing RT–11 by Using the FEP Installation Diskette

This section explains procedures for using the FEP Installation Diskette to:

- copy itself
- copy any other diskette
- install an SJ or FB monitor

Note that Section 2.5.2 explains another use for the FEP Installation Diskette; that is, how to use that diskette for installing the XM monitor.

Throughout the dialogs listed in this section and in Section 2.5.2, note that:

- The terminal displays an asterisk (*) before each line that contains a question.
- The terminal displays a question mark (?) before each line that contains a completion message or error message.
- The terminal displays a semicolon (;) before each line that does not contain a question, completion message, or error message.
- Brackets ([]) enclose information about possible answers to a question. For example, if a question is followed by [Y/N], you can answer YRET or NRET. If a question is followed by [S], your answer can be any string (consisting of one or more characters).

To install the SJ or FB monitor:

1. Since you have powered on the MINC computer system according to the procedure listed in Section 2.1, the following dialog is displayed on your terminal. Copy the FEP Installation Diskette by typing the commands in this dialog:

```
RT-11/FEP V2.1
;
;
; Welcome to RT-11/FEP version 2.1.
;
;
; The diskette in DY0: is a distribution diskette.
; The first thing you must do is to copy the files
; on that diskette to an unused diskette in DY1:. The
; following procedure formats and initializes an
; unused diskette and copies the distribution
; diskette.
;
; Insert an unused diskette in DY1:
* Are you ready to continue? [Y/N]: Y(RET)
;
; Formatting DY1:
?. FORMAT-I-Formatting complete
;
; Initializing DY1:
? DUP-I-No bad blocks detected DY1:
;
; Copying files from DY0: to DY1:
;
; The diskette in DY1: now contains a bootable
; installation diskette which will help you to
; install FEP/RT.
;
; Remove the diskette from DY0: and store it in
; a safe place.
;
; Remove the diskette from DY1: and boot it in DY0:.
;
;
```

2. Make copies of the four RT–11 distribution diskettes by typing the commands in the following dialog:

```
RT-11/FEP V2.1
;
;
; Welcome to RT-11/FEP version 2.1.
;
; This installation diskette provides automatic
; procedures for software installation.
;
; You must use the RT-11/FEP and RT-11/FRP Installation and
; User's Guide in conjunction with these procedures.
;
;
Please enter
Today's date dd-mmm-yy:
```

Enter a date in which `dd` represents the day (a decimal number from 1 to 31); `mmm` represents the first three characters of the name of the month; and `yy` represents the year (a decimal number from 73 to 99): `dd-mmm-yy(RET)`

```
;
; Select the procedure you want to perform by typing one
; of the following codes:
;
; RT      Build an example RT-11 development system diskette
; SYSGEN  Sysgen an XM monitor
; COPY    Copy (preserve) any diskette
```

```
;
* Select a procedure [S]: COPY(RET)
;
; This procedure copies a diskette and begins by
; formatting and initializing an output diskette.
;
; Insert an unused diskette in DY1:
* Are you ready to continue? [Y/N]: Y(RET)
;
* Any data on DY1: will be lost. Continue? [Y/N]: Y(RET)
;
; Formatting DY1:
? FORMAT-I-Formatting complete
;
; Initializing DY1:
? DUP-I-No bad blocks detected DY1:
;
; For the copy procedure,
;
; The output volume is already in drive 1.
; The input volume is the diskette you want to copy.
; The system volume is the FEP installation diskette.
;
Mount output volume in DY1:; Continue? Y(RET)
Mount input volume in DY0:; Continue? Y(RET)
Mount system volume in DY0:; Continue? Y(RET)
;
; Finished copying
;
* Do you want to copy the bootstrap? [Y/N]:
```

If the diskette in drive 1 is copied RT–11 distribution diskette number 1 or RT–11 Autopatch diskette number 1, type Y(RET); otherwise, type N(RET). If you type Y(RET), the following question also appears.

```
;
* Enter filename of monitor [S]:
```

If the diskette in drive 1 is copied RT–11 distribution diskette number 1, type RT11BL.SYS(RET). If the diskette in drive 1 is RT–11 Autopatch diskette number 1, type AUTOP.SYS(RET).

```
;
; Bootstrap copied.
;
* Do you want to copy another diskette? [Y/N]: Y(RET)
;
```

Repeat the COPY procedure until you have copied all four RT–11 distribution diskettes. Store the RT–11 distribution and FEP Installation diskettes in a safe place. Use only the copies for developing and executing programs.

NOTE

You can use the COPY procedure found on the FEP Installation Diskette to copy any diskette (including RT–11, FORTRAN IV, and FEP distribution diskettes as well as the RT–11 Autopatch Kit).

3. Patch the copies of the RT–11 distribution diskettes by using the following procedure, which may require as much as one-and-one-half hours to perform:

   - Bootstrap the copy of RT–11 Distribution Diskette 1 in drive 0. Remove protection from files on this diskette by typing the following command. (Use the NOLOG option to prevent a list of renamed files from appearing on the terminal.)

     ```
     .RENAME/SYSTEM/NOPROTECTION/NOLOG  DY0:*.*  DY0:*.*RET
     ```

   - Mount a copy of one of the remaining RT–11 distribution diskettes in drive 1. Remove protection from files on the copied diskette by typing the following command. (Use the NOLOG option to prevent a list of renamed files from appearing on the terminal.)

     ```
     .RENAME/SYSTEM/NOPROTECTION/NOLOG  DY1:*.*  DY1:*.*RET
     ```

     Remove protection from each of the copied RT–11 distribution diskettes.

   - Install the patches for RT–11 by following instructions for doing so in the *RT–11 Autopatch User's Guide.*

   - Bootstrap the copy of RT–11 Distribution Diskette 1, which you have now patched, in drive 0. Protect files on this diskette by typing the following command:

     ```
     .RENAME/SYSTEM/PROTECTION/NOLOG  DY0:*.*  DY0:*.*RET
     ```

   - Mount a patched copy of another RT–11 distribution diskette in drive 1. Protect files on the patched diskette by typing the following command. (The NOLOG option prevents a list of renamed files from appearing on the terminal.)

     ```
     .RENAME/SYSTEM/PROTECTION/NOLOG  DY1:*.*  DY1:*.*RET
     ```

     Protect files on each of the patched diskettes.

   - The chapter titled Procedures Following an Autopatch Session in the *RT–11 Autopatch User's Guide* instructs you to make copies of the patched diskettes. Use the FEP Installation Diskette copy procedure, as described in step 2 of this section, to make the copies.

4. Use the following procedure to create a development system diskette:

   - Bootstrap the FEP Installation Diskette in drive 0.

   - Type the commands in the following dialog:

     ```
     RT-11/FEP V2.1
     ;
     ;
     ; Welcome to RT-11/FEP version 2.1.
     ;
     ; This installation diskette provides automatic
     ; procedures for software installation.
     ```

```
;
; You must use the RT-11/FEP and RT-11/FRP Installation and
; User's Guide in conjunction with these procedures.
;
;

Please enter
Today's date (dd-mmm-yy):
```

Enter a date in which `dd` represents the day (a decimal number from 1 to 31), `mmm` represents the first three characters of the name of the month, and `yy` represents the year (a decimal number from 73 to 99): `dd-mmm-yy`RET

```
;
; Select the procedure you want to perform by typing one
; of the following codes:
;
; RT      Build an RT-11 development system diskette
; SYSGEN  Sysgen an XM monitor
; COPY    Copy (preserve) any diskette
;
* Select a procedure [S]: RT
; RT-11 INSTALLATION
;
; This procedure creates an RT-11 system that
; includes a standard monitor contained on the RT-11
; distribution kit. You will need a blank diskette that the
; procedure will format and initialize and onto which it
; will copy the RT-11 system.
;
; Insert an unused diskette in DY1:
* Are you ready to continue? [Y/N]: Y
;
* Any data on DY1: will be lost. Continue? [Y/N]: Y
;
; Formatting DY1:
? FORMAT-I-Formatting complete
;
; Initializing DY1:
? DUP-I-No bad blocks detected DY1:
;
; For the next operation,
;
; The output volume is already in DY1:.
; The input volume is diskette 1 of the RT-11 distribution kit.
; The system volume is the FEP installation diskette.
;
Mount output volume in DY1:; Continue? Y
Mount input volume in DY0:; Continue? Y
Mount system volume in DY0:; Continue? Y
;
; You can install either the RT-11 SJ or the RT-11 FB monitor
;
* Do you want the RT-11 FB monitor? [Y/N]:
```

Type YRET or NRET.

```
;
* Do you have a serial printer ? [Y/N]:
```

Type YRET or NRET. If you type YRET, the following question also appears.

```
;
* Which serial line unit is it connected to (0, 1, or 2) ?
```

Normally, the line printer is connected to serial line unit number 2 (labelled SLU2) at the rear of the MINC chassis. However, the line-printer cable may instead be connected to SLU0 or SLU1. Type 0(RET), 1(RET), or 2(RET).

```
* Do you have a parallel line printer ? [Y/N]:
```

Type Y(RET) or N(RET).

```
;
* Do you have a video terminal [Y/N]:
```

Type Y(RET) or N(RET).

```
;
; Working ...
; Working ...
; Working ...
; Working ...
;
; For the next operation,
;
; The input volume is diskette 2 of the RT-11 distribution kit.
; The output volume is already in DY1.
; The system volume is the FEP installation diskette.
;
Mount input volume in DY0:; Continue? Y(RET)
Mount output volume in DY1:; Continue? Y(RET)
Mount system volume in DY0:; Continue? Y(RET)
;
; Now squeezing DY1:
;
; Your diskette now has an installed RT-11 development system
; which is equivalent to that recommended in Table 2-2
; in the FEP/RT Installation and User's Guide.
;
; To proceed with the installation of FORTRAN IV, and
; the remaining FEP components, remove the diskette
; from DY1:, boot it in DY0:, and follow the procedures
; in Chapter 3 and Chapter 4 of the RT-11/FEP and RT-11/FRP
; Installation and User's Guide.
```

Table 2–2: Contents of the RX02 Development System Diskette

File	Comment
RT11SJ.SYS	If you have installed the SJ monitor.
RT11FB.SYS	If you have installed the FB monitor.
DY.SYS	
LP.SYS or LS.SYS	If you have a line printer.
TT.SYS	
SWAP.SYS	
CREF.SAV	
DIR.SAV	
DUP.SAV	
KED.SAV or EDIT.SAV	Both are text editors. EDIT is on the disk if you have a hardcopy terminal. KED is on the disk if you have a video terminal.
FORMAT.SAV	
HELP.SAV	
LIBR.SAV	
LINK.SAV	
MACRO.SAV	
PIP.SAV	
RESORC.SAV	
SYSMAC.SML	
STARTS.COM	If you have installed RT11SJ.SYS.
STARTF.COM	If you have installed RT11FB.SYS.
SYSLIB.OBJ	
DEMOED.TXT	
DEMOX1.MAC	
DEMOBG.MAC	
DEMOFG.MAC	
DEMOF1.FOR	

2.3.2 Building an RX02 Utilities Diskette

In an ideal installation, one disk is large enough to contain an entire working system. However, RX02 diskettes are not large enough to do this. Therefore, even though you have now created a development system diskette, you may also want to create an additional system diskette, one that contains infrequently used utilities. To begin creating a utilities diskette, read the following chapters of the *RT–11 Installation and System Generation Guide*:

Introduction

Preparing for Installation

Installing a System Distributed on RX02 to Run on RX02 (if you have only RX02 diskette drives)

As you build the utilities diskette, keep in mind the following:

- Many commands in the instructions for installing RT–11 contain the expression XXn:, XX0:, and XX1:. For XX, type DY. For n, type 0 or 1 for diskette drive 0 or 1. (Such instructions include FORMAT, INITIALIZE, SQUEEZE, COPY, and RENAME.)

  When you build your utilities diskette, copy to an RX02–formatted, initialized diskette at least the files found in Table 2–3. (Those and other files are listed and defined in the table of RT–11 Software Components in the *RT–11 Installation and System Generation Guide.*)

- The chapter entitled Installing a System Distributed on RX02 to Run on RX02 in the *RT–11 Installation and System Generations Guide* contains discussions of the following procedures; read these discussions and perform the procedures described:

  1. Bootstrapping the distribution disk
  2. Creating the system from chosen components
  3. Installing the bootstrap on the disk

Table 2–3: Recommended Contents of an RX02 Utilities Diskette

File	Comment
RT11SJ.SYS	If you want to use the SJ monitor.
RT11FB.SYS	If you want to use the FB monitor.
DY.SYS	
LP.SYS	If you have a parallel line printer, such as an LA180.
LS.SYS	If you have a serial line printer, such as an LA120.
TT.SYS	
SWAP.SYS	
CREF.SAV	
DIR.SAV	
DUP.SAV	
KED.SAV or EDIT.SAV	Both are text editors. Include EDIT if you have a hardcopy terminal. Include KED if you have a video terminal.
FORMAT.SAV	
HELP.SAV	
LIBR.SAV	
LINK.SAV	
MACRO.SAV	
PATCH.SAV	
PAT.SAV	
PIP.SAV	
RESORC.SAV	
SIPP.SAV	
SLP.SAV	
SRCCOM.SAV	
SYSMAC.SML	
BINCOM.SAV	
DUMP.SAV	
STARTS.COM	If you have included RT11SJ.SYS.
STARTF.COM	If you have included RT11FB.SYS.
SYSLIB.OBJ	

2.4 Customizing the System

The *RT–11 Installation and System Generation Guide* lists many optional procedures for customizing the system without performing the system generation procedure. A common procedure of this type is the addition of commands to the startup indirect command files.

An RT–11 monitor looks for a startup indirect command file whenever you bootstrap the system. Each monitor looks for a particular file. The single-job monitor looks for STARTS.COM; the foreground/background monitor looks for STARTF.COM; the extended-memory monitor looks for STARTX.COM. If the monitor finds a startup command file, it executes the commands in that file. If you want to add commands to a startup command file, add them to the file that your monitor uses. (For information on indirect command files, see the *RT–11 System User's Guide.*)

If you have no listing device and want the terminal to be the default listing device, add the following command to the startup indirect command file:

```
ASSIGN TT LP
```

The *RT–11 Installation and System Generation Guide* contains discussions of the following procedures:

1. Customizing the system (optional)
2. Compressing the disk
3. Preserving the working system
4. Testing the working system

If you have RL01/RL02 disks and want to customize your system, read those discussions in the chapter entitled Installing a System Distributed on Hard Disk to Run on a Hard Disk. If you have RX02 diskettes, read the discussions in the chapter entitled Installing a System Distributed on RX02 to Run on RX02. Perform the procedures described.

2.5 Performing the System-generation Process

Before you execute the system-generation process, note the following:

- You should now have created a system disk by using the RT option on the FEP Installation Diskette. If you have not, do so now by following the procedures listed in Section 2.2 (if you have two RL01/RL02 disk drives) or Section 2.3 (if you have only RX02 diskette drives). You must have created a system disk before you perform a SYSGEN.
- For a list of the features in the monitor you have installed (SJ or FB), consult the table titled Features Available in Distributed Monitor in the *RT–11 Installation and System Generation Guide.* Perform the system-generation process only if you need additional features not included in the standard monitors.

- To install the XM monitor, use the SYSGEN option on the FEP Installation Disk. (See Appendix B for a list of the answers that the SYSGEN option uses in executing the system generation procedure.) To use the SYSGEN option, follow the instructions in Section 2.5.1 (if you have RL01/RL02 disks) or in Section 2.5.2 (if you have RX02 diskettes).
- Users of the IBS software may need to perform a SYSGEN for the following reasons:

  Distributed RT–11 monitors contain 16 device slots. If your hardware system has fewer than 16 different types of devices (other than MNC–series modules) connected to it when it is booted, the IB handler will be installed automatically as long as the IBV11–A unit itself is connected to the hardware system. If all of the device slots are occupied, you can remove a device handler from the system to make a device slot available for the IB handler. If you cannot remove any device handlers and if you have no extra device slots, you will have to regenerate your RT–11 monitor. See Section 2.8.13 of the *RT–11 Installation and System Generation Guide* for more information.

  NOTE

  Most MINC–family systems use fewer than eight device slots.

- For a list of other modifications available only through SYSGEN, consult the table of Features Available Only through System Generation Process in the introductory chapter of the *RT–11 Installation and System Generation Guide.*
- If you decide to perform a SYSGEN for any reason other than to install the XM monitor, follow the instructions in the *RT–11 Installation and System Generation Guide*, and see Section 2.5.3 of this manual.
- If you decide not to perform a SYSGEN, skip the remainder of this chapter and proceed to Chapter 3.

2.5.1 Using the SYSGEN Option with Two RL01/RL02 Disks

To use the FEP distribution disk to perform a SYSGEN that creates an XM monitor, perform the following procedures:

1. Bootstrap the FEP distribution disk in drive 0.
2. Type the commands in the following dialog (for a list of documentation conventions observed in this dialog, see Section 2.2.1):

```
RT-11/FEP V2.1
;
;
; Welcome to RT-11/FEP version 2.1.
;
; This disk provides automatic procedures for
; software installation.
;
```

```
; You must use the RT-11/FEP and RT-11/FRP Installation and
; User's Guide in conjunction with these procedures.

Please enter
Today's date (dd-mmm-yy):
```

Enter a date in which dd represents the day (a decimal number from 1 to 31), mmm represents the first three characters of the name of the month, and yy represents the year (a decimal number from 73 to 99) :dd-mmm-yy(RET)

```
;
; Select the procedure you want to perform by typing one
; of the following procedures:
;
; RT      Build an RT-11 development system disk
; SYSGEN  Sysgen an XM monitor
; COPY    Copy (preserve) any disk
;
* Select a procedure [S]: SYSGEN(RET)
;
; RT-11 XM SYSGEN
;
; This procedure creates an XM monitor, which can be used
; only with an 11/23 processor.
;
; Before doing the sysgen, you should have installed an
; RT-11 monitor (for example, by using the RT procedure on
; this disk) and should also have become familiar with RT-11.
; Continue with the sysgen only if you have already done both.
;
* Do you want to continue with the SYSGEN? [Y/N]:
```

Type Y(RET) or N(RET).

```
;
; Insert the RT-11 distribution disk in DL1:
;
* Ready? [Y/N]: Y(RET)
;
; ASSEMBLING AND LINKING SYSTEM FILES
;
; THIS PROCESS WILL RUN UNATTENDED FOR ABOUT 20 MINUTES
;
; Remove the RT-11 Distribution Disk from DL1:
;
* Continue? [Y/N]: Y(RET)
;
; This procedure will copy the system files, which resulted
; from the sysgen procedure, to a target system (an RT-11 system
; disk of your choice). If you used the FEP distribution
; disk to create an RT-11 development system, you can use
; the disk that resulted from that procedure as a target
; system.
;
; NOTE: The target system must have at least 110 blocks
; of free space.
;
* Do you want to copy the new system files at this time? [Y/N]: Y(RET)
;
; Insert the disk that contains the target system in DL1:
;
* Ready? [Y/N]: Y(RET)
;
```

```
; Your RT-11 XM sysgen is complete. The current system
; disk, which is a copy of the FEP distribution disk,
; contains object and map files that you will find
; useful in the future. It also contains the system
; files resulting from the sysgen process. You can copy
; these files to a system disk of your own at any time.
; The system files created by the sysgen have the .SYG file
; type.
;
@ <EOF>

.
```

2.5.2 Using the SYSGEN Option with Only RX02 Diskettes

To use the FEP Installation Diskette to perform a SYSGEN that creates an XM monitor, perform the following procedures:

1. Bootstrap the FEP Installation Diskette in drive 0.
2. Type the commands in the following dialog (for a list of documentation conventions observed in this dialog, see Section 2.3.1):

```
RT-11/FEP V2.1
;
;
; Welcome to RT-11/FEP version 2.1.
;
; This installation diskette provides
; automatic procedures for software installation.
;
; You must use the RT-11/FEP and RT-11/FRP Installation and
; User's Guide in conjunction with these procedures.
;
Please enter
Today's date (dd-mmm-yy):
```

Enter a date in which `dd` represents the day (a decimal number from 1 to 31), `mmm` represents the first three characters of the name of the month, and `yy` represents the year (a decimal number from 73 to 99): `dd-mmm-yy`RET

```
;
; Select the procedure you want to perform by typing one
; of the following codes.
;
; RT      Build an RT-11 development system diskette
; SYSGEN  Sysgen an XM monitor
; COPY    Copy (preserve) any diskette
;
* Select a procedure [S]: SYSGENRET
;
; RT-11 XM SYSGEN
;
; This procedure creates an XM monitor, which can
; be used only with an 11/23 processor.
;
; Before doing the sysgen, you should have installed an
; RT-11 monitor (for example, by using the RT procedure on
; this diskette) and should also have become familiar with
; RT-11. Continue with the sysgen only if you have already
; done both.
;
```

```
* Do you want to continue with the SYSGEN? [Y/N]:
```
Type Y(RET) or N(RET).
```
;
; For this procedure, you will need one blank diskette.
; This diskette, which is referred to as the work
; volume, will receive the object modules and link map
; which result from the RT-11 system generation. You
; should retain the work volume for future reference.
;
; Insert the diskette to be used as the work volume in DY1:
;
* Ready? [Y/N]: Y(RET)
;
* Any data on DY1: will be lost. Continue? [Y/N]: Y(RET)
;
; Formatting DY1:
? FORMAT-I-Formatting complete
;
; Initializing DY1:
? DUP-I-No bad blocks detected DY1:
;
; For the next operation:
;
; The input volume is diskette 1 of the RT-11 distribution kit.
; The output volume is the work volume (already in DY1:).
; The system volume is the FEP installation diskette.
;
Mount input volume in DY0:; Continue? Y(RET)
Mount output volume in DY1:; Continue? Y(RET)
Mount system volume in DY0:; Continue? Y(RET)
;
; Insert diskette 3 of the RT-11 distribution kit in DY1:
;
* Ready? [Y/N]: Y(RET)
;
; For the next operation:
;
; The input volume is the work volume.
; The system volume is the FEP installation diskette.
;
; (The system volume will be requested only if an error occurs)
;
Mount input volume in DY0:; Continue? Y(RET)

RT-11/FEP V2.1
;
; ASSEMBLING AND LINKING SYSTEM FILES
;
; THIS PROCESS WILL RUN UNATTENDED FOR ABOUT 30 MINUTES
;
; Remove the RT-11 distribution diskette from DY1:
;
* Continue? [Y/N]: Y(RET)
;
; This procedure will copy the system files, which resulted
; from the sysgen procedure, to a target system (an RT-11 system
; diskette of your choice). If you used the FEP installation
; diskette to create an RT-11 development system, you can use
; the diskette which resulted from that procedure as a target
; system.
;
; NOTE: The target system must have at least 110 blocks
; of free space.
;
```

```
* Do you want to copy the new system files at this time? [Y/N]: Y(RET)
;
; Insert the diskette that contains the target system in DY1:
;
* Ready? [Y/N]: Y(RET)
;
; Your RT-11 XM sysgen is complete. Save the sysgen work
; diskette (the current system diskette) for future
; reference. It contains object and map files which you
; will find useful in the future. It also contains the
; system files resulting from the sysgen process. You
; can copy these files to a system diskette of your own
; at any time. The system files created by the sysgen have
; the .SYG file type.
```

2.5.3 Performing a SYSGEN Using the RT–11 Installation and System Generation Guide

If you perform a SYSGEN by following instructions in the *RT–11 Installation and System Generation Guide*, keep the following in mind:

- If you have RX02 diskette drives and perform a SYSGEN, be sure to enable support of diskettes in both RX01 and RX02 format. (You need RX01 support to access software that has been distributed in RX01 format.)
- If you perform a SYSGEN, answer Y to the question that asks whether you want all the keyboard monitor commands. Many of the installation procedures in this manual require you to use those commands, such as SQUEEZE, RENAME, and LINK.
- The *RT–11 Installation and System Generation Guide* contains an appendix that lists answers to SYSGEN questions. One section of this appendix lists answers that duplicate the standard single-job and foreground/background monitors. When you perform a SYSGEN, use these answers to enable standard features; supply your own answers to enable nonstandard features. For example, to create a monitor that is identical to a standard monitor except that it also includes multi-terminal support, use those answers but answer Y to the question that asks if you want multi-terminal support. (Note that answering Y to that question causes SYSGEN to ask two additional questions.)
- If you choose multi-terminal support, do not assign an SLU for use by a terminal if you plan to use that SLU for REAL–11/MNC serial I/O (see Section A.3.12).
- Read the sections in the *RT–11 Installation and System Generation Manual* on performing the system generation process if you plan to perform that process.

Chapter 3
Installing FORTRAN IV

The FORTRAN IV distribution media contain a standard compiler and OTS (Object–Time System) as well as programs for building the FORTRAN IV software and tailoring it to meet your needs.

DIGITAL strongly recommends that you build the software rather than use the standard FORTRAN IV. One reason for doing so is that only by building the software yourself can you enable FORTRAN IV programs to use the floating-point hardware. In addition, instructions for installing some FORTRAN IV patches may require your performing the FORTRAN IV build procedures.

If you decide to build FORTRAN IV yourself, use the installation and build procedures given in this chapter. If you decide to install the standard FORTRAN IV software, follow instructions given in the *RT–11 FORTRAN IV Installation Guide* and ignore the remainder of this chapter.

Sections 3.1 and 3.2 contain instructions for building FORTRAN IV. (See Figure 3–1 for an overview of the installation process.) If you have two RL01 or RL02 disks, perform the procedures described in Section 3.1. If you have only RX02 diskettes, perform the procedures described in Section 3.2.

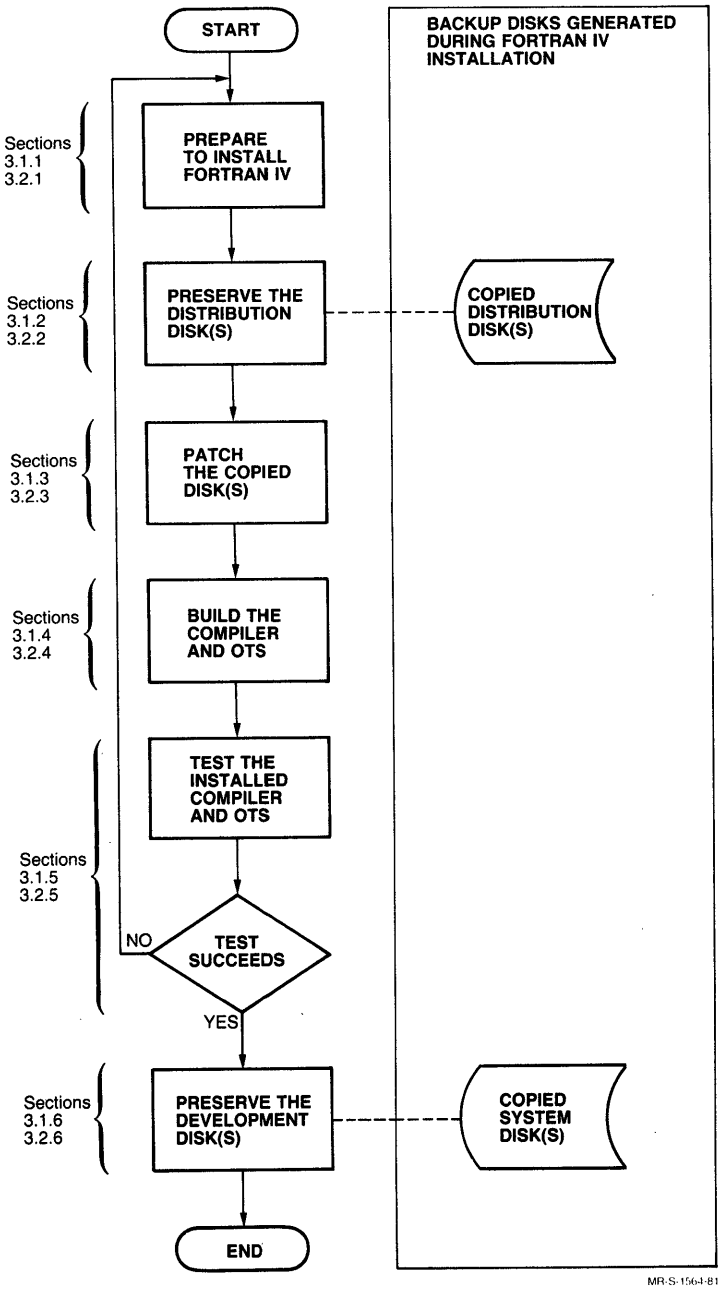
3.1 Installing FORTRAN IV When Using Two RL01/RL02 Disk Drives

Read this section before you begin the FORTRAN IV installation procedures. This section contains information that will help you install and build FORTRAN IV. For further information, consult the *RT–11 FORTRAN IV Installation Guide.*

The FORTRAN IV installation procedure involves the following operations:

1. Preparing for installation
2. Preserving the distribution disk
3. Patching the copied disk
4. Building the compiler and OTS (Object Time System) on your development system
5. Testing the installed compiler and OTS
6. Preserving the development system disk

Figure 3–1: Building FORTRAN IV



3.1.1 Preparing to Install FORTRAN IV

Before beginning the FORTRAN IV installation procedure:

- Gather the following:

  RT–11 FORTRAN IV Installation Guide

  The RT–11 Autopatch Kit

  FORTRAN IV distribution disk

  One blank disk

  The disk onto which you have backed up your development system

- Read the *FORTRAN IV Installation Guide*
- See Table 3–1 to verify that you have the minimal software needed for installing FORTRAN IV.

Table 3–1: Minimal Software Needed for Installing FORTRAN IV

File	Comment
RT11SJ.SYS	If you want to use the SJ monitor.
RT11FB.SYS	If you want to use the FB monitor.
DL.SYS	If you have two RL01/RL02 disk drives.
DY.SYS	If you have only RX02 diskette drives.
LP.SYS	If you have a parallel line printer, such as an LA180.
LS.SYS	If you have a serial line printer, such as an LA120.
TT.SYS	
SWAP.SYS	
CREF.SAV	
DIR.SAV	
DUP.SAV	
FORMAT.SAV	
LIBR.SAV	
LINK.SAV	
MACRO.SAV	
PIP.SAV	
RESORC.SAV	
SYSMAC.SML	
STARTS.COM	If you have included RT11SJ.SYS.
STARTF.COM	If you have included RT11FB.SYS.
SYSLIB.OBJ	

3.1.2 Preserving the Distribution Disk

Preserve the FORTRAN IV distribution disk by executing the following instructions. (Use this procedure to preserve any disk that does not contain the operating system.)

1. Write-protect drive 0.

   Bootstrap your development system disk in drive 0.

   Mount a blank disk in drive 1.

   Initialize the disk in drive 1 by typing the following commands:

```
.INITIALIZE/BADBLOCKS  DL1:RET
DL1:/INITIALIZE; Are you sure? Y RET
?DUP-I-No bad blocks detected DL1:
```

   NOTE

   If the system finds bad blocks, see the discussion of the INITIALIZE command in the *RT–11 System User's Guide* for additional information.

2. Copy the disk by typing:

```
.SQUEEZE/OUTPUT:DL1:/WAIT  DL0:RET
Mount output volume in DL1:; Continue?Y RET
Mount input volume in DL0:; Continue?
```

   Remove the development system disk from drive 0.

   Mount the disk to be copied in drive 0.

   When the READY light is lit, type Y RET.

3. When copying is complete, the terminal displays:

```
Mount system volume in DL0:; Continue?
```

   Remove the disk from drive 0 and store it in a safe place.

   Mount the development system disk in drive 0. When the READY light is lit, type Y RET.

3.1.3 Patching the Copied Disk

1. Be sure that the development system disk is bootstrapped in drive 0 and the copied FORTRAN distribution disk is mounted in drive 1.

2. Remove protection from files on the copied disk by typing the following command. (Use the NOLOG option to prevent a list of renamed files from appearing on the terminal.)

```
.RENAME/SYSTEM/NOPROTECTION/NOLOG  DL1:*.*  DL1:*.*RET
```

3. Apply the patches for FORTRAN IV by following instructions in the *RT–11 Autopatch User's Guide.*

4. Bootstrap the development system disk in drive 0, and mount the patched disk in drive 1. Protect files on the patched disk by typing the following command. (The NOLOG option prevents a list of renamed files from appearing on the terminal.)

```
.RENAME/SYSTEM/PROTECTION/NOLOG  DL1:*.*  DL1:*.*RET
```

3.1.4 Building the Compiler and OTS

Use the following sample dialog as a guide to answering questions asked by the system as it builds the compiler and OTS. Before executing the dialog, be sure that your development system disk is on drive 0 and that your copy of the FORTRAN IV distribution disk is on drive 1. Write-enable drive 0, and write-protect drive 1.

(This dialog contains answers likely to be useful to you. For alternative answers, consult the *RT–11 FORTRAN IV Installation Guide.*)

```
.ASSIGN DL0: OUPRET

.ASSIGN DL1: INPRET

.RUN INP:FORGENRET
Answer questions with:
"?" or carriage return (<CR>) for information,
YES(Y) for affirmation, anything else for NO.


A maximum of 56 lines are allowed per listing page.
Is this acceptable? YRET
A maximum of 136 characters are allowed in a formatted
(ASCII) record. Is this acceptable? YRET
A maximum of 6 channels may be open at a given time.
Is this acceptable? YRET
The FORTRAN compiler can compile system-specific OPEN and CLOSE
statement keywords for RT-11, RSX-11, and RSTS/E systems. These
statements are currently compiled for the following system: RT-11
Is this acceptable? YRET
Based on your system's hardware configuration the default
type of code generated by the FORTRAN compiler will be xxx.
Is this acceptable? YRET
```

(where xxx is FIS if you have an 11/03 processor or EIS if you have an 11/23 processor).

```
Do you wish an inline only compiler? NRET
```

(You need a threaded compiler to use FDT.)

```
Compiler options selection complete.

Type: @OUP:FORBLD to initiate the actual building
of the compiler.

.@OUP:FORBLDRET

@INP:F4LINK
```

```
.R LINK
*OUP:FORTRA=INP:FROOT///S
*INP:F0,OUP:DEFLTS/O:1
*INP:F1/O:1
*INP:F2/O:1
*INP:F3/O:1
*INP:F4/O:1
*INP:F5/O:1
*INP:F6/O:1
*INP:F7/O:1
*INP:F8/O:1
*INP:F9/O:1
*INP:F10/O:1
*INP:LOOP/O:1
*INP:F11/O:1
*INP:F12/O:1
*INP:CONVRT/O:1
*INP:REGALO/O:1
*INP:F14/O:1
*INP:F20/O:1
*INP:F19/O:1
*INP:F21/O:1
*INP:PEEP/O:1
*INP:OBJGSD/O:1
*INP:F18/O:1
*INP:F13/O:1
*INP:F15/O:1
*INP:F16/O:1
*INP:F17/O:1
*INP:CDUMP/O:1//
*^C

.RUN INP:OTSGEN(RET)
Answer questions with:
"?" or carriage return (<CR>) for information,
YES(Y) for affirmation, anything else for NO.


The following questions refer to the building of the
default FORTRAN library.

The FORTRAN library components are normally added to
the default RT-11 system library file, SYSLIB.OBJ.
Is this acceptable? Y(RET)
The default FORTRAN library will be found in the file SYSLIB.OBJ.

The default FORTRAN library will be configured for the
xxx hardware found on this machine. Is this acceptable? Y(RET)
```

(where xxx is FIS if you have an 11/03 processor and FPU if you have an 11/23 processor).

```
Is support for VIRTUAL arrays desired?
```

If you have an 11/23 processor and more than 32k words of memory, answer to this question; otherwise, answer N. If you answer Y, the following question also appears.

```
Since you are running under the SJ monitor,
and you have a KT-11 memory management unit, the
VIRTUAL support selected for the default library
is: SJ/FB
Is this acceptable? Y(RET)
Do you wish an inline only library? N(RET)
Should run-time checks be made to insure all array
references are contained within program bounds? N(RET)
```

```
Do you want SIMRT support in your library? N(RET)
The default FORTRAN library has been configured.

Are any other FORTRAN libraries required? N(RET)

Type: @OUP:OTSBLD to initiate the actual building of the
FORTRAN library(s).

STOP --

.@OUP:OTSBLD(RET)

.R LIBR
*OUP:SYSLIB[-1]:=OUP:SYSLIB,INP:FPU,
INP:OTSCOM,INP:VIRNP,INP:V2S/G
Global? $ERRS
Global? $ERRTB
Global? $OVRH
Global?
*^C
```

3.1.5 Testing the Installed Compiler and OTS

Perform the procedures for installation verification in the *RT–11 FORTRAN IV Installation Guide.*

3.1.6 Preserving the Development System Disk

Preserve your development system disk, which contains the FORTRAN IV compiler and OTS, by executing the following instructions:

1. Write-protect drive 0.

   Write-enable drive 1.

   Mount your development system backup disk in drive 1.

2. Initialize the output disk by typing:

   ```
   .INITIALIZE/BADBLOCKS  DL1:(RET)
   DL1:/INITIALIZE; Are you sure? Y(RET)
   ?DUP-I-No bad blocks detected DL1:
   ```

   NOTE

   If the system finds bad blocks, see the discussion of the INITIALIZE command in the *RT–11 System User's Guide* for additional information.

   Copy the development system disk by typing:

   ```
   .SQUEEZE/OUTPUT:DL1:  DL0:(RET)
   ```

   Copy the bootstrap by typing:

   ```
   .COPY/BOOT  DL1:RT11yy.SYS  DL1:(RET)
   ```

   where *yy* is the name of the monitor in operation.

   Remove the disk from drive 1, and store it in a safe place.

3.2 Installing FORTRAN IV When Using Two RX02 Diskette Drives

Read this section before you begin the FORTRAN IV installation procedures. This section contains information that will help you install and build FORTRAN IV. For further information, see the *RT–11 FORTRAN IV Installation Guide.*

The FORTRAN IV installation procedure involves the following operations:

1. Preparing for installation
2. Preserving the distribution diskettes
3. Patching the copied diskettes
4. Building the compiler and OTS (Object Time System) on your development system diskette
5. Testing the installed compiler and OTS
6. Preserving the development system diskette

The FORTRAN IV distribution kit contains three diskettes. They are:

- The compiler diskette, which contains files for building the FORTRAN IV compiler
- The OTS diskette, which contains files for building the FORTRAN IV OTS
- The standard-software diskette, which contains a pre-built compiler and OTS library as well as optional FORTRAN IV modules

3.2.1 Preparing to Install FORTRAN IV

Before beginning the FORTRAN IV installation procedure:

- Gather the following:

  RT–11 FORTRAN IV Installation Guide

  The RT–11 Autopatch Kit

  FORTRAN IV distribution diskettes

  Three blank RX02 diskettes

  The diskette onto which you have backed up your development system
- Read the *FORTRAN IV Installation Guide.*
- Create additional free space on the development system disk by deleting the RT–1 HELP file, HELP.SAV:

1. Bootstrap the development system disk in drive 0.
2. Type the following command. (The NOQUERY option prevents the system from printing a confirmation message before it deletes the file.)

```
.DELETE/NOQUERY HELP.SAV(RET)
```

3. Collect free space on the disk into one contiguous area by typing:

```
.SQUEEZE SY:(RET)
SQUEEZE:/Squeeze; Are you sure? Y(RET)

RT-11xx V04.00
```

(followed by any startup file commands)

where `dv` is `DL` if you have two RL01/RL02 disk drives or `DY` if you have only RX02 diskette drives

`xx` is the name of the monitor in operation

- Before installing FORTRAN IV software, be sure your development system disk contains at least the files listed in Table 3–1.

3.2.2 Preserving the Distribution Diskettes

Preserve the FORTRAN IV distribution kit by executing the following instructions for each distribution diskette. (Use this procedure to preserve any diskette that does not contain the operating system.)

1. Bootstrap your development system diskette in drive 0.

   Mount a blank diskette in drive 1.

   Format and initialize the disk in drive 1 by typing the following commands:

```
.FORMAT  DY1:(RET)
DY1:/FORMAT-Are you sure?Y(RET)
?Format-I-Formatting complete

.INITIALIZE/BADBLOCKS  DY1:(RET)
DY1:/Initialize; Are you sure? Y(RET)
?DUP-I-No bad blocks detected DY1:
```

NOTE

If the system finds bad blocks, see the *RT–11 System User's Guide* for additional information on the INITIALIZE command.

2. Type the following commands to copy the distribution diskette:

```
.SQUEEZE/OUTPUT:DY1:/WAIT  DY0:(RET)
Mount output volume in DY1:; Continue? Y(RET)
Mount input volume in DY0:; Continue?
```

   Remove the diskette from drive 0.

Mount the next diskette to be copied in drive 0.

Type Y(RET).

When the diskette is copied, the following message appears:

```
Mount system volume in DY0:? Continue?
```

Remove the diskette from drive 0 and store it in a safe place.

Remove the diskette from drive 1.

Mount the system diskette in drive 0 and type Y(RET).

3.2.3 Patching the Copied Diskettes

1. Be sure that the development system diskette is bootstrapped in drive 0 and a copied FORTRAN distribution diskette is mounted in drive 1.
2. Remove protection from files on the copied diskette by typing the following command. (Use the NOLOG option to prevent a list of renamed files from appearing on the terminal.)

   ```
   .RENAME/SYSTEM/NOPROTECTION/NOLOG  DY1:*.*  DY1:*.*(RET)
   ```

   Remove protection from each of the other copied FORTRAN distribution diskettes.
3. Apply the patches for FORTRAN IV by following instructions in the *RT–11 Autopatch User's Guide.*
4. Bootstrap the development system diskette in drive 0, and mount a patched diskette in drive 1. Protect files on the patched diskette by typing the following command. (The NOLOG option prevents a list of renamed files from appearing on the terminal.)

   ```
   .RENAME/SYSTEM/PROTECTION/NOLOG  DY1:*.*  DY1:*.*(RET)
   ```

   Protect files on each of the other patched diskettes.

3.2.4 Building the Compiler and OTS

Bootstrap the development system disk in drive 0.

If you installed RT–11 by using the RT option of the FEP Installation Diskette, the development system diskette does not contain enough free space to accomodate the FORTRAN IV installation procedure; create free space on the development system diskette by typing the following commands:

```
RT-11xx V04.00
(followed by any startup file commands)
```

where xx is the name of the monitor in operation

```
.DELETE HELP.SAV(RET)
.SQUEEZE DY0:(RET)
```

Mount copied, patched FORTRAN IV distribution diskette 1 in drive 1.

Use the following sample dialog as a guide to answering questions asked by the program that builds the compiler and OTS. (The dialog contains answers likely to be useful to you. For alternative answers, consult the *RT–11 FORTRAN IV Installation Guide*.)

```
.ASSIGN DY0: OUP(RET)

.ASSIGN DY1: INP(RET)

.RUN INP:FORGEN(RET)
Answer questions with:
"?" or carriage return (<CR>) for information,
YES(Y) for affirmation, anything else for NO.


A maximum of 56 lines are allowed per listing page.
Is this acceptable? Y(RET)
A maximum of 136 characters are allowed in a formatted
(ASCII) record. Is this acceptable? Y(RET)
A maximum of 6 channels may be open at a given time.
Is this acceptable? Y(RET)
The FORTRAN compiler can compile system-specific OPEN and CLOSE
statement keywords for RT-11, RSX-11, and RSTS/E systems. These
statements are currently compiled for the following system: RT-11
Is this acceptable? Y(RET)
Based on your system's hardware configuration the default
type of code generated by the FORTRAN compiler will be xxx.
Is this acceptable? Y(RET)
```

(where xxx is FIS if you have an 11/03 processor or EIS if you have an 11/23 processor).

```
Do you wish an inline only compiler? N(RET)
```

(You need a threaded compiler to use FDT.)

```
Compiler options selection complete.

Type: @OUP:FORBLD to initiate the actual building
of the compiler.

.@OUP:FORBLD(RET)

@INP:F4LINK

.R LINK
*OUP:FORTRA=INP:FROOT///S
*INP:F0,OUP:DEFLTS/O:1
*INP:F1/O:1
*INP:F2/O:1
*INP:F3/O:1
*INP:F4/O:1
*INP:F5/O:1
*INP:F6/O:1
*INP:F7/O:1
*INP:F8/O:1
*INP:F9/O:1
*INP:F10/O:1
*INP:LOOP/O:1
*INP:F11/O:1
*INP:F12/O:1
*INP:CONVRT/O:1
*INP:REGALO/O:1
*INP:F14/O:1
```

```
*INP:F20/O:1
*INP:F19/O:1
*INP:F21/O:1
*INP:PEEP/O:1
*INP:OBJGSD/O:1
*INP:F18/O:1
*INP:F13/O:1
*INP:F15/O:1
*INP:F16/O:1
*INP:F17/O:1
*INP:CDUMP/O:1//
*^C
```

Remove the diskette from drive 1. Mount the copy of the second FORTRAN IV distribution diskette in drive 1.

```
.RUN INP:OTSGEN(RET)
Answer questions with:
"?" or carriage return (<CR>) for information,
YES(Y) for affirmation, anything else for NO.


The following questions refer to the building of the
default FORTRAN library.

The FORTRAN library components are normally added to
the default RT-11 system library file, SYSLIB.OBJ.
Is this acceptable? Y(RET)
The default FORTRAN library will be found in the file SYSLIB.OBJ.

The default FORTRAN library will be configured for the
xxx hardware found on this machine. Is this acceptable? Y(RET)
```

(where xxx is FIS if you have an 11/03 processor and FPU if you have an 11/23 processor).

```
Is support for VIRTUAL arrays desired?
```

If you have an 11/23 processor and more than 32k words of memory, answer Y to this question; otherwise, answer N. If you answer Y, the following question also appears.

```
Since you are running under the SJ monitor,
and you have a KT-11 memory management unit, the
VIRTUAL support selected for the default library
is: SJ/FB

Is this acceptable? Y(RET)
Do you wish an inline only library? N(RET)
Should run-time checks be made to insure all array
references are contained within program bounds? N(RET)
Do you wish SIMRT support in your library? N(RET)
The default FORTRAN library has been configured.

Are any other FORTRAN libraries required? N(RET)

Type: @OUP:OTSBLD to initiate the actual building of the
FORTRAN library(s).

STOP --
```

```
.@OUP:OTSBLD(RET)

.R LIBR
*OUP:SYSLIB[-1]:=OUP:SYSLIB,INP:FPU,
INP:OTSCOM,INP:VIRNP,INP:V2S/G
Global? $ERRS
Global? $ERRTB
Global? $OVRH
Global?
*^C
```

3.2.5 Testing the Installed Compiler and OTS

Perform the procedures for installation verification in the *RT–11 FORTRAN IV Installation Guide.*

3.2.6 Preserving the Development System Diskette

Preserve your development system diskette, which contains the FORTRAN IV compiler and OTS, by executing the following instructions:

1. Mount your development system backup diskette in drive 1.
2. Initialize the output diskette by typing:

```
.INITIALIZE/BADBLOCKS  DY1:(RET)
DY1:/INITIALIZE; Are you sure? Y(RET)
?DUP-I-No bad blocks detected DY1:
```

NOTE

If the system finds bad blocks, see the *RT–11 System User's Guide* for additional information on the INITIALIZE command.

Copy the development system diskette by typing:

```
.SQUEEZE/OUTPUT:DY1:  DY0:(RET)
```

Copy the bootstrap by typing:

```
.COPY/BOOT  DY1:RT11yy.SYS  DY1:(RET)
```

where `yy` is the name of the monitor in operation.

Remove the diskette from drive 1, and store it in a safe place.

Chapter 4
Installing the FEP/FRP Laboratory Software

This chapter contains detailed step-by-step instructions for installing the FEP/FRP Laboratory Software. All FEP/FRP component manuals, except the *REAL–11/MNC FORTRAN Programmer's Reference Manual* and the *RGL/11 Programmer's Reference Manual*, also contain similar installation instructions. Follow the instructions in this chapter rather than those in the individual FEP/FRP component manuals; the instructions in this chapter provide help in using RL01/RL02 disks and RX02 diskettes whereas the instructions in the FEP/FRP component manuals do not.

To install the FEP/FRP Laboratory Software, perform the following procedures, all of which are described in this chapter:

1. Preserve all distribution media that contain FEP/FRP Laboratory Software.
2. Patch and preserve the copied disks.
3. Install and verify a FEP/FRP Laboratory Software component.
4. Preserve your development system disk.

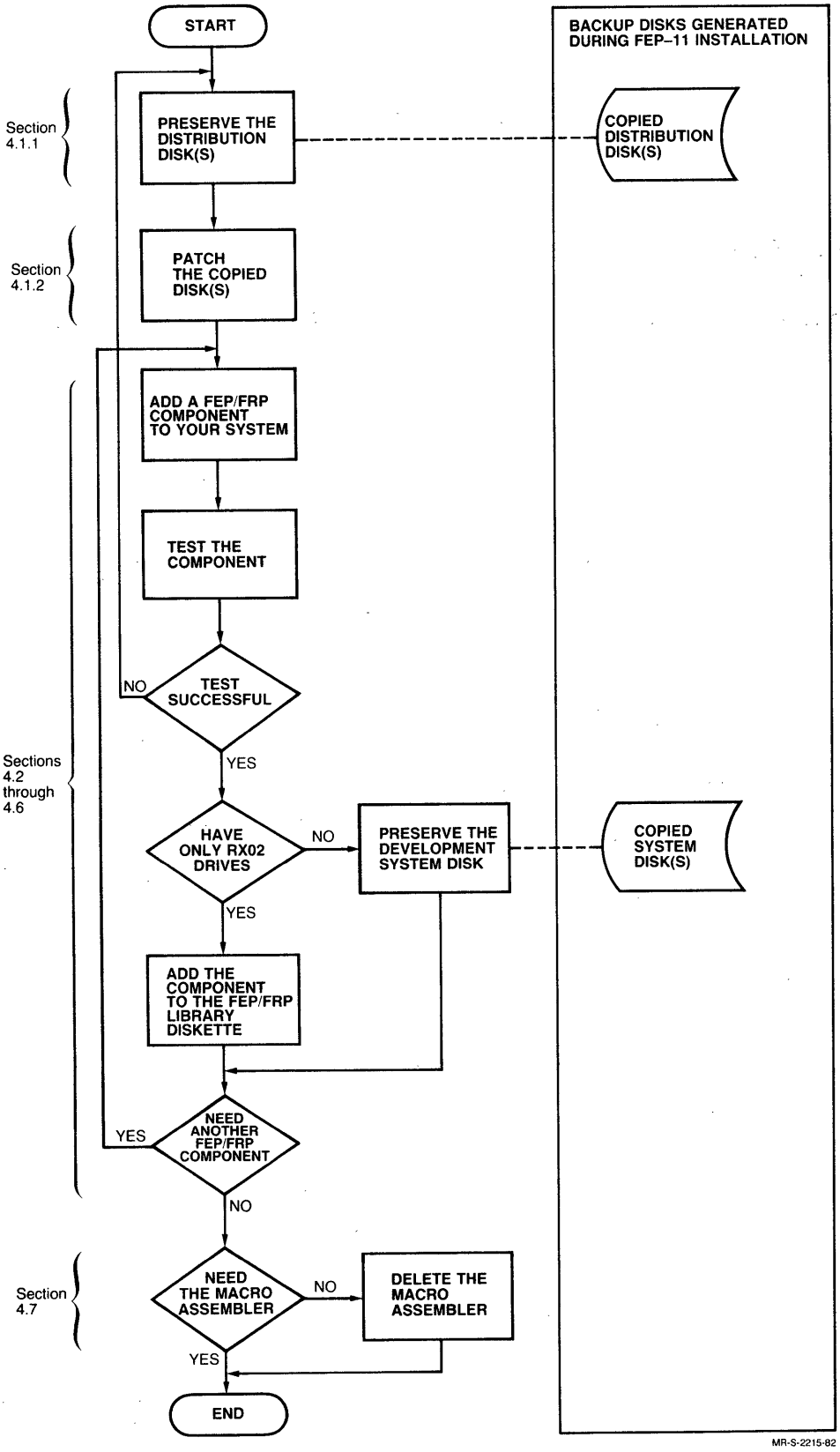
Repeat steps 3 and 4 until you have installed all the components you need. (See Figure 4–1 for an overview of the installation procedure.)

NOTE

> The FEP/FRP Laboratory Software components are independent of each other. Install only the components you need; if you will use only one of them, you need to install only that one.

Section 4.1 tells you how to copy and patch the distribution media. Sections 4.2 through 4.7 tell you how to install, verify, and preserve each FEP/FRP component. Section 4.8 contains optional procedures for deleting the MACRO assembler.

Figure 4–1: Installing FEP/FRP



4.1 Preparing to Install the FEP/FRP Laboratory Software

Before you begin the FEP/FRP Laboratory Software installation procedures:

- Gather the following:

  Copies of the RT–11 software dispatch

  FEP/FRP Laboratory Software distribution media

  One blank RL01 or RL02 disk (if you have RL01 or RL02 disk drives)

  Fifteen blank RX02 diskettes (if you have RX02 diskette drives)

  The disk onto which you have backed up your development system

- Be sure your development system disk contains at least the files listed in Table 4–1.

- If you have only RX02 diskette drives, create a new development system diskette by performing the following instructions:

  1. Bootstrap the development system disk in drive 0.

     Mount a blank diskette in drive 1.

     Format the diskette in drive 1 by typing the following commands:

     ```
     .FORMAT  DY1:RET
     DY1:/FORMAT-Are you sure?YRET
     ?Format-I-Formatting complete
     ```

  2. Follow the instructions in Section 3.2.6 for preserving the development system diskette.

     Remove the diskette from drive 0 and store it in a safe place. After you have installed all needed FEP/FRP software, you can use this diskette as a system diskette for program development.

     Move the other diskette from drive 1 to drive 0.

     Bootstrap the diskette in drive 0.

  3. Create additional free space on the diskette by using the DELETE command to delete all files not listed in Table 4–1. Use this diskette as the development system diskette for installing FEP/FRP Laboratory Software.

- Collect free space on the disk into one contiguous area by typing:

  ```
  .SQUEEZE SY:RET
  dv0:/Squeeze; Are you sure? YRET

  RT-11xx V04.00
  ```
  (followed by any startup file commands)

where `dv` is `DL` if you have two RL01/RL02 disk drives or `DY` if you have only RX02 diskette drives

`xx` is the name of the monitor in operation

- If you have only RX02 diskettes, create the FEP Library Diskette by performing the following procedures:

1. Mount a blank diskette in drive 1.

   Format and initialize the disk in drive 1 by typing the following commands:

```
.FORMAT  DY1:(RET)
DY1:/FORMAT-Are you sure?Y(RET)
?Format-I-Formatting complete

.INITIALIZE/BADBLOCKS  DY1:(RET)
DY1:/Initialize;Are you sure?Y(RET)
?DUP-I-No bad blocks detected DY1:
```

NOTE

If the system finds bad blocks, see the *RT–11 System User's Guide* for additional information on the INITIALIZE command.

2. Remove the diskette from drive 1. This is the FEP Library Diskette. Installation instructions for all FEP/FRP Laboratory Software include copying installed software to this diskette.

Table 4–1: Minimal Software Needed for Installing FEP/FRP

File	Comment
RT11SJ.SYS	If you want to use the SJ monitor.
RT11FB.SYS	If you want to use the FB monitor.
DL.SYS	If you have two RL01/RL02 disk drives.
DY.SYS	If you have only RX02 diskette drives.
LP.SYS	If you have a parallel line printer, such as an LA180.
LS.SYS	If you have a serial line printer, such as an LA120.
TT.SYS	
SWAP.SYS	
CREF.SAV	
DIR.SAV	
DUP.SAV	
FORMAT.SAV	
LIBR.SAV	
LINK.SAV	

Table 4–1: Minimal Software Needed for Installing FEP/FRP (Cont.)

File	Comment
MACRO.SAV	
PIP.SAV	
RESORC.SAV	
SYSMAC.SML	
STARTS.COM	If you have included RT11SJ.SYS.
STARTF.COM	If you have included RT11FB.SYS.
SYSLIB.OBJ	Which includes the FORTRAN OTS if you have not added FORLIB.OBJ as a separate library.
FORLIB.OBJ	If you have not added the FORTRAN OTS to SYSLIB.
FORTRA.SAV	

4.1.1 Preserving the Distribution Media

If you have two RL01/RL02 disk drives, you have already preserved the FEP/FRP distribution disk by following procedures in Section 2.2.1.

If you have only RX02 diskette drives, preserve the distribution diskettes that contain the FEP/FRP Laboratory Software by following the instructions in Section 3.2.2. However, when initializing diskettes onto which you will copy the SSP software, use the following form of the INITIALIZE command:

```
.INITIALIZE/BADBLOCKS/SEGMENTS:5  DY1:RET
```

This form of the command creates a directory large enough to accommodate the SSP software. For more information on the INITIALIZE command and the SEGMENTS option, see the *RT–11 System User's Guide.*

4.1.2 Patching the FEP/FRP Laboratory Software

Install all FEP/FRP patches contained in the RT–11 Software Dispatch. Before installing patches, remove protection from files on the copied distribution disk by typing the following command. (Use the NOLOG option to prevent a list of renamed files from appearing on the terminal.)

```
.RENAME/SYSTEM/NOPROTECTION/NOLOG  dv1:*.*  dv1:*.*RET
```

After installing patches, protect files on the patched disk by typing the following command. (Use the NOLOG option to prevent a list of renamed files from appearing on the terminal.)

```
.RENAME/SYSTEM/PROTECTION/NOLOG  dv1:*.*  dv1:*.*(RET)
```

where `dv` is `DL` if you have two RL01/RL02 disk drives or `DY` if you have only RX02 diskette drives.

4.2 Installing REAL–11/MNC

The FEP/FRP distribution kit contains standard REAL–11/MNC library files as well as programs for customizing the REAL–11/MNC software. Read Section 4.2.1 and decide if one of the distributed library files fulfills your needs. If one of the distributed REAL–11/MNC library files is satisfactory, perform the installation procedures described in this chapter. Otherwise, perform the REAL–11/MNC Software Generation Procedure (MNCGEN), described in Appendix A.

NOTE

MNCGEN is a time-consuming procedure; perform it only if the distributed software does not satisfy your needs.

4.2.1 Selecting a REAL–11/MNC Library File

The FEP/FRP distribution kit contains two REAL–11/MNC library files:

- MNCMUL.OBJ — supports multiple (simultaneous) clock-driven sweeps, which are two or more sweeps controlled or "driven" by the real-time clock. For example, if the real-time clock drives A/D and D/A sweeps that are running simultaneously, the sweeps are said to be multiple clock-driven sweeps. However, if the real-time clock drives an A/D sweep that is running concurrently with an interrupt-driven digital-input sweep, the sweeps are *not* multiple clock-driven sweeps. For further information, see the *REAL–11/MNC FORTRAN Programmer's Reference Manual.*
- MNCSNG.OBJ — does not support multiple clock-driven sweeps

The following list describes features common to both distributed REAL–11 library files. Both files:

- May be added to an existing library.
- Check for the presence of required MNC–series hardware each time a REAL–11/MNC subprogram is executed.
- Require you to set CSR and vector address switches on MNC–series modules to the default settings (see the chapter on system configuration in the *DECLAB–11/MNC User's Guide.*)
- Support any number of MNC–series modules up to the maximum.

- Support two Serial Line Units (SLUs).
- Stop sweeps on a data overrun.
- Use a value of 37777_8, which is decimal 16383, to indicate an error in the execution of any routine that acquires analog data. (REAL–11 places 37777_8 in the input buffer if the execution of any routine that acquires analog data results in a data overrun.)
- Use a value of 177777_8 to indicate an error in the execution of DISWP. REAL–11 places 177777_8 in the input buffer if the execution of DISWP results in a data overrun.
- Contain the GTHIST subprogram.
- Are supported under the RT–11 single-job, foreground/background, and extended-memory monitors. (See Section 6.6 for restrictions related to the use of the extended-memory monitor.)
- Supply a default gain value of 0.50 for the subprograms that acquire analog data.
- Use an input ring buffer of 82 bytes for the CIN subprogram.
- Use an output ring buffer of 40 bytes for the COUT subprogram.

Select a REAL–11/MNC library file to use at your installation. Choose MNCMUL.OBJ only if you will execute multiple clock-driven sweeps. Choose MNCSNG.OBJ if you will not execute multiple clock-driven sweeps. Choose carefully; using MNCMUL.OBJ causes any clock-driven sweep to execute more slowly than it would if you used MNCSNG.OBJ. If neither MNCMUL.OBJ nor MNCSNG.OBJ appears to satisfy your needs, consult Appendix A of this manual and perform the software-generation procedure to customize the REAL–11/MNC software. The version of REAL–11 produced by the software-generation procedure is named MNCLIB.OBJ.

4.2.2 Adding a Distributed REAL–11/MNC Library to the System Disk

To include distributed REAL–11/MNC software on the system disk, perform the following procedures:

1. See Table 4–1 to verify that you have the minimal software needed for installing FEP/FRP.
2. Bootstrap the development system disk in drive 0.

   If you have two RL01/RL02 disk drives, mount the copied, patched FEP/FRP distribution disk in drive 1.

   If you have only RX02 diskette drives, mount copied, patched REAL–11/MNC distribution diskette number 1 in drive 1.

3. Copy a distributed REAL–11/MNC library to the development system disk by typing *one* of the following:

   (In the following instructions, for dv, type DL if you have two RL01/RL02 disk drives or DY if you have only RX02 diskette drives.)

```
.COPY  dv1:MNCMUL.OBJ  dv0:MNCMUL.OBJ(RET)
.COPY  dv1:MNCSNG.OBJ  dv0:MNCSNG.OBJ(RET)
```

4. When you have installed all needed FEP/FRP software, see Section 5.6 for information on using the librarian to add files to SYSLIB.

4.2.3 Verifying Distributed REAL–11/MNC Software

This section contains instructions for verifying a library that contains one of the two distributed REAL–11/MNC library files. For instructions on verifying REAL–11/MNC software that you have generated yourself (by performing the MNCGEN procedure), see Appendix A.

To verify distributed REAL–11/MNC software:

1. You must be using the SJ or FB monitor. You also must execute the procedure on the system console terminal.
2. In drive 0, mount a system disk that contains both a distributed REAL–11/MNC library file and at least 100 contiguous free blocks.

   If you have two RL01/RL02 disk drives, mount the copied, patched FEP distribution disk in drive 1.

   If you have only RX02 diskette drives, mount copied, patched REAL–11/MNC diskette number 1 in drive 1.
3. The clock Schmitt triggers can generate outputs in response to electrical noise when no device is connected to the input. To avoid such spurious responses from unused Schmitt triggers, set the mode selector of each MNCKW clock module to VAR and turn the associated potentiometer to either of its extreme settings.

   Also, be sure no instruments are connected to the MNC–series modules.
4. Assign logical device names needed by the verification procedure by typing:

```
.ASSIGN  dvn:  N:(RET)
.ASSIGN  dv1:  I:(RET)
```

   where `dv` is `DL` if you have two RL01/RL02 disk drives and `DY` if you have only RX02 diskette drives.

   `n` is 0 if you have two RL01/RL02 disk drives and 1 if you have only RX02 diskette drives.

5. The verification procedure creates (and deletes) on device N: files named MVERN.OBJ and MTMP.RUN. Therefore, if you have any files with those names on device N:, rename them before you begin the verification procedure.

6. Execute the verification procedure that tests the distributed REAL–11/MNC library file installed on device N:. If you have installed MNCSNG.OBJ, verify that software by running MNCSNG.VER. If you have installed MNCMUL.OBJ, run MNCMUL.VER. Type:

```
•@I:filename.VER(RET)
```

   where `filename` is MNCSNG or MNCMUL

7. As the verification procedure tests the REAL–11/MNC software, it prints messages on the console terminal. The following list contains the types of messages and an example of each:

   - Confirmation messages, which verify that an MNC–series module and some related REAL–11/MNC software have been exercised successfully. For example:

     ```
     The primary clock (1st MNCKW) and some related REAL-11 software
     have been exercised successfully!
     ```

   - Level–W error messages, which indicate that hardware named in the message is absent. For example:

     ```
     ?MVER-W-The REAL-11 software has NOT detected an analog to
     digital input module (MNCAD) in this system.
     ```

   - Level–F error messages, which are fatal error messages indicating that the software has malfunctioned. One type of fatal error message names the verification module that has failed. For example:

     ```
     ?MVER-F-REAL Software Error - Installation failure at PSDI
     ```

     Another type of fatal error message lists trouble-shooting aids for DIGITAL support personnel. For example:

     ```
     ?MVER-F-***UNEXPECTED ERROR DURING MNCAA RELATED PHASE***
             ***CHECK HARDWARE I/O SIGNALS***

             Trouble-shooting aids for DIGITAL support:
             a) IND =       1
             b) INFO(1) =   334
     ```

   - The completion message:

     ```
     STOP -- VERIFICATION PROCESS COMPLETE
     ```

8. When the console terminal displays the system dot, evaluate the results of the verification procedure:

   - If the terminal has displayed only a series of confirmation messages followed by the completion message, then the REAL–11/MNC software and MNC–series hardware are correctly installed.

- If the terminal has displayed any level–W messages, then MNC–series hardware is absent. If a level–W message names an MNC–series module that is present in your system, be sure that module's vector and CSR address switches are correctly set, then execute the verification procedure again. Ignore level–W messages that name MNC–series hardware not present in your system.
- If the terminal has displayed any level–F messages, then the software has malfunctioned; call the MINC Product Services Center for further information.
- If the terminal has displayed a series of messages but has not printed the completion message, then the software has malfunctioned; call the MINC Product Services Center for additional information.

4.2.4 Completing the Installation of REAL–11/MNC

If you have two RL01/RL02 disk drives, follow the instructions in Section 3.1.6 for preserving your development system disk.

If you have only RX02 diskette drives, add the installed REAL–11/MNC software to the FEP Library Diskette and remove that software from the development system diskette:

1. Remove the diskette from drive 1.

   Mount the FEP Library Diskette in drive 1.

2. Type:

   ```
   .COPY DY0:filename.OBJ  DY1:filename.OBJ(RET)
   .DELETE/NOQUERY DY0:filename.OBJ(RET)
   ```

   where `filename` is MNCMUL or MNCSNG

4.3 Installing the Instrument Bus Subroutines (IBS)

The FEP/FRP distribution kit contains five standard IBS files. These files are the IBS library (IBLIB.OBJ) and the four device handlers (IBMNC.SYS, IBNMNC.SYS, IBXMNC.SYS and IBXNMC.SYS).

4.3.1 Adding the IBS Library and Driver to Your System

To install the IBS software:

1. See Table 4–1 to verify that you have the minimal software needed for installing FEP/FRP.
2. Under some circumstances you must build a customized device handler and IBS library. You must do so if one or more of the following conditions are true:

- You have performed the RT–11 SYSGEN process by using the *RT–11 Installation and System Generation Guide*, which is mentioned in Section 2.5.3 of this manual. (If you have installed the XM monitor by using the SYSGEN option described in Sections 2.5.1 and 2.5.2, you need not build a customized device handler and IBS library.)
- You have more than one IBV11–A bus unit.
- The IBV11–A vector address switch is not set to the standard value, octal 420.
- The IBV11–A CSR address switch is not set to the standard value, octal 171420.

If you need to build a customized device handler and IBS library, do so now by following the procedures in Appendix C, Building a New IBV11–A Device Handler and IBS Library, in the *Instrument Bus Subroutines Programmer's Reference Manual.*

3. Bootstrap the development system disk in drive 0.

   Mount the disk that contains the device handler in drive 1.

4. If you already have an IB handler on your development system disk, remove it by typing:

```
.UNLOAD IB
.REMOVE IB
```

5. Copy the device handler to the development system disk, but before doing so, be sure that the development system contains enough contiguous, free blocks to accommodate the device handler. If you are using the SJ or FB monitor, the development system disk must contain at least nine contiguous, free blocks. If you are using the XM monitor, the development system disk must contain at least ten contiguous, free blocks. To copy the device handler to the development system disk, type:

```
.COPY/SYSTEM  dv1:devhnd  dv0:driv.SYS
```

   where `dv` is `DL` if you have two RL01/RL02 disk drives or `DY` if you have only RX02 diskette drives

   `devhnd` is one of the following:

   `IBMNC.SYS` if you are installing the distributed device handler and are using the distributed SJ or FB monitor;

   `IBXMNC.SYS` if you are installing the distributed device handler and are using the distributed XM monitor;

   `IB.SYG` if you have built a customized device handler.

driv is one of the following:

IB if you are using the SJ or FB monitor;

IBX if you are using the XM monitor.

The following message may appear on your terminal after you have copied the device handler:

```
?PIP-W-Reboot
```

Ignore this message.

6. Be sure the disk that contains the IBS library file, IBLIB.OBJ, is mounted in drive 1.

7. Copy the library file to the development system disk, but before you do so, be sure that the development system disk contains enough contiguous, free blocks to accommodate the library file. If you are using the SJ or FB monitor, the development system disk must contain at least 48 contiguous, free blocks. If you are using the XM monitor, the development system disk must contain at least 56 contiguous, free blocks. To copy the library file to the development system disk, type:

```
.COPY dv1:IBLIB.OBJ dv0:IBLIB.OBJ<RET>
```

where dv is DL if you have two RL01/RL02 disk drives or DY if you have only RX02 diskette drives.

You have now installed IBLIB.OBJ, the library of instrument-bus subroutines. IBLIB.OBJ is the name of the library of instrument-bus subroutines regardless of whether you have customized that library or have installed the distributed library. IBLIB.OBJ is also the name of the library regardless of whether you are using the SJ, FB, or XM monitor.

8. Install and load the device handler by typing:

```
.INSTALL IB<RET>
.LOAD IB<RET>
```

9. Add the following command to the startup command file for the monitor you are using:

```
LOAD IB
```

If you are using the SJ monitor, add the command to STARTS.COM. If you are using the FB monitor, add the command to STARTF.COM. If you are using the XM monitor, add the command to STARTX.COM.

The LOAD IB command will execute correctly only if the IB handler is installed. Each time you boot your system, the IB handler is automatically installed if all the following conditions are true:

- The IB handler is on the system disk.
- The IBV11–A bus unit is connected to the system.

- The monitor's device table has enough slots for each type of device connected to the system.

  See Section 2.5 for information on the possible need for IBS users to increase the number of slots in the monitor's device table by performing a SYSGEN.

10. When you have installed all needed FEP/FRP software, see Section 5.6 for information on using the librarian to add files to SYSLIB.

4.3.2 Verifying IBS

To verify IBS, perform the procedures in Section 4.4, Verifying Your Instrument Bus Subroutines Software, in the *Instrument Bus Subroutines Programmer's Reference Manual.*

4.3.3 Completing the Installation of IBS

If you have two RL01/RL02 disk drives, follow the instructions in Section 3.1.6 for preserving your development system disk. If you have only RX02 diskette drives, follow the instructions in Section 3.2.6 for preserving your development system diskette.

If you have only RX02 diskette drives, add the installed IBS library to the FEP Library Diskette and remove that library from the development system diskette:

1. Remove the diskette from drive 1.

   Mount the FEP Library Diskette in drive 1.

2. Type:

```
.COPY  DY0:IBLIB.OBJ  DY1:IBLIB.OBJ<RET>
.DELETE/NOQUERY  DY0:IBLIB.OBJ<RET>
```

4.4 Installing the Scientific Subroutines Package (SSP)

The SSP distribution media contain three different types of files needed for installing and verifying SSP. The files are:

- FORTRAN IV source files (file type .FOR), some containing SSP subroutines, and others containing test programs executed by the indirect command files
- Indirect command files (file type .COM), each of which verifies at least one SSP subroutine
- Data files (file type .DAT), which contain test data used by the test programs

The *Scientific Subroutines Programmer's Reference Manual* contains lists of these files. For an alphabetical list of the Scientific Subroutines, see Appendix C, Alphabetical Index of Subroutines, in that manual. For a list of indirect command files, related SSP source files, test program source files, and data files, see Table A–1, The Indirect Command Files, in that manual.

4.4.1 Preparing to Add SSP to Your System

Before you add SSP to your system:

1. See Table 4–1 to verify that you have the minimal software needed for installing FEP/FRP.
2. Choose the SSP subroutines you need. Add to your system and verify only those subroutines you actually need. SSP contains more than 100 subroutines; verifying all of them is a lengthy procedure. In addition, adding unneeded software to your development disk wastes space.
3. Write-enable drive 0 (if you have two RL01/RL02 disk drives).

   Write-protect drive 1 (if you have two RL01/RL02 disk drives).

   Bootstrap the development system disk in drive 0.
4. The verification procedure described in Section 4.4.2 requires you to assign input and temporary storage devices. To assign those devices, type:

```
.ASSIGN dv0: TM(RET)
.ASSIGN dv1: IN(RET)
```

   where `dv` is `DL` if you have RL01/RL02 disks or `DY` if you have RX02 diskettes

   The verification procedures use available space on the development system disk as temporary storage space. (When the verification procedure is complete, that space again is free.)

4.4.2 Verifying and Compiling SSP Subroutines

To verify an SSP subroutine:

1. Choose an SSP subroutine you plan to install. Mount in drive 1 the copied distribution disk containing that subroutine. (This disk also contains indirect command files, test programs, and data files needed to verify the subroutine.)
2. Consult Table A–1, The Indirect Command Files, in the *Scientific Subroutines Programmer's Reference Manual.* In that table, find the name of the subroutine you plan to install. Determine how many contiguous blocks must be on the development system disk for verifying the subroutine. If the disk does not have enough contiguous space, create more contiguous space by typing:

```
.SQUEEZE dv0:RET
```

where `dv` is `DL` if you have two RL01/RL02 disk drives or `DY` if you have only RX02 diskette drives

3. The verification procedure deletes all files on the development system disk named TMPSSP. Therefore, if you have any files on your development system disk with that name, rename them. (See the *RT–11 System User's Guide* for information on the RENAME command.)

4. Again, see Table A–1, The Indirect Command Files, in the *Scientific Subroutines Programmer's Reference Manual.* In that table, find the name of the subroutine you plan to install and the name of the indirect command file that verifies it. Run the indirect command file by typing:

```
.@dv1:fileRET
```

where `file` is the name of the indirect command file

The indirect command file runs a test program that performs computations and prints data and a message on your terminal. The indirect command file also deletes temporary files it has created.

5. If the test program executes successfully, it prints a confirmation message such as:

```
STOP -- file verification procedure successful
```

where `file` is the name of the indirect command file you ran

If that message has appeared, add the tested subroutine to the development system disk by typing:

```
.FORTRAN/OBJECT:dv0:  dv1:filenameRET
```

where `dv` is `DL` if you have two RL01/RL02 disk drives or `DY` if you have only RX02 diskette drives

`filename` is the name of the tested SSP subroutine

If the test program has printed an error message, determine the cause of the error. Some conditions that can cause errors are:

- A required program such as the FORTRAN IV compiler, the RT–11 linker, PIP, or DUP is not on your system device.
- You assigned the input device incorrectly. That is, the device you assigned to be the input device does not contain the volume where input files from the Scientific Subroutines reside.
- You assigned the output device incorrectly. That is, the device you assigned to be the output device does not have enough contiguous storage space.
- Your system device or your input or output device was write-protected when you tried to run an indirect command file.

If an indirect command file cannot run even though you have not made errors of this kind, you may have received a defective copy of the Scientific Subroutines software. If that is the case, contact your MINC phone support center for more information.

Verify and compile the other SSP subroutines you plan to use.

4.4.3 Completing the Installation of SSP

If you have two RL01/RL02 disk drives,see Section 5.6 for information on using the librarian to add files to SYSLIB after you have installed all needed FEP/FRP software. Follow the directions in Section 3.1.6 for preserving the system disk.

If you have only RX02 diskette drives, mount the FEP library diskette in drive 1. Use the COPY command to add the installed SSP object files to the FEP Library Diskette, then use the DELETE command to remove SSP object files from the development system diskette.

4.5 Installing the Laboratory Subroutine Package (LSP)

The FEP/FRP distribution kit contains standard LSP software as well as files for installing and customizing that software. To install the software, execute the LSP interactive build procedure, LSPMAK.SAV, which is described in Section 4.5.1. Select from the following types of options as you perform the procedure:

- Library options, which let you place the subroutines in a new library with a name of your choosing.
- Hardware options, which let you make use of your extended-arithmetic hardware.
- Subroutine and algorithmic options, which let you specify the subroutines and algorithmic options you need.

The interactive build procedure creates the following files on the development system disk:

- LSPCND.MAC, which sets switches that enable the options you have requested.
- LSPBLD.COM, an indirect command file that builds the subroutines. ("Building" means assembling subroutines with switches set to enable the algorithmic options you have chosen and optionally adding those subroutines to a library you have selected.)
- LSPVER.COM, an indirect command file that verifies the installed LSP software by running an example program for each assembled subroutine.

4.5.1 Building the LSP Software

To execute the LSP interactive build procedure:

1. See Table 4–1 to verify that you have the minimal software needed for installing FEP.

2. Write-enable drive 0 (if you have two RL01/RL02 disk drives).

   Write-protect drive 1 (if you have two RL01/RL02 disk drives).

   Mount the development system disk in drive 0.

   Mount the copy of the FEP/FRP distribution disk that contains LSP in drive 1.

3. The build procedure requires you to assign input and output devices. To assign those devices, type:

   ```
   .ASSIGN  dv0:  OU:(RET)
   .ASSIGN  dv1:  IN:(RET)
   ```

   where `dv` is `DL` if you have RL01/RL02 disks or `DY` if you have RX02 diskettes

4. Begin the interactive build procedure by typing:

   ```
   .RUN IN:LSPMAK(RET)
   ```

   Answer all questions displayed on your terminal during the procedure. If you have only RX02 diskette drives, answer Y to the question that asks whether you want to build a new library; name the library DY:LSPLIB.OBJ. Regardless of which type of disk drives you have, answer Y to the question that asks whether your machine has the EIS option or a floating point option. For a listing of the dialog, see Appendix C (Sample of the Interactive Build Procedure for RT–11, LSPMAK.SAV) in the *Laboratory Subroutines Programmer's Reference Manual.*

5. When the dialog is complete and your terminal displays the monitor dot, run the indirect command file LSPBLD.COM, which builds the software. Type:

   ```
   .@OU:LSPBLD(RET)
   ```

4.5.2 Verifying the LSP Software

To verify the LSP software:

1. Run the indirect command file, LSPVER.COM, which verifies each LSP subroutine built by the indirect command file, LSPBLD.COM. Type:

   ```
   .@OU:LSPVER(RET)
   ```

LSPVER executes example programs that test the installed LSP software. As each program is executed, the terminal displays the name of the program and some output. These programs and the correct output are listed throughout the *Laboratory Subroutines Programmer's Reference Manual.*

To determine whether your software is correctly installed, compare the results of LSPVER with the results of the example programs in the *Laboratory Subroutines Programmer's Reference Manual.* If your results are the same as those in that manual, LSP is correctly installed. If not, install the LSP software again.

4.5.3 Completing the Installation of LSP

If you have two RL01/RL02 disk drives, follow the instructions in Section 3.1.6 for preserving your development system disk.

If you have only RX02 diskette drives, add the installed LSP software to the FEP Library Diskette and remove that software from the development system diskette:

1. Remove the diskette from drive 1.

   Mount the FEP library diskette in drive 1.

2. Type:

```
.COPY  DY0:LSPLIB.OBJ  DY1:LSPLIB.OBJ(RET)
.DELETE/NOQUERY  DY0:LSPLIB.OBJ(RET)
```

4.6 Installing the FORTRAN Debugging Technique Software (FDT)

The FEP/FRP distribution kit contains three FDT files:

- FDT.MAC, the software in source form
- FDT.OBJ, the software in assembled form
- FDTVER.FOR, a FORTRAN source file used for testing FDT

4.6.1 Adding FDT to Your System

To install FDT:

1. See Table 4–1 to verify that you have the minimal software needed for installing FEP/FRP.
2. Bootstrap the development system disk in drive 0. Mount the copy of the FEP/FRP distribution disk that contains FDT in drive 1.

3. Copy the FDT object file to the development system disk by typing the following command:

```
.COPY  dv1:FDT.OBJ  dv0:FDT.OBJ<RET>
```

where dv is DL if you have two RL01/RL02 disk drives or DY if you have only RX02 diskette drives.

4.6.2 Verifying the FDT Software

To verify your FDT software:

1. Bootstrap the working system disk in drive 0.

   Mount the patched copy of the FEP/FRP distribution disk that contains FDTVER.FOR in drive 1.

2. Type the commands in the following dialog:

   (In the following instructions, for dv, type DL if you have RL01/RL02 disks; type DY if you have RX02 diskettes.)

```
.FORTRAN/CODE:THREADED/LIST:TT:/OBJECT:FDTVER dv1:FDTVER<RET>
FORTRAN IV    V02.5-11

0001           DATA B,C,J,M/1.,0.,1,1000/
0002        1  A=C
0003           N=M
0004        2  A=A+B
0005           N=N-1
0006           IF(N.GT.0) GO TO 2
0008           TYPE 1000,A,B,M
0009     1000  FORMAT(///' FINAL VALUE = ',1PE15.5,' INCREMENT = 'E15.5,
            1'  STEPS = ' I5,//)
0010           IF(J.NE.0) GO TO 1
0012           STOP 'FINISHED PROGRAM'
0013           END
.MAIN.


FORTRAN IV    Storage Map for Program Unit .MAIN.

Local Variables, .PSECT $DATA, Size = 000024 (   10. words)

Name  Type  Offset  Name  Type  Offset  Name  Type  Offset
A     R*4   000016  B     R*4   000002  C     R*4   000006
J     I*2   000012  M     I*2   000014  N     I*2   000022

.LINK FDTVER,FDT<RET>

.RUN FDTVER<RET>
FDT   V02-02

FDT PAUSE AT ISN 2 IN .MAIN.

!NAME J,12<RET>
!NAME M,14<RET>
!PAUSE ,10<RET>
!START<RET>

FINAL VALUE = 1.00000E+03 INCREMENT = 1.00000E+00 STEPS = 1000
```

```
FDT PAUSE AT ISN 10 IN .MAIN.
!ACCEPT M=5000RET

!CONTINUERET

FINAL VALUE = 5.00000E+03 INCREMENT = 1.00000E+00 STEPS = 5000

FDT PAUSE AT ISN 10 IN .MAIN.
!ACCEPT J=0RET

!CONTINUERET

STOP - FINISHED PROGRAM
```

3. Compare the computer responses printed by your printer with the computer responses in the preceding dialog. If the responses do not match and if you are sure you have correctly installed FDT, then you may have received unreliable software. Consult the MINC Product Services Center for further information.

4.6.3 Completing the Installation of FDT

If you have two RL01/RL02 disk drives, follow the instructions in Section 3.1.6 for preserving your development system disk.

If you have only RX02 diskette drives, add the installed FDT software to the FEP Library Diskette and remove that software from the development system diskette:

1. Remove the diskette from drive 1.

   Mount the FEP Library Diskette in drive 1.

2. Type:

```
.COPY  DY0:FDT.OBJ  DY1:FDT.OBJRET

.DELETE/NOQUERY  DY0:FDT.OBJRET
```

4.7 Installing the ReGIS Graphics Library for FEP Users (RGL/11)

RGL/11 is the ReGIS Graphics Library software for PDP–11s. This software enables a user to draw figures and plots on the VT125 terminal by means of subroutine calls in user-created FORTRAN programs. Information about the capabilities of the RGL/11 software, the syntax of subroutine calls, example programs, and so forth, is in a single manual called the RGL/11 Programmer's Reference Manual.

This manual contains only the RGL/11 installation procedures.

Installing RGL/11 involves copying the RGL/11 library and some other files from the distribution volume to your system disk, unless you have an RX02 system disk, in which case RGL/11 is installed on separate floppies. The installation procedure is usually done in three stages: (1) you copy the distribution volume to a backup disk, (2) using the copy you just made, you run the RGL/11 installation procedure, and (3) you verify that the installation was successful.

4.7.1 Background Information for Installing RGL/11 Files

When you install RGL/11, you need to know how the RGL/11 programs will be run: overlaid or non-overlaid. The next sections give background information for that decision.

4.7.1.1 RGL/11 Overlays — RGL/11 programs can be run in one of two ways: overlaid or non-overlaid. An overlaid program consists of two or more program segments. When an overlaid program is executed, only one program segment is in low memory at any one time. There are two kinds of overlays: disk resident and extended memory resident. If the program will run under an SJ or FB monitor, the segments are stored on disk. If it will run under XM, the segments can be stored on disk or in extended memory.

Overlays provide a way to execute a program that would otherwise exceed the size of memory available to you, and they are a way of trading execution speed for program size. Using overlays, programmers can build programs that are larger than the 32K word limit imposed by the PDP–11 architecture. But, when overlays involve disk I/O, as they do when they are disk-resident, the execution speed of an overlaid program may be slower than it would be if not overlaid.

One type of RGL/11 application program that usually must be overlaid because of size is one that calls many RGL/11 data-plotting subroutines. A program that calls only RGL/11 picture-drawing subroutines probably does not need to be overlaid.

Information on overlays is available in Section 11.4, Creating an Overlay Structure, in the *RT–11 System User's Guide* or the *PDP–11 FORTRAN Language Reference Manual.* Information about whether to use disk-resident or extended-memory-resident overlays is in Sections 4.4.1.2 and 4.4.1.3 of the *RGL/11 Programmer's Reference Manual.*

4.7.1.2 RGL/11 Files — When you install RGL/11, you install the appropriate files for the scheme or schemes selected for your site: non-overlaid, overlaid as either disk-resident or extended-memory-resident, or overlaid with an alternate extended-memory overlay scheme. Table 4–2 lists the files for each scheme.

Table 4–2: RGL/11 Files on Distribution Kit

Section 1 **Files Needed to Run Non–Overlaid RGL/11 Programs**		
RGLLIB.OBJ	BLODAT.OBJ	ERRTXT.TXT
GREEK.FNT	RGLLNK.COM	RGLLNK.COM
RGLVF1.FOR	RGLVF2.FOR	

Table 4–2: RGL/11 Files on Distribution Kit (Cont.)

Section 2 Files Needed to Run Standard (Disk–Resident or Extended–Memory) Overlaid RGL/11 Programs		
PRMLIB.OBJ	BLODAT.OBJ	ERRTXT.TXT
LFIXED.OBJ	LOCAT2.OBJ	LFREE.OBJ
LOCATE.OBJ	PPOINT.OBJ	PDATA.OBJ
DPAPER.OBJ	LNAXIS.OBJ	LTAXIS.OBJ
LNNICE.OBJ	LINMIN.OBJ	LINMAX.OBJ
PRINUM.OBJ	PRISTR.OBJ	TEXT.OBJ
XLABEL.OBJ	YLABEL.OBJ	FMINMX.OBJ
LGNICE.OBJ	EXPTST.OBJ	DPALOG.OBJ
DPALIN.OBJ	DEFALT.OBJ	DRWBOX.OBJ
LINE.OBJ	MARKER.OBJ	MOVE.OBJ
SWINDO.OBJ	GENOVR.OBJ	INITGR.OBJ
CLIPIT.OBJ	SSTATE.OBJ	GREEK.FNT
RGLLNK.COM	RGLLNK.SAV	RGLVF1.FOR
RGLVF2.FOR		

Table 4–2: RGL/11 Files on Distribution Kit (Cont.)

Section 3 Files Needed to Run RGL/11 Programs Using the Alternate Extended–Memory Overlay Scheme		
REXLIB.OBJ	BLODAT.OBJ	ERRTXT.TXT
LFIXED.OBJ	LOCAT2.OBJ	LFREE.OBJ
LOCATE.OBJ	PPOINT.OBJ	PDATA.OBJ
DPAPER.OBJ	LNAXIS.OBJ	LTAXIS.OBJ
LNNICE.OBJ	LINMIN.OBJ	LINMAX.OBJ
PRINUM.OBJ	PRISTR.OBJ	TEXT.OBJ
XLABEL.OBJ	YLABEL.OBJ	FMINMX.OBJ
LGNICE.OBJ	EXPTST.OBJ	DPALOG.OBJ
DPALIN.OBJ	DEFALT.OBJ	DRWBOX.OBJ
LINE.OBJ	MARKER.OBJ	MOVE.OBJ
SWINDO.OBJ	GENOVR.OBJ	INITGR.OBJ
CLIPIT.OBJ	SSTATE.OBJ	ARCOVR.OBJ
RELOVR.OBJ	SAVOVR.OBJ	GN2OVR.OBJ
GREEK.FNT	RGLOVR.COM	RGLVFY.COM
RGLVF1.FOR	RGLVF2.FOR	

In each category, the first file listed is a library:

- RGLLIB, the library you use with non-overlaid programs
- PRMLIB, the library you use with overlaid programs, either disk-resident or extended-memory-resident
- REXLIB, the library you use for the alternate extended-memory overlay scheme

You can install any or all of these libraries during the same session. You will notice that some files are mentioned in each of the three categories.If you copy more than one category, only one version of a file will be copied to your media.

4.7.2 Installation of RGL/11 Software

The procedure for installing RGL/11 varies, depending upon your distribution media:

- If your distribution media is a hard disk (RL01 or RL02), follow the instructions in Section 4.7.2.1.
- If your distribution media is RX02 floppy diskettes, follow the instructions in Section 4.7.2.2.

4.7.2.1 Hard Disk Distribution Kits — To install RGL/11, your system must have either 2 RL01 disk drives or 2 RL02 disk drives. Install the RGL/11 software by following these steps:

1. Bootstrap the system disk; these examples assume it is bootstrapped in drive 0.

   Type the following command, which prevents the monitor from displaying commands in indirect command files:

   ```
   .SET TT QUIET RET
   ```

2. Mount the copied RGL/11 distribution disk in drive 1 and write-protect it.

   Assign input and output devices by typing:

   ```
   .ASSIGN DL1: IN: RET

   where dd1 is the device containing your
   RGL/11 distribution media

   .ASSIGN DL0: OU: RET

   where dd0 is the device containing your system disk
   ```

3. Run the indirect command file that installs the basic RGL/11 software for overlaid and non-overlaid programs by typing:

   ```
   .@IN:RGLINS RET
   ```

   This command file copies RGLLIB and the files common to all three sections of Table 4–2 to your system disk.

4. If you decided to execute the RGL/11 software with overlaid programs (either disk-resident or extended-memory-resident), run the indirect command file that installs the RGL/11 software related to overlays by typing:

   ```
   .@IN:RGLINS.OVR RET
   ```

   This command file copies PRMLIB, REXLIB, and all other files in Sections 2 and 3 of Table 4–2.

5. Now you are ready to verify the RGL/11 software by following procedures described in Section 4.7.3.

In addition to the files copied during installation, the RGL/11 distribution media contains RGL/11 object files and source files supplied for backup purposes.

4.7.2.2 RX02 Floppy Diskette Distribution Kits — Copying the diskettes as described in Section 4.1.1 is all you needed to do for installation. The contents of the disks are as follows:

1. Disk 1/5 (first of five) contains RGLLIB and all the files needed to run non-overlaid RGL/11 programs (see the first section of Table 4–2).
2. Disk 2/5 contains PRMLIB and all the files needed to run standard overlaid (disk-resident or extended-memory-resident) RGL/11 programs (see Section 2 of Table 4–2).
3. Disk 3/5 contains REXLIB and all the files needed to run RGL/11 programs with the alternate overlay scheme, as shown in Section 3 of Table 4–2 and described in Section 4.3.2.3 of the *RGL/11 Programmer's Reference Manual.*
4. Disks 4/5 and 5/5 contain RGL/11 object files and source files supplied for backup purposes.

Proceed to Section 4.7.3, Verification.

4.7.3 Verification

When you have installed the RGL/11 software according to the directions given in the previous section, the software is ready for verification. Follow the instructions listed below for running the RGL/11 verification programs.

Remember that RGL/11 verification must be done from a VT125 terminal.

To verify the RGL/11 software:

1. Be sure you have bootstrapped the system disk. These examples assume it is bootstrapped in drive 0.
2. If you are using the FB or XM monitor or SJ with multi-terminal support, type the following command, which must be in effect whenever you execute a program that uses RGL/11 software:

```
.SET TT NOCRLF RET
```

3. If you have an RX02 distribution kit, follow these steps (otherwise, continue at step 4):

   - Be sure that you have mounted in drive 1 the copied RGL/11 distribution diskette you plan to use. If you are not planning to use overlays, the diskette you want is labeled 1/5 (first of the five distribution diskettes). RGLLIB and all other files from Section 1 of Table 4–2 are on this diskette. If you are planning to use standard overlays (either disk-resident or extended-memory-resident), the diskette you want is labeled 2/5 (the second of five). PRMLIB and all other files from Section 2 are on this diskette. If you are planning on using the alternate overlay scheme described in Section 4.3.2.3 of the *RGL/11 Programmer's Reference Manual*, the diskette you want is labeled 3/5. REXLIB and all other files from Section 3 are on this diskette.

   - Type:

     ```
     >ASSIGN DY1: DK:RET
     ```

   Go to step 5 to verify RGLLIB or PRMLIB; go to step 6 to verify REXLIB.

4. If you have a hard disk distribution kit, assign your system disk to be your default disk. Type:

   ```
   >ASSIGN DL0: DK: RET

   where DL0 is the device name and number from which you
   bootstrapped your system
   ```

   Go to step 5 to verify RGLLIB or PRMLIB; go to step 6 to verify REXLIB.

5. This step verifies RGLLIB or PRMLIB. Follow this step if you are planning to run either non-overlaid or standard overlaid programs (disk-resident or extended-memory-resident). If you have an RX02 distribution disk, you must have RGL/11 disk 1/5 or 2/5 mounted to run this step. Do not follow this step if you are planning to use the alternate overlay scheme and REXLIB. To verify REXLIB, follow step 6.

   Execute the indirect command file, RGLLNK.COM, to compile, link, and run the two verification programs, RGLVF1.FOR and RGLVF2.FOR. (Information on the RGLLNK program is available in Chapter 4 of the *RGL/11 Programmer's Reference Manual.*) Type:

   ```
   >@RGLLNKRET
   ```

When RGLLNK asks you for a program name, specify RGLVF1. RGLLNK also asks:

```
Do you want to compile [Y] ?
Do you want listings [N] ?
What device do you want the listings to go to [LP:] ?
Do you want to link [Y] ?
Do you want a map [N] ?
What device do you want the listings to go to [LP:] ?
Do you want overlays [Y] ?
Do you want extended memory overlays [N] ?
Do you want to run [Y] ?
```

If you have installed the software needed to run RGL/11 programs with standard overlays, answer Yes to the first question about overlays. If you have an RX02 distribution kit and you have RGL/11 disk 2/5 mounted, you must answer Yes to this question. (Answering Yes to the overlay question verifies PRMLIB.) If you have installed the software needed to run RGL/11 programs non-overlaid, answer No to that question. (Answering No verifies RGLLIB.) If you have an RX02 distribution kit and you have RGL/11 disk 1/5 mounted, you must answer No to this question.

If you are booted under the XM monitor and have installed the software needed to run RGL/11 programs with overlays, you can answer Yes to both overlay questions. (Answering Yes to both overlay questions verifies PRMLIB, but the overlays are extended-memory-resident rather than disk-resident.)

While RGLVF1 is running, read step 7, which describes the verification programs. Then execute RGLLNK.COM again, specifying RGLVF2 for the program name. If you have an RX02 distribution kit, before running RGLVF2, delete RGLVF1.OBJ, RGLVF1.SAV, and VFY.TMP to provide enough space for the files RGLVF2 creates.

6. Follow this step if you are planning to use the alternate overlay scheme. If you have an RX02 distribution kit, you must have RGL/11 disk 3/5 mounted to run this step. To verify the REXLIB library, which will be used for the alternate overlay scheme, you must be booted under the XM monitor. Type:

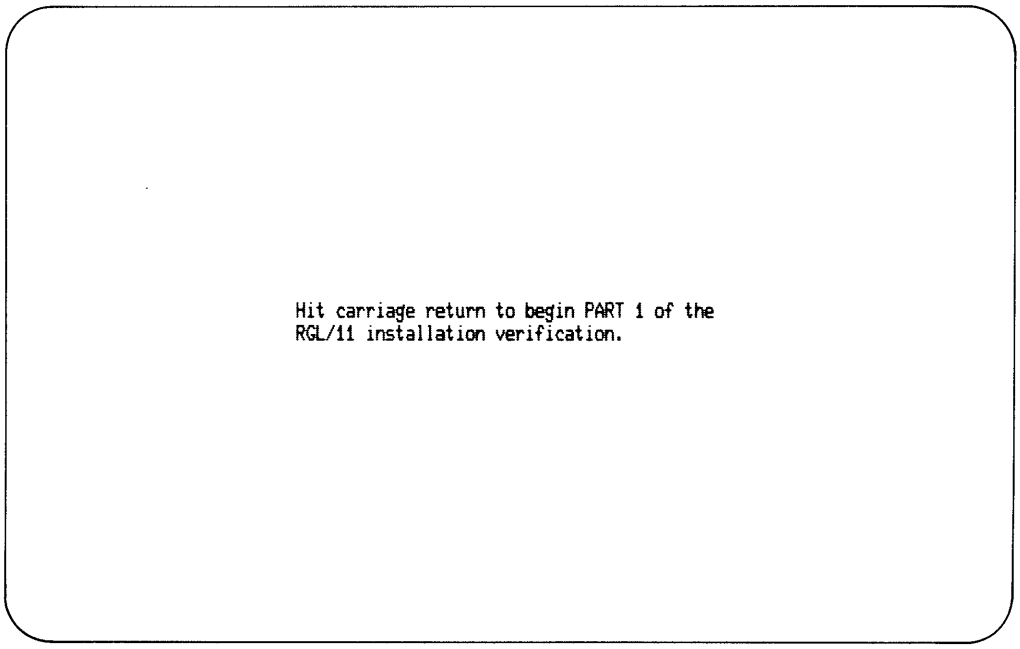
```
.@RGLVFY RET
```

When you invoke RGLVFY, it compiles, links, and runs both verification programs, RGLVF1.FOR and RGLVF2.FOR.

While RGLVF1 and RGLVF2 are running, read step 7, which describes the verification programs. At the end of RGLVF1, RGLVF2 is compiled and linked, so there is a pause before RGLVF2 is run.

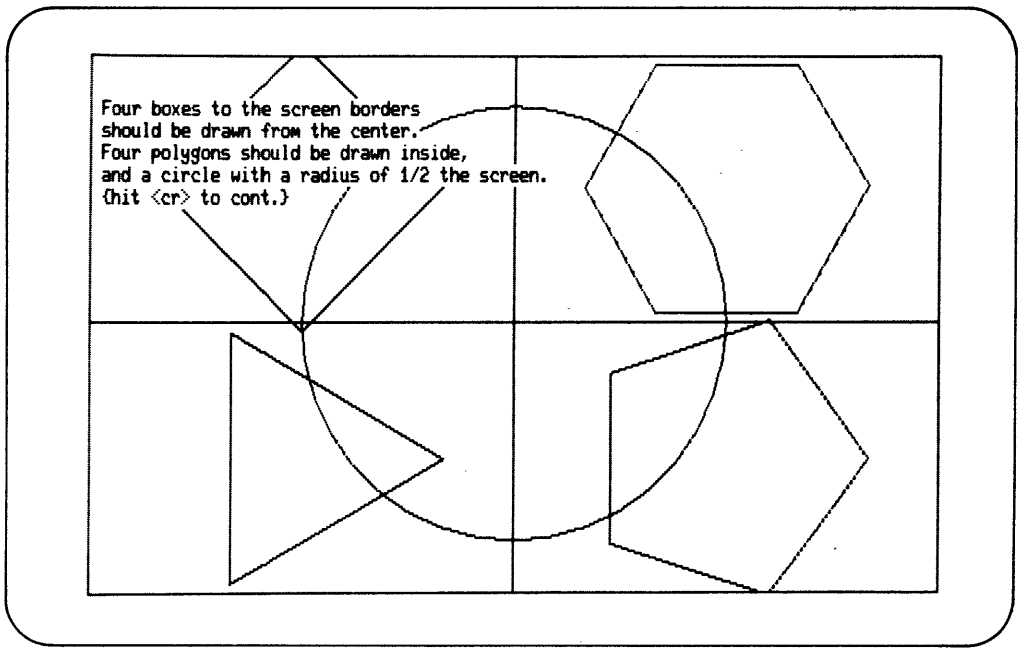
7. The verification programs, RGLVF1 and RGLVF2, call some subroutines from the RGL/11 library to verify that the subroutines work correctly. The verification programs draw graphic illustrations on the terminal screen to demonstrate the subroutine called, display a few lines explaining what the subroutine does, and then wait for you to type (RET) before they continue. Figures 4–2 through 4–12 show the pictures the verification programs draw.

Figure 4–2: Verification Program — Screen 1



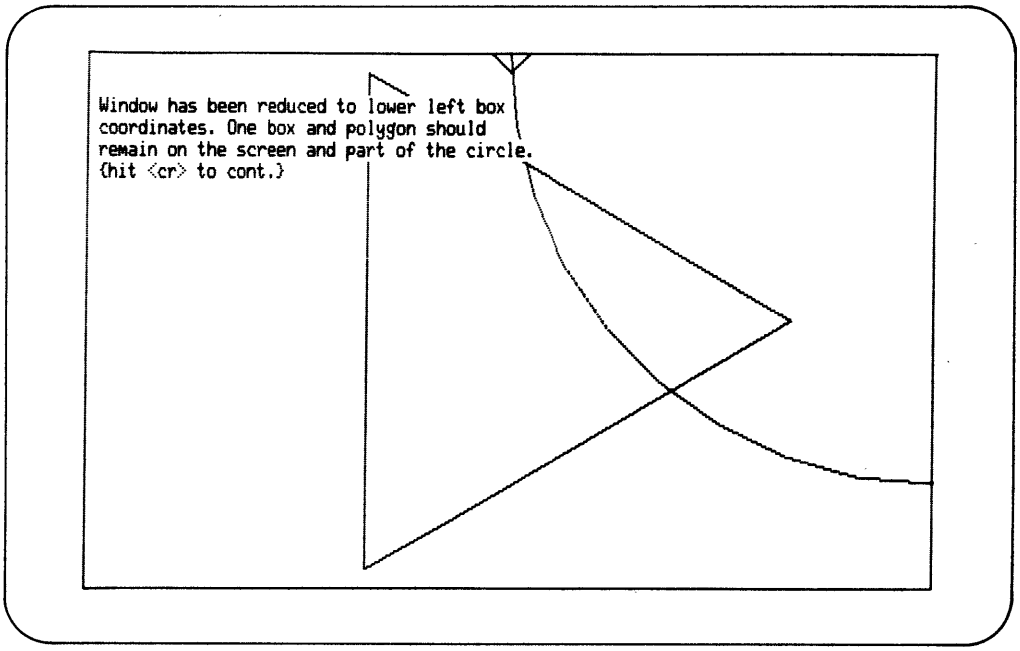
MR-S-2200-82

Figure 4–3: Verification Program — Screen 2



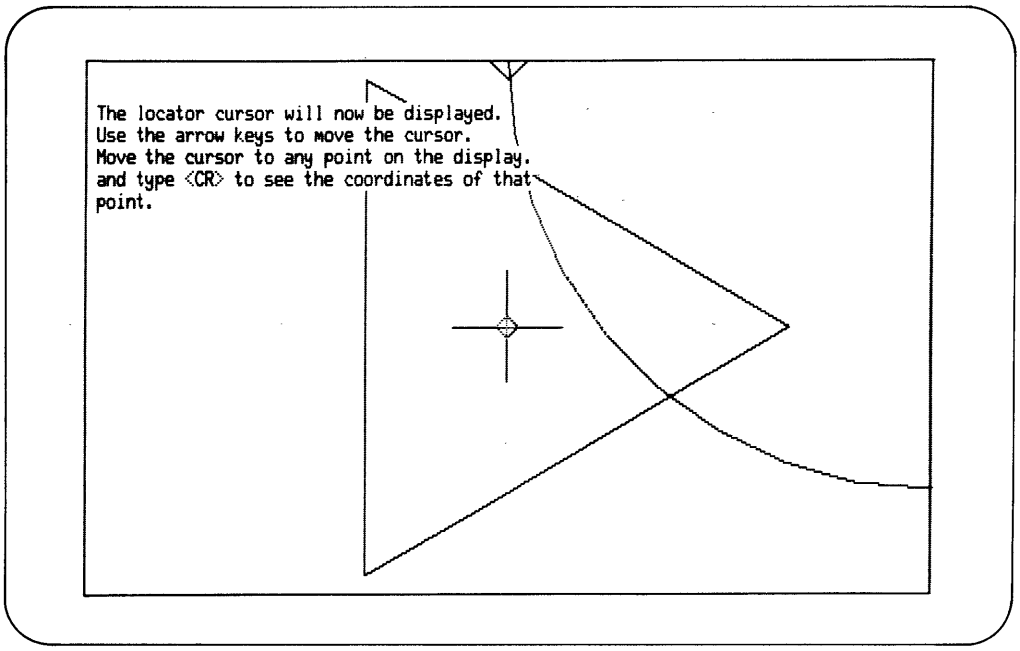
MR-S-2201-82

Figure 4–4: Verification Program — Screen 3



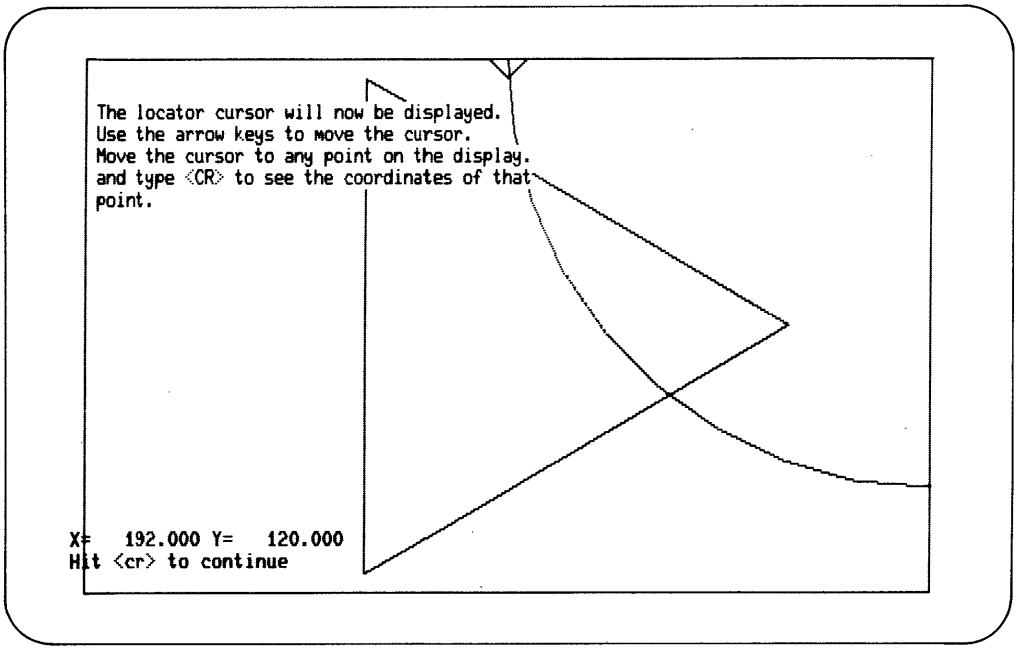
MR-S-2202-82

Figure 4–5: Verification Program — Screen 4



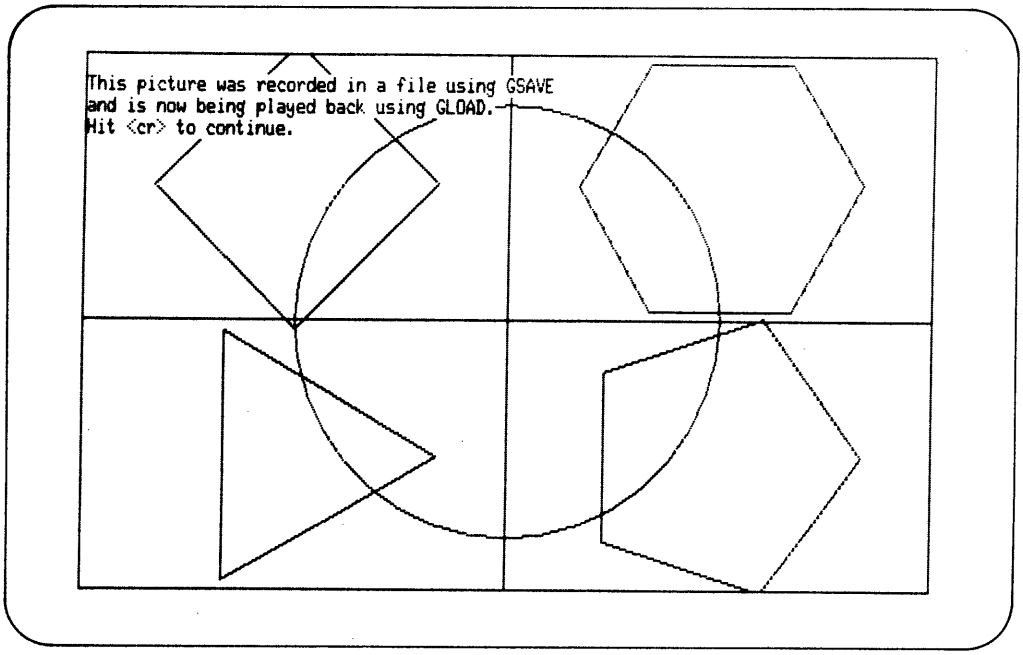
MR-S-2203-82

Figure 4–6: Verification Program — Screen 5



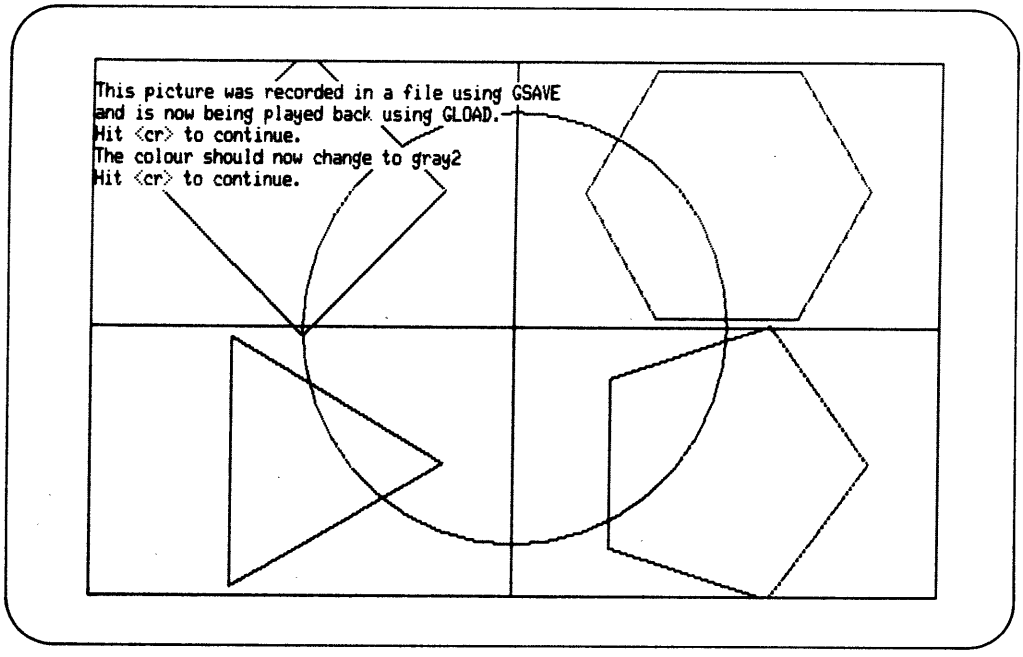
MR-S-2204-82

Figure 4–7: Verification Program — Screen 6



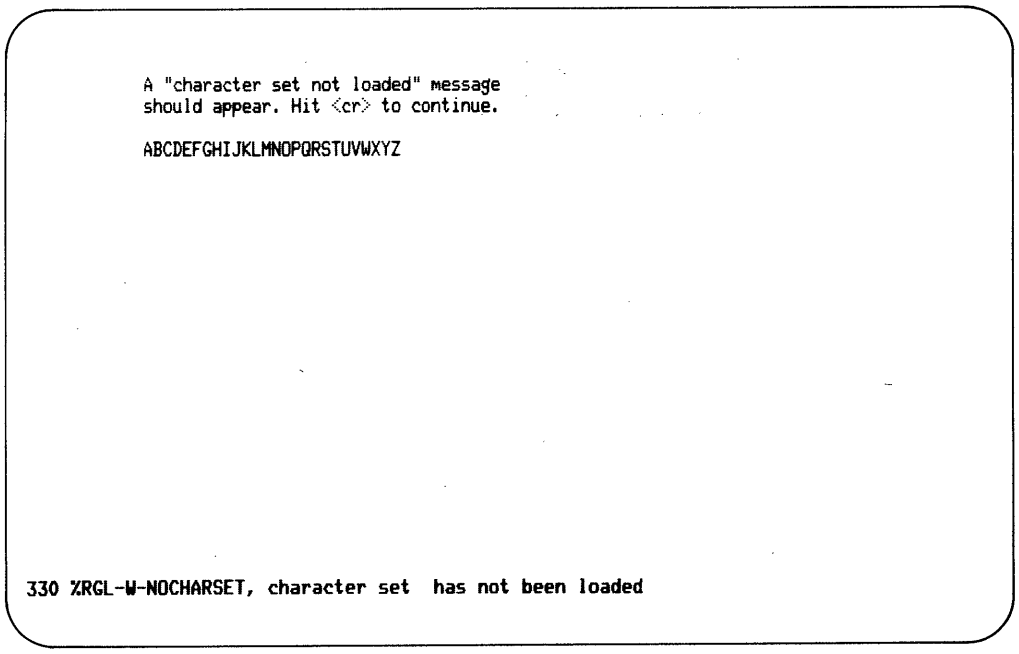
MR-S-2205-82

Figure 4–8: Verification Program — Screen 7



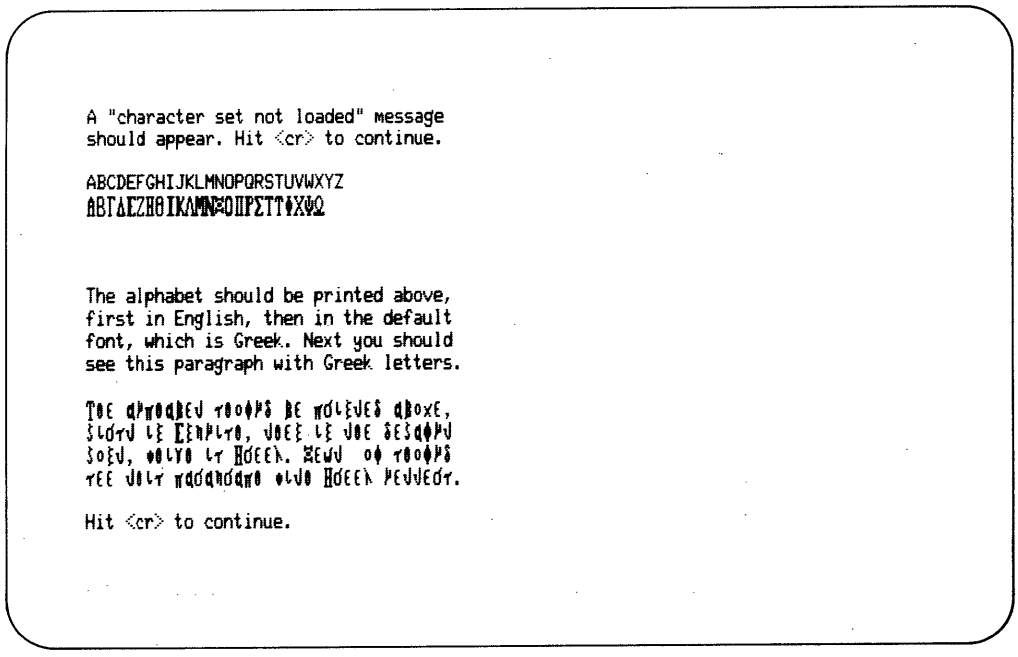
MR-S-2206-82

Figure 4–9: Verification Program — Screen 8



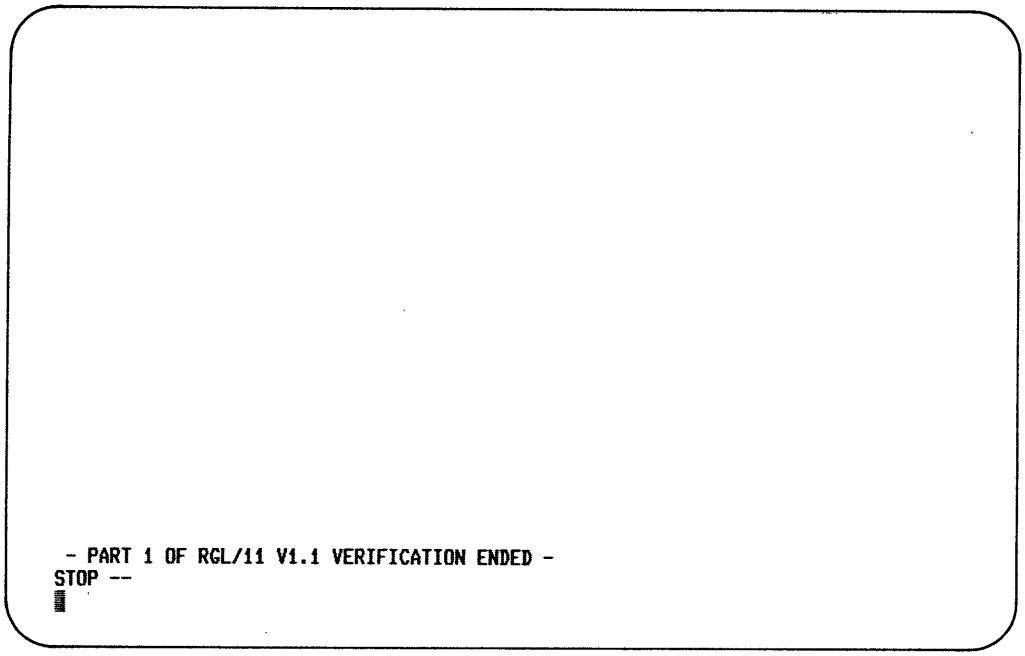
MR-S-2207-82

Figure 4–10: Verification Program — Screen 9



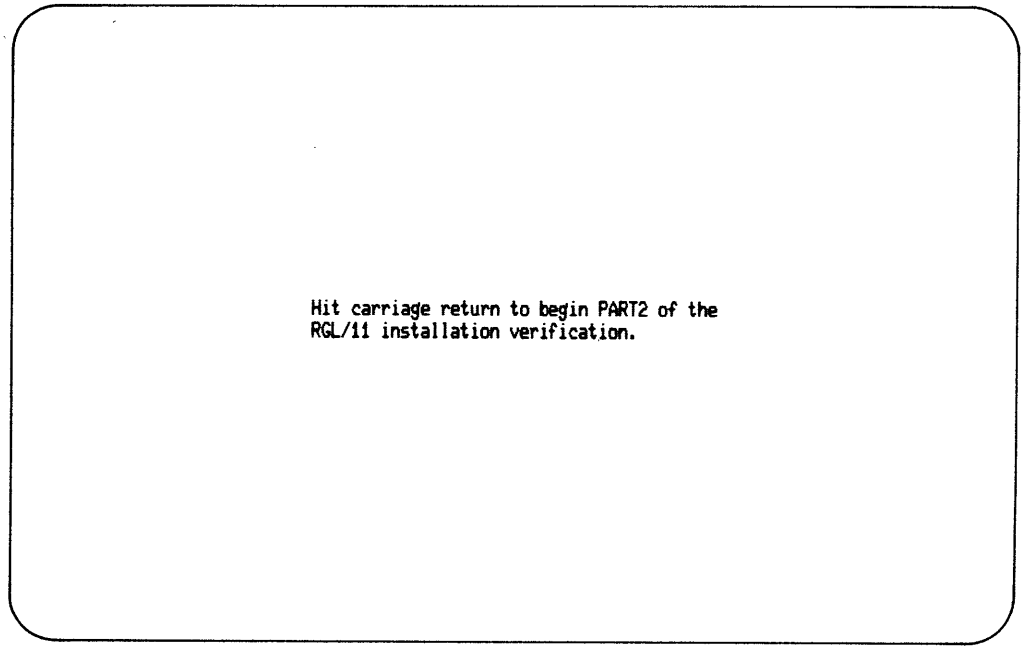
MR-S-2208-82

Figure 4–11: Verification Program — Screen 10



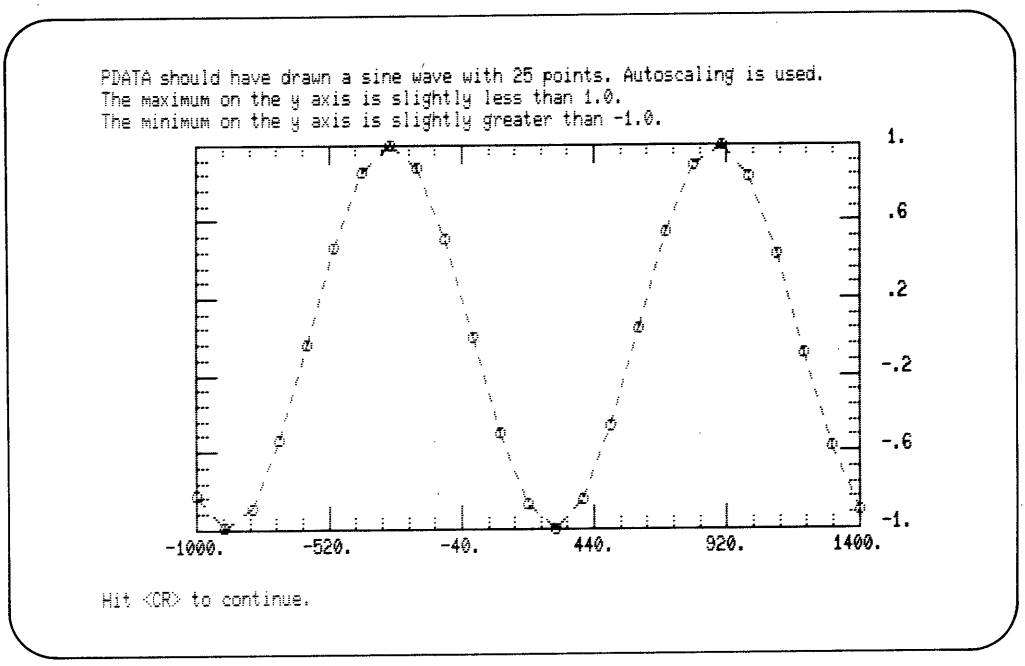
MR-S-2209-82

Figure 4–12: Verification Program — Screen 11



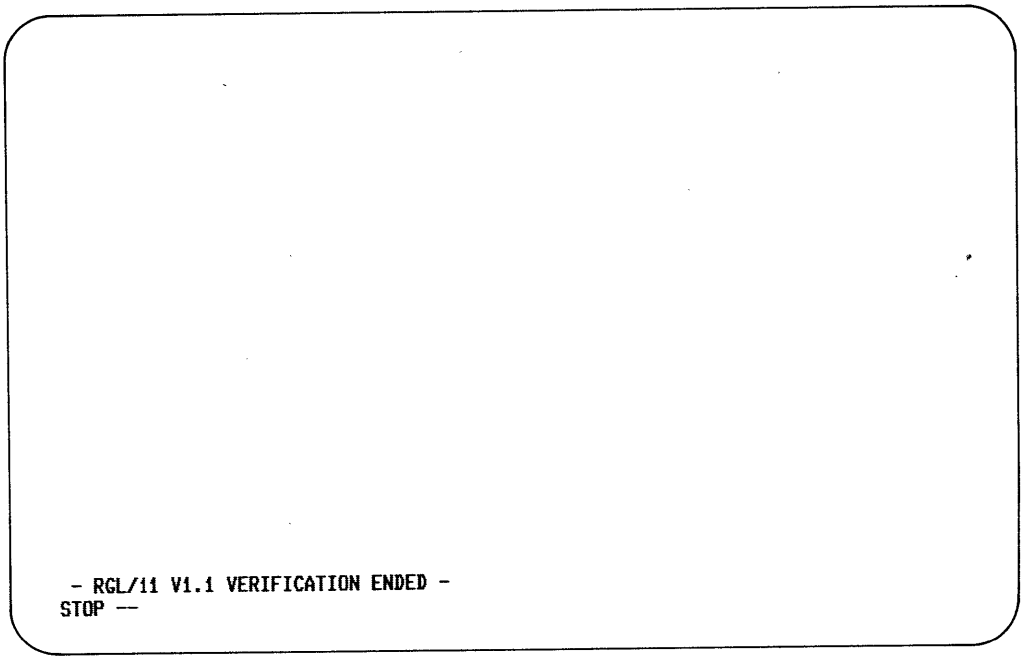
MR-S-2210-82

Figure 4–13: Verification Program — Screen 12



MR-S-2211-82

Figure 4–14: Verification Program — Screen 13



MR-S-2212-82

When RGLVF1 finishes, it displays the message displayed in Figure 4–11:

When RGLVF2 finishes, it displays the message displayed in Figure 4–14:

If you receive the completion messages, the RGL/11 library is properly installed.

If you do not receive the completion messages and you cannot finish the verification procedure, exit the verification program by typing CTRL/C. Then reset the terminal by pressing the SET–UP key and the RESET key (the zero key on the main keyboard). Install RGL/11 again and run the verification programs again from the VT125. If they again fail to complete, contact DIGITAL for assistance.

8. To verify any other library that you have installed, repeat the whole verification procedure for that library.

   If you have an RX02 distribution kit, before you run a new verification, you should delete RGLVF1.OBJ, RGLVF2.OBJ, RGLVF1.SAV, RGLVF2.SAV, and VFY.TMP to ensure enough space on the disk for the files the verification procedure creates.

4.8 Deleting the MACRO Assembler

If you have only RX02 diskette drives and are not planning to use the MACRO assembler, you can delete the files MACRO.SAV and SYSMAC.SML from your development system diskette. Doing so will free disk space occupied by those files.

NOTE

Installation procedures for FEP/FRP components use the MACRO assembler. Therefore, any time you need to rebuild a FEP/FRP component, be sure that MACRO.SAV and SYSMAC.SML are on your development diskette.

To delete the MACRO assembler, type the commands in the following dialog:

```
.DEL MACRO.SAV,SYSMAC.SML<RET>
 Files deleted:
DK:SYSMAC.SML? Y<RET>
DK:MACRO.SAV? Y<RET>
```

Part III
Using the Software

Part III contains chapters on program development, system precautions and use of the RGL/11 and PLOT55 software.

Chapter 5
Program Development

Once you have installed RT–11, FORTRAN IV, and the FEP/FRP components you need, you are ready to write your own FORTRAN programs. This chapter introduces you to techniques for developing programs and guides you to other documentation of program development techniques.

NOTE

All RT–11 commands in this chapter are presented in their simplest keyboard-command forms. To see other forms of these commands, consult documentation recommended in this chapter.

5.1 Terminals

Read Section 5.1.1 if you have a video terminal; read Section 5.1.2 if you have a hard-copy terminal.

5.1.1 Video Terminal

You can alter the video terminal's performance. You can set some terminal characteristics before using the terminal and can make other settings while the terminal is in use. This section describes how to make these settings.

RT–11 requires you to set AUTO XOFF/XON. You should already have made this setting. (See Section 2.1.)

If you have a VT105 terminal and plan to use PLOT55, put the terminal in VT52 mode before using PLOT55. (See Chapter 8.)

When the video terminal displays a file, it displays data continuously until the end of the file is reached. The terminal provides two mechanisms for altering this continuous display. One mechanism temporarily halts the display. The other mechanism suppresses the display.

To temporarily halt the display, press NO SCROLL , which is located at the bottom left of the keyboard. To cause the display to resume, again press NO SCROLL . The display resumes where it left off when you first pressed NO SCROLL . No data has been lost.

To suppress the display of data, type CTRL/O. To cause the display to resume, again type CTRL/O. The display resumes, but all data transmitted to the terminal since you first typed CTRL/O is lost.

For further information, see the *VT125 User's Guide* or the *VT105 Graphic Terminal User's Manual.*

5.1.2 Hard-copy Terminal

You can alter the performance of the terminal while it is in use. When the terminal displays a file, it displays data continuously until the end of the file is reached. The terminal provides two mechanisms for altering this continuous display. One mechanism temporarily halts the display. The other mechanism suppresses the display.

To temporarily halt the display, type CTRL/S. To cause the display to resume, type CTRL/Q. The display resumes where it left off when you typed CTRL/S. No data has been lost.

To suppress the display of data, type CTRL/O. To cause the display to resume, again type CTRL/O. The display resumes, but all data transmitted to the terminal since you first typed CTRL/O is lost.

For further information, see the *DECLAB–11/MNC User's Guide.*

5.2 Choosing a Text Editor

RT–11 provides two text editors, EDIT and KED, that are commonly used for creating and editing source programs. EDIT is designed for use with any terminal. KED is designed for use only with VT100–series terminals and makes efficient use of video-terminal hardware features.

NOTE

If you use KED with a VT105 terminal, be sure that terminal is not in VT52 mode (see Chapter 8).

If you have a hard-copy terminal, use EDIT.

If you have a video terminal, use KED, and be sure that your startup indirect command file contains the commands that cause the interactive EDIT commands to invoke KED. These commands are:

```
SET TT SCOPE
SET EDIT KED
```

5.3 Creating a Program

To create a file for a FORTRAN program, type:

```
.EDIT/CREATE  prog.FOR↵
```

where `prog` is the name of your program.

If you have a video terminal, RT–11 will execute KED. For detailed information on the KED editor, consult the *PDP–11 Keypad Editor User's Guide.*

If you have a hard-copy terminal, RT–11 will execute EDIT. For detailed information on EDIT, consult the *RT–11 System User's Guide.*

As you type your FORTRAN IV program, be sure that the program lines have a maximum length of 72 characters. Do so because the compiler, unless instructed otherwise, does not process characters beyond position 72. The presence of any characters beyond that position can cause the compiler to print confusing error messages and can cause your program to execute incorrectly even though it has been compiled correctly. (If you have a VT100–series terminal, you can enable the margin bell to sound when you reach position 72. See SETUP mode in the *VT105 Graphic Terminal User's Manual* or the *VT125 User's Guide* for instructions on enabling the margin bell.)

Before creating a program that calls RGL/11 subprograms, read Section 7.1, RGL/11 Files Needed for Program Development.

5.4 Compiling a Program

When you are satisfied that you have accurately entered your source program, exit from the editor and compile the program by typing:

```
.FORTRAN prog↵
```

where `prog` is the name of your program. Consult the *RT–11 System User's Guide* for more information on the interactive FORTRAN command.

The FORTRAN compiler attempts to translate your source program into an object program (machine language instructions). If the compiler encounters syntax errors in your source program, it prints error messages. For help in understanding these messages and in removing syntax errors from your program, see the *RT–11,RSTS/E FORTRAN IV User's Guide* or the *RT–11 System Message Manual.*

If the compiler encounters no syntax errors in your source program, the compiler produces an object file, prog.OBJ. Your disk then contains two versions of the program:

- prog.FOR, the source program
- prog.OBJ, the object program

Note that the compiler only determines whether or not each statement in your program can be executed: the compiler finds syntax errors but does not find errors in program logic. Therefore, after you have written your program, read it carefully to be certain that when you execute the program, it will do what you want it to do.

The following example contains a logic error but no syntax errors. Even though the compiler finds these statements free of syntax errors, the statements will produce undesirable results when they are executed:

```
1 I=1
  GO TO 1
2 CONTINUE
```

5.5 Linking a Program

The linker associates your program with the subprograms that your program calls. These subprograms may be:

- In SYSLIB.OBJ (the default system library)
- In other libraries named in the linker command string
- Individual object modules named in the linker command string

NOTE

Before linking programs that use the REAL–11/MNC software, see Section 6.5, Special Considerations in Linking REAL–11/MNC. If you are using the XM monitor, also see Section 6.6, Using REAL–11/MNC Software under the XM Monitor.

Before linking a program that calls RGL/11 software, see Section 7.3, Linking a Program that calls the RGL/11 Software.

When you have successfully compiled your program, link it by typing the following command string:

```
.LINK prog,filespecs
```

where `prog` is the name of your program and `filespecs` are the names of libraries or other object files separated by commas.

The linker attempts to translate your object program into an executable load module. However, if the linker encounters inadequately defined variables, it may not produce a load module and also prints an error message such as:

```
?LINK-W-Undefined globals
```

If you encounter this error message, you probably have forgotten to include the names of one or more libraries or object modules in the linker command string. If this is the case, invoke the linker again, and be sure to include all necessary files in the linker command string.

If the linker encounters no fatal errors, it produces an executable load module called prog.SAV. Your disk then contains three versions of your program:

- prog.FOR, the source program
- prog.OBJ, the object program
- prog.SAV, the load module

For further information on the linker, see the *RT–11 System User's Guide.*

5.6 Using the Librarian

The linker automatically searches SYSLIB.OBJ for variables called global symbols that remain unresolved after the linker has searched the files you have specified. For this reason you should use the librarian to add frequently used subprograms to SYSLIB.OBJ. Doing so simplifies use of the linker. For example, suppose your program, MYPROG.OBJ, calls subprograms contained in MNCLIB.OBJ, and MYLIB.OBJ. To link MYPROG, you would type:

```
. LINK MYPROG,MNCLIB,MYLIB(RET)
```

However, if you added MNCLIB and MYLIB to the default library SYSLIB.OBJ, you could link MYPROG by typing:

```
. LINK MYPROG(RET)
```

5.6.1 Adding Files to SYSLIB

After you have installed all needed FEP/FRP components, you may want to use the librarian to add them to SYSLIB. As you do so, you must also remove certain global symbols from the library directory for the librarian to function properly. To add FEP/FRP component libraries to SYSLIB, use the LIBRARY keyboard command with the REMOVE option. Type:

```
.LIBRARY/REMOVE SYSLIB filea[,fileb,...,filez]RET
GLOBAL? globalaRET
GLOBAL? globalbRET
          .
          .
          .
GLOBAL? globalzRET
GLOBAL? RET
```

where `filea` through `filez` are the names of object (OBJ.) files, such as libraries of FEP/FRP components, that are to be added to SYSLIB

`globala` through `globalz` are global symbols to be removed from the library directory

One global symbol you must remove every time you add files to SYSLIB is $OVRH. You must also remove $ERRS and $ERRTB if you have added the FORTRAN OTS library to SYSLIB. (See Sections 3.1.4 and 3.2.4 for information on adding the FORTRAN OTS to SYSLIB.) If you are adding IBLIB.OBJ to SYSLIB, also remove the global symbol IB$ERR. If you fail to remove those global symbols from the SYSLIB directory, you will receive an error message of the following type as you attempt to add files to SYSLIB:

```
?LIBR-W-Illegal insert of globala
```

where `globala` is a global symbol that should be removed

If you forget which global symbols to remove, create a dummy library file by typing a command such as the following:

```
.LIBRARY/CREATE dummy SYSLIBRET
```

where `dummy` is the name of a dummy library file

Execution of this command will produce error messages that list global symbols. These are the symbols that should be removed. After creating the dummy library file, delete it and type the correct command for adding files to SYSLIB.

5.6.2 Using the Librarian with Two RX02 Diskette Drives

If you have enough room on your development system diskette, you can add the installed FEP/FRP Laboratory Software components to SYSLIB on that diskette. To do so, follow instructions listed in this section. If your development system diskette has too little free space to accomodate an enlarged SYSLIB, you may want to create a single library of FEP Laboratory Software components on the FEP Library Diskette. To do so, follow instructions for using the LIBRARY keyboard command in the *RT–11 System User's Guide.*

When adding files to an existing library, such as SYSLIB, the librarian builds a new copy of the library that incorporates the new files, then deletes the old copy of the library. For this reason the disk that is to contain the expanded library must have enough contiguous free space available to accommodate the expanded library. When the desired output diskette has too little free space, you can use an extra diskette for temporary storage of the expanded library. One way to do so is:

1. Place SYSLIB and all files to be added to it on the development system diskette.
2. Preserve the development system diskette by following the instructions in Section 3.2.6.
3. Mount a formatted, initialized diskette in DY1:
4. Use the OBJECT option when executing the librarian. The OBJECT option allows you to specify an output disk on which the system will place the new library. For example, to add libraries of REAL–11/MNC and LSP files to SYSLIB on DY1:, type:

```
.LIBRARY/REMOVE/OBJECT:DY1:SYSLIB  SYSLIB,MNCMUL,IBLIB<RET>
GLOBAL? globala<RET>
GLOBAL? globalb<RET>
      .
      .
      .
GLOBAL? globalz<RET>
GLOBAL? <RET>
```

   where `globala` through `globalz` are global symbols to be removed from the library directory

5. Create free space on drive 0 by deleting both SYSLIB and the files you have added to the copy of SYSLIB that is on drive 1. Consolidate this free space on drive 0 by using the SQUEEZE command.
6. Copy the new SYSLIB from DY1: to DY0:

```
.COPY DY1:SYSLIB DY0:SYSLIB<RET>
```

5.7 Executing a Program

When you have successfully linked your program, execute it by typing:

```
.RUN prog
```

where `prog` is the name of your program. However, before executing a program that calls RGL/11 subprograms, see Section 7.4, Running a Program that Calls RGL/11 Software.

RT–11 attempts to execute the load module, prog.SAV. If RT–11 encounters a condition that prohibits it from executing your program, RT–11 prints a FORTRAN OTS (Object Time System) error message. For help in understanding FORTRAN OTS error messages, see the *RT–11,RSTS/E FORTRAN IV User's Guide.*

When you have determined what part of your program has produced the error during execution, repeat all the steps listed in this chapter:

1. Use the editor to correct the mistakes in your source program. (The editor changes the name of prog.FOR to prog.BAK and gives the name prog.FOR to the new file that contains your corrections.)
2. Compile the source program.
3. Link the object program.
4. Execute the load module.

To help find program errors, you can use the FORTRAN Debugging Technique (FDT). To use FDT:

1. Compile your program and produce threaded code by typing:

   ```
   .FORTRAN/LIST  prog/CODE:THR
   ```

2. Link your program and include FDT in the linker command string, for example:

   ```
   .LINK  prog,FDT
   ```

3. Execute your program by typing:

   ```
   .RUN  prog
   ```

See the *FORTRAN Debugging Technique Reference Manual* for detailed information on using FDT. For discussions on threaded FORTRAN object code, see the *RT–11 User's Guide* and the *RT–11,RSTS/E FORTRAN IV User's Guide.*

An alternate way to execute FORTRAN IV programs is to use the EXECUTE command in place of the FORTRAN, LINK, and RUN commands. The EXECUTE command combines the functions of those commands. To use the EXECUTE command, type:

```
.EXECUTE  prog[/LINKLIBRARY:filespecs]RET
```

where `prog` is the name of your program.

filespecs are the names of libraries or other object files separated by commas.

5.8 Aids for Program Development

The RT–11/FEP software includes two aids for program development. These aids are PREP.COM, an indirect command file, and PROGD.FOR, an interactive program that creates an indirect command file. Both PREP.COM and the indirect command file created by PROGD.FOR contain command lines that execute instructions useful during program development.

5.8.1 Indirect Command File for Program Development

PREP.COM is an indirect command file that serves as a model for the compile, assemble, link, and run phases of program development. This file contains a series of command lines that you can use to perform each of those phases. The command lines illustrate many options and capabilities of the FORTRAN IV compiler, MACRO assembler, and RT–11 linker.

To create a copy of PREP.COM that you can use for program development:

1. If you have two RL01/RL02 disk drives, mount the copied FEP distribution disk in drive 0. If you have only RX02 diskettes, mount the copied FEP Installation Diskette in drive 0.

   In drive 1, mount a system disk that contains the FORTRAN IV compiler (FORTRA.SAV) and OTS (SYSLIB.OBJ—if SYSLIB includes the OTS—or FORLIB.OBJ—if SYSLIB does not include the OTS).

2. Copy PREP.COM to the disk in drive 1 by typing:

   ```
   .COPY  DK0:PREP.COM  DK1:filename.COMRET
   ```

   where `filename` is the name of the copy of PREP.COM. (You can conveniently create and modify a copy of PREP.COM for each program you develop. By doing so you will have a command file that you can run each time you need to compile or assemble, link, and run one of your programs. Therefore, for filename use the name of the program which this copy of PREP.COM works with.)

Remove the disk from drive 0.

Remove the disk from drive 1, and bootstrap it in drive 0.

3. Read the indirect command file and choose the command lines that suit your application.
4. Edit the indirect command file and replace the example filenames (PROG1, PROG2, PROG3, PREFIX, and MYLIB) with the names of your files.
5. Delete the exclamation point at the start of each command line that you want to execute.
6. If any of the files you are referring to are on drive 1 (DK1:), use the editor to change the device names in those filespecs from DK: to DK1:.
7. Read in this section discussions of the command lines you are using, and determine whether you need to make any additional changes. If any changes are required, use the editor to make them.

To run the indirect command file, type:

```
.@filename
```

where `filename` is the name of your indirect command file.

For additional information on the RT–11 utilities, see the RT–11 help files or documentation. To see RT–11 help files on the FORTRAN IV compiler, the MACRO–11 assembler, the RT–11 linker, and the RT–11 RUN commands, type:

```
.HELP topic
```

where `topic` is `FORTRAN`, `MACRO`, `LINK`, `RUN`, or `FRUN`

For additional documentation, see the *RT–11,RSTS/E FORTRAN IV User's Guide*, the *PDP–11 MACRO–11 Language Reference Manual*, and the *RT–11 System User's Guide.*

The remainder of this section lists each command line in PREP.COM and describes the results of executing that line:

```
!FORTRAN/LIST/WARNINGS/ONDEBUG/OBJECT:DK: DK:PROG1
```

The FORTRAN IV compiler creates the object file PROG1.OBJ and stores it on drive DK:. The compiler includes debug lines (those lines in the source file with a D in column one) in the compilation. The compiler also prints a listing on the line printer and includes warnings among the FORTRAN IV diagnostic error messages.

```
!FORTRAN/WARNINGS/OBJECT:DK DK:PROG1/LIST:DK:
```

The FORTRAN IV compiler creates the files, PROG1.OBJ and PROG1.LST, and stores them on drive DK:. The compiler also includes warnings among the FORTRAN IV diagnostic error messages.

```
!FORTRAN/LIST:DK:/ALLOCATE:NN,/OBJECT:DK: DK:PROG1+DK:PROG2
```
The compiler integrates the source files, PROG1 and PROG2, into a single object file, PROG1.OBJ. The compiler stores this object file on drive DK: and creates the listing file PROG1.LST, which is stored in the allocated space on drive DK:. (See the discussion of compiler options in the *RT–11,RSTS/E FORTRAN IV User's Guide* for help in choosing a value for NN. Use the editor to change NN to that value.)

```
!FORTRAN/LIST/HEADER/NOLINENUMBERS/OBJECT:DK:
DK:PROG1,DK:PROG2,DK:PROG3
```
FORTRAN IV creates three object files — PROG1.OBJ, PROG2.OBJ, and PROG3.OBJ — and stores them on drive DK:. the compiler also produces a listing of each file on the line printer. Each listing names the compiler options currently in effect but does not include sequence numbers.

```
!FORTRAN/LIST:DK: DK:PROG1+DK:PROG2/OBJECT:DK:
```
The FORTRAN IV compiler integrates the source files, PROG1.FOR and PROG2.FOR into a single object file, PROG2.OBJ. The compiler also creates the listing file PROG1.LST and stores the object and listing files on drive DK:.

```
!MACRO/LIST/CROSSREFERENCE:S/OBJECT:DK:DK:PROG1,DK:MYLIB/LIBRARY
```
MACRO–11 assembles the source file PROG1.MAC and uses the library MYLIB.MAC to satisfy references made with the .MCALL directive. MACRO–11 stores the object file on drive DK: and sends to the lineprinter a listing that includes a cross-reference of user-defined symbols.

```
!MACRO/ENABLE:LSB:GBL DK:PROG1+DK:PROG2/OBJECT:DK:
```
The MACRO–11 assembler creates PROG2.OBJ as an assembly of both PROG1.MAC and PROG2.MAC. During the process, MACRO–11 enables a local symbol block and treats all undefined symbol references as default global references.

```
!MACRO DK:PREFIX/PASS:1+DK:PROG2/LIST:DK:/NOSHOW:BEX
```
The MACRO–11 assembler creates PROG2.OBJ from the source file PROG2.MAC. During the first pass of the assembly, MACRO–11 processes PREFIX.MAC, a prefix file that contains macro definitions. The assembler also creates a listing file, PROG2.LST, on drive DK: and suppresses the binary extensions.

```
!LINK/MAP/DEBUG/EXECUTE:DK: DK:PROG1
```
The RT–11 linker creates the executable load module, PROG1.SAV, by linking the object file, PROG1.OBJ, with ODT, the On-line Debugging Technique. The linker stores PROG1.SAV on drive DK: and creates a load map listing that is printed on the lineprinter. (For more information on ODT, see the *RT–11 System User's Guide.*)

```
!LINK/MAP:DK:/EXECUTE:DK: DK:PROG1,DK:PROG2,DK:PROG3
```
The linker creates the executable load module, PROG1.SAV, by linking PROG1.OBJ, PROG2.OBJ, and PROG3.OBJ. The linker also creates PROG1.MAP, a load-map listing file, and stores both PROG1.MAP and PROG1.SAV on drive DK:.

```
!LINK/MAP/FOREGROUND:256./EXECUTE:DK DK:PROG1
```
The RT–11 linker creates file PROG1.REL, which can be executed as a foreground job under either the FB or XM monitor. The linker allocates 256 (decimal) bytes of stack space for the job. The linker also sends a load-map listing to the lineprinter.

```
!LINK/MAP:DK: DK:PROG1,DK:PROG2/EXECUTE:DK:
```
The linker creates the executable load module, PROG2.SAV, by linking PROG1.OBJ and PROG2.OBJ. The linker also creates PROG1.MAP, a load-map listing file, on drive DK:.

```
!LINK/EXECUTE:DK:/RUN DK:PROG1,DK:PROG2,DK:PROG3
```
The linker creates the executable load module, PROG1.SAV, by linking PROG1.OBJ, PROG2.OBJ, and PROG3.OBJ. The linker also initiates execution of PROG1.SAV as a background job.

```
!RUN DK:PROG1
```
This RUN command loads PROG1.SAV into memory and executes it as a background job.

```
!FRUN DK:PROG1/BUFFER:n
```
This command executes PROG1.REL as a foreground job and reserves more memory space than the program size. The /BUFFER option must be used to run a FORTRAN IV program as a foreground job. Use the following formula to determine the size of n, an octal value that specifies the number of words of memory to be allocated to PROG1.REL:

$$n = [1/2\ [504 + (33*c) + (r-136) + (a*512)]]$$

where a is the number of files open at one time. (If double buffering is in effect, multiply the number of files by 2.)

c is the number of channels (logical unit numbers).

r is the maximum record length. (The default is 136 characters.)

```
!FRUN DK:PROG1/PAUSE
```
This command loads PROG1.REL as a foreground job, prints the load address, and delays execution until the RESUME command is issued. During the pause, you may enter ODT commands to help debug PROG1. Note that should PROG1 be a FORTRAN IV program, you must specify the /BUFFER:n option.

```
!FRUN DK:PROG1/TERMINAL:m
```
This command assigns a particular terminal to interact with the foreground job. Use the editor to change m to the logical unit number of the terminal you have chosen. Note that if PROG1 is a FORTRAN IV program, you must specify the /BUFFER:n option.

5.8.2 Interactive Procedure for Program Development

PROGD is an interactive program that creates an indirect command file that is useful during program development. PROGD asks you questions and displays valid answers in brackets ([ ]). Any time you are not sure how to answer a question, type HELP(RET) or H(RET), which causes PROGD to display additional information about valid responses.

Install PROGD by executing the following procedure:

1. In drive 0, mount a system disk that contains the FORTRAN IV compiler and OTS.

   If you have two RL01/RL02 disk drives, mount the copied FEP distribution disk in drive 1. If you have only RX02 diskettes, mount the copied FEP Installation Diskette in drive 1.

2. Copy PROGD to the disk in drive 0 by typing:

```
.COPY  DK1:PROGD.FOR  DK0:PROGD.FOR(RET)
```

3. Remove the disk from drive 1.

Run PROGD by typing:

```
.RUN PROGD(RET)
```

The following sample PROGD dialog contains answers that elicit all introductory and help messages:

```
.RUN PROGD(RET)

Do you want introductory messages [Y/N]? Y(RET)

PROGD is an interactive program that creates indirect command files
that help you develop programs. After creating one or more source
files, in either FORTRAN or MACRO, you can use PROGD to assist you
in the next three phases of program development. These phases are:

• COMPILING OR ASSEMBLING
• LINKING
• RUNNING



Type (RET) to continue.(RET)

PROGD asks you for the names of files you would like to compile or
assemble, link, and run. (As you enter each filename, enter the
filetype if the filetype is not the default.) PROGD also asks you
for compiler, assembler, and linker options.
```

```
PROGD repeatedly asks you to enter filenames and options. To
tell the program that you have entered all desired responses to a
filename or option prompt, press only RET in response
to that prompt. To obtain a list of acceptable responses to
a prompt, type HELP or H in answer to that prompt. To abort
the program, type CTRL/Z in response to any prompt. Typing RET in re-
sponse to any question that lists Y and N as valid answers is the
same as typing NRET.

PROGD constructs command lines that are stored in an indirect com-
mand file. When you execute the indirect command file, it compiles
or assembles, links, and runs the files you have named during the
interactive phase of the program. Note that although this indirect
command file can link two or more object files, it can produce only
one executable load module.

(RET to continue)RET

Indicate the device on which your files for
program development reside.
1. SY:
2. DL0:
3. DL1:
4. DY0:
5. DY1:

[1/2/3/4/5]: 3RET

FORTRAN compilation, MACRO assembly, or LINKAGE?[F/M/L]? JRET

The only valid responses are "F", "M", or "L".

FORTRAN compilation, MACRO assembly, or LINKAGE? [F/M/L]? FRET

**** FORTRAN COMPILATION PHASE ****

File: HRET

• Normally, PROGD causes a separate object file to be
  created for each source file you specify. However, you
  can specify source filenames in such a way that a single
  object file is created for two or more source files. To
  cause this to happen, specify source files as a string of
  filenames separated by plus signs.

• You can enter one filename or group of filenames per line.
  You can also enter two or more filenames or groups of
  filenames on the same line by entering them as a string
  of filenames or groups of filenames separated by commas.

• Include compiler options that affect the file or group
  of preceding files separated by plus signs.

• The FORTRAN default input file type is .FOR, the MACRO
  default input file type is .MAC, and the default output file
  type is .OBJ for both the compiler and assembler.

Do you also want a listing of options? [Y/N] YRET

**** FORTRAN OPTIONS ****

/ALLOCATE:size
        Reserves space for an output file
/CODE:type
        Specifies the type of machine code to be generated
/DIAGNOSE
        Expands compiler crash dump information
```

```
/EXTEND
        Enables 80-column source input
/HEADER
        Includes the "options-in-effect" header in the listing
        file
/I4
        Allocates 4 bytes for single-precision integers
/LINENUMBERS (default)
        Includes internal sequence numbers in compiled programs
/LIST[:filespec]
        Produces a listing file
/NOLINENUMBERS
        Supresses internal sequence numbers in compiled programs
/NOOBJECT
        Suppresses output of the binary object file

Type (RET) to continue(RET)
/NOOPTIMIZE[:type]
        Disables a specific optimization
/NOSWAP
        Causes a compiled program to keep the USR resident during
        execution
/NOVECTORS
        Specifies array access by multiplication operations
/NOWARNINGS
        Supresses warning messages during compilation
/OBJECT[:filespec]
        Specifies a file name for the object file
/ONDEBUG
        Includes D lines in compilation
/OPTIMIZE[:type]
        Enables a specific optimization
/RECORD:length
        Alters the default record length for sequential ASCII
        formatted I/O
/SHOW:value
        Controls the listing content
/STATISTICS
        Includes compilation statistics in the listing file

Type (RET) to continue, "R" to re-read.(RET)
/SWAP (default)
        Causes a compiled program to swap the USR during execution
/UNITS:n
        Specifies the number of logical units that can be open at
        once
/VECTORS (default)
        Specifies array access by tabular lookup
/WARNINGS
        Includes warning messages in the listing file
```

```
Type (RET) to continue, "R" to re-read (RET)
File: TEMP1(RET)
File: TEMP2,TEMP3(RET)
File: (RET)
Option: H(RET)

OPTION ENTRY:
• Multiple options may be entered singly or as a string.
• Enter options that qualify the entire command string.
```

```
Do you also want a listing of the options? [Y/N] N(RET)
Option: /WA(RET)
Option: /LIS(RET)
Option:(RET)

Your Command:

FORT/WA/LIS
Files? DL1:TEMP1,DL1:TEMP2,DL1:TEMP3

Any Corrections [Y/N]? M(RET)

The only valid responses are YES [Y], or NO [N].

Any Corrections [Y/N]? (RET)
Another compilation/assembly [Y/N]? Y(RET)

FORTRAN compilation, MACRO assembly, or LINKAGE? [F/M/L]? M(RET)

**** MACRO ASSEMBLY PHASE ****

File: TEMP4+TEMP5(RET)
File:(RET)
Option: HELP(RET)


OPTION ENTRY:
• Multiple options may be entered singly or as a string.
• Enter options that qualify the entire command string.


Do you also want a listing of the options? [Y/N] Y(RET)


**** MACRO OPTIONS ****

/ALLOCATE:size
        Reserves space for an output file
/CROSSREFERENCE[:type[,...:type]]
        Generates a cross-reference table in the listing file
/DISABLE:value[,...:value]
        Specifies a .DSABL directive
/ENABLE:value[,...:value]
        Specifies an .ENABL directive
/LIBRARY
        Identifies a file as a macro library file
/LIST[:filespec]
        Produces a listing file
/NOOBJECT
        Supresses output of the binary object file
/NOSHOW:value
        Specifies an .NLIST directive

Type (RET) to continue.(RET)
/OBJECT[:filespec]
        Specifies a file name for the object file
/PASS:1
        Specifies a file as a prefix macro or conditional file
/SHOW:value
        Specifies a .LIST directive

Type (RET) to continue, "R" to re-read.(RET)
Option: /ENA:GBL(RET)
Option: /SH:MEB(RET)
Option:(RET)
```

```
Your Command:

MAC/ENA:GBL/SH:MEB
Files? DL1:TEMP4+DL1:TEMP5

Any corrections [Y/N]?RET
Another compilation/assembly [Y/N]?RET

**** LINKAGE PHASE ****

Are you linking to REAL-11/MNC [Y/N]? YRET

In response to the Option: prompt be sure to
include the following, where dvn: is the device on
which the REAL-11 library exists and filespec is the
name of the REAL-11/MNC library you are using.

     /LIB:dvn:filespec

File: TEMP1,TEMP2,TEMP3RET
File: TEMP4RET
File: HELPRET

• Input filenames singly or as a string
  of names separated by commas.
• The LINKER default input filetype is .OBJ, and the
  default output filetype is .SAV for background jobs,
  and .REL for foreground jobs.
• Include linker options that affect each file.
• To bypass the linkage and run phases, press RET
  in response to the first prompt for a filename.

Do you also want a listing of options? [Y/N] YRET


**** LINKER OPTIONS ****

/ALLOCATE:size
  Reserves space for an output file
/ALPHABETIZE
  Lists global symbols on the link map in alphabetical order
/BITMAP (default)
  Outputs the program bit map
/BOTTOM:n
  Specifies the lowest address to be used by the linked program
/BOUNDARY:value
  Starts a specific program section on a particular
  address boundary
/DEBUG[:filespec]
  Links ODT or the debugging program you specify with your
  program
/EXECUTE[:filespec]
  Specifies the name of the memory image file
/EXTEND:n
  Extends a program section to the size you specify
/FILL:n
  Initializes unused locations in the memory image file
  to the value you specify

Type RET to continue.RET
/FOREGROUND[:stacksize]]
  Links the program for foreground execution
/INCLUDE
  Accepts specific global symbols by name for
  inclusion in the memory image file
```

```
/LDA
  Produces a file in absolute binary loader format
/LINKLIBRAR[:filespec]
  Includes the file you specify as a library in the link
  operation
/MAP[:filespec]
  Generates a load map
/NOBITMAP
  Does not output the program bit map if there is
  code below location 400
/NOEXECUTE
  Supresses creation of the memory image file
/PROMPT
  Indicates that there is more than one line of input to the linker
/ROUND:n
  Rounds up a specific program section so that the root is a
  multiple of the value you specify

Type (RET) to continue, "R" to re-read. (RET)
/RUN
  Initiates execution of the linked program
/SLOWLY
  Uses the largest possible area for the symbol table during the
  link operation
/STACK[:n]
  Specifies the stack address for the linked program
/SYMBOLTABLE[:filespec]
  Generates a symbol table file
/TOP:n
  Specifies the highest address to be used by the linked program
/TRANSFER[:n]
  Specifies the starting address of the linked program
/WIDE
  Produces a load map that is 132 columns wide
/XM
  Specifies that a job to be run under the XM monitor requires the
  special SETTOP features



Type (RET) to continue, "R" to re-read. (RET)
File: (RET)
Option: /LIB:DL0:MNCLIB/MAP:DL1:(RET)
Option: (RET)

Your Command:

LIN/LIB:DL0:MNCLIB/MAP:DL1:
Files? DL1:TEMP1,DL1:TEMP2,DL1:TEMP3,DL1:TEMP4

Any Corrections [Y/N]? (RET)

**** RUN PHASE ****

       R [R]
     RUN [U]
    FRUN [F]
 NEITHER [N]
[R/U/F/N]? H(RET)

The R command loads a memory image file from
the system device into memory and starts execution.
Note the only valid device is SY:. There are
no options affecting this command.
```

```
Use the R command for:
• a background job
• a background privileged job under XM
• a background virtual job under XM

The RUN command loads and starts a program that has
the filetype .SAV and resides on any RT-11 block-
replaceable device. Any argument or input and output
list you include is passed to the program in Command
String Interpreter format.

Use the RUN command for:
• a background job
• a background privileged job under XM

Type (RET) to continue.(RET)
The FRUN command loads and starts execution of a
foreground program with the filetype .REL. Only one
foreground program can be run at a time.

**** FRUN OPTIONS ****

/BUFFER:n
  Reserves n words in memory over and above normal program size
/NAME:Jobname
  Assigns the specified logical job name to the job;
  system job monitors only
/PAUSE
  Prints the load address and waits; a RESUME command will start the
  job
/TERMINAL:n
  Assigns terminal unit n to interact with the foreground
  job; multi-terminal monitors only



Type (RET) to continue, "R" to re-read.(RET)

**** RUN PHASE ****

       R [R]
     RUN [U]
    FRUN [N]
 NEITHER [N]
[R/U/F/N]? R(RET)

                Your commands are filed under the name

                        **** TEMP1.COM ****

                    To view, type: TYP TEMP1.COM
                    To execute, type: @TEMP1.COM


STOP - - END

.TYP TEMP1.COM(RET)

FORT/WA/LIS
DL1:TEMP1,DL1:TEMP2,DL1:TEMP3
MAC/ENA:GBL/SH:MEB
DL1:TEMP4+DL1:TEMP5
LIN/LIB:DL0:MNCLIB/MAP:DL1:
DL1:TEMP1,DL1:TEMP2,DL1:TEMP3,DL1:TEMP4
R TEMP1

.
```

Chapter 6
System Precautions for Users of REAL–11/MNC

When using the REAL–11/MNC software, you should be familiar with certain features of RT–11 and recognize some limitations on its use with that software. This chapter describes these features and limitations.

6.1 Special Keys: CTRL/C and BREAK

The effect of typing CTRL/C while a FORTRAN IV program is running depends on which RT–11 monitor you are using: the FB monitor or the SJ monitor.

When you are using the RT–11 SJ monitor, typing CTRL/C causes a QBUS INIT function to occur. This RT–11 function clears and initializes all devices on the system. The SJ monitor executes a QBUS INIT reset when a program terminates. See the RT–11 documentation for details.

When you are using the RT–11 FB monitor, typing CTRL/C causes the monitor to clear all the Control and Status Registers (CSRs) related to MNC–series modules for the job interrupted by the CTRL/C. Typing CTRL/C while using the FB monitor does not cause a QBUS INIT function to occur. If QBUS INIT were to execute while another job is running, it could interfere with the successful running of that other job.

Another special key is BREAK . As discussed in Chapter 2, you can type BREAK whenever you want to halt the processor. When you type BREAK , the processor halts, the terminal displays a six-digit octal number, and all system activity ceases. Since system activity ceases, any data being transmitted after the CPU has halted is lost. If you have typed BREAK accidentally, you can restart the processor immediately by typing P. If you were using KED when you typed BREAK and some unwanted characters have appeared on your terminal, you can remove those characters by typing CTRL/W.

6.2 Completion Routines

Completion routines are user-written, FORTRAN–callable subprograms that usually execute asynchronously with a main subprogram (that is, when the main program has been interrupted). This section describes completion routines only as they are used with asynchronous sweeps. (Burst-mode sweeps also use completion routines but execute them sequentially rather than asynchronously. For information on differences between asynchronous and burst-mode sweeps, see the *REAL–11/MNC Programmer's Reference Manual.*) Completion routines are handled differently by the SJ and FB monitors. If you plan to write completion routines in your programs, you should be familiar with these differences.

Under the SJ monitor, completion routines can interrupt one another. If no other completion routine is executing when a completion routine is called, the SJ monitor executes the requested completion routine immediately. If a completion routine is executing when another is called, the second completion routine interrupts the first, executes, then returns control to the first.

The SYNCH ERROR code is not detected under the RT–11 SJ monitor. (With the REAL–11/MNC software, the SYNCH ERROR code is returned as status-code 337 in the sweep-status word; see the *REAL–11/MNC FORTRAN Programmer's Reference Manual* for more information on the sweep-status word.) Therefore, a completion routine is not prevented from interrupting itself. This situation is almost always fatal. In general, you should use great care if completion routines are used in your program and your program is running under the SJ monitor.

Under the FB monitor, completion routines cannot interrupt one another. Consequently, if a completion routine is executing when a second one is called, the first completion routine continues to execute until finished, regardless of whether it was called from a main program operating in the foreground or background of memory. After the first completion routine has finished its work, the second completion routine executes, followed by the main program.

With two or more completion routines active, along with main programs requiring the foreground and background, the FB monitor schedules the completion routines and main programs according to the priorities listed below.

Highest Priority	Interrupt Service Routines (priority determined by the hardware)
	RT–11 FB monitor
	Foreground completion routines
	Background completion routines
	Foreground main programs
	Background main programs

Lowest Priority Null job

As the list shows, completion routines, whether running in the foreground or background, have priority over all main programs. Foreground completion routines have priority over background completion routines, and foreground main programs have priority over background main programs.

Though one completion routine cannot interrupt another completion routine when running under the FB monitor, a completion routine can interrupt a main program in order to execute.

For additional information on completion routines, see the *REAL–11/MNC FORTRAN Programmer's Reference Manual.*

6.3 Virtual Arrays

You can use FORTRAN IV virtual arrays with the REAL–11/MNC software. However, you cannot use virtual arrays as REAL–11/MNC buffers or subprogram arguments.

To use virtual arrays, you must have an 11/23 processor and more than 32k words of memory. You must also have installed the FORTRAN IV OTS to support virtual arrays under the SJ, FB, or XM monitor. (See discussions of the OTSGEN dialog in Sections 3.1.4 and 3.2.4.)

If you are using virtual arrays with the SJ or FB monitor, you must install the patch that causes the monitor to bootstrap into 28k words, instead of 30k words. Instructions for installing this patch are listed in the *RT–11 Installation and System Generation Guide*, Section 2.8.19, Running RT–11 in Less Memory Than is Available. As you follow those instructions, install the following patch instead of the one listed in the *RT–11 Installation and System Generation Guide*:

```
.R SIPP(RET)
*monitr.SYS(RET)
Base? 1000(RET)
Offset? 30(RET)

Base            Offset          Old          New?
 001000          000030         000407       000240(RET)
 001000          000032         013704       012704(RET)
 001000          000034         177570       157776(RET)
 001000          000036         042704       (CTRL/y)(RET)
*(CTRL/c)

.COPY/BOOT dv0:monitr.SYS dv0:(RET)
```

where `monitr` is `RT11SJ` if you are installing a patch to the SJ monitor or `RT11FB` if you are installing a patch to the FB monitor

`dv` is `DL` if you have two RL01/RL02 disk drives or `DY` if you have only RX02 diskette drives

6.4 INTSET and .PROTECT

There are two RT–11 facilities that you should not use to access the MNC–series modules when you are simultaneously using the REAL–11/MNC software. These facilities are the FORTRAN IV Interrupt Service Routine (ISR) called INTSET and the programmed request called .PROTECT.

You can, however, use these facilities to access hardware devices other than the MNC–series modules. (For more information on INTSET and .PROTECT see the RT–11 documentation.)

6.5 Special Considerations in Linking REAL–11/MNC Software

This section lists some considerations you should be aware of when linking REAL–11/MNC subprograms with your calling FORTRAN IV program. Section 6.5.1 contains information on linking REAL–11/MNC software under any RT–11 monitor. Section 6.5.2 contains information on linking REAL–11/MNC software under the RT–11 SJ and FB monitors. For additional considerations involving the linking of REAL–11/MNC software under the XM monitor, see Section 6.6, Using REAL–11/MNC Software under the XM Monitor.

6.5.1 Linking REAL–11/MNC Software under any RT–11 Monitor

You can place REAL–11/MNC subprograms in either the root segment of your FORTRAN IV program or in a disk overlay segment.

If you place REAL–11/MNC subprograms in a disk overlay segment, you must observe the restrictions for creating virtual overlays that are listed in Section 6.6.4. You can also use the sample procedure described in that section. However, as you link the program, use the /O:n (overlay) option instead of the /V:n option. Use the /Q option mentioned in Section 6.6.4 only if you are using the XM monitor.

6.5.2 Linking REAL–11/MNC Software under RT–11 SJ or RT–11 FB

This section contains information pertinent to users of the RT–11 SJ and FB monitors. For information on linking programs under the RT–11 XM monitor, see Section 6.6, Using REAL–11/MNC Software Under the XM Monitor.

Several I/O operations, such as the opening and closing of files, require the use of RT–11 User Service Routines (USRs). The USR portion of the RT–11 SJ and FB monitors is not permanently resident in memory. Usually the USR is swapped over a defined section of memory whenever a program requires any of its services (for example, during a sweep operation).

However, an untimely swapping of the USR can cause a fatal system error. You can avoid such an error by locking the USR into memory before running a program that requires its services. The following command locks the USR into memory:

```
SET USR NOSWAP(RET)
```

If you cannot lock the USR into memory (if to do so, for example, requires too much memory), and you want to avoid the fatal system error, you must identify certain areas of memory that the USR must not swap over. You can then direct the USR to other areas of memory that do not involve the risk of a fatal error.

You can identify the areas of memory that the USR must not swap over by looking at a link map, which gives the addresses of these areas. Such areas include the sections containing completion routines.

For further details regarding the swapping algorithm and areas of memory affected by swapping the USR, see the *RT–11 Software Support Manual* and the *RT–11,RSTS/E FORTRAN IV User's Guide.*

6.6 Using REAL–11/MNC Software under the XM Monitor

The RT–11 Monitor offers special features to support extended memory. These features are only occasionally useful for REAL–11/MNC applications. Therefore, DIGITAL recommends using the RT–11 SJ or FB monitor with the REAL–11/MNC software. However, as long as you use the procedures listed in this section, you can use the XM monitor with REAL–11/MNC. Also, note that to use the XM monitor, you must have an 11/23 processor and should have at least 64k words of memory.

For a detailed discussion of extended memory, see Chapter 4 of the *RT–11 Software Support Manual.*

6.6.1 Privileged and Virtual Jobs

Under the XM monitor, jobs may be either privileged or virtual. The terms "privileged" and "virtual" refer to the way in which physical memory is allocated to a job and to that job's rights to access other memory areas, such as the buffers and hardware registers for the MNC–series modules.

Unless you specify otherwise, all jobs under the XM monitor are privileged. Furthermore, any jobs that use REAL–11/MNC data-transfer subprograms must be privileged. However, a job that uses only computational routines (for example, the REAL–11/MNC subprogram FLT16), can be run as a virtual job. To use any procedures described in this section, you must run jobs as privileged.

6.6.2 Linking Background Jobs under XM

To link a background job that uses the REAL–11/MNC software under the XM monitor, type:

```
.R LINK(RET)
*prog[,map]=prog,reallib/Q(RET)
Load section:address? aaaaa:nnnnn(RET)
Load section:address?(RET)
```

where `prog` is the name of your program

`map` is the name of the output map file, which is optional

`reallib` is `MNCSNG`, `MNCMUL`, or `MNCLIB` — the name of the REAL–11/MNC library that you are using

`aaaaa` is `$CODE` if prog uses completion routines and `$DATA` if prog does not use completion routines

`nnnnn` is `60000` if your system uses the MQ handler and `40000` if your system does not use the MQ handler

This link procedure may produce one of the following warning messages, which should be ignored:

```
?LINK-W-load address too low PSECT $DATA
?LINK-W-load address too low PSECT $CODE
```

NOTE

Only advanced users should attempt the following additional procedures.

If the link procedure did not produce the error message indicating that the load address is too low (as listed above), you may be able to link your program to use memory more efficiently. To do so:

1. Link your program again by using the procedure listed earlier in this section, but do not use the /Q option. This procedure produces a load map.
2. Study the load map and determine whether any of the .KSCT p-sects can be moved to the area of memory between p-sect SYS$S and octal address 20000. (The area of memory between octal addresses 20000 and 37777 is a restricted area for PAR1.)
3. If you can move one or more .KSCT p-sects, generate a dummy file with only p-sect directives in the new sequence. Be sure that the attributes are identical to the original and that you include all p-sects listed in the load map produced in step 1.
4. Assemble the dummy file.

5. Link again. As you do so, specify the dummy file first. Then link again as you normally would for the XM monitor and include the /Q option. If your program uses completion routines or the MQ handler, make suitable adjustments.

6.6.3 Linking and Executing Foreground Jobs under XM

To link and execute a foreground program that uses the REAL–11/MNC software under the XM monitor, use the following procedures:

1. When linking your program, use the /FOREGROUND and /LIBRARY options. For information on these options, see the discussion of the LINK keyboard command in the *RT–11 System User's Guide*. Type:

   ```
   .LINK/FOR:256.[/MAP:DK:]/LIB:reallib prog
   ```

   where `reallib` is MNCSNG, MNCMUL, or MNCLIB — the name of the REAL–11/MNC library that you are using.

   `prog` is the name of your program.

2. To execute your program, use the FRUN keyboard command with the /PAUSE and /BUFFER options. The FRUN command with the /PAUSE option loads a program and prints the load address before the program is executed. The /BUFFER option allows you to specify additional program workspace required for internal buffers. Type:

   ```
   .FRUN prog/PAUSE/BUFFER:nnn
   ```

   where `prog` is the name of your program.

   `nnn` is an octal number that specifies the number of words of memory reserved for your program. (For information on how to choose a value for `nnn`, see the discussion of the FRUN keyboard command in the *RT–11 System User's Guide*.)

   RT–11 responds with a message that tells whether your program was loaded into memory.

   If the following message appears, your program was loaded, and you can proceed to step 3 of this procedure.

   ```
   Loaded at nnnnn
   ```

   where `nnnnn` is the load address

   If the following message appears, RT–11 has failed to load your program:

   ```
   ?KMON-F-Not enough memory
   ```

If that message has appeared, use the SHOW DEVICES and SHOW JOBS keyboard commands to determine which device drivers and programs are loaded in memory. If you do not need some of the device handlers and jobs displayed, use the UNLOAD keyboard command to remove them, and retype the FRUN command line to execute your job. If there still is too little memory for your job, it is too big to run as a foreground job under the XM monitor. If that is the case, you may be able to rearrange your program to run in overlays, or you may be able to run the program under the FB monitor (after relinking it).

For more information on the FRUN, SHOW, and UNLOAD keyboard commands, see the *RT–11 System User's Guide.*

3. If your program is in memory and RT–11 has printed a load address, determine if the program was loaded in a portion of memory where it can be executed:

   If the load address is at least 42000 (octal), you should be able to execute the program.

   If you are using the MQ handler, you probably cannot execute your program unless the load address is at least 62000 (octal).

   For information on the mapping problems that make some areas of memory unavailable for program execution, see the section entitled "Interrupt Service Routines in XM Systems" in the *RT–11 Software Support Manual.*

4. If your program is in an area of memory where it can be executed, type:

   ```
   .RESUME[ prog ]RET
   ```

   where `prog` is the name of your program. You must specify a value for prog if system-job support is enabled on your monitor.

   The RESUME command causes your program to be executed. (The /PAUSE option used with the FRUN command suspended execution.)

   If you have specified too small a buffer value for nnn in the FRUN command line, RT–11 displays a message such as one of the following:

   ```
   ?ERR 62 Fortran start fail
   ?ERR 0, non-Fortran error call
   ```

   If one of those messages appears, retype the FRUN command and use a larger value for nnn.

   For information on system-job support, see the *RT–11 Installation and System Generation Guide.* For more information on the RESUME keyboard command, see the *RT–11 System User's Guide.*

5. If you are using the console terminal as a shared console for foreground and background jobs, press CTRL/F prior to entering any data that the job requests.

6. When your job has completed, remove it from memory to free space for other jobs. Type:

```
.UNLOAD prog(RET)
```

where `prog` is the name of your program

6.6.4 Virtual Overlays

Your FORTRAN IV program can use a virtual overlay segment that calls REAL–11/MNC subprograms. (For more information on virtual overlays, see Chapter 11, Linker, in the *RT–11 System User's Guide.*) However, you must observe the following restrictions as you write the program:

- All REAL–11/MNC buffer-management and I/O subprograms that your program calls must be in the same segment.
- The segment that calls the REAL–11/MNC subprograms must be the only segment in its region while the REAL–11/MNC software is transferring data. In other words, that segment should not be competing with any other segment for its region while the segment is active.
- Your program must be linked with MNCOVR.OBJ, which causes multiple references to the global ISR p-sects and forces them into the root segment.
- You must link completion routines called by REAL–11/MNC subprograms to the root.
- Your program must conform to all restrictions normally related to the use of overlays.
- Foreground programs that call REAL–11/MNC subprograms cannot use virtual arrays.

The following procedure illustrates how to create and execute a program that uses virtual overlays. This program is a modification of the program BMEX1.FOR, which is included on the FEP distribution disk and is also presented in Chapter 5 of the *REAL–11/MNC Programmer's Reference Manual.* Some comments in this program refer to "REAL–11/MNC subprograms described in this chapter." The "chapter" referred to is Chapter 5 of the *REAL–11/MNC Programmer's Reference Manual.*

NOTE

Although the procedure instructs you to use the editor to create the root segment and virtual overlays, they are also contained on the FEP distribution disk.

1. Use the editor to create the root segment, VOEX1.FOR (see Figure 6–1).

The root segment declares the virtual overlays, VOEX2.FOR and VOEX3.FOR (see Figures 6–2 and 6–3) in a FORTRAN IV EXTERNAL statement. The root segment also contains a FORTRAN IV COMMON statement that declares the REAL–11/MNC data buffers and sweep-status array. Finally, the root segment calls each virtual overlay and ends.

Figure 6–1: The Root Segment, VOEX1.FOR

```
C       VOEX1.FOR               VIRTUAL OVERLAY EXAMPLE FOR REAL/RT
C                               RT-11 XM ONLY
C
C       THIS IS THE ROOT SEGMENT
C
C       THE ENTIRE ROUTINE SHOULD BE LINKED FOR XM AS FOLLOWS:
C
C       .R LINK
C       *VOEX1,VOEX1=VOEX1,MNCOVR,MNCSNG/Q//
C       *VOEX2/V:1
C       *VOEX3/V:1
C       *//
C       *Load section:address? $DATA:40000
C       *Load section:address? RET
C       *^C
C
C
C
C       DECLARE THE SUBROUTINE SEGMENTS AS EXTERNAL
C
        EXTERNAL VOEX2
        EXTERNAL VOEX3
C
C
C       CREATE A COMMON BLOCK FOR SWEEP STATUS AND DATA
        DIMENSION INFO(40),IBUF(100,4),ISTAT(4)
        COMMON /REAL/INFO,IBUF,ISTAT
C
C
C
C       NOW CALL THE FIRST OVERLAY SEGMENT
        CALL VOEX2
C
C
C       NOW CALL SECOND OVERLAY SEGMENT
        CALL VOEX3
C
C
C       ALL DONE
C
        STOP 'VOEX1 COMPLETED'
        END
```

2. Use the editor to create the virtual overlays, VOEX2.FOR and VOEX3.FOR (see Figures 6–2 and 6–3).

   Note that each virtual overlay contains exactly the same COMMON statement used in the root segment and that each virtual overlay calls various REAL–11/MNC subprograms.

Figure 6–2: The Virtual Overlay, VOEX2.FOR

```
        SUBROUTINE VOEX2
C
C       VIRTUAL OVERLAY EXAMPLE FOR REAL/RT
C
C       THIS IS OVERLAY SEGMENT ONE FOR REGION ONE
```

```
C
C       THIS ROUTINE IS AN ADAPTATION OF THE BMEX1 EXAMPLE ROUTINE
C
C       CHANGES ARE:
C       1)      SUBROUTINE INSTEAD OF MAIN ROUTINE (INCLUDE RETURN)
C       2)      COMMON BLOCK FOR STATUS AND DATA
C       3)      FIRST PORTION OF ORIGINAL BMEX1 ROUTINE
C
C
C
C
C
C       !BUFFER MANAGEMENT SUBPROGRAMS: EXAMPLE
C
C This example demonstrates the use of the subprograms described in
C this chapter. It assumes that your hardware configuration contains
C at least one MNCDI and one MNCKW, and that you have generated your
C REAL-11/MNC library accordingly. The hardware is required since
C buffer management is always involved in the transfer of data through
C the MNC-series modules.
C
C The primary aim of this example is to highlight the use of the
C REAL-11/MNC subprograms described in this chapter. The example
C should not detect any errors if your hardware configuration and
C your REAL-11/MNC library match the requirements stated in the pre-
C vious paragraph. Therefore, in many instances throughout this
C example, error checking is omitted. In practice, especially in
C programs which have not yet been debugged, you should include more
C error checking than is demonstrated by this program.
C
C The example begins by defining certain arrays required by the sweep
C process. INFO is used as the sweep-information array, IBUF(1,N+1)
C is used as the Nth buffer, and ISTAT is used to obtain buffer-status
C information.
C
C
        COMMON /REAL/INFO,IBUF,ISTAT
        DIMENSION INFO(40),IBUF(100,4),ISTAT(4)
C
C Next, using the SETIBF subroutine, the program directs the buffer-
C management software to use INFO as a sweep-information array and to
C use IBUF(1,N+1) as the Nth buffer associated with that sweep.
C
   CALL SETIBF(INFO,IND,,IBUF(1,1),IBUF(1,2),IBUF(1,3),IBUF(1,4)
C
C The example now, using the RLSBUF subprogram, puts, or releases,
C buffers number 2, 1, and 0 into the device queue in that order.
C
   CALL RLSBUF(INFO,IND,2,1,0)
C
C Buffer #2 is now taken out of the device queue with the RMVBUF
C subprogram.
C
   CALL RMVBUF(INFO,2,IND)
C
C The buffers remaining in the device queue, buffers 1 and 0, are now
C arranged in ascending order in the queue using the INXTBF
C subprogram.
C
   CALL INXTBF(INFO,0,IND)
C
C The program now starts the sweep, acquiring a buffer of data every
C 20 seconds. The sweep will continue unitl the operator asks to call
C the STPSWP subroutine. Note that MNCDI unit 0 will be sampled.
C
   CALL CLOCKA(5,-20,IND)
   CALL DISWP(INFO,100)
   IF(INFO(1).NE.0) STOP 'DISWP - LIBGEN OR CONFIGURATION ERROR.'
C
C The program now releases the remaining buffers to the device queue
C for this sweep. Note that this action is not required at this time.
C
```

```
     1 CALL RLSBUF(INFO,IND,2,3)
C
C
C       { BALANCE OF ORIGINAL ROUTINE IS NOW IN VOEX3, WHICH
C         IS OVERLAY SEGMENT 2, REGION 1. }
C
        RETURN
        END
```

Figure 6–3: The Virtual Overlay, VOEX3.FOR

```
        SUBROUTINE VOEX3
C
C       VIRTUAL OVERLAY EXAMPLE FOR REAL/RT
C
C       THIS IS OVERLAY SEGMENT TWO FOR REGION ONE
C
C       THIS ROUTINE IS AN ADAPTATION OF THE BMEX1 EXAMPLE ROUTINE
C
C       CHANGES ARE:
C       1)          SUBROUTINE INSTEAD OF MAIN ROUTINE (INCLUDE RETURN)
C       2)          COMMON BLOCK FOR STATUS AND DATA
C       3)          SECOND PORTION OF ORIGINAL BMEX1 ROUTINE
C
C
C
C
C
C       !BUFFER MANAGEMENT SUBPROGRAMS: EXAMPLE
C
C       { FIRST HALF OF ORIGINAL BMEX1 ROUTINE IS NOW IN VOEX2.FOR,
C         WHICH IS OVERLAY SEGMENT 1, REGION 1. }
C
C
C
        COMMON /REAL/INFO,IBUF,ISTAT
        DIMENSION INFO(40),IBUF(100,4),ISTAT(4)
C
C
C The operator now controls the sweep-related activity, by indicating
C the next function to be performed, as follows:
C a) typing a "1" causes the subprogram IGTBUF to be called,
C b) typing a "2" causes the subprogram IWTBUF to be called,
C c) typing a "3" causes the subprogram IBFSTS to be called, and
C d) typing a "4" causes the subprogram STPSWP to be called, thus
C    stopping the sweep and the program.
C
C If any errors occur during the sweep, they will be reported when
C they are detected and the program will stop.
C
C Prompt the operator.
C
    2 TYPE 1000
 1000 FORMAT(/' TYPE 1 FOR IGTBUF, 2 FOR IWTBUF, 3 FOR IBFSTS OR 4 FOR
     1 STPSWP  '$)
      ACCEPT 2000,NEXT
 2000 FORMAT(I1)
      IF(NEXT.LT.1.OR.NEXT.GT.4) GO TO 2
      TYPE 1001
 1001 FORMAT(/)
      GO TO (10,20,30,40),NEXT


C
C This program section is executed if the operator indicates to call
C the IGTBUF subprogram. First check for a buffer in the user queue.
C Report if one is found and on the current status of sweep. If a
C buffer is found, report the buffer number and release it to the
C device queue, since IGTBUF has already extracted it from the user
C queue.
C
```

```
   10 IF(IGTBUF(INFO,I,IND).LT.0) GO TO 12
   11 TYPE 1002,I
 1002 FORMAT(' BUFFER #'I1' EXTRACTED FROM USER QUEUE.')
      CALL RLSBUF(INFO,IND,I)
      GO TO 12
   12 TYPE 1003
 1003 FORMAT(' NO BUFFERS CURRENTLY IN USER QUEUE.')
   13 TYPE 1004,INFO(1)
 1004 FORMAT(' CURRENT SWEEP STATUS CODE IS 'I3'.')
      IF(INFO(1).NE.0) STOP 'SWEEP HAS ENDED.'
      GO TO 2
C
C This program section is executed if the operator indicates to call
C the IWTBUF subprogram. Wait for a buffer to be placed into the user
C queue and then report on the buffer found and the current sweep
C status.
C
   20 CALL IWTBUF(INFO,,I,IND)
      GO TO 11
C
C This program section is executed if the operator indicates to call
C the IBFSTS subprogram. Print the current status of each buffer
C and the current status of the sweep.
C
   30 CALL IBFSTS(INFO,ISTAT,IND)
      TYPE 1005,(I-1,ISTAT(I),I=1,4)
 1005 FORMAT(' THE BUFFER STATUS CODE FOR BUFFER #'I1' IS 'I2'.')
      GO TO 13
C
C This program section is executed if the operator indicates to call
C the STPSWP subprogram. The program first calls STPSWP for a stop at
C the end of the next buffer, and then flushes the user queue in order
C to check on the final status of the sweep.
C
   40 CALL STPSWP(INFO,1,IND)
   41 IF(IWTBUF(INFO,,I,IND).GE.0) GO TO 41
      GO TO 13
C
           RETURN
           END
```

3. Compile the root segment and virtual overlays by typing:

```
.FORTRAN VOEX1RET
.MAIN.

.FORTRAN VOEX2RET
VOEX2

.FORTRAN VOEX3RET
VOEX3
```

4. Use the linker to create the virtual overlay structure and produce a background job by typing:

```
.R LINKRET
*VOEX1,VOEX1=VOEX1,MNCOVR,MNCSNC/Q//RET
*VOEX2/V:1RET
*VOEX3/V:1RET
*//RET
Load section:address? $DATA:40000RET
Load section:address?RET
*CTRL/c
```

5. Execute VOEX1 by typing the following. (The results printed here are a sample of those you can obtain by running the program.)

```
.RUN VOEX1(RET)

TYPE 1 FOR IGTBUF, 2 FOR IWTBUF, 3 FOR IBFSTS OR 4 FOR STPSWP 1(RET)

NO BUFFERS CURRENTLY IN USER QUEUE.
CURRENT SWEEP STATUS CODE IS 0.

TYPE 1 FOR IGTBUF, 2 FOR IWTBUF, 3 FOR IBFSTS OR 4 FOR STPSWP 3(RET)

THE BUFFER STATUS CODE FOR BUFFER #0 IS -1.
THE BUFFER STATUS CODE FOR BUFFER #1 IS  2.
THE BUFFER STATUS CODE FOR BUFFER #2 IS  2.
THE BUFFER STATUS CODE FOR BUFFER #3 IS  2.
CURRENT SWEEP STATUS CODE IS 0.

TYPE 1 FOR IGTBUF, 2 FOR IWTBUF, 3 FOR IBFSTS OR 4 FOR STPSWP 2(RET)

BUFFER #0 EXTRACTED FROM USER QUEUE.
CURRENT SWEEP STATUS CODE IS 0.

TYPE 1 FOR IGTBUF, 2 FOR IWTBUF, 3 FOR IBFSTS OR 4 FOR STPSWP 4(RET)

CURRENT SWEEP STATUS CODE IS 315.
STOP -- SWEEP HAS ENDED.

.
```

6.7 RT–11 System Errors Related to MNC–Series Modules

This section describes errors that may indicate the absence of MNC–series modules required by the REAL–11/MNC software. These error messages are printed at the system terminal.

6.7.1 Illegal Memory Reference

The FORTRAN IV OTS (Object Time System) displays this error as:

```
?Err 61 Illegal memory reference
```

If you are executing REAL–11/MNC software when the error occurs, the executing program stops and control returns to the monitor. The error may indicate that an MNC–series module required by the software is not present; to recover from this condition, determine the missing MNC–series module, and install it. (Be sure to power off your MINC computer system before removing or inserting an MNC–series hardware module; see the *DECLAB–11/MNC User's Guide* for more information.) An illegal memory reference may also indicate that CSR and vector address switches are set incorrectly; see the *DECLAB–11/MNC User's Guide* for more information.

6.7.2 Processor Halt

If the processor halts, the terminal displays the following message and no further program or system activity occurs:

```
nnnnnn
@
```

where `nnnnnn` is the address plus 2 at which the processor stopped.

If this happens while you are executing REAL–11/MNC software, an MNC–series module required by the software may not be present. To recover, reboot the system (see Section 2.1), determine the missing MNC–series module, and install it. (Be sure to power off your MINC computer system before removing or inserting an MNC–series hardware module; see the *DECLAB–11/MNC User's Guide* for more information.)

6.8 REAL–11/MNC Performance with the FB and XM Monitors

When you are using the FB or XM monitor, a program that calls REAL–11 MNC subprograms can be running at the same time another program is running. For example, a program that uses REAL–11 /MNC subprograms to acquire data can run as a foreground job while a data-reduction program is running as a background job. Because active programs compete for a share of system resources, however, the REAL–11/MNC program executes more slowly than it would if no other program were active. This situation, may produce data-overrun, buffer-overrun, or buffer-underrun errors during sweeps that transfer data at high rates. (For more information on these errors, see Section 2.2.10, Status Codes for the Sweep–Status Word, in the *REAL–11/MNC Programmer's Reference Manual.*) To obtain the maximum performance from a program that calls REAL–11/MNC subprograms, run that program alone.

Chapter 7
Using RGL/11

This chapter contains information on developing, linking, and executing programs that call RGL/11 software.

7.1 RGL/11 Files Needed for Program Development

The RGL/11 software includes three files that are the minimum you need when you develop a program that calls RGL/11 subroutines. The three files are:

- RGLLIB.OBJ, the RGL/11 library
- BLODAT.OBJ, the block data module
- ERRTXT.TXT, the RGL/11 error messages

In addition, if you plan to use the Greek character font, you also need GREEK.FNT.

If you have two RL01/RL02 disk drives, you have installed these files on the development system disk. (See Section 4.7 on installing RGL/11.) You may also have placed RGLLIB.OBJ in SYSLIB.

If you have only RX02 diskettes, you have installed these files on the separate diskette 1/5 that contains only the RGL/11 software needed to run non-overlaid RGL/11 programs.

The files listed in this section are sufficient only to execute RGL/11 subroutines with programs that are not overlaid. To execute RGL/11 software with overlaid programs, you also need additional object modules. However, if you have installed RGL/11 for use with overlays, you have already installed those files, which are on the disk that contains PRMLIB.OBJ, BLODAT.OBJ and ERRTXT.TXT.

7.2 Developing a Program that Calls the RGL/11 Software

Execute the following procedures when you develop a program that calls any RGL/11 subroutines:

If you have RL01/RL02 disks:

1. Bootstrap the development system disk in drive 0.
2. Create the source file for your program on the development system disk, which also contains the files listed in Section 7.1.

If you have RX02 diskettes:

1. Bootstrap the development system diskette in drive 0.
2. Mount in drive 1 the diskette that contains the files listed in Section 7.1. Use the following diskette:
   - diskette 1/5 for a non-overlaid RGL/11 program
   - diskette 2/5 for a standard overlaid RGL/11 program
   - diskette 3/5 for an alternate extended-memory overlaid program

   If your program calls software that is part of other FEP–11/FRP components, such as REAL–11 or IBS, those components must also be on the diskette.
3. Designate the diskette in drive 1 as the default diskette by typing:

   ```
   .ASSIGN  DY1:  DK:RET
   ```

   Create the source file for your program on the diskette in drive 1.

7.3 Linking a Program that Calls the RGL/11 Software

There are three methods for linking programs that call RGL/11 subroutines. These methods are:

- Running the command file RGLLNK.COM, which you can use whether or not you overlay your program. RGLLNK also compiles and executes your program.
- Running the linker, which you should use only if you do not overlay your program.
- Editing and running the command file RGLOVR.COM, which you should use only if you use the alternate extended-memory overlay structure.

The three methods are described below.

7.3.1 The Indirect Command File, RGLLNK

RGLLNK creates an executes a command file called RGL.COM that enables you to compile, link, and execute your program. If you are running under the XM monitor and you want to use overlays, you have the option of using low memory overlays or extended-memory overlays. If you choose extended-memory overlays, your program will be created as a virtual job. If you are running under FB or SJ and want to use overlays, you can use only low memory overlays. You can default all answers to RGLLNK except the program name. The default for each question is given in square brackets at the end of each question.

To invoke RGLLNK, type:

```
.@RGLLNK(RET)
```

A sample RGLLNK dialog follows. It compiles, links, and runs a program named prog:

```
.@RGLLNK(RET)
Do you want to compile [Y] ? (RET)
Do you want listings [N] ? (RET)
```

Type Y(RET) if you want listings and (RET) if you do not. If you answer Yes, RGLLNK prompts with the further question:

```
What device do you want the listings to go to [LP:] ?
```

Type (RET) to have the listing go to the line printer, or type the name of the device, such as DL0: where you want the listings sent.

```
Do you want to link [Y] ? (RET)
Do you want a map [N] ? (RET)
```

Type Y(RET) if you want a load map; type (RET) if you do not. If you answer Yes, RGLLNK prompts with the further question:

```
What device do you want the listings to go to [LP] ?
```

Type (RET) to have the load map go to the line printer, or type the name of the device, such as DL0:, where you want it sent.

```
Do you want overlays [Y] ? (RET)
```

Type (RET) to overlay your programs; type N(RET) to avoid overlays. A Yes answer link your program to PRMLIB.OBJ., and RGLLNK prompts with the further question:

```
Do you want extended memory overlays [N] ? (RET)
```

Type Y(RET) if you are running under XM and want extended-memory overlays, other wise type (RET).

```
Do you want to run [Y] ? (RET)
```

Type (RET) if you have a VT125 and want to run the program; type N(RET) if you do not.

If you are running under SJ, FB, or XM and want to compile, link, and run a program called PROG1 using low memory overlays, RGL.COM would look like:

```
FORT/CODE:THR PROG1
R LINK
PROG1=PROG1,BLODAT,PRMLIB/B:1200/A/W/P:1000//
LFIXED,LOCAT2/O:1
LFREE,LOCATE/O:1
PPOINT/O:1
PDATA/O:1
DPAPER/O:1
LNAXIS/O:1
LTAXIS/O:1
LNNICE,LINMIN,LINMAX/O:2
PRINUM,PRISTR,TEXT,XLABEL,YLABEL/O:2
FMINMX,LGNICE,EXPTST/O:2
DPALOG,DPALIN,DEFALT,DRWBOX/O:2
LINE,MARKER,MOVE,SWINDO/O:3
GENOVR/O:3
INITGR,CLIPIT,SSTATE/O:3//
^C
RUN PROG1
```

If you are running under XM and want to compile, link, and run PROG1 using extended memory overlays, RGL.COM would look like:

```
FORT/CODE:THR PROG1
R LINK
SY:PROG1=PROG1,BLODAT,PRMLIB/I/A/W/P:1000//
LFIXED,LOCAT2/V:1
LFREE,LOCATE/V:1
PPOINT/V:1
PDATA/V:1
DPAPER/V:1
LNAXIS/V:1
LTAXIS/V:1
LNNICE,LINMIN,LINMAX/V:2
PRINUM,PRISTR,TEXT,XLABEL,YLABEL/V:2
FMINMX,LGNICE,EXPTST/V:2
DPALOG,DPALIN,DEFALT,DRWBOX/V:2
LINE,MARKER,MOVE,SWINDO/V:3
GENOVR/V:3
INITGR,CLIPIT,SSTATE/V:3//
$QBLK

^C
R PROG1
```

7.3.2 Linking a Program that Calls RGL/11 Software without Using Overlays

If you have decided not to overlay your program, you can link it by running the linker yourself instead of running RGLLNK. Running the linker yourself makes the process of linking your program faster.

To link a program that calls RGL/11 subroutines, type the following:

```
,R LINK(RET)
*myprog = myprog, BLODAT,RGLLIB/B:1200/P:1000 [,filespec](RET)

where myprog is the name of your program

BLODAT is the name of the object file that defines the initial
graphic defaults

RGLLIB is the name of the graphics library for non-overlaid
programs

/B:1200 ensures there will be enough room in the monitor stack

/P:1000 allows table space for the linker program

filespec is the name or names (separated by commas) for optional
subroutines, or for optional libraries (which require the /LB
switch)
```

If the linker encounters no fatal errors, it produces an executable program or "load module" called myprog.SAV. The RGLLNK command file can also produce a non-overlaid program, so you have a choice which procedure to use.

7.3.3 RGLOVR.COM

Using the indirect command file RGLOVR is the third way to link a program. Use it for programs that use extended-memory overlays. This overlay structure will overlay additional RGL/11 routines. You should use this overlay structure only if you are running under XM, are going to use extended memory overlays, and:

1. Your program is too large using the extended memory overlay structure provided by RGLLNK and you would like to overlay more RGL/11 routines, or
2. Your program is too large and you would like to overlay more RGL/11 routines and add your own virtual overlay segments to overlay region 4.

The contents of RGLOVR.COM are:

```
R LINK
SY:MAINPR=MAINPR,BLODAT,REXLIB/I/A/W/P:1000//
LFIXED,LOCAT2/V:1
LFREE,LOCATE/V:1
PPOINT/V:1
PDATA/V:1
```

```
DPAPER/V:1
LNAXIS/V:1
LTAXIS/V:1
ARCOVR/V:1
RELOVR/V:1
SAVOVR/V:1
GN2OVR/V:1
LNNICE,LINMIN,LINMAX/V:2
PRINUM,PRISTR,TEXT,XLABEL,YLABEL/V:2
FMINMX,LGNICE,EXPTST/V:2
DPALOG,DPALIN,DEFALT,DRWBOX/V:2
LINE,MARKER,MOVE,SWINDO/V:3
GENOVR/V:3
INITGR,CLIPIT,SSTATE/V:3//
$QBLK

^C
```

If you want to just overlay more RGL/11 subroutines, do the following:

1. Make a copy of RGLOVR.COM
2. Edit the copy by replacing all occurrences of "MAINPR" with the name of your own program.

After the copy has been edited, it is ready to be run.

If you want to overlay more RGL/11 subroutines and overlay parts of your application program, do the following:

1. Make a copy of RGLOVR.COM
2. Edit the copy to:
   - Replace all occurrences of MAINPR with the name of your program
   - Include your segments in virtual overlay region 4. Remember to remove the "//" at the end of line "INITGR,CLIPIT, SSTATE/V:3//" and place it after your last user segment. If you choose to do this, you reduce the address space of the root by 4K (see the RT System User's Guide, Extended Memory Overlays).

An example of a RGLOVR command file containing user overlay segments follows. The file is an edited copy of RGLOVR and is named PROG1.COM. The name PROG1 replaces MAINPR throughout, and two user overlay segments are added to region 4.

```
R LINK
SY:PROG1=PROG1,BLODAT,REXLIB/I/A/W/P:1000//
LFIXED,LOCAT2/V:1
LFREE,LOCATE/V:1
PPOINT/V:1
PDATA/V:1
DPAPER/V:1
LNAXIS/V:1
LTAXIS/V:1
ARCOVR/V:1
RELOVR/V:1
```

```
SAVOVR/V:1
GN2OVR/V:1
LNNICE,LINMIN,LINMAX/V:2
PRINUM,PRISTR,TEXT,XLABEL,YLABEL/V:2
FMINMX,LGNICE,EXPTST/V:2
DPALOG,DPALIN,DEFALT,DRWBOX/V:2
LINE,MARKER,MOVE,SWINDO/V:3
GENOVR/V:3
INITGR,CLIPIT,SSTATE/V:3
USER1/V:4
USER2/V:4//
$QBLK

^C
```

Execute PROG1 by typing:

```
.@PROG1(RET)
```

When PROG1 successfully completes, it produces an executable program or load module with the name you supplied and a .SAV extension on the system device.

RGL/11 extended-memory overlays are run as virtual jobs and use the R command, rather than the RUN command. The R command can load programs only from the system disk. To run a file named PROG1.SAV that was created by PROG1.COM, you would type:

```
.R PROG1(RET)
```

7.4 Running a Program that Calls RGL/11 Software

If you are running under the FB or XM monitor or SJ with multi-terminal support, you must set your terminal to prevent RT–11 from sending carriage-return/line-feed characters. To do so, type:

```
.SET TT NOCRLF(RET)
```

To avoid typing this command every time you use RGL/11, add it to your startup indirect command file, which is either STARTS.COM, STARTF.COM, or STARTX.COM.

To run most RGL/11 programs, type:

```
.RUN prog (RET)

where prog is the name of your program
```

If your program uses RGL/11 extended-memory overlays, however, it must be run from the system disk. The reason for this is that RGL/11 creates the program as a virtual job and RT–11 looks on the system disk for virtual jobs. The command line to run a virtual job uses the "R" form of the RUN command:

```
.R prog (RET)

where prog is the name of the program using RGL/11 extended-memory
overlays
```

When you issue either form of the RUN command, RT–11 attempts to execute the load module, prog.SAV. If it encounters a condition that prevents it from doing so, it prints a FORTRAN OTS (Object Time System) error message. For help in understanding the error message, see the RT–11,RSTS/E FORTRAN IV User's Guide.

Chapter 8
Using PLOT55 with a VT105

PLOT55 is a FORTRAN–callable subroutine distributed with the RT–11 operating system. The PLOT55 software is designed for use with the VT55 terminal, the predecessor of the VT105 terminal. However, PLOT55 can also be used with the VT105 because the hardware designs of the two terminals are compatible. By using the PLOT55 package, you have access to the capabilities of the VT105 terminal that are also found in the VT55 terminal. This subset contains the majority of the VT105 features.

NOTE

PLOT55 is designed for use with a terminal that has 236 vertical positions. However, if you use the VT105 in rectangular mode, as the *VT105 Graphic Terminal User's Manual* recommends, the terminal has only 230 vertical positions.

Do not attempt to use PLOT55 with a VT125 terminal. PLOT55 can be used only with a VT55 or another terminal, such as a VT105, that can emulate a VT55. The VT125, even though it can emulate a VT105, cannot emulate a VT55.

The PLOT55 software sends commands to the terminal as if it were a VT55 terminal. Therefore, to ensure that the VT105 terminal correctly interprets commands from the PLOT55 software, put the VT105 terminal into VT52 mode before any part of the PLOT55 software is executed. You may do this either manually or under program control.

After running programs using PLOT55, always return the VT105 to ANSI mode.

8.1 Manually Changing the VT105 Mode

To manually change the VT105 operating mode:

1. Press the SET–UP key to put the terminal into SET–UP A mode.
2. Press the 5 key on the main keyboard to put the terminal into SET–UP B mode.
3. The terminal displays four groups of binary values. Look at the third entry in the second group of values. If that value is zero, the terminal already is in VT52 mode; press the SET–UP key again to return the terminal to normal operation, and ignore steps 4, 5, and 6 below. If that value is one, put the terminal in VT52 mode by performing steps 4, 5, and 6 below.
4. Repeatedly press the right-arrow (→) key, which is located at the upper right corner of the main keyboard, until the cursor is positioned over the third entry in the second group of switches.
5. When the cursor is correctly positioned, press the 6 key on the main keyboard to change the value from a one to a zero.
6. Press the SET–UP key again to return the terminal to normal operation, which is now VT52 mode.

To return the VT105 terminal to ANSI Mode, which is the default VT105 operating mode, perform the same procedure, but change the third entry in the second group of binary values to a one. Note that when the VT105 is in SET–UP B mode, pressing the 6 key changes the value over which the cursor is positioned from one to zero or from zero to one.

8.2 Changing the VT105 Mode under Program Control

To change the VT105 operating mode under program control:

Send the terminal-specific escape sequences. (An escape sequence is a series of ASCII characters, the first of which must be the escape character, 33 octal.)

When you send the following five-character escape sequence to the VT105 terminal, its operating mode changes from ANSI to VT52:

ESC [? 2 1

When you send the following two-character escape sequence to the VT105 terminal, its operating mode changes from VT52 to ANSI:

ESC >

NOTE

The terminal automatically returns to its predefined operating mode whenever the terminal is powered on.

For detailed information on operating the VT105 terminal, see the *VT105 Graphic Terminal User's Manual.*

The FEP–11 distribution media that contain the REAL–11/MNC software also include a MACRO–11 routine called VTMODE. This routine consists of two FORTRAN–callable subroutines that provide programmable control of the mode of your VT105. You can use these subroutines either as they are presented here or as a model for creating your own FORTRAN–IV or MACRO–11 subroutines.

NOTE

If you write a FORTRAN IV program that sends escape sequences to the terminal, remember that the first character in the sequence does not go directly to the terminal. The first character is normally intercepted by FORTRAN–IV and used as a form-control character.

A program can use the VT52 subroutine to change the current operating mode of the VT105 from ANSI to VT52 mode. Invoke this subroutine with the following FORTRAN statement:

```
CALL VT52
```

A program can use the ANSI subroutine to change the current operating mode of the VT105 from VT52 to ANSI mode. Invoke this subroutine with the following FORTRAN statement:

```
CALL ANSI
```

Following are the contents of VTMODE.MAC:

```
        .TITLE VTMODE       ;SUBROUTINES TO CHANGE THE OPERATING
                            ; MODE OF THE VT100 (VT105) TERMINAL.
        .MCALL .TTYOUT      ;ROUTINE FOUND IN FILE "SYSMAC.SML"
                            ;
        .GLOBL VT52,ANSI    ;DEFINE SUBROUTINE NAMES
                            ;
VT52:   MOV  #VT52ES,R1     ;GET ADDRESS OF ESC. SEQ. FOR VT52 MODE
        BR   START          ; GO SEND VT100 ESCAPE SEQUENCE
                            ;
ANSI:   MOV  #ANSIES,R1     ;GET ADDRESS OF ESC. SEQ. FOR VT100 MODE
        BR   START          ; GO SEND VT100 ESCAPE SEQUENCE
                            ;
LOOP:   .TTYOUT             ;SEND A SINGLE CHARACTER TO THE TERMINAL
START:  MOVB (R1)+,R0       ;GET NEXT CHARACTER IN ESCAPE SEQUENCE
        BNE  LOOP           ; IF IT IS NOT ZERO, GO SEND IT.
        RTS  PC             ; IF IT IS ZERO, WE ARE DONE, SO RETURN.
                            ;
VT52ES: .BYTE  33,133,77,62,154,0  ;ESC. SEQ. FOR VT52 MODE
ANSIES: .BYTE  33,74,0,0           ;ESC. SEQ. FOR ANSI MODE
                            ;
        .END
```

Assemble VTMODE.MAC and use the object file VTMODE.OBJC as you would use any other object file.

For example, assume that you have a FORTRAN–IV program called MYPROG that uses the PLOT55 subroutine. LINK your program by typing the following:

```
.LINK/EXECUTE:dv#MYPROG dv#: (MYPROG,PLOT55,VTMODE)RET
```

This RT–11 interactive command assumes the following:

1. The FORTRAN–IV library file, FORLIB.OBJ, has been included in the system library file, SYSLIB.OBJ.
2. The system library file resides on your system device.
3. The files MYPROG.OBJ, PLOT55.OBJ, and VTMODE.OBJ also reside on device dv#.
4. You want the resulting executable file, MYPROG.SAV, to reside on device dv#.

Appendixes

Appendix A
REAL–11/MNC Software Generation Procedure

This appendix explains the procedures for building and using the REAL–11/MNC software. The chapter describes the REAL–11/MNC distribution kit, the dialog for generating the REAL–11/MNC software that is appropriate for your application, and the procedure for linking the REAL–11/MNC software with your FORTRAN IV program. The chapter also provides examples of MNCGEN.

A.1 The Distribution Media for REAL–11/MNC

The REAL–11/MNC software is distributed on one RL01/RL02 hard disk or two RX02 diskettes.

Each medium contains all the files required to generate the REAL–11/MNC software suitable for your application, including all the source files for the REAL–11/MNC software. The RL01/RL02 disk contains all the following files; RX02 diskette number 1 contains the first group of files; RX02 diskette number 2 contains the second group:

```
ADEX1 .FOR   ITEX1.FOR   MGEN07.MAC   MGEN18.FOR   MNCSNG.VER
ADEX2 .FOR   KWEX1.FOR   MGEN8.FOR    MGEN19.FOR   MSEX1 .FOR
BMEX1 .FOR   KWEX2.FOR   MGEN9.FOR    MGEN20.FOR   MSEX2 .FOR
BTEX1 .FOR   MGENBD.COM  MGEN10.FOR   MNCLIB.VER   MSEX3 .FOR
BTEX2 .FOR   MGEN00.FOR  MGEN11.FOR   MNCMUL.DSC   MTMP  .DIS
CIOEX1.FOR   MGEN01.FOR  MGEN12.FOR   MNCMUL.OBJ   MVERN .DIS
DAEX1 .FOR   MGEN02.FOR  MGEN13.FOR   MNCMUL.VER   VOEX1 .FOR
DIEX1 .FOR   MGEN03.FOR  MGEN14.FOR   MNCOVR.MAC   VOEX2 .FOR
DOEX1 .FOR   MGEN04.MAC  MGEN15.MAC   MNCOVR.OBJ   VOEX3 .FOR
FTN2  .DAT   MGEN05.FOR  MGEN16.FOR   MNCSNG.DSC   VTMODE.MAC
FTN2  .FOR   MGEN06.FOR  MGEN17.FOR   MNCSNG.OBJ
```

```
MDSC  .MAC   MNC09 .MAC   MNC30 .MAC   MNC51 .MAC   MNC72 .MAC
MMLB  .MAC   MNC10 .MAC   MNC31 .MAC   MNC52 .MAC   MNC73 .MAC
MNCA  .MAC   MNC11 .MAC   MNC32 .MAC   MNC53 .MAC   MNC74 .MAC
MNCB  .FOR   MNC12 .MAC   MNC33 .MAC   MNC54 .MAC   MNC75 .MAC
MNCF00.FOR   MNC13 .MAC   MNC34 .MAC   MNC55 .MAC   MNC76 .MAC
MNCGEN.SAV   MNC14 .MAC   MNC35 .MAC   MNC56 .MAC   MNC77 .MAC
MNCG2F.FOR   MNC15 .MAC   MNC36 .MAC   MNC57 .MAC   MNC78 .MAC
MNCG2F.SAV   MNC16 .MAC   MNC37 .MAC   MNC58 .MAC   MNC79 .MAC
MNCG2S.FOR   MNC17 .MAC   MNC38 .MAC   MNC59 .MAC   MNC80 .MAC
MNCG2S.SAV   MNC18 .MAC   MNC39 .MAC   MNC60 .MAC   MNC81 .MAC
MNCLST.BLD   MNC19 .MAC   MNC40 .MAC   MNC61 .MAC   MNC82 .MAC
MNCNOL.BLD   MNC20 .MAC   MNC41 .MAC   MNC62 .MAC   MNC83 .MAC
MNC00 .MAC   MNC21 .MAC   MNC42 .MAC   MNC63 .MAC   MNC84 .MAC
MNC01 .MAC   MNC22 .MAC   MNC43 .MAC   MNC64 .MAC   MNC85 .MAC
MNC02 .MAC   MNC23 .MAC   MNC44 .MAC   MNC65 .MAC   MNC86 .MAC
MNC03 .MAC   MNC24 .MAC   MNC45 .MAC   MNC66 .MAC   MNC87 .MAC
MNC04 .MAC   MNC25 .MAC   MNC46 .MAC   MNC67 .MAC   MNC88 .MAC
MNC05 .MAC   MNC26 .MAC   MNC47 .MAC   MNC68 .MAC   MNC89 .MAC
MNC06 .MAC   MNC27 .MAC   MNC48 .MAC   MNC69 .MAC   MNC90 .MAC
MNC07 .MAC   MNC28 .MAC   MNC49 .MAC   MNC70 .MAC   MNC91 .MAC
MNC08 .MAC   MNC29 .MAC   MNC50 .MAC   MNC71 .MAC   MNC92 .MAC
```

A.2 Introduction to MNCGEN

MNCGEN is the software-generation process for REAL–11/MNC. The name of the program that performs the process is MNCGEN.SAV. It allows you to customize the distributed REAL–11/MNC software to match the specific needs of your application.

MNCGEN is an optional procedure. Normally you can use one of the distributed REAL–11/MNC library files, rather than performing the MNCGEN procedure. Therefore, before performing MNCGEN, read Section 4.2.1, Selecting a REAL–11/MNC Library File, and determine whether you actually need to perform the MNCGEN.

A.2.1 Planning for the MNCGEN Dialog

Before beginning the software-generation procedure, you should gather all the information required to respond appropriately to the dialog questions. The information you need includes the following:

- What the requirements are for performing the MNCGEN (see Section A.3.3)
- What options (for example, expanded comments) are available in the dialog
- Which MNC–series modules are contained in the target configuration
- What the vector and Control Status Register (CSR) addresses are for each MNC–series module

Other required information is requested in the MNCGEN dialog itself.

Before running the MNCGEN program, you should read an entire dialog. Examples are given in Section A.5 of this appendix. Understanding the MNCGEN dialog and the implications of the various responses is important for making appropriate choices.

Also, before running MNCGEN, see Section 4.1 to verify that you have the minimal software needed for installing FEP–11.

A.2.2 Executing the MNCGEN Program

The MNCGEN program consists of a series of questions to which you must respond in order to create the REAL–11/MNC software for the MNC–series configuration in your target system.

The procedure for running MNCGEN has two phases. The objective of Phase 1 is to create a new file called MDSC.MAC, which symbolically describes the MNC–series configuration of your target system. The steps required to create MDSC.MAC are explained in detail in Section A.3.

The objective of Phase 2 of MNCGEN is to create a new REAL–11/MNC library file called MNCLIB.OBJ, which is based on your responses in Phase 1. This file is customized to match your target configuration according to the answers you supply in the dialog. Phase 2 is described in Section A.4.

It is not necessary to complete Phase 2 at the same session in which you complete Phase 1. Because the assembly process takes about 30 minutes with two RL01/RL02 disk drives or one hour with RX02 diskette drives, you may prefer to run Phase 2 of MNCGEN at a later time. You also have the option to produce a hard-copy listing of the assembled files. The dialog in Phase 1 allows you to choose from these options.

You must be using the SJ or FB monitor when you execute the MNCGEN procedure. You also must execute the procedure on the system console terminal. To start the MNCGEN dialog, make the device assignments discussed in Section A.3.3, and type:

```
.RUN I:MNCGEN
```

A.3 The MNCGEN Dialog: Phase 1

This section explains in detail the MNCGEN dialog for Phase 1. It describes all text, questions, and responses displayed during the dialog. However, not all of these questions appear in every MNCGEN dialog because some questions depend on your responses to previous questions.

A.3.1 Choice of Dialog Form

The MNCGEN dialog has two forms: expanded (or long) and short. The expanded form provides an explanation of each dialog question before the question itself appears. The short form limits the dialog to the questions themselves; it provides no explanatory information, unless you specifically request "help" by typing H.

You must choose either the expanded or short form of the dialog. If you have run similar system-generation programs before and understand each question of the dialog and its appropriate response, you may find that the short form saves time. However, if you are unfamiliar with the dialog, the long form is recommended.

You indicate the form you wish to use by answering the first question in the dialog with Y (for yes) or N (for no):

```
DO YOU WANT THE MNCGEN DIALOG IN THE EXPANDED FORM?[Y/N]:N
```

Your response to this question governs the remainder of the session.

A.3.2 Question Types and Defaults

The MNCGEN dialog for Phase 1 consists of a series of questions that you can answer in one of three ways (explained in the expanded form):

- With a Y (for yes) or N (for no). This form usually concerns options of this procedure (see the first dialog question given above) or of the REAL–11/MNC software (for example, `DOES YOUR TARGET SYSTEM CONTAIN AN MNCAG?`).
- With a decimal number. This form usually pertains to the number of certain hardware items (for example, `HOW MANY MNCDIs ARE CONTAINED IN YOUR TARGET SYSTEM?`).
- With an octal number. This form usually pertains to the vector and CSR addresses of the MNC–series modules (for example, `WHAT IS THE VECTOR ADDRESS OF MNCAD #n?` and `WHAT IS THE CSR ADDRESS OF MNCAA #n?`).

Terminate all responses to these questions by pressing the RETURN key on your terminal.

Following each question is a range of acceptable answers enclosed in square brackets. Each range is first categorized by one of the following letters:

- Y/N for yes or no. Your response must be Y or N.
- O for octal. Your response must be an octal number.
- D for decimal. Your response must be a decimal number.

Following O or D is a colon and a range of acceptable values. Following the closing square bracket is a second colon and the default answer that the program uses if you respond by pressing only the RETURN key. Examples follow:

```
DOES YOUR TARGET SYSTEM CONTAIN AN MNCAG?[Y/N]:Y
```

In this case, your choice is Y or N, and the default value is N.

```
HOW MANY MNCDIs ARE CONTAINED IN YOUR TARGET SYSTEM? [D:0-8]:8
```

In this case, you must specify a decimal number in the range 0 to 1. The default value is 1.

```
WHAT IS THE VECTOR ADDRESS OF MNCAD #n?[O:120-474]:400
```

In this case, you must specify an octal value in the range 120 to 474. The default value is 400.

If you respond to such questions with a number that is out of the acceptable range, MNCGEN issues the following message:

```
**ERROR**
RESPONSE IS OUT OF RANGE!
```

It then repeats the request for the appropriate address.

If you respond to any of these questions with a number that matches a number given in response to an earlier question, the following message appears:

```
**ERROR**
ADDRESS GIVEN CONFLICTS WITH PREVIOUS RESPONSES.
```

MNCGEN then repeats the request for the appropriate address.

A.3.3 Requirements for MNCGEN

For the MNCGEN procedure to function properly, you must satisfy certain requirements, as explained below and in the MNCGEN dialog.

NOTE

You cannot execute the MNCGEN procedure or verify generated REAL–11/MNC software (see Section A.5.5) under the XM monitor. Therefore, if you plan to use REAL–11/MNC under the XM monitor, the SJ or FB monitor must be in operation while you perform the MNCGEN and verification procedures.

The first requirement is to copy your distributed REAL–11/MNC software and your latest MDSC.MAC file onto a compatible storage medium. This measure ensures that you do not lose these files.

You then must assign specific names to the devices that will contain your various files. You make these assignments using the RT–11 ASSIGN command, as follows:

I If you plan to complete Phase 2 as part of the current procedure and in that way create a new REAL–11/MNC library file (MNCLIB.OBJ), then you must assign the name I to the device that contains a copy of the distributed REAL–11/MNC software. For example:

```
ASSIGN DLn: I   (for RL01s)

ASSIGN DYn: I   (for RX02s)
```

where n is the unit number of the device.

D You must assign the name D to the device that contains the copy of the MDSC.MAC file. This file is used in the current procedure for default answers. Usually it is either the distributed MDSC.MAC file or the one created by a previous MNCGEN procedure. For example:

```
ASSIGN DLn: D   (for RL01s)

ASSIGN DYn: D   (for RX02s)
```

where n is the unit number of the device.

N You must have assigned the name N to the device that will contain the new MDSC.MAC file and (after Phase 2 is completed) the new MNCLIB.OBJ file. MNCLIB.OBJ is the actual REAL–11/MNC library file. This device will also be used in Phase 2 for intermediate file storage. Depending on the attributes of the REAL–11/MNC object file being created, between 150 and 350 blocks of contiguous storage space are required.

```
ASSIGN DLn: N   (for RL01s)

ASSIGN DYn: N   (for RX02s)
```

where n is the unit number of the device.

L You have the option during the MNCGEN procedure to produce assembly listings. If you plan to choose this option, you must assign the name L to the device that will contain your listing files. For example:

```
ASSIGN LP: L
```

Do not assign the name N to the same device that you have named I or D. By creating new files on a device that contains old files with the same names, you replace the old files.

The MNCGEN dialog prompts you regarding these various requirements with the following question:

```
HAVE YOU MET THE REQUIREMENTS TO PERFORM THIS PROCEDURE?[Y/N]:N
```

If you answer with N to this question, the system issues the following message:

```
STOP - REAL-11 GENERATION REQUIREMENTS NOT MET!
```

If you answer with Y but have not assigned the names N or D to appropriate devices, the following system error message appears:

```
?Err 27 Attempt to use device not in system
```

If you answer with Y but the MDSC.MAC file is not on the device named D, the following system error message appears:

```
?Err 28 Open Failed for file
```

You cannot continue with the dialog until you have met the requirements. You must begin the dialog again.

Similar messages appear during Phase 2 of the MNCGEN dialog if you have not assigned the names I, N, and (when you request listings) L to the appropriate devices. Phase 2 cannot continue until you have made these required assignments.

A.3.4 Option to Postpone the Assembly Process (Phase 2)

As mentioned earlier, Phase 2 of the MNCGEN procedure consists of the creation of a new REAL–11/MNC library file called MNCLIB.OBJ. This file, which is based on your responses in the MNCGEN dialog, is customized to match your target configuration.

Because the process requires several minutes to complete, however, you have the option to postpone Phase 2 of the MNCGEN procedure. The system asks the question:

```
SHOULD THIS PROCEDURE EXECUTE PHASE 2?[Y/N]:Y
```

If you respond with N, the dialog continues only until Phase 1 is complete. Then, at a later time, you can complete Phase 2 by executing one of two indirect files supplied with your distributed REAL–11/MNC software. These files are MNCNOL.BLD, which you use if you do not want listings of your assembled files, and MNCLST.BLD, which you use if you do want such listings.

These indirect command files, though run at a later time, require the device assignments that you specify in Phase 1. If you decide to postpone Phase 2, therefore, be sure that you assign the name N to the device containing the MDSC.MAC file created during Phase 1.

If you respond with Y to the question, indicating that you want to produce your new object file now, the system asks you a related question:

```
DO YOU WANT TO PRODUCE ASSEMBLY LISTING FILES?[Y/N]:N
```

If you respond with Y to this question, the system produces your assembled source listing files on the device that you named as L. If device L is not a hard-copy device, the file names are of the form MNCxx.LST, where xx corresponds to the related MNCxx.MAC or MNCxx.FOR source file distributed with the software.

These listing files are numerous and large. In general, you should avoid producing these files.

A.3.5 MNCGEN Hardware Checking

The MNCGEN dialog attempts to determine if the target system that you are specifying in the current MNCGEN session matches the actual configuration of the MNC–series modules that are presently installed in your system:

```
IS THIS SYSTEM, AS PRESENTLY CONFIGURED, THE TARGET SYSTEM OF THIS
SOFTWARE-GENERATION PROCEDURE?[Y/N]:N
```

If you answer with Y to this question, MNCGEN attempts to verify that the hardware-selection information provided in the dialog corresponds to the hardware on the system.

As noted in the dialog, this additional checking feature requires that your system be used only for the MNCGEN procedure during Phase 1.

A.3.6 Run–Time Hardware Checking

The REAL–11/MNC software includes an optional feature that checks for the presence of required MNC–series hardware every time a REAL–11/MNC subprogram is executed. If required hardware is absent, the software returns an error code to the calling program. In some cases, the software returns the error code 305 in the sweepstatus word; in other cases, it returns a value of 0 in the variable ind. See the *REAL–11/MNC Programmer's Reference Manual* for more information.

To enable run-time hardware checking, answer Y to the following question:

```
DO YOU WANT THE REAL-11 SOFTWARE TO MAKE RUN-TIME CHECKS
FOR THE PRESENCE OF REQUIRED LAB MODULES? [Y/N]:Y
```

Run-time hardware checking slows execution of REAL–11/MNC subprograms and occupies additional memory. Therefore, if possible, avoid using this feature. You do not need the feature if your hardware configuration rarely changes or if you are confident that you know which MNC–series hardware is needed by the REAL–11 subprograms you are using.

A.3.7 Specifying MNCAD–Related Information

The REAL–11/MNC system can support one MNCAD. You specify the number of MNCADs by responding to the following question:

```
HOW MANY MNCADs ARE CONTAINED IN YOUR TARGET SYSTEM?[D:0-1]:1
```

If you answer 0, the following REAL–11/MNC subprograms will not be included in the resulting MNCLIB library file:

ADINP (IADINP), ADSWP, ADFAST, ADSUM and MADSTS.

(See Chapter 7 in the *REAL–11/MNC Programmer's Reference Manual* for information on these subprograms). Also if you answer 0, the dialog begins the section of questions related to the MNCKW.

If your answer to this question is 1, you must specify the assigned vector and CSR addresses for your MNCAD. The default values for these addresses are the standard locations, and usually they correspond to the switch settings on the module. However, if you have reason to think that these addresses are not set to the standard locations, see the *DECLAB–11/MNC User's Guide* for instructions on how to verify or change the switch settings.

For the MNCAD, the system requests these addresses as follows:

```
WHAT IS THE VECTOR ADDRESS OF MNCAD #0?[D:120-474]:400

WHAT IS THE CSR ADDRESS OF MNCAD #0?[D:171000-171400]:171000
```

The MNCAG is a preamplifier that provides differential-mode amplification or attenuation of signals on four A/D input channels. The number of allowable MNCAGs depends on the number of MNCAMs in your system. The maximum number is two for each MNCAM and one for the MNCAD.

MNCGEN prompts you as follows:

```
DOES YOUR TARGET SYSTEM CONTAIN AN MNCAG ?[Y/N]:Y
```

If an MNCAG is connected to an MNCAD or MNCAM in your system, you can specify the gain value that you want for channels that will be used in A/D sampling. You can also default this gain value. If the gain-selection switch on the front panel of the MNCAG is set to a fixed gain, the hardware defines the default gain value.

However, if the gain-selection switch on the MNCAG is set to P (for "programmable"), the REAL–11/MNC software must use the default gain value that you define during the MNCGEN dialog.

You set a default gain value for that situation by responding to the following question:

```
WHAT SHOULD BE THE DEFAULT GAIN VALUE WHEN THE GAIN-SELECTION
SWITCH OF AN MNCAG IS SET TO PROGRAMMABLE?:
,5(0), 5(1), 50(2), OR 500(3)? [D:0-3]:0
```

To specify the default gain value, you enter the number given in parentheses after the desired gain value. For example, to specify a default gain value of 5, you enter the number 1.

The MNCTP module allows ANSI standard thermocouples to be connected to the A/D.

```
DOES YOUR TARGET SYSTEM CONTAIN AN MNCTP?[Y/N]:Y
```

Occasionally a sweep can cause a data overrun, which occurs when the system cannot sample input at the requested rate. This situation is not indicated until the sweep is underway. The MNCGEN dialog requires that you select one of three ways to handle a data overrun. You can:

1. Immediately terminate the sampling process when the data overrun is detected. This is the usual way to handle a data overrun. It is the only way that REAL–11/MNC can handle a data overrun for a clock-driven sweep involving any module other than MNCAD.

   If you select this option and a data overrun occurs, the value 37777 (octal) is stored in the data buffer, and the sweep-status word returns an error code. (See Appendix A of the *REAL–11/MNC Programmer's Reference Manual.*)

2. Indicate by a predetermined value in the data buffer that a data overrun has occurred. This option is useful when the sampling process is not driven by clock overflows.

   As with the first option, a data overrun causes a data-overrun indicator to be stored in the data buffer, and the sweep-status word returns an error code. (See Appendix A of the *REAL–11/MNC Programmer's Reference Manual.*) However, unlike the first option, this option allows the sweep to continue, and the data-overrun indicator is placed in the data buffer whenever a data overrun occurs. (See below.)

3. Ignore all data overruns. This choice, which slightly enhances speed, is nonetheless dangerous because you may lose data without any indication that it has been lost.

The system prompts you with one question at a time. If you respond to the first question with Y, the system does not ask the other two. If you respond to the first question with N and the second with Y, the system does not ask the third. The dialog reads as follows:

```
HOW DO YOU WANT REAL-11/MNC TO HANDLE DATA OVERRUNS DURING
AN ANALOG INPUT SWEEP ("ADSWP")?

1) IMMEDIATELY TERMINATE THE SAMPLING PROCESS[Y/N]:Y

2) INDICATE DATA OVERRUN BY A PREDETERMINED BIT PATTERN IN THE DATA
   BUFFER[Y/N]:N

3) IGNORE DATA OVERRUNS[Y/N]:N
```

You must respond with Y to one of these quesions. If you respond to all three with N, the system responds as follows:

```
YOU MUST ANSWER "Y" TO ONE OF THE QUESTIONS!
```

It then repeats the first question.

If you select the second option, the MNCGEN dialog also requests that you specify a numeric value that the system can return to you as a signal indicating a data overrun has occurred. This value should be an unlikely input value and one that you can easily recognize as an indicator and not a datum in the data buffer.

MNCGEN asks:

```
WHAT VALUE SHOULD BE USED TO INDICATE AN ANALOG DATA-OVERRUN?
[0:0-177777]:37777
```

Tip: The four leftmost bits in the data buffer (bits 15 to 12) are used to indicate gain and type of input. Therefore, you could choose a data-overrun value that does not match, in these four bits, the quantity being measured in the experiment.

Once you have supplied this value, MNCGEN adds the following comment:

```
NOTE: THIS VALUE WILL ALSO BE USED TO REPORT
"(I)ADINP", "ADFAST", AND "ADSUM" ERRORS.
```

If you are using the ADINP subprogram and you do not select the second option for handling data overruns, the software returns a value of 37777 (octal) to indicate an error. If you are using the ADINP subprogram and you do select the second option, the value you specify in the question above is the value returned to indicate an error.

A.3.8 Specifying MNCKW–Related Information

The REAL–11/MNC software can support one or two MNCKWs. However, in applications that require two MNCKWs, the operations performed by each MNCKW are distinct and separate. Before responding to MNCGEN questions regarding MNCKWs, therefore, you should know the different ways in which the two MNCKWs are used.

In general, the first MNCKW (associated with the CLOCKA subprogram) is used for clock-driven sampling and for gathering data for time-interval histograms. This MNCKW is referred to as the "primary" clock, or MNCKW #0 in this MNCGEN procedure.

The second MNCKW (associated with the CLOCKB subprogram) is used to maintain a software fine-resolution time base that supports time-stamping operations. This MNCKW is referred to as the "secondary" clock, or MNCKW #1 in this MNCGEN procedure.

When two MNCKWs are used, the primary clock must be installed to the left of the secondary clock, and to the right of any MNCAD that may be present. As the MNCGEN dialog warns, if this convention is not observed, the software will malfunction, causing unpredictable results.

For details regarding one–MNCKW and two–MNCKW configurations, refer to the *REAL–11/MNC Programmer's Reference Manual* and the *DECLAB–11/MNC User's Guide.*

The MNCGEN dialog prompts you as follows:

```
HOW MANY MNCKWs ARE CONTAINED IN YOUR TARGET SYSTEM? [D:0-2]:2
```

If you answer 0, the following REAL–11/MNC subprograms are not included in the resulting MNCLIB library:

CLOCKA, CLOCKB, and GTHIST.

(See Chapter 6 in the *REAL–11/MNC Programmer's Reference Manual* for information on these subprograms.) In addition, any requests for clock-driven sweeps will result in an error. The dialog now begins the section on information related to the MNCDI.

If your answer to this question is not 0, the system requests a vector address and a CSR address for each MNCKW, as follows:

```
WHAT IS THE VECTOR ADDRESS OF MNCKW #0?[O:120-474]:440

WHAT IS THE CSR ADDRESS OF MNCKW #0?[O:171000-171400]:171020
```

MNCGEN then asks if you want to include the GTHIST subprogram in your library. This subprogram collects data for time-interval histograms. Because of the size of this subprogram and because it is not required for many applications, you can omit it. If you do not intend to use GTHIST, respond with N to the following question:

```
DO YOU WANT THE "GTHIST" SUBPROGRAM IN YOUR LIBRARY?[Y/N]:Y
```

You also have the option to exclude from your customized library the capability of performing clock-driven data transfers through several MNC–series modules operating in parallel. This feature requires additional software overhead, even when an operation activates only one clock-driven sampling sweep. Therefore, if your application does not require parallel clock-driven sampling through several MNC–series modules, omit this feature by responding with N to the following question:

```
DO YOU REQUIRE PARALLEL CLOCK-DRIVEN TRANSFERS THROUGH SEVERAL
MNC-SERIES MODULES?[Y/N]:N
```

A.3.9 Specifying MNCDI–Related Information

The REAL–11/MNC software supports a maximum of eight MNCDIs. You indicate how many MNCDIs you plan for your target system by responding to the following question:

```
HOW MANY MNCDIs ARE CONTAINED IN YOUR TARGET SYSTEM?[D:0-8]:8
```

If you answer 0, the REAL–11/MNC subprogram DIGO is not included in the resulting MNCLIB library. Furthermore, any call to DISWP, DINP, or MDISET will produce an error message that indicates a needed device is not present. The dialog now proceeds to the section on information related to the MNCDO.

The software identifies each MNCDI by a unit number. The software assigns the unit numbers according to the order in which you supply vector addresses and CSR addresses for the MNCDIs in responding to the MNCGEN questions. For example, assume that you have responded with 2 to the question shown above. MNCGEN then requests two sets of addresses, one set for MNCDI #0 and one set for MNCDI #1.

```
WHAT IS THE VECTOR ADDRESS OF MNCDI #0?[O:120-474]:120

WHAT IS THE CSR ADDRESS OF MNCDI #0?[O:171000-171400]:171160

WHAT IS THE VECTOR ADDRESS OF MNCDI #1?[O:120-474]:130

WHAT IS THE CSR ADDRESS OF MNCDI #1?[O:171000-171400]:171170
```

Because data overruns can occur in digital-input sweeps, MNCGEN asks how you want the software to handle such data overruns. The options are identical to those mentioned earlier regarding input from A/D sweeps (see Section A.3.7). You must respond with Y to one of the following questions:

```
HOW DO YOU WANT REAL-11/MNC TO HANDLE DATA OVERRUNS DURING
A DIGITAL-INPUT SWEEP ("DISWP")?

1) IMMEDIATELY TERMINATE THE SAMPLING PROCESS[Y/N]:Y

2) INDICATE DATA OVERRUN BY A PREDETERMINED BIT PATTERN IN THE DATA
   BUFFER[Y/N]:N

3) IGNORE DATA OVERRUNS[Y/N]:N
```

As with A/D sweeps, if you select the second option, you must specify a numeric value that will appear in a data buffer to indicate the occurrence of a data overrun. This value should be one that you can easily recognize as an indicator and not a datum.

```
WHAT VALUE SHOULD BE USED TO INDICATE A DIGITAL DATA OVERRUN?
[0:0-177777]:177777
```

Tip: Each bit in the data word is associated with an input line. If any input line is not used to indicate a datum, then set to 1 the bit corresponding to that line as an indicator of a data-overrun condition. If there is a known range of valid data, you can also choose a value outside that range to indicate the data-overrun condition.

A.3.10 Specifying MNCDO–Related Information

The REAL–11/MNC software supports a maximum of eight MNCDOs. With the exception of questions related to data overruns, the MNCGEN questions for MNCDOs are identical to those asked for MNCDIs (see Section A.3.9). As with MNCDIs, the software recognizes each MNCDO by a unit number that you assign according to the order in which you supply addresses for the MNCDOs in response to MNCGEN questions.

The questions read as follows:

```
HOW MANY MNCDOs ARE CONTAINED IN YOUR TARGET SYSTEM?[0:0-8]:8
```

If you answer 0, any call to DOUT and DOSWP will produce an error message that indicates a needed device is not present, and the dialog proceeds to the section on information related to the MNCAA.

```
WHAT IS THE VECTOR ADDRESS OF MNCDO #0?[0:120-474]:340

WHAT IS THE CSR ADDRESS OF MNCDO #0?[0:171000-171400]:171260
```

A.3.11 Specifying MNCAA–Related Information

The REAL–11/MNC software supports a maximum of eight MNCAAs. You indicate the number of MNCAAs for your target system by responding to the following MNCGEN question:

```
HOW MANY MNCAAs ARE CONTAINED IN YOUR TARGET SYSTEM?[0:0-8]:8
```

Each MNCAA has four channels for output. The software addresses a particular channel on a particular MNCAA according to global channel numbers. The software assigns these numbers according to the order in which you supply CSR addresses for the MNCAAs. (For example, the first MNCAA has channels 0 to 3, the second has channels 4 to 7, and so forth.)

To indicate the CSR address for each MNCAA, respond to the following question, once for each MNCAA:

```
WHAT IS THE CSR ADDRESS OF MNCAA #0?[O:171000-171400]:171060
```

If you answer 0, the REAL–11/MNC subprograms DASWP and AOUT are not included in the resulting MNCLIB library file.

A.3.12 Specifying SLU–Related Information

The REAL–11/MNC software supports up to 20 SLU ports, Serial Line Units dedicated to instruments. These ports are contained in the DLV11J module(s) in the restricted part of the MNC chassis. The standard MINC hardware configuration contains four SLUs; one for the console, one for a serial line printer, and two for instruments. You may not use an SLU that is being used for the RT–11 multi-terminal support option.

```
HOW MANY SLUs IN YOUR TARGET SYSTEM WILL BE DEDICATED
TO INSTRUMENTS?[D:0-20]:2
```

Each Serial Line Unit input channel uses an internal ring buffer for storage of input data. This data is entered into the buffer asynchronously; your program removes data from the buffer. Therefore, the larger the size of the buffer, the less frequently your program must remove data from it. The size of the buffer should be at least as large as the longest message that your program will read, and it must be an even number.

```
WHAT IS THE SIZE, IN BYTES, OF THE INPUT RING BUFFER
FOR EACH CHANNEL? [D:32-1024]: 82
```

Each Serial Line Unit output channel uses an internal ring buffer for storage of data to be transmitted. This buffer can be smaller than the length of the largest message; however, the larger the buffer, the less frequently your program must remove data from it.

```
WHAT IS THE SIZE, IN BYTES, OF THE OUTPUT RING
BUFFER FOR EACH CHANNEL?[D:16-1024]: 40
```

In the software, an SLU is identified by a unit number. The software assigns these numbers according to the *order* in which you provide the vector and CSR addresses. If you have only one DLV11J module, obtain the SLU vector and CSR addresses from the *DECLAB–11/MNC User's Guide*. If you have more than one DLV11J module, obtain the SLU vector and CSR addresses from DIGITAL.

```
WHAT IS THE VECTOR(recv) ADDRESS OF SLU #0?[O:120-370]:300

WHAT IS THE CSR(recv) ADDRESS OF SLU #0?
[O:160000-177770]:176500

WHAT IS THE VECTOR(recv) ADDRESS OF SLU #1? [O:120-370]:310

WHAT IS THE CSR(recv) ADDRESS OF SLU #1?
[O:160000-177770]:176510
```

Data overrun occurs when the system cannot transfer data at the rate requested. It is not detected until the sampling process has begun. Three methods of handling this situation are available in REAL–11/MNC for serial I/O (CIN/COUT). You must select one.

These methods are:

1. Immediately terminate the sampling process when data overrun is detected. This is the usual way to handle the problem.
2. Indicate by a predetermined bit pattern in the data buffer that a data overrun has occurred. This method is useful when the sampling process is not clock driven.
3. Ignore all data-overrun conditions. This is a fast but dangerous choice since data may be lost with no indication.

```
HOW DO YOU WANT REAL-11/MNC TO HANDLE DATA OVERRUNS DURING A SERIAL
INPUT SWEEP

1) IMMEDIATELY TERMINATE THE SAMPLING PROCESS[Y/N]:Y

2) INDICATE DATA OVERRUN BY A PREDETERMINED BIT PATTERN IN THE DATA
   BUFFER[Y/N]:N

3) IGNORE DATA OVERRUNS[Y/N]:N
```

Phase 1 of the MNCGEN procedure ends after you have supplied information related to the SLUs. To indicate the end of Phase 1, MNCGEN prints the following message at the terminal:

```
*************** END REAL-11/MNC GENERATION PHASE 1 ***************
```

A.4 The MNCGEN Procedure: Phase 2

The objective of Phase 2 of MNCGEN is to assemble the source files distributed with your REAL–11/MNC software, together with the source file created in Phase 1. The resulting object files are used to create a new REAL–11/MNC library file called MNCLIB.OBJ. This file is customized to match your target configuration according to the answers you supplied in Phase 1.

Unless you indicated in Phase 1 that you wanted to postpone the assembly process, MNCGEN immediately begins Phase 2 after Phase 1 ends.

If you indicated in Phase 1 that you wanted to postpone the assembly process, you can complete Phase 2 at the time you select by executing one of the following indirect command files:

MNCNOL.BLD	Use this file if you do not want listings of your assembled files.
MNCLST.BLD	Use this file if you do want listings of your assembled files.

These indirect command files are supplied with your distributed REAL–11/MNC software. Though run at a later time, these files require the device assignments that you specified in Phase 1 (see Section A.3:3). For example, you must have assigned the new MDSC.MAC file created during Phase 1 to the device named N.

To indicate the beginning of Phase 2, MNCGEN prints the following message at the terminal:

```
*************** REAL-11/MNC RT GENERATION PHASE 2 ***************
```

To indicate the end of Phase 2, MNCGEN prints the following message at the terminal:

```
************* END REAL-11/MNC RT GENERATION PHASE 2 *************
```

When Phase 2 has ended, you can link your new REAL–11/MNC library file with the object file of your FORTRAN IV program(s).

A.5 Sample MNCGEN Dialogs

This section provides three sample MNCGEN dialogs. All three dialogs result in the generation of exactly the same REAL–11/MNC library. However, each uses a different form that demonstrates various features of the MNCGEN procedure.

- The first sample demonstrates the use of the expanded, or long, form of the MNCGEN dialog.
- The second sample demonstrates the use of the short form of the MNCGEN dialog.
- The third sample demonstrates the "help" feature and various nonfatal error reports, which are usually caused by inappropriate responses to MNCGEN questions.

A.5.1 Sample MNCGEN Dialog: Long Form

The sample MNCGEN dialog given below demonstrates use of the expanded (or long) form of the dialog.

```
.DATE 20-AUG-81RET
.TIME 10:45RET
.ASSIGN DL0: NRET
.ASSIGN DL1: IRET
.ASSIGN DL1: DRET
.RUN I:MNCGENRET

REAL-11/MNC/RT SOFTWARE GENERATION PROCEDURE
```

```
********** REAL-11/MNC GENERATION PHASE 1 **********

The system time is  10:45:29

The system date is  20-AUG-81


Some additional information concerning this procedure and its
effects is available in Appendix A of the RT-11/FEP and RT-11/FRP
Installation and User's Guide.


You can select either an expanded form or a short form of this
dialogue.  The expanded form explains each step in this gener-
ation procedure.  The short form, on the other hand, does not
explain any question.  However, if you do not select the expand-
ed form and do not understand a particular question, respond with
"H" for help.  This response will cause the expanded comment for
that particular question to be printed.


DO YOU WANT THE MNCGEN DIALOGUE IN THE EXPANDED FORM?[Y/N]:N     Y (RET)


This procedure is executed in two phases.  The end result is a new
file containing a library of the REAL-11 software routines which
has been tailored to your system configuration.  This file can be
used as is, or added to an existing library file.  See Appendix A
of the RT-11/FEP and RT-11/FRP Installation and User's Guide
for further information.
The purposes of the two phases of this procedure are:

a) In Phase 1, to create a NEW file called "MDSC.MAC", which symbol-
   ically describes the configuration of the MNC-series modules for
   your target system.  Phase 1 also creates a  verification program,
   "MVERN.FOR", which is tailored specifically to your configuration. See
   Appendix A of the RT-11/FEP and RT-11/FRP Installation and User's Guide
   for software  verification instructions.

b) In Phase 2, to create a NEW library file called "MNCLIB.OBJ",
   using the description provided in Phase 1 of this procedure.
   The new file contains the customized REAL-11 software that matches
   your target hardware configuration and software needs.


Phase 1 of the generation procedure consists of a series of questions
that you may answer in one of three ways:

1) With "Y" or "N", to indicate yes or no.  These questions usually
   concern features of this procedure or of the REAL-11/MNC software.

2) With a decimal number.  These questions usually pertain to the
   number of certain hardware items.

3) With an octal number.  These questions usually concern the vector
   and control-status-register (CSR) addresses of the MNC-series
   modules.

Terminate all answers by striking the RETURN key (<RET>).


Each question in the dialogue is followed by an acceptable range of
answers.  This range is indicated by values enclosed within square
brackets.  If the range is given in octal, indicated by an "0:" pre-
ceding the actual range, then your response must also be an octal
```

```
value.  If the range is given in decimal, indicated by a "D:" preced-
ing the actual range, then your response must also be a decimal value.
Following the range is the response that the procedure will use
if you respond to a question by typing only a RETURN.

EXAMPLES: [Y/N]:Y          => range is Y or N; default is Y
          [D:0-8]:1        => range is decimal 0 to 8; default is 1
          [O:130-474]:130 => range is octal 130 to 474; default is 130


The following is a list of requirements that MUST be satisfied before
the software-generation procedure can function properly:

1) You must have made a copy of your distributed REAL-11/MNC software
   and your latest MDSC.MAC MNC-series-module descriptor file.

2) You must have assigned the name "D" to the device that contains the
   copy of MDSC.MAC to be used in this procedure for the default an-
   swers.  This file is usually the distributed file MDSC.MAC, or the
   latest one created by a previous execution of Phase 1 of the generation
   procedure.  You can assign a logical name to a physical device
   by using the RT-11 "ASSIGN" command.
   EXAMPLE: .ASSIGN DL0 D

3) You must have assigned the name "N" to the device that will contain
   the NEW files MDSC.MAC, and MVERN.FOR.  When Phase 2 is executed,
   MNCLIB.OBJ will also be directed to this device.

      NOTE: This device, in Phase 2, will also be used for intermediate
            file storage.  Depending on the attributes of the REAL-11/MNC
            software being created, between 150 and 350 blocks of
            CONTIGUOUS storage space are required.

When Phase 2 of this procedure is executed, the following additional
requirements apply:

4) You must have assigned the name "I" to the device that contains a
   COPY of the distributed REAL-11/MNC software.

5) If you intend to get assembly listings of the REAL-11 files, you
   must assign the name "L" to the device that will receive your
   listing files.

CAUTION: NEW files created on a device having OLD files with the same
         names REPLACE the old files.  Therefore, devices "I" and "D"
         should differ from device "N".

HAVE YOU MET THE REQUIREMENTS TO PERFORM THIS PROCEDURE?[Y/N]:N  Y (RET)

The software-generation procedure is performed in two phases:

1) In Phase 1, a new MDSC.MAC file is created on device "N".  This
   file symbolically represents the configuration of MNC-series
   modules of your target system and any optional software support
   that you have selected during the generation dialogue.

2) Through a lengthy assembly process, a new REAL-11/MNC library
   file, MNCLIB.OBJ, is created, which reflects the description given
   in Phase 1 of your configuration of MNC-series modules and software
   requirements.
```

You may perform this second phase independently at some other time. To do so you must execute the indirect command file "MNCNOL.BLD" if you do not want assembly listings of the REAL-11 source files, or the indirect command file "MNCLST.BLD" if you do want listings. Both files are part of your REAL-11/MNC software distribution kit.

These command files assume that device names "I", "N", and, if required, "L" have been assigned exactly as if the second phase were being performed as part of this current procedure.

If you decide to perform Phase 2 at a later time, be sure that the file MDSC.MAC, created during Phase 1 of this procedure, is on device "N".

SHOULD THIS PROCEDURE EXECUTE PHASE 2?[Y/N]:Y (RET)

The software-generation procedure involves the assembly of many MACRO source files. These sources are uncommented, and the resulting listing files are many and lengthy.

Therefore, in general, you should NOT choose to obtain listing files.

DO YOU WANT TO PRODUCE ASSEMBLY LISTING FILES?[Y/N]:N (RET)

If this system is the target system for the REAL-11/MNC software and if the MNC-series modules are presently installed, then the procedure will verify that your responses match the installed hardware.

CAUTION: This additional checking feature requires that your system be used solely for this procedure during Phase 1.

IS THIS SYSTEM, AS PRESENTLY CONFIGURED, THE TARGET SYSTEM OF THIS MNCGEN PROCEDURE?[Y/N]:N Y(RET)

Selecting the following feature allows the REAL-11 software to check for the presence of required devices when a program is run. If selected, when a device required for a REAL-11 procedure is not present in the system (for example, an MNCAD is required for the ADSWP routine), an error code is returned to the calling program. If this feature is not selected, a system error results when a program requiring a missing device is run. Such an error causes control to return to the system monitor.

Although this feature is convenient, it is very costly in terms of the software overhead required to support it. Thus, if your hardware configuration does not change very often, or if users of the software are fairly familiar with the hardware required to use the various parts of REAL-11, it is recommended that this feature not be included in your REAL-11 software.

DO YOU WANT THE REAL-11 SOFTWARE TO MAKE RUN-TIME CHECKS FOR THE PRESENCE OF REQUIRED LAB MODULES? [Y/N]:Y (RET)

********** MNCAD-RELATED INFORMATION **********

The REAL-11/MNC software can support one MNCAD having up to 64
input channels, single-ended or differential, connected to it.

HOW MANY MNCADs ARE CONTAINED IN YOUR TARGET SYSTEM?[D:0-1]:1(RET)

The vector address of each MNCAD is determined by the state of a bank
of switches located on the module. The factory switch settings usu-
ally correspond to the default responses provided in this dialogue.
However, information regarding the changing or verification of these
switch settings is provided in the DECLAB-11/MNC user's guide.

WHAT IS THE VECTOR ADDRESS OF MNCAD #0?[O:120-474]:400(RET)

The CSR address of each MNCAD is determined by the state of a bank
of switches located on the module. The factory switch settings usu-
ally correspond to the default responses provided in this dialogue.
However, information regarding the changing or verification of these
switch settings is provided in the DECLAB-11/MNC user's guide.

WHAT IS THE CSR ADDRESS OF MNCAD #0?[O:171000-171400]:171000(RET)

The MNCAG module allows variable gain settings for input on any
of the A/D channels with which it is associated. An MNCAG can
be associated with a block of four channels on either an MNCAD
or an MNCAM.

DOES YOUR TARGET SYSTEM CONTAIN AN MNCAG?[Y/N]:Y(RET)

If your system has an MNCAG assciated with any MNCAD, you will have to
inform the A/D subprograms of your desired gain value when sampling
the affected channels. This value may be defaulted when a subprogram
is called, and the front-panel gain-selector switch on the MNCAG is
set to a fixed gain. In this case the hardware determines the gain
selections.

However, if the gain-selection switch is set to "P", for programmable,
the subprograms need to be 'told a default gain value to use.

The following response will define what gain value is to be used in
that situation.

WHAT SHOULD BE THE DEFAULT GAIN VALUE WHEN THE GAIN-SELECTION SWITCH
ON AN MNCAG IS SET TO PROGRAMMABLE?:
.5(0), 5(1), 50(2), OR 500(3)? [D:0-3]:0(RET)

The MNCTP module allows ANSI standard type thermocouples to be
connected to the A/D.

DOES YOUR TARGET SYSTEM CONTAIN AN MNCTP?[Y/N]:Y(RET)

"Data overrun" occurs when the system cannot transfer data at the rate requested. It is not detected until the sampling process has begun. Three methods of handling this situation are available in REAL-11/MNC for analog-input sweeps (ADSWP). You must select one.

These methods are:

1) Immediately terminate the sampling process when data overrun is detected. This is the usual way to handle the problem.In fact, this is the only way that REAL-11/MNC handles data overruns for all clock-driven sweeps involving any device except the MNCAD.

2) Indicate in the data buffer, via a bit pattern (numeric value) to be defined later, when a datum was missed during the sampling process. This method is useful when the sampling process is not clock driven.

3) Ignore all data-overrun conditions. This is a fast but dangerous choice since data may be lost with no indication.

HOW DO YOU WANT REAL-11/MNC TO HANDLE DATA OVERRUNS DURING
AN ANALOG INPUT SWEEP ("ADSWP")?

1) IMMEDIATELY TERMINATE THE SAMPLING PROCESS[Y/N]:Y N(RET)

2) INDICATE DATA OVERRUN BY A PREDETERMINED BIT PATTERN IN THE
DATA BUFFER[Y/N]:N Y(RET)

The bit pattern, put in a data buffer to indicate the occurrence of a data-overrun condition, should be a numeric value that is a very unlikely input value. Thus this value will be easily recognized later as an indicator and not a datum.

WHAT VALUE SHOULD BE USED TO INDICATE AN ANALOG DATA OVERRUN?
[0:0-177777]:37777 (RET)

NOTE: THIS VALUE WILL ALSO BE USED TO REPORT
"(I)ADINP", "ADFAST", AND "ADSUM" ERRORS.

********** MNCKW-RELATED INFORMATION **********

The REAL-11/MNC software can support one or two MNCKWs. However, if two MNCKWs are installed, each is used by the software to perform separate and distinct functions. MNCKW #0 described in this procedure is referred to as the "primary" clock. It is used for clock-driven sampling or for gathering time-interval histogram data. MNCKW #1 described in this procedure is referred to as the "secondary" clock. It is used by the software to maintain a fine-resolution software time base, which may be used to time stamp asynchronous digital input.

Note: If two MNCKWs are present in your system, the primary module must be installed to the LEFT of the secondary module and to the RIGHT of any MNCADs. If this convention is not observed, the software will malfunction, causing unpredictable results.

```
HOW MANY MNCKWs ARE CONTAINED IN YOUR TARGET SYSTEM?[D:0-2]:2 (RET)

WHAT IS THE VECTOR ADDRESS OF MNCKW #0?[O:120-474]:440 (RET)

WHAT IS THE CSR ADDRESS OF MNCKW #0?[O:171000-171400]:171020 (RET)

WHAT IS THE VECTOR ADDRESS OF MNCKW #1?[O:120-474]:450 (RET)

WHAT IS THE CSR ADDRESS OF MNCKW #1?[O:171000-171400]:171024 (RET)

The subprogram 'GTHIST', which collects time-interval histogram data,
is an optional subprogram.  Because of its size, you may choose to
exclude it from your library.  If you do not intend to use 'GTHIST',
respond with 'N' to the following question.

DO YOU WANT THE 'GTHIST' SUBPROGRAM IN YOUR LIBRARY?[Y/N]:Y (RET)

It is possible to allow the software to perform clock-driven I/O
through several different MNC-series modules in parallel.  However,
this feature requires additional software overhead, even when only
a single clock-driven transfer subroutine is active.  Therfore, if
you do not require parallel clock-driven transfers through several
MNC-series modules, do not include this feature in your library.

DO YOU REQUIRE PARALLEL CLOCK-DRIVEN TRANSFERS THROUGH SEVERAL
MNC-SERIES MODULES?[Y/N]:N (RET)



********** MNCDI-RELATED INFORMATION **********

The REAL-11/MNC software supports up to eight MNCDIs.  These modules
should be located to the right of any MNCADs.  In the software, an
MNCDI is identified by a 'unit number'.  The software assigns these
numbers according to the ORDER in which you provide the vector and
CSR addresses.

HOW MANY MNCDIs ARE CONTAINED IN YOUR TARGET SYSTEM?[D:0-8]:8   1 (RET)

WHAT IS THE VECTOR ADDRESS OF MNCDI #0?[O:120-474]:120 (RET)

WHAT IS THE CSR ADDRESS OF MNCDI #0?[O:171000-171400]:171160 (RET)

'Data overrun' occurs when the system cannot transfer data at the rate
requested.  It is not detected until the sampling process has begun.
Three methods of handling this situation are available in REAL-11/MNC
for digital-input sweeps (DISWP) that are not driven  by the clock
(MNCKW).  You must select one.
```

```
These methods are:

1) Immediately terminate the sampling process when data overrun is
   detected.  This is the usual way to handle the problem.In fact,
   this is the only way that REAL-11/MNC handles data overruns for
   all clock-driven sweeps involving any device except the MNCAD.


2) Indicate in the data buffer, via a bit pattern (numeric value)
   to be defined later, when a datum was missed during the sampling
   process.  This method is useful when the sampling process is not
   clock driven.

3) Ignore all data-overrun conditions.  This is a fast but dangerous
   choice since data may be lost with no indication.


HOW DO YOU WANT REAL-11/MNC TO HANDLE DATA OVERRUNS DURING
A DIGITAL INPUT SWEEP ("DISWP")?


1) IMMEDIATELY TERMINATE THE SAMPLING PROCESS[Y/N]:Y (RET)



********** MNCDO-RELATED INFORMATION **********


The REAL-11/MNC software supports up to eight MNCDOs.  These modules
should be located to the right of any MNCADs.  In the software, an
MNCDO is identified by a "unit number".  The software assigns these
numbers according to the ORDER in which you provide the vector and
CSR addresses.


HOW MANY MNCDOs ARE CONTAINED IN YOUR TARGET SYSTEM?[D:0-8]:8   1(RET)


WHAT IS THE VECTOR ADDRESS OF MNCDO #0?[O:120-474]:340 (RET)


WHAT IS THE CSR ADDRESS OF MNCDO #0?[O:171000-171400]:171260 (RET)



********** MNCAA-RELATED INFORMATION **********


The REAL-11/MNC software will support up to eight MNCAAs.  Each
module has four channels for output.  The software will address a
particular channel on a particular module via a global channel-number
assignment.  The software makes this assignment in accordance with the
ORDER in which your supply the CSR addresses for the modules.  The
software assigns channel numbers 0-3 to the first MNCAA described,
channel numbers 4-7 to the second MNCAA, and so forth.


HOW MANY MNCAAs ARE CONTAINED IN YOUR TARGET SYSTEM?[D:0-8]:8   1(RET)
```

```
WHAT IS THE CSR ADDRESS OF MNCAA #0?[O:171000-171400]:171060 (RET)


********** SLU-RELATED INFORMATION **********

The REAL-11/MNC software supports up to  20 SLU
Serial Line Units (ports) dedicated to instruments.
These ports are contained in the DLV11J module(s) in
the restricted part of the MNC chassis. The standard
MINC hardware configuration contains 4 SLUs; one for
the console, one for a serial line printer, and two
for instruments.  In the software, an SLU is
identified by a unit number.  The software
assigns these numbers according to the ORDER in which
you provide the vector and CSR addresses.

You may not use a SLU that is concurrently being used
for system terminal support or for a serial line
printer.


HOW MANY SLUs IN YOUR TARGET SYSTEM WILL BE DEDICATED
TO INSTRUMENTS?[D:0-20]:  2    3 (RET)


Each Serial Line Unit input channel uses an internal
ring buffer for storage of input data. This data
is entered into the buffer asynchronously; your program
removes data from the buffer.  Therefore, the larger the
size of the buffer, the less frequently your program must
remove data from it.  The size of the buffer should be at
least as large as the the longest message that your
program will read, and it must be an even number.


WHAT IS THE SIZE, IN BYTES, OF THE INPUT RING BUFFER
FOR EACH CHANNEL? [D:32-1024]:   82 (RET)


Each Serial Line Unit output channel uses an internal
ring buffer for storage of data to be output.  This
buffer can be smaller than the length of the largest
message; however, the larger the buffer, the less
frequently your program must enter data into it.


WHAT IS THE SIZE, IN BYTES, OF THE OUTPUT RING
BUFFER FOR EACH CHANNEL?[D:16-1024]:   40 (RET)


WHAT IS THE VECTOR(recv) ADDRESS OF SLU # 0?[O:120-370]:300 (RET)


WHAT IS THE CSR(recv) ADDRESS OF SLU # 0?[O:160000-177770]:176500 (RET)
*NOTE- the transmit CSR, 176504 and vector  304 are actually being checked


WHAT IS THE VECTOR(recv) ADDRESS OF SLU # 1?[O:120-370]:310 (RET)
```

```
WHAT IS THE CSR(recv) ADDRESS OF SLU # 1?[0:160000-177770]:176510 (RET)
*NOTE- the transmit CSR, 176514 and vector  314 are actually being checked


WHAT IS THE VECTOR(recv) ADDRESS OF SLU # 2?[0:120-370]:320 (RET)


WHAT IS THE CSR(recv) ADDRESS OF SLU # 2?[0:160000-177770]:176520 (RET)
*NOTE- the transmit CSR, 176524 and vector  324 are actually being checked


"Data overrun" occurs when the system cannot transfer data at the rate
requested.  It is not detected until the sampling process has begun.
Three methods of handling this situation are available in REAL-11/MNC
for serial I/O (CIN/COUT).   You must select one.

These methods are:

1) Immediately terminate the sampling process when data overrun is
   detected.  This is the usual way to handle the problem.

2) Indicate in the data buffer, via a bit pattern (numeric value)
   to be defined later, when a datum was missed during the sampling
   process.  This method is useful when the sampling process is not
   clock driven.

3) Ignore all data-overrun conditions.  This is a fast but dangerous
   choice since data may be lost with no indication.


HOW DO YOU WANT REAL-11/MNC TO HANDLE DATA OVERRUNS DURING
A SERIAL  INPUT SWEEP

1) IMMEDIATELY TERMINATE THE SAMPLING PROCESS[Y/N]:Y (RET)



********** END REAL-11/MNC GENERATION PHASE 1 **********


STOP --

.SET TT QUIET


********** REAL-11/MNC RT GENERATION PHASE 2 **********

The system time is 10:49:35

The system date is 20-AUG-81


ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
```

```
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
```

```
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ITEMPL
ITEMPF
IWHICH
VOLTS
CELSIU
CONVER
PSCLKA
PSHIST
PSCLKB
PSAD
PSDI
PSDO
PSDA
PSSL
.MAIN.
?FORTRAN-I-[.MAIN.] Errors: 0, Warnings: 1


********** END REAL-11/MNC RT GENERATION  PHASE2 **********

The system time is 11:11:08


The system date is 20-AUG-81
```

A.5.2 Sample MNCGEN Dialog: Short Form

The sample MNCGEN dialog given below demonstrates use of the short form of the dialog. The resulting library is identical to the one created by the dialog given in Section 3.5.1.

```
.DATE 20-AUG-81 RET

.TIME 10:45 RET

.ASSIGN DL0: N RET

.ASSIGN DL1: I RET

.ASSIGN DL1: D RET

.RUN I:MNCGEN RET

REAL-11/MNC/RT SOFTWARE GENERATION PROCEDURE

********** REAL-11/MNC GENERATION PHASE 1 **********

The system time is  10:45:22

The system date is  20-AUG-81

Some additional information concerning this procedure and its
effects is available in Appendix A of the RT-11/FEP and RT-11/FRP
Installation and User's Guide.

You can select either an expanded form or a short form of this
dialogue.  The expanded form explains each step in this gener-
ation procedure.  The short form, on the other hand, does not
explain any question.  However, if you do not select the expand-
ed form and do not understand a particular question, respond with
"H" for help.  This response will cause the expanded comment for
that particular question to be printed.

DO YOU WANT THE MNCGEN DIALOGUE IN THE EXPANDED FORM?[Y/N]:N RET

HAVE YOU MET THE REQUIREMENTS TO PERFORM THIS PROCEDURE?[Y/N]:N  Y RET

SHOULD THIS PROCEDURE EXECUTE PHASE 2?[Y/N]:Y RET

DO YOU WANT TO PRODUCE ASSEMBLY LISTING FILES?[Y/N]:N RET

IS THIS SYSTEM, AS PRESENTLY CONFIGURED, THE TARGET SYSTEM OF THIS
MNCGEN PROCEDURE?[Y/N]:N  Y RET

CAUTION: This additional checking feature requires that your system
         be used solely for this procedure during Phase 1.

DO YOU WANT THE REAL-11 SOFTWARE TO MAKE RUN-TIME CHECKS
FOR THE PRESENCE OF REQUIRED LAB MODULES? [Y/N]:Y RET
```

```
********** MNCAD-RELATED INFORMATION **********

HOW MANY MNCADs ARE CONTAINED IN YOUR TARGET SYSTEM?[D:0-1]:1 (RET)

WHAT IS THE VECTOR ADDRESS OF MNCAD #0?[O:120-474]:400 (RET)

WHAT IS THE CSR ADDRESS OF MNCAD #0?[O:171000-171400]:171000 (RET)

DOES YOUR TARGET SYSTEM CONTAIN AN MNCAG?[Y/N]:Y (RET)

WHAT SHOULD BE THE DEFAULT GAIN VALUE WHEN THE GAIN-SELECTION SWITCH
ON AN MNCAG IS SET TO PROGRAMMABLE?:
.5(0), 5(1), 50(2), OR 500(3)? [D:0-3]:0 (RET)

DOES YOUR TARGET SYSTEM CONTAIN AN MNCTP?[Y/N]:Y (RET)

HOW DO YOU WANT REAL-11/MNC TO HANDLE DATA OVERRUNS DURING
AN ANALOG INPUT SWEEP ('ADSWP')?

1) IMMEDIATELY TERMINATE THE SAMPLING PROCESS[Y/N]:Y   N(RET)

2) INDICATE DATA OVERRUN BY A PREDETERMINED BIT PATTERN IN THE
   DATA BUFFER[Y/N]:N   Y(RET)

WHAT VALUE SHOULD BE USED TO INDICATE AN ANALOG  DATA OVERRUN?
[O:0-177777]:37777 (RET)

NOTE: THIS VALUE WILL ALSO BE USED TO REPORT
'(I)ADINP', 'ADFAST', AND 'ADSUM' ERRORS.



********** MNCKW-RELATED INFORMATION **********

HOW MANY MNCKWs ARE CONTAINED IN YOUR TARGET SYSTEM?[D:0-2]:2 (RET)

WHAT IS THE VECTOR ADDRESS OF MNCKW #0?[O:120-474]:440 (RET)

WHAT IS THE CSR ADDRESS OF MNCKW #0?[O:171000-171400]:171020 (RET)

WHAT IS THE VECTOR ADDRESS OF MNCKW #1?[O:120-474]:450 (RET)

WHAT IS THE CSR ADDRESS OF MNCKW #1?[O:171000-171400]:171024 (RET)

DO YOU WANT THE 'GTHIST' SUBPROGRAM IN YOUR LIBRARY?[Y/N]:Y (RET)

DO YOU REQUIRE PARALLEL CLOCK-DRIVEN TRANSFERS THROUGH SEVERAL
MNC-SERIES MODULES?[Y/N]:N (RET)
```

```
********** MNCDI-RELATED INFORMATION **********

HOW MANY MNCDIs ARE CONTAINED IN YOUR TARGET SYSTEM?[D:0-8]:8   1 (RET)

WHAT IS THE VECTOR ADDRESS OF MNCDI #0?[O:120-474]:120 (RET)

WHAT IS THE CSR ADDRESS OF MNCDI #0?[O:171000-171400]:171160 (RET)

HOW DO YOU WANT REAL-11/MNC TO HANDLE DATA OVERRUNS DURING
A DIGITAL INPUT SWEEP ('DISWP')?

1) IMMEDIATELY TERMINATE THE SAMPLING PROCESS[Y/N]:Y (RET)


********** MNCDO-RELATED INFORMATION **********

HOW MANY MNCDOs ARE CONTAINED IN YOUR TARGET SYSTEM?[D:0-8]:8   1(RET)

WHAT IS THE VECTOR ADDRESS OF MNCDO #0?[O:120-474]:340 (RET)

WHAT IS THE CSR ADDRESS OF MNCDO #0?[O:171000-171400]:171260 (RET)


********** MNCAA-RELATED INFORMATION **********

HOW MANY MNCAAs ARE CONTAINED IN YOUR TARGET SYSTEM?[D:0-8]:8   1 (RET)

WHAT IS THE CSR ADDRESS OF MNCAA #0?[O:171000-171400]:171060 (RET)


********** SLU-RELATED INFORMATION **********

HOW MANY SLUs IN YOUR TARGET SYSTEM WILL BE DEDICATED
TO INSTRUMENTS?[D:0-20]: 2   3 (RET)

WHAT IS THE SIZE, IN BYTES, OF THE INPUT RING BUFFER
FOR EACH CHANNEL? [D:32-1024]:  82 (RET)

WHAT IS THE SIZE, IN BYTES, OF THE OUTPUT RING
BUFFER FOR EACH CHANNEL?[D:16-1024]:  40 (RET)
```

```
WHAT IS THE VECTOR(recv) ADDRESS OF SLU # 0?[0:120-370]:300 (RET)

WHAT IS THE CSR(recv) ADDRESS OF SLU # 0?[0:160000-177770]:176500 (RET)
*NOTE- the transmit CSR, 176504 and vector  304 are actually being checked

WHAT IS THE VECTOR(recv) ADDRESS OF SLU # 1?[0:120-370]:310 (RET)

WHAT IS THE CSR(recv) ADDRESS OF SLU # 1?[0:160000-177770]:176510 (RET)
*NOTE- the transmit CSR, 176514 and vector  314 are actually being checked

WHAT IS THE VECTOR(recv) ADDRESS OF SLU # 2?[0:120-370]:320 (RET)

WHAT IS THE CSR(recv) ADDRESS OF SLU # 2?[0:160000-177770]:176520 (RET)
*NOTE- the transmit CSR, 176524 and vector  324 are actually being checked

HOW DO YOU WANT REAL-11/MNC TO HANDLE DATA OVERRUNS DURING
A SERIAL  INPUT SWEEP

1) IMMEDIATELY TERMINATE THE SAMPLING PROCESS[Y/N]:Y (RET)



********** END REAL-11/MNC GENERATION PHASE 1 **********

STOP --

.SET TT QUIET


********** REAL-11/MNC RT GENERATION PHASE 2 **********

The system time is 10:46:55


The system date is 20-AUG-81
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
```

```
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
ERRORS DETECTED:   0
```

```
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ITEMPL
ITEMPF
IWHICH
VOLTS
CELSIU
CONVER
PSCLKA
PSHIST
PSCLKB
PSAD
PSDI
PSDO
PSDA
PSSL
.MAIN.
?FORTRAN-I-[.MAIN.] Errors: 0, Warnings: 1


********** END REAL-11/MNC RT GENERATION  PHASE2 **********

The system time is 11:11:08


The system date is 20-AUG-81
```

A.5.3 Sample MNCGEN Dialog: Correcting Dialog Input Errors

The sample MNCGEN dialog given below demonstrates how to use the "help" feature to recover from errors made in responding to MNCGEN questions. It also selects and performs Phase 2 as a separate operation. The resulting library, however, is identical to the one created by the examples given in Sections 3.5.1 and 3.5.2.

```
. DATE 20-AUG-81 (RET)

. TIME 10:45 (RET)

. ASSIGN DL0: N (RET)

. ASSIGN DL1: I (RET)

. ASSIGN DL1: D (RET)

. RUN I:MNCGEN (RET)

REAL-11/MNC/RT SOFTWARE GENERATION PROCEDURE


********** REAL-11/MNC GENERATION PHASE 1 **********

The system time is  10:45:16

The system date is  20-AUG-81


Some additional information concerning this procedure and its
effects is available in Appendix A of the RT-11/FEP and RT-11/FRP
Installation and User's Guide.


You can select either an expanded form or a short form of this
dialogue.  The expanded form explains each step in this gener-
ation procedure.  The short form, on the other hand, does not
explain any question.  However, if you do not select the expand-
ed form and do not understand a particular question, respond with
"H" for help.  This response will cause the expanded comment for
that particular question to be printed.


DO YOU WANT THE MNCGEN DIALOGUE IN THE EXPANDED FORM?[Y/N]:N    Q (RET)


**ERROR**
RESPONSE IS SYNTACTICALLY INCORRECT!


DO YOU WANT THE MNCGEN DIALOGUE IN THE EXPANDED FORM?[Y/N]:N    H (RET)


You can select either an expanded form or a short form of this
dialogue.  The expanded form explains each step in this gener-
ation procedure.  The short form, on the other hand, does not
explain any question.  However, if you do not select the expand-
ed form and do not understand a particular question, respond with
"H" for help.  This response will cause the expanded comment for
that particular question to be printed.
```

```
DO YOU WANT THE MNCGEN DIALOGUE IN THE EXPANDED FORM?[Y/N]:N (RET)

HAVE YOU MET THE REQUIREMENTS TO PERFORM THIS PROCEDURE?[Y/N]:N   Y (RET)

SHOULD THIS PROCEDURE EXECUTE PHASE 2?[Y/N]:Y (RET)

DO YOU WANT TO PRODUCE ASSEMBLY LISTING FILES?[Y/N]:N (RET)

IS THIS SYSTEM, AS PRESENTLY CONFIGURED, THE TARGET SYSTEM OF THIS
MNCGEN PROCEDURE?[Y/N]:N   Y (RET)

CAUTION: This additional checking feature requires that your system
         be used solely for this procedure during Phase 1.

DO YOU WANT THE REAL-11 SOFTWARE TO MAKE RUN-TIME CHECKS
FOR THE PRESENCE OF REQUIRED LAB MODULES? [Y/N]:Y    N (RET)



********** MNCAD-RELATED INFORMATION **********

HOW MANY MNCADs ARE CONTAINED IN YOUR TARGET SYSTEM?[D:0-1]:1 (RET)

WHAT IS THE VECTOR ADDRESS OF MNCAD #0?[O:120-474]:400      488 (RET)

**ERROR**
RESPONSE IS SYNTACTICALLY INCORRECT!



WHAT IS THE VECTOR ADDRESS OF MNCAD #0?[O:120-474]:400 (RET)

WHAT IS THE CSR ADDRESS OF MNCAD #0?[O:171000-171400]:171000   171020 (RET)

**ERROR**
THE ATTEMPT TO SIMULATE AN INTERRUPT USING CSR ADDRESS 171020 FAILED.

WHAT IS THE VECTOR ADDRESS OF MNCAD #0?[O:120-474]:400 (RET)

WHAT IS THE CSR ADDRESS OF MNCAD #0?[O:171000-171400]:171000 (RET)

DOES YOUR TARGET SYSTEM CONTAIN AN MNCAG?[Y/N]:Y (RET)

WHAT SHOULD BE THE DEFAULT GAIN VALUE WHEN THE GAIN-SELECTION SWITCH
ON AN MNCAG IS SET TO PROGRAMMABLE?:
.5(0), 5(1), 50(2), OR 500(3)? [D:0-3]:0 (RET)

DOES YOUR TARGET SYSTEM CONTAIN AN MNCTP?[Y/N]:Y (RET)
```

```
HOW DO YOU WANT REAL-11/MNC TO HANDLE DATA OVERRUNS DURING
AN ANALOG INPUT SWEEP ('ADSWP')?

1) IMMEDIATELY TERMINATE THE SAMPLING PROCESS[Y/N]:Y   N (RET)

2) INDICATE DATA OVERRUN BY A PREDETERMINED BIT PATTERN IN THE
   DATA BUFFER[Y/N]:N   Y (RET)

WHAT VALUE SHOULD BE USED TO INDICATE AN ANALOG  DATA OVERRUN?
[O:0-177777]:37777 (RET)

NOTE: THIS VALUE WILL ALSO BE USED TO REPORT
'(I)ADINP', 'ADFAST', AND 'ADSUM' ERRORS.



********** MNCKW-RELATED INFORMATION **********

HOW MANY MNCKWs ARE CONTAINED IN YOUR TARGET SYSTEM?[D:0-2]:2   3 (RET)

**ERROR**
RESPONSE IS OUT OF RANGE!


HOW MANY MNCKWs ARE CONTAINED IN YOUR TARGET SYSTEM?[D:0-2]:2   1 (RET)

WHAT IS THE VECTOR ADDRESS OF MNCKW #0?[O:120-474]:440 (RET)

WHAT IS THE CSR ADDRESS OF MNCKW #0?[O:171000-171400]:171020   171000 (RET)

**ERROR**
ADDRESS GIVEN CONFLICTS WITH PREVIOUS RESPONSES.

WHAT IS THE CSR ADDRESS OF MNCKW #0?[O:171000-171400]:171020 (RET)

DO YOU WANT THE 'GTHIST' SUBPROGRAM IN YOUR LIBRARY?[Y/N]:Y (RET)

DO YOU REQUIRE PARALLEL CLOCK-DRIVEN TRANSFERS THROUGH SEVERAL
MNC-SERIES MODULES?[Y/N]:N (RET)



********** MNCDI-RELATED INFORMATION **********

HOW MANY MNCDIs ARE CONTAINED IN YOUR TARGET SYSTEM?[D:0-8]:8   1 (RET)

WHAT IS THE VECTOR ADDRESS OF MNCDI #0?[O:120-474]:120 (RET)
```

```
WHAT IS THE CSR ADDRESS OF MNCDI #0?[O:171000-171400]:171160 (RET)

HOW DO YOU WANT REAL-11/MNC TO HANDLE DATA OVERRUNS DURING
A DIGITAL INPUT SWEEP ('DISWP')?

1) IMMEDIATELY TERMINATE THE SAMPLING PROCESS[Y/N]:Y (RET)



********** MNCDO-RELATED INFORMATION **********

HOW MANY MNCDOs ARE CONTAINED IN YOUR TARGET SYSTEM?[D:0-8]:8   -1 (RET)

**ERROR**
RESPONSE IS SYNTACTICALLY INCORRECT!


HOW MANY MNCDOs ARE CONTAINED IN YOUR TARGET SYSTEM?[D:0-8]:8   1 (RET)

WHAT IS THE VECTOR ADDRESS OF MNCDO #0?[O:120-474]:340      100 (RET)

**ERROR**
RESPONSE IS OUT OF RANGE!

WHAT IS THE VECTOR ADDRESS OF MNCDO #0?[O:120-474]:340 (RET)

WHAT IS THE CSR ADDRESS OF MNCDO #0?[O:171000-171400]:171260 (RET)



********** MNCAA-RELATED INFORMATION **********

HOW MANY MNCAAs ARE CONTAINED IN YOUR TARGET SYSTEM?[D:0-8]:8   1 (RET)

WHAT IS THE CSR ADDRESS OF MNCAA #0?[O:171000-171400]:171060 (RET)



********** SLU-RELATED INFORMATION **********

HOW MANY SLUs IN YOUR TARGET SYSTEM WILL BE DEDICATED
TO INSTRUMENTS?[D:0-20]: 2   3 (RET)

WHAT IS THE SIZE, IN BYTES, OF THE INPUT RING BUFFER
FOR EACH CHANNEL? [D:32-1024]:  82 (RET)
```

```
WHAT IS THE SIZE, IN BYTES, OF THE OUTPUT RING
BUFFER FOR EACH CHANNEL?[D:16-1024]:  40 RET


WHAT IS THE VECTOR(recv) ADDRESS OF SLU # 0?[O:120-370]:300 RET


WHAT IS THE CSR(recv) ADDRESS OF SLU # 0?[O:160000-177770]:176500  176510 RET
*NOTE- the transmit CSR, 176514 and vector  304 are actually beins checked
**ERROR**
SIMULATING AN INTERRUPT USING CSR ADDRESS 176514
CAUSES AN INTERRUPT THROUGH VECTOR ADDRESS 314
RATHER THAN THROUGH 304 AS WAS INDICATED


WHAT IS THE VECTOR(recv) ADDRESS OF SLU # 0?[O:120-370]:300 RET


WHAT IS THE CSR(recv) ADDRESS OF SLU # 0?[O:160000-177770]:176500 RET
*NOTE- the transmit CSR, 176504 and vector  304 are actually beins checked


WHAT IS THE VECTOR(recv) ADDRESS OF SLU # 1?[O:120-370]:310 RET


WHAT IS THE CSR(recv) ADDRESS OF SLU # 1?[O:160000-177770]:176510 RET
*NOTE- the transmit CSR, 176514 and vector  314 are actually beins checked


WHAT IS THE VECTOR(recv) ADDRESS OF SLU # 2?[O:120-370]:320 RET


WHAT IS THE CSR(recv) ADDRESS OF SLU # 2?[O:160000-177770]:176520 RET
*NOTE- the transmit CSR, 176524 and vector  324 are actually beins checked


HOW DO YOU WANT REAL-11/MNC TO HANDLE DATA OVERRUNS DURING
A SERIAL  INPUT SWEEP

1) IMMEDIATELY TERMINATE THE SAMPLING PROCESS[Y/N]:Y RET



********** END REAL-11/MNC GENERATION PHASE 1 **********


STOP --


********** REAL-11/MNC RT GENERATION PHASE 2 **********

The system time is 10:48:58


The system date is 20-AUG-81
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
```

```
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
```

```
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ERRORS DETECTED:  0
ITEMPL
ITEMPF
IWHICH
VOLTS
CELSIU
CONVER
PSCLKA
PSHIST
PSCLKB
PSAD
PSDI
PSDO
PSDA
PSSL
.MAIN.
?FORTRAN-I-[.MAIN.] Errors: 0, Warnings: 1


********** END REAL-11/MNC RT GENERATION  PHASE2 **********

The system time is 11:11:08


The system date is 20-AUG-81
```

A.5.4 Installing Generated REAL–11/MNC Software

Before you verify REAL–11/MNC software that has been generated by the MNCGEN procedure, be sure the software is on the development system disk. If you did not specify the development system disk as device N: for the MNCGEN dialog, copy the REAL–11/MNC software to the development system disk by performing the following procedures:

1. Bootstrap the development system disk in drive 0.

   Mount the disk that contains the generated REAL–11/MNC software in drive 1. This disk is the one that was in device N: during the MNCGEN procedure.

2. Copy the software by typing:

   ```
   .COPY  dv1:MNCLIB.OBJ  dv0:MNCLIB.OBJ(RET)
   ```

   where `dv` is `DL` if you have two RL01/RL02 disks or `DY` if you have only RX02 diskettes.

A.5.5 Verifying Generated REAL–11/MNC Software

This section contains instructions for verifying REAL–11/MNC software that you have generated by performing the REAL–11/MNC Software Generation Procedure (MNCGEN). For instructions on verifying distributed REAL–11/MNC software, see Section 4.2.3.

To verify generated REAL–11/MNC software:

1. Be sure disks are in the same drives they were in during the MNCGEN procedure and that those drives still have the same logical device names they had during the MNCGEN procedure. Also be sure that device N: contains at least 100 contiguous free blocks; if necessary, use the SQUEEZE command to consolidate free space on device N:.

2. The clock Schmitt triggers can generate outputs in response to electrical noise when no device is connected to the input. To avoid such spurious responses from unused Schmitt triggers, set the mode selector of each MNCKW clock module to VAR and turn the associated potentiometer to either of its extreme settings.

   Also be sure no instruments are connected to the MNC–series modules.

3. The verification procedure creates (and deletes) files named MVERN.OBJ and MTMP.RUN. Therefore, if you have any files with those names, rename them before you begin the verification procedure.

4. You must be using the SJ or FB monitor when you execute the REAL–11/MNC verification procedure. You also must execute the procedure on the system console terminal.

5. Execute the verification procedure by typing:

   ```
   .@I:MNCLIB.VER(RET)
   ```

6. As the verification procedure tests the REAL–11/MNC software, it prints messages on the console terminal. The following list contains the types of messages and an example of each:

   - confirmation messages, which verify that an MNC–series module and some related REAL–11/MNC software have been exercised successfully. For example:

     ```
     The primary clock (1st MNCKW) and some related REAL-11 software
     have been exercised successfully!
     ```

   - level–W error messages, which indicate that hardware named in the message is absent. For example:

     ```
     ?MVER-W-The REAL-11 software has NOT detected an analog to
     digital input module (MNCAD) in this system.
     ```

   - level–F error messages, which are fatal error messages indicating that the software has malfunctioned. One type of fatal error message names the verification module that has failed. For example:

     ```
     ?MVER-F-REAL Software Error - Installation failure at PSDI
     ```

     Another type of fatal error message lists trouble-shooting aids for DIGITAL support personnel. For example:

     ```
     ?MVER-F-***UNEXPECTED ERROR DURING MNCAA RELATED PHASE***
             ***CHECK HARDWARE I/O SIGNALS***

             Trouble-shooting aids for DEC support:
             a) IND =         1
             b) INFO(1) =     334
     ```

     A third type of fatal error message indicates either that required MNC–series hardware is not present or that vector and/or CSR address switch settings do not agree with values you gave for those settings during the MNCGEN dialog. For example:

     ```
     ?MVER-F-MNCAD CSR/VECTOR addressing problem
     ```

   - the completion message:

     ```
     STOP--VERIFICATION PROCESS COMPLETE
     ```

7. When the console terminal displays the system dot, evaluate the results of the verification procedure:

   - If the terminal has displayed only a series of confirmation messages followed by the completion message, then the REAL–11/MNC software and MNC–series hardware are correctly installed.

   - If the terminal has displayed any level–W messages, then MNC–series hardware is absent. If a level–W message names an MNC–series module that is present in your system, be sure that module's vector and CSR address switches are correctly set, then execute the verification procedure again. Ignore level–W messages that name MNC–series hardware not present in your system.

- If the terminal has displayed any level–F messages that indicate missing hardware or inaccurately set vector and/or CSR address switches, then again execute the MNCGEN procedure and install and verify the generated software.
- If the terminal has displayed any level–F messages that indicate software malfunction, call the MINC Product Services Center for further information.
- If the terminal has displayed a series of messages but has not printed the completion message, then the software has malfunctioned; call the MINC Product Services Center for additional information.

A.5.6 Adding a Library of Generated REAL–11/MNC Software to SYSLIB

When you have installed all needed FEP–11 software, see Section 5.6 for information on using the librarian to add files to SYSLIB.

Appendix B
SYSGEN Answers that Will Duplicate the Installed XM Monitor

The following page lists the SYSGEN answers that you can use to duplicate the XM monitor built by the SYSGEN option of the FEP Installation Disk.

1	N	40	177170	79	skipped	118	skipped
2	N	41	264	80	skipped	119	skipped
3	N	42	skipped	81	skipped	120	skipped
4	N	43	skipped	82	skipped	121	skipped
5	Y	44	N	83	skipped	122	skipped
6	skipped	45	skipped	84	skipped	123	skipped
7	Y	46	skipped	85	skipped	124	skipped
8	skipped	47	skipped	86	skipped	125	skipped
9	N	48	skipped	87	skipped	126	skipped
10	N	49	skipped	88	N	127	skipped
11	N	50	N	89	Y	128	skipped
12	N	51	N	90	N	129	skipped
13	skipped	52	N	91	skipped	130	skipped
14	skipped	53	skipped	92	skipped	131	skipped
15	40	54	N	93	Y	132	skipped
16	134	55	skipped	94	176500	133	skipped
17	N	56	N	95	300	134	skipped
18	Y	57	Y	96	N	135	skipped
19	skipped	58	2	97	N	136	skipped
20	skipped	59	N	98	N	137	skipped
21	skipped	60	skipped	99	Y	138	skipped
22	N	61	N	100	N	139	skipped
23	N	62	N	101	skipped	140	skipped
24	Y	63	skipped	102	skipped	141	skipped
25	Y	64	skipped	103	skipped	142	skipped
26	N	65	skipped	104	10	143	skipped
27	N	66	skipped	105	skipped	144	skipped
28	Y	67	skipped	106	skipped	145	skipped
29	N	68	skipped	107	skipped	146	skipped
30	skipped	69	skipped	108	skipped	147	skipped
31	N	70	skipped	109	skipped	148	skipped
32	skipped	71	skipped	110	skipped	149	skipped
33	skipped	72	skipped	111	skipped	150	skipped
34	skipped	73	skipped	112	skipped	151	skipped
35	skipped	74	skipped	113	skipped	152	DL1 or DY1
36	skipped	75	skipped	114	skipped	153	DL0 or DY0
37	Y	76	skipped	115	skipped	154	DL0 or DYO
38	N	77	skipped	116	skipped	155	N
39	N	78	skipped	117	skipped		

Index

READER'S COMMENTS

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

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

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

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

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

Name _____ Date _____

Organization _____ Telephone _____

Street _____

City _____ State _____ Zip Code _____
or Country

Do Not Tear – Fold Here and Tape

digital

No Postage
Necessary
if Mailed in the
United States

BUSINESS REPLY MAIL
FIRST CLASS PERMIT NO. 33 MAYNARD MASS.

POSTAGE WILL BE PAID BY ADDRESSEE

SOFTWARE PUBLICATIONS
200 FOREST STREET MR01–2/L12
MARLBOROUGH, MA 01752

Do Not Tear – Fold Here and Tape