



iRMX[®]
ICU User's Guide and
Quick Reference

Order Number: 611036-003

In the United States, additional copies of this manual or other Intel literature may be obtained by writing:

Literature Distribution Center
Intel Corporation
P.O. Box 7641
Mt. Prospect, IL 60056-7641

Or you can call the following toll-free number: 1-800-548-4725

In locations outside the United States, obtain additional copies of Intel documentation by contacting your local Intel sales office. For your convenience, international sales office addresses are printed on the last page of this document. Contact your local sales office to obtain the latest specifications before placing your order.

Intel Corporation (Intel) makes no warranty of any kind with regard to this material, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. Intel assumes no responsibility for any errors that may appear in this document. Intel makes no commitment to update nor to keep current the information contained in this document. No part of this document may be copied or reproduced in any form or by any means without prior written consent of Intel. Intel retains the right to make changes to these specifications at any time, without notice.

Intel software products are copyrighted by and shall remain the property of Intel Corporation. Use, duplication or disclosure is subject to restrictions stated in Intel's Software License Agreement.

U.S. GOVERNMENT RESTRICTED RIGHTS: These software products and documentation were developed at private expense and are provided with "RESTRICTED RIGHTS." Use, duplication, or disclosure by the Government is subject to restrictions as set forth in FAR 52.227-14 and DFAR 252.227-7013 et seq. or its successor.

The Intel logo, i960, Pentium, and iRMX are registered trademarks of Intel Corporation, registered in the United States of America and other countries. Above, i287, i386, i387, i486, Intel287, Intel386, Intel387, Intel486, Intel487 and EtherExpress are trademarks of Intel Corporation.

Adaptec is a registered trademark of Adaptec, Inc. AT, IBM and PS/2 are registered trademarks and PC/XT is a trademark of International Business Machines Corporation. All Borland products are trademarks or registered trademarks of Borland International, Inc. CodeView, Microsoft, MS, MS-DOS and XENIX are registered trademarks of Microsoft Corporation. Comtrol is a registered trademark and HOSTESS is a trademark of Comtrol Corporation. DT2806 is a trademark of Data Translation, Inc. Ethernet is a registered trademark of Xerox Corporation. Hayes is a registered trademark of Hayes Microcomputer Products. Hazeltine and Executive 80 are trademarks of Hazeltine Corporation. Hewlett-Packard is a registered trademark of Hewlett-Packard Co. Maxtor is a registered trademark of Maxtor Corporation. MIX® is a registered trademark of MIX Software, Incorporated. MIX is an acronym for Modular Interface eXtension. MPI is a trademark of Centralp Automatismes (S.A.). NetWare and Novell are registered trademarks of Novell Corp. NFS is a trademark of Sun Microsystems, Inc. Phar Lap is a trademark of Phar Lap Software, Inc. Soft-Scope is a registered trademark of Concurrent Sciences, inc. TeleVideo is a trademark of TeleVideo Systems, Inc. UNIX is a registered trademark in the United States and other countries, licensed exclusively through X/Open Company Limited. VAX is a registered trademark and VMS is a trademark of Digital Equipment Corporation. Visual Basic and Visual C++ are trademarks of Microsoft Corporation. All Watcom products are trademarks or registered trademarks of Watcom International Corp. Windows, Windows 95 and Windows for Workgroups are registered trademarks and Windows NT is a trademark of Microsoft in the U.S. and other countries. Wyse is a registered trademark of Wyse Technology. Zentec is a trademark of Zentec Corporation. Other trademarks and brands are the property of their respective owners.

Copyright © 1991, 1992 and 1995 Intel Corporation, All Rights Reserved

REVISION HISTORY

		DATE
-001	Original Issue	12/91
-002	Update for Release 2.0 of the OS	08/92
-003	Update for Release 2.2 of the OS	11/95

Quick Contents

- Chapter 1. Introduction**
- Chapter 2. ICU Operations**
- Chapter 3. Example ICU Session**
- Chapter 4. ICU Quick Reference**
- Appendix A. Error Messages**
- Appendix B. Development Environment**
- Appendix C. Using the ICU to Configure Custom Device Drivers**
- Appendix D. Standard Definition Files for the iRMX III OS**
- Index**
- Service Information**

Notational Conventions

Most of the references to system calls in the text and graphics use C syntax instead of PL/M (for example, the system call **send_message** instead of **send\$message**). If you are working in C, you must use the C header files, *rmx_c.h*, *udi_c.h* and *rmx_err.h*. If you are working in PL/M, you must use dollar signs (\$) and use the *rmxplm.ext* and *error.lit* header files.

This manual uses the following conventions:

- Syntax strings, data types, and data structures are provided for PL/M and C respectively.
- All numbers are decimal unless otherwise stated. Hexadecimal numbers include the H radix character (for example, 0FFH). Binary numbers include the B radix character (for example, 11011000B).
- Bit 0 is the low-order bit. If a bit is set to 1, the associated description is true unless otherwise stated.
- Data structures and syntax strings appear in this font.
- **System call names and command names appear in this font.**
- PL/M data types such as BYTE and SELECTOR, and iRMX data types such as STRING and SOCKET are capitalized. All C data types are lower case except those that represent data structures.
- The following OS layer abbreviations are used. The Nucleus layer is unabbreviated.

AL	Application Loader
BIOS	Basic I/O System
EIOS	Extended I/O System
HI	Human Interface
UDI	Universal Development Interface

- Whenever this manual describes I/O operations, it assumes that tasks use BIOS calls (such as **rq_a_read**, **rq_a_write**, and **rq_a_special**). Although not mentioned, tasks can also use the equivalent EIOS calls (such as **rq_s_read**, **rq_s_write**, and **rq_s_special**) or UDI calls (**dq_read** or **dq_write**) to do the same operations.

Contents

1 Introduction

Reader Level.....	13
Overview of ICU-configurable Options.....	14
ICU File Descriptions.....	15
ICU Internal Files.....	15
Screen Master and Template Files	15
Definition Files	16
Generation Files	16
Practice Session: Invoking the ICU	17
Screen Display Command Characters.....	19
Main Menu Commands	19
Change Command.....	20
Generate Command.....	20
List Command.....	21
Save Command	21
Quit Command.....	22
Exit Command	22
Replace Command	23
Detail-level Command	23
Backup Command.....	24

2 ICU Operations

Editing ICU Screens.....	25
Screen Formats.....	25
Fixed Screen Format.....	26
Repetitive Screen Format	27
Repetitive-fixed Screen Format	28
Explanation of Screen Elements	29
Entering New Values.....	30

Screen Display/Editing Commands.....	31
Back Command	32
Next Command.....	32
Cancel Command	32
Find Command.....	32
Help Command.....	32
Redisplay Command.....	33
Search Command.....	33
Delete Command.....	33
Insert Command	35
Copy Command.....	36
Generating and Testing the New System	36
Creating Generation Files	36
Generating with Prefix Option	37
Executing the Generation Submit File	38
Testing the System	38
Restoring a Definition File	39
Log File.....	40

3 Example ICU Session

Step 1: Getting Started	42
Step 2: Entering Screen-editing Mode	43
How To Access ICU Help Screens	44
Step 3: Editing the Definition File.....	45
Exploring Device Driver Screens.....	48
Setting Automatic Boot Device Recognition	51
Step 4: Saving and Listing the Edited Definition File.....	52
Step 5: Generating the System.....	53

4 ICU Quick Reference

Screen Names and Acronyms	55
Application Loader Screen	58
BIOS Screen.....	58
Comments for Build File Screen.....	60

Device Driver Screens.....	61
Intel Device Drivers Screen.....	61
PC Bus Device Drivers Screen	61
Driver Screen Composite.....	62
Unit Information Screen Composite	62
Device-Unit Information Screen Composite	63
User Devices Screen.....	64
UDS Device Driver Modules Screen	65
DOS Extender Dispatcher Screens	65
DOS Extender Reserved Interrupts Screen.....	66
EIOS Screens	67
EIOS Screen.....	67
Automatic Boot Device Recognition Screen.....	67
Logical Names Screen.....	68
I/O Users Screen	69
I/O Jobs Screen	69
File Drivers Screens	71
BIOS File Drivers Screen	71
File Driver Screen Composites	71
Generate File Names Screen	72
Hardware Screens	73
Hardware Screen	73
Multibus II Hardware Screen.....	74
Human Interface Screens.....	76
Human Interface Screen	76
Human Interface Jobs Screen	77
Resident/Recovery User Screen.....	78
Prefixes Screen.....	79
HI Logical Names Screen.....	79
Includes and Libraries Screen.....	80
Interrupts Screens.....	82
Interrupts Screen	82
Slave Interrupt Levels Screen.....	82
Memory Screens	83
Memory for System Screen	84
Memory for Free Space Manager Screen.....	84
Paging Identity Mapped Memory Screen.....	84

Network Screens	85
Network Subsystem Screen	85
iNA MIP Driver Job Screen	86
MIP Configuration for Multibus I Screen	87
MIP Configuration for Multibus II Screen	89
COMMengine Board Instance Load Method for MBII Screen	90
COMMengine Board Names Screen	90
MIP Configuration for PC Bus Screen	91
iNA 960 COMMputer Job Screen	92
iRMX-NET Client Job Screen	93
File Consumer Configuration Screen	94
Remote File Access Screen	95
iRMX-NET Server Job Screen	96
File Server Configuration Screen	97
Apex File Access Screen	98
AFA User Screen	99
Public Devices Screen	100
Public Directory Screen	101
User Definition File Screen	102
Client Definition File Screen	103
Name Server Configuration Screen	104
Name Server Search Domains Screen	105
TCP/IP Job Screen	106
Parameters for TCP/IP Network Interfaces	107
IP Router Information Screen	107
TCP/IP Tunable Parameters 1 Screen	108
TCP/IP Tunable Parameters 2 Screen	109
TCP/IP Tunable Parameters 3 Screen	111
Streams Parameters Screen	112
X.25 Server Job Screen	113
X.25 Client Job Screen	114
Nucleus Screens	115
Nucleus Screen	115
Nucleus Communication Service Screen	117
OS Extension Screen	118
Query Screen	118
ROM Code Screen	119
Shared C Library Screen	120
Subsystems Screen	121
System Debugger Screens	123
System Debugger Screen	123
System Debug Console Screen for Multibus I/II	123

System Jobs Screens.....	125
System Jobs Screen	125
PCI Server Job Screen.....	126
PCI Server Controller Configuration Screen.....	127
Multibus II Downloader Job Screen.....	128
ATCS/279/ARC Server Job Screen	128
ATCS/450 Server Job Screen	129
MSA BootServer Job Screen	130
FPI Server Job Screen	131
Soft-Scope Kernel Job Screen	132
Third Party Tools-Generated Job Screen	133
User Jobs Screens.....	135
User Jobs Screen	135
User Modules Screen.....	136

A Error Messages

Invocation Messages	138
Version Control Messages	138
Automatic Restore and Update Messages	139
Interactive Error and Warning Messages	140
Interactive Error Messages	140
Interactive Warning Messages.....	143
Internal ICU Errors	144
System Generation Error and Warning Messages	145
Builder Warnings	145
Builder Errors.....	145

B Development Environment

Hardware Platforms	149
iRMX Target System.....	149
PC Target System.....	149
Software Requirements	150
Testing and Debugging Tools.....	150

C Using the ICU to Configure Custom Device Drivers

Adding Drivers with the UDS and ICUMRG Utilities	151
UDS Utility	153
Creating the Input File for UDS	153
Device Information Screens	158
Unit Information Screens	158
Device-Unit Information Screens	159
Invoking the UDS Utility	159
UDS Error Messages	160
ICUMRG Utility	163
UDS Modules Screen in the ICU	164
Adding Your Driver Without the UDS and ICUMRG	165
Example of Adding an Existing Driver as a Custom Driver	168
Contents of the Duib.inc File Specified in the (DPN) Parameter ...	169
Contents of the File Specified in the (TUP) Parameter	172
Portion of System Generation Submit File as Changed by this Process	174

D Standard Definition Files for the iRMX III OS

Standard Definition File Names	178
Multibus I Standard Definition Files	179
File Applications	179
Default OS Layer and Jobs for Multibus I Definition Files	181
Multibus I Device Driver Support	183
Additional Notes	184
SCSI Support for SBC 386/12S and 486/12S Boards	185
Networking Definition Files	185
Interrupt Levels Used in the Standard Definition Files	186
I/O Controller Board Support	191
Multibus I Operating System Layer Configuration	192
Multibus II Standard Definition Files	195
Multibus II File Applications	196
Default OS Layer and Jobs for Multibus II Definition Files	201
Memory Addresses Used in Multibus II Standard Definition Files	204
Multibus II Operating System Layer Configuration	205
PC Standard Definition Files	206
PC Applications	206
Default OS Layer and Jobs for PC Definition Files	209
PC Operating System Layer Configuration	210

Service Information

Inside Back Cover

Tables

D-1.	Multibus I Application Types and Definition Files.....	180
D-2.	OS Layer and Jobs in Multibus I Definition Files.....	182
D-3.	Multibus I Device Driver Support.....	183
D-4.	Interrupt Level Assignments for SBC 386/12(S) and SBC 486/12(S).....	187
D-5.	Interrupt Level Assignments for SBC 386/2X and SBC 386/3X	188
D-6.	Interrupt Level Assignments for the SBC PCP4DX(2) and PCP4SX Boards	189
D-7.	I/O Controller Support for Multibus I Definition Files	191
D-8.	Terminal Support for the SBC 386/2X/3X Boards	194
D-9.	Terminal Support for the SBC 386/12(S) and 486/12(S) Boards	194
D-10.	Multibus II Application Types and Definition Files	198
D-11.	OS Layer and Jobs in Multibus II Definition Files	202
D-12.	PC Application Types and Definition Files.....	208
D-13.	OS Layer and Jobs in PC Definition Files	210

Figures

1-1.	Layers of the iRMX OS	14
C-1.	Adding Drivers with UDS and ICUMRG.....	152
C-2.	Syntax of UDS Input File	154
C-3.	Example User Devices Screen	167
C-4.	Computing Device and Device-Unit Numbers	170
C-5.	Public Declarations Needed for the DINFO and UINFO Tables	173
C-6.	Portion of the Modified Submit File.....	174

Introduction 1

The *ICU User's Guide and Quick Reference* is a guide to using the iRMX[®] Interactive Configuration Utility (ICU). The ICU is a software tool for building an iRMX Operating System (OS) for your hardware platform. This manual provides operation guidelines, an example ICU session tutorial, quick reference information on screens and parameters, error message descriptions and instructions for adding your custom device drivers to the ICU.

This chapter introduces you to the ICU and provides some basic information that prepares you for using it including:

- An overview of ICU-configurable options
- ICU file descriptions
- How to invoke the ICU, including a practice session
- ICU command descriptions

Reader Level

This manual is written for engineers doing real-time system design and implementation. This manual assumes you are familiar with:

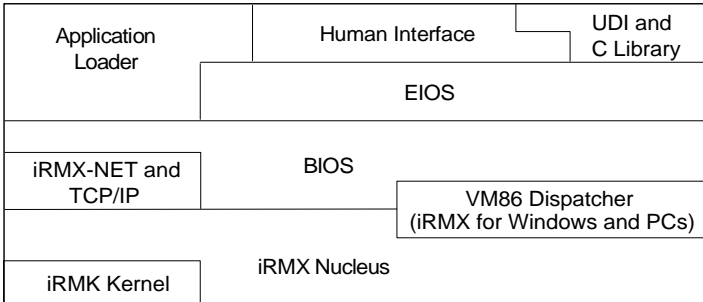
- iRMX OS concepts
See also: *Introducing the iRMX Operating Systems*
- iRMX OS configuration
See also: *System Configuration and Administration*
- Developing your own iRMX application jobs
See also: *Programming Techniques and AEDIT Text Editor; System Concepts*
- Hardware-specific parameters as documented in hardware reference manuals for your system components

Overview of ICU-configurable Options

The ICU enables you to configure a variety of options in an iRMX system, such as parameters related to the hardware, OS layers, application jobs, device drivers, system memory requirements, or networking. Using the ICU, you can:

- Fine-tune hardware-related parameters to enhance the performance of the system
- Reduce the overall size of the OS by removing layers
- Add support for your own application jobs, enabling the system to automatically invoke your job when booting
- Add or remove support for Intel-provided device and file drivers or add your own custom drivers to the OS
- Adjust system memory requirements to suit a particular configuration
- Add support for various local area networking architectures

Figure 1-1 illustrates the layers of the iRMX OS, and how they relate to each other.



iRMX is a registered trademark of Intel Corporation.

OM02727

Figure 1-1. Layers of the iRMX OS

ICU File Descriptions

There are three kinds of files associated with the ICU executable program file, *icu386*:

Internal files	Describe all of the configurable parameters of the system to the ICU.
Definition files	Describe a particular system configuration.
Generation files	Create a bootable iRMX OS when executed.

ICU Internal Files

From the internal files, the ICU creates a definition file that you can modify interactively; from these files, the ICU creates the generation files for your system.

These internal files, supplied with the ICU, are associated with the operation of the ICU:

Filename	Description
<i>icu386.scm</i>	Screen master file
<i>icu386.tpl</i>	Template file for system generation

The iRMX OS installation process places these files in the *icu* subdirectory. If you are not using the installation directory as your working directory, you must specify a different pathname when invoking the ICU.

See also: Default file structure, *Installation and Startup*

Screen Master and Template Files

The ICU presents configuration parameters grouped in screens on your terminal. The *icu386.scm* screen master file defines all of the allowable screens for any ICU definition file.

The ICU uses the *icu386.tpl* template file to produce generation files.

See also: Screen formats, creating generation files, in this manual

The *icu386.scm* and *icu386.tpl* files are very important to ICU operation; do not modify them directly. Use only the User Device Support (UDS) and ICU Merge (ICUMRG) utilities to add screens. These utilities enable you to add support for your custom drivers to the ICU, although you can add the drivers in loadable form.

See also: Making a driver loadable, *Driver Programming Concepts*;
Adding drivers with the UDS and ICUMRG utilities, in this manual

Definition Files

Definition files specify all of the screens and parameter values that apply to a particular iRMX system configuration. The OS provides various definition files for you to use as starting points in your system configuration. These are installed in the same directory as the other ICU files; look at the **.bck* files to see what is currently available.

The ICU presents the definition file to you, one screen at a time. You change the value of individual screen parameters to tell the ICU how to modify the definition file. Only the ICU can create and modify a definition file.

In a definition file, screens are arranged according to the various subsystems of the system configuration. Screen parameters represent two types of information:

- Initialization parameters, which determine how the system boots
- System description parameters, which specify the OS layers and drivers that form your system, and options pertaining to these layers and devices

Generation Files

Generation files are the end result of running the ICU. The ICU produces ASM386 source (**.a38*), ASM286 source (**.a28*), binder/builder configuration (**.cf*), BLD386 (**.bld*), and submit (**.csd*) files.

Executing the generation submit file runs the appropriate compilers and utilities to create the new iRMX OS. This file has the same filename base as your definition file but with a *.csd* extension.

Practice Session: Invoking the ICU

The syntax for invoking the ICU is:

```
[pathname]ICU386 [in_file TO] out_file
```

Where:

pathname Specifies the directory that contains the ICU internal files. You can omit it if this is your current working directory, or if the ICU directory is in your search path.

in_file Specifies the name of the definition file where the ICU obtains starting configuration information. This is typically an existing file from a previous session or an OS-supplied definition file. For an OS-supplied definition file, pick one that closely matches your hardware.

See also: Appendix D

out_file Specifies the name of the definition file to which the ICU writes updated configuration information.

Always specify an input and output file; this guarantees a backup copy of your starting point. If you omit *in_file*, *out_file* is both the input and output file.

Now, try out the ICU for yourself. Use one of the supplied definition files in the installation directory as the input file and *test.def* as the output file. For example, enter this:

```
icu386 48612scp.bck to test.def
```

You will see the main menu screen, meaning you are in command mode:

```
For general help in any screen enter H <CR>.
```

```
The following commands are available
```

```
Change
```

```
Generate
```

```
List
```

```
Save
```

```
Quit
```

```
Exit
```

```
Replace
```

```
Detail-Level
```

```
Backup
```

```
ENTER COMMAND :
```

Other messages may appear if there is a problem.

Enter the character `c` followed by a `<CR>`. This enters the **change** command, which puts you into screen-editing mode. You will see the first screen, **HARD**:

```
(HARD)      Hardware

(BUS) System Bus Type [ 1=MBI / 2 = MBII ]
(TP) 8254 Timer Port [0-0FFFFH]
(CIL) Clock Interrupt Level [0-7]
(CN) Timer Counter Number [0,1,2]
(CIN) Clock Interval [0-65535 msec]
(KTR) Kernel Tick Ratio [1-65535]
(CF) Clock Frequency [0-65535 khz]
(TPS) Timer Port Separation [0-0FFH]
(EMU) Emulate Numeric Processor [Yes/No]
(IF) Initialize On-board Functions [0=No / 1-0FFH]]
(BIP) Board Initialization Procedure [1-45 Chars]

Enter [Abbreviation = new value / Abbreviation ? / H]
:
```

Press `<CR>` again and you will see the next screen.

Screen Display Command Characters

In screen-editing mode, enter various characters in upper or lower case to activate screen display commands:

Character	Command	Action
B	Back	Move back one screen.
N	Next	Move to the next logical screen.
C	Cancel	Exit screen-editing mode; go back to the main menu.
F	Find	Display the list of all ICU screens; enter a screen abbreviation to jump to that screen.
H	Help	Display the list of screen-editing commands that apply to the current screen format.
R	Redisplay	Redisplay the current screen with changes.

On some systems, precede a command character by the ^ (caret) character.

To get back to the main menu, enter `c` to cancel the editing session. To leave the ICU without saving any changes to your definition file, enter `q` from the main menu.

See also: Editing ICU screens, in this manual, for more about screen display/editing commands;
 Example ICU session, in this manual, for step by step practice

Main Menu Commands

Activate main menu commands by entering the first letter of the command or the full command name at the command prompt:

```
ENTER COMMAND :
```

The ICU is not case sensitive; enter commands in upper or lower case.

If you enter any command that is not on the main menu (such as **help**), help information for the main menu screen appears.

If you enter `<Ctrl-C>` during the execution of an ICU command, the ICU stops executing the current command, and returns to the main menu screen. If you enter `<Ctrl-C>` in command mode, or during **save**, **quit**, **exit** or **backup** command operations, it is ignored.

In the command descriptions, parameters shown inside [] (square brackets) are optional.

Change Command

```
C[hange] [screen_abbrev] <CR>
```

Where:

- | | |
|---------------|---|
| C or Change | Puts the ICU into screen-editing mode. |
| screen_abbrev | Specifies the screen you want to edit. If you enter an abbreviation incorrectly, the help screen for abbreviations appears. |
| C? | Displays the names and abbreviations screen. |

Change enables you to edit the definition file. After executing this command, you can edit the current screen or move to the next screen by pressing <CR>.

If you enter a valid screen name but it is not in your definition file, the ICU displays the next screen and this warning:

```
*** Warning - The screen requested cannot be displayed
```

Generate Command

```
G[enerate] <CR>
```

Where:

- | | |
|---------------|---|
| G or Generate | Starts the process that produces the generation files necessary to create a bootable iRMX system. |
|---------------|---|

After using the **save** command to save the changes made in your ICU session, enter the **generate** command to create the system files required to build your system.

See also: Creating generation files, in this manual

List Command

```
L[list] [name] <CR>
```

Where:

L or List	Lists the contents of all screens in ASCII characters to a file or device.
name	Specifies the file or device. If you omit the name, the console device (:co:) is assumed. List to a filename rather than to the terminal since the display scrolls rapidly. If you want to use your terminal to review your definition file, use the change command to view just those screens you want.

Use **list** to record or review your ICU entries.

After this command executes, it notifies you that the definition file has been listed with a message like this, followed by the main menu screen:

```
The Definition File has been listed to file: TEST.LST
```

Save Command

```
S[ave] [pathname] <CR>
```

Where:

S or Save	Saves all the changes made in this session.
pathname	Specifies the pathname of a file to save definition file changes to, instead of the default <i>out_file</i> .

Save updates your definition file with all of the changes made during the current ICU session. Then it notifies you that the specified file has been updated with a message like this, followed by the menu screen:

```
The Definition File has been written to file: TEST.DEF
```

Quit Command

Q[uit] <CR>

Where:

Q or Quit Terminates your current ICU session without updating the definition file. In screen-editing mode, use this command to return to the main menu.

When you enter the **quit** command, the ICU may display the prompt:

```
Do you want to quit without saving your changes? y/[n]
```

The ICU only displays this prompt if you use the **quit** command after editing the definition file without saving any changes.

Exit Command

E[xit] [pathname] <CR>

Where:

E or Exit Exits the ICU, saving all the changes made in this session.

pathname Specifies the pathname of a file to save definition files to instead of *out_file*.

The **exit** command terminates the ICU session while saving the definition file.

Replace Command

R[eplace] <CR>

Where:

R or Replace Changes the character that precedes screen display/editing commands. The default is ^. If your terminal does not support ^, or you prefer a different character, use this command to change it.

After you enter the **replace** command, the ICU displays this:

```
Input new control character :
```

Enter the new character, followed by a <CR>. You may use any character except a UDI delimiter. If you enter a character that is unacceptable, you will receive this message:

```
*** ERROR. The control character cannot be a UDI delimiter.
```

```
Input new control character :
```

See also: *Delimiters, System Concepts*;
 dq_get_argument call, *System Call Reference*

Detail-level Command

D[etail-Level] <CR>

Where:

D or Detail-Level Enables selective screen displays.

Rather than viewing all the screens, you can elect to see only screens of a particular type:

Option	Description
All	Shows all screens
Devices	Shows only device screens
Operating System	Shows all non-hardware related screens (default)
Jobs	Shows only job screens, such as User, I/O and User Modules screen

Enter the first character of the selections shown. The ICU redisplay this screen until you enter a valid response.

Backup Command

```
B[ackup] filename <CR>
```

Where:

B or Backup Writes a backup file.

filename Name of the file that will contain the backup information.

The **backup** command writes an ASCII backup file containing a list of all the parameter abbreviations and their current values. The backup file is used as input to the ICU during a restore process.

See also: Restoring a definition file, in this manual

When the **backup** command has completed, the ICU indicates successful completion. For example, if you backed up your definition file to a file named *update.bck*, this would be displayed:

```
The Definition File has been backed up to file: UPDATE.BCK
```



ICU Operations **2**

This chapter describes these ICU operations:

- Editing ICU screens
- Generating and testing the new system
- Restoring a definition file from a backup file

Editing ICU Screens

Each ICU screen contains a related set of configuration parameters with default values. You can change any of the parameters by entering a new value at the screen prompt.

Start screen-editing mode by entering the **change** command from the main menu. The first screen appears. Pressing <CR> enables you to proceed to the next screen. The changes you make to screens are recorded in the definition file when you use:

- The **save** command while still in the ICU
- The **exit** command when you leave the ICU

Screen Formats

There are three ICU screen formats; you edit them using a set of special screen display/editing commands. The screen format determines which display/editing commands are applicable. The three screen formats are:

- Fixed format
- Repetitive format
- Repetitive-fixed format

Fixed Screen Format

Most screens use fixed screen format to display information. Fixed screen format enables you to make changes by entering the parameter abbreviation, the equal (=) sign, and the new value, then pressing <CR>. This HARD screen is typical of the fixed screen format:

```
(HARD)      Hardware

(BUS) System Bus Type [ 1=MBI / 2 = MBII ]
(TP) 8254 Timer Port [0-0FFFFH]
(CIL) Clock Interrupt Level [0-7]
(CN) Timer Counter Number [0,1,2]
(CIN) Clock Interval [0-65535 msec]
(KTR) Kernel Tick Ratio [1-65535]
(CF) Clock Frequency [0-65535 khz]
(TPS) Timer Port Separation [0-0FFH]
(EMU) Emulate Numeric Processor [Yes/No]
(IF) Initialize On-board Functions [0=No / 1-0FFH]]
(BIP) Board Initialization Procedure [1-45 Chars]

Enter [Abbreviation = new value / Abbreviation ? / H]
:
```

Repetitive Screen Format

In a repetitive screen, the ICU repeats the same prompt many times. You add or delete lines on the screen at the screen prompt. This PREF screen uses a repetitive screen format:

```
(PREF)          Prefixes

          Prefix = 1-45 characters
[ 1] Prefix = :PROG:
[ 2] Prefix = :UTILS:
[ 3] Prefix = :UTIL286:
[ 4] Prefix = :SYSTEM:
[ 5] Prefix = :LANG:
[ 6] Prefix = :ICU:
[ 7] Prefix = :$:
[ 8] Prefix =

Enter Changes [Line Number = new value / ^D Number / ? / H]
:
```

Each time you enter a line of information in this screen, you define a logical name for a directory. Identifying numbers precede each line of information. To make an addition to this screen, you type the line number, the equal sign (=), the new value, and <CR>; the ICU displays the screen again to show the added line.

Repetitive-fixed Screen Format

The repetitive-fixed screen format, like this User Jobs screen, repeats a full fixed screen of information any number of repetitions:

```
(USERJ)      User Jobs
(NAM) Job Name [0-14 Chars]
(SEQ) Job Sequence [Before/After]
(ODS) Object Directory Size [0-3840]
(PMI) Pool Minimum [20H-0FFFFFFFH]
(PMA) Pool Maximum [20H-0FFFFFFFH]
(MOB) Maximum Objects [1-0FFFFFFH]
(MTK) Maximum Tasks [1-0FFFFFFH]
(MPR) Maximum Priority [0-255]
(EHS) Exception Handler Entry Point [1-31] Chars]

(EM)  Exception Mode [Never/Prog/Environ/All]
(PV)  Parameter Validation [Yes/No]
(TP)  Task Priority [0-255]
(TSA) Task Entry Point [1-31 Chars]

(VAR) Public Variable Name [1-31 Chars]

(SSA) Stack Segment Address [SS:SP]
(SSS) Stack Size [0-0FFFFFFH]
(NPX) Numeric Processor Extension Used [Yes/No]

Enter Changes [Line Number = new value / ^D Number / ? / H]
:
```

When you complete a repetitive-fixed screen like User Jobs, a one-line query screen asks:

Do you have any/more user Jobs?

If you answer Yes, the ICU presents another USERJ screen. Each time you make entries to one of these screens you define a new user job. The ICU repeats this screen until you respond with No, and/or <CR> to the prompt.

Device driver screens are also repetitive-fixed format.

Explanation of Screen Elements

The following defines various screen parts:

```
( SCABV)  SCREEN_NAME

(ABV) PARAMETER DEFINITION [range of values]      XXX
(ABV) PARAMETER DEFINITION [range of values]      XXX

      •
      •
      •

Enter [Abbreviation = new value / Abbreviation ? / H / ? ]
: <prompt line>
```

Where:

(SCABV) Screen abbreviation. The characters enclosed in parentheses uniquely identify the screen being displayed.

SCREEN_NAME
The full name of the screen.

(ABV) Parameter abbreviation. The characters enclosed in parentheses identify the screen parameter whose existing value can be changed.

PARAMETER DEFINITION
A brief description of the parameter.

[range of values]
The range of acceptable values for this parameter.

XXX The default value of this parameter. If the existing value is not what you want, replace it with any other value within the range of values. This manual does not illustrate defaults since they vary by system bus type and definition file.

<prompt line>
The cursor appears at the beginning of this line ready for you to enter screen display/editing commands or make changes to the screen by entering one of these:

- An abbreviation, =, and a new value
- An abbreviation and ? (question mark) for parameter help
- H, for screen format and display/editing command help
- ?, for a description of the screen and usage information

Entering New Values

You can enter a new value for a parameter at the prompt line, depending on the range of values for that parameter:

Device/filename	A device or filename can be any device or filename acceptable to the OS.
Integer constants	Constants must be unsigned integers that you can enter in decimal (no radix), hexadecimal (H), milliseconds, microseconds or kilobytes (K).
Addresses	Enter address values in the form SELECTOR:OFFSET. Specify the radix, either explicitly or by default, for both portions of an address. For example, specify the selector of 900H and an offset address of 384H as 900H:384H.

The ICU accepts values within range without checking their reasonableness. So it is possible to generate a nonfunctional version of the OS without seeing any errors. If you enter a value outside its range, you will receive an appropriate error message with the current value of the parameter remaining unchanged.

When you leave screen-editing mode and return to the main menu, the ICU performs a number of logical tests, such as checking that the locations reserved for the System Memory and the Free Space Manager do not overlap. If it detects such an error, the ICU issues an error message. You must make the necessary corrections to your definition file or you will not be able to generate a working system.

See also: Interactive error messages, in Appendix A

Screen Display/Editing Commands

Display commands (**back**, **next**, **cancel**, **help**, **redisplay**) enable you to move around in the definition file and control the display of screens. Editing commands (**search**, **delete**, **insert**, **copy**) let you manipulate repetitive and repetitive-fixed screens. Enter these commands while in screen-editing mode:

Entered	Command	Action
B	Back	Takes you to the previous screen
N	Next	Takes you to the next logical screen
C	Cancel	Takes you back to the main menu
F	Find	Takes you to the specified screen, or presents a list of screens
H	Help	Displays a list of screen-editing commands that apply to the current screen format
R	Redisplay	Redisplays the screen you are currently editing, with changes
S	Search	Searches repetitive-fixed screens for the specified string
^D	Delete	For repetitive screens, deletes a specified line within; for repetitive-fixed screens, deletes the current screen and all associated screens
^I	Insert	For repetitive screens, inserts a new line; for repetitive-fixed screens, inserts a new screen in front of the current one
^CO	Copy	Copies the current repetitive-fixed screen

Depending on your system, you may precede commands with a ^ (carat), for example:

^S

Back Command

- B Enables you to move backwards from the current screen to the previous screen. Moving backwards beyond the beginning of the definition file returns you to command mode.

Next Command

- N Displays the next logical screen. For example, if you are entering data on a UINFO screen and enter No, the first DUIB screen for that driver is displayed. If you enter No again, the DINFO screen of the next driver is displayed, and so on. If you enter No in the last screen, the ICU returns to command mode.

Cancel Command

- C Cancels the screen editing session. Returns you to command mode from any ICU screen. The ICU then displays the main menu.

Find Command

- F [`<screen abbreviation>`]
Finds and displays the screen you indicated in the screen abbreviation. This command enables you to jump from one screen to another.

To let you know that the specified screen is not a part of this definition file, the ICU shows you the next logical screen, and displays this message:

```
*** WARNING - The screen requested cannot be displayed
```

To display the help screen for abbreviations, use the **find** command without specifying the screen abbreviation. When this screen appears, you see this prompt:

```
ENTER screen abbreviation or <CR> for next list:
```

Enter a screen abbreviation and you will jump to that screen. Press <CR> only and you will see the second part of the abbreviations help screen. This second part includes abbreviations for device driver screens followed by this prompt:

```
Enter screen abbreviation:
```

If you want to exit this command without entering an abbreviation, press <CR> to return to the screen you were in when you invoked the **find** command.

Help Command

- H Displays the list of screen-editing commands that apply to the current screen format.

Redisplay Command

- R Redisplays the current screen, showing any changes made. If you are in a help screen, this command returns you to the last definition file screen you were working on.

Search Command

S <string>

Searches repetitive-fixed screens of the same logical type for the specified string. When you enter this command, the search begins in the next screen of that logical type and searches all parameter fields that contain character values, as specified by the range of allowable values. The search continues until the ICU finds a match. If it finds no match, the cursor remains at its current position and the ICU displays this message:

```
No next match found
```

Suppose you have 20 DUIB (Device-unit Information Block) screens for the MSC (Mass Storage Controller) driver and you want to find the screen that defines the NAM (device name) parameter as w0. First, you get to the first MSC screen using the **find** command. Then you enter:

```
S w0 <CR>
```

The ICU searches all the DUIB screens for this device until it finds w0. It then displays that screen.

Delete Command

^D [<number>]

Enables you to delete a line in a repetitive screen. The number you enter identifies the entry to delete. After the line is deleted, the remaining lines are renumbered and the screen is displayed again. To replace a line you must first delete the existing line, and then insert the new line.

The **delete** command also enables you to delete the current repetitive-fixed screen and/or associated screens. For example, if you want to delete a device driver, just delete the DINFO screen for that device. The ICU automatically deletes all the other screens associated with it.

Here is how to delete a line in a repetitive screen, using line 8 on the PREF screen:

```
(PREF)      Prefixes

      Prefix = 1-45 characters
[1] Prefix = :PROG:
[2] Prefix = :UTILS:
[3] Prefix = :UTILS286:
[4] Prefix = :SYSTEM:
[5] Prefix = :LANG:
[6] Prefix = :ICU:
[7] Prefix = :$:
[8] Prefix = :WORK1:
[9] Prefix =

Enter Changes [Number = new value / ^D Number / ? / H ]
: ^d 8 <CR>
```

After the ICU deletes line 8, the screen is displayed again.

```
      Prefix = 1-45 characters
[1] Prefix = :PROG:
[2] Prefix = :UTILS:
[3] Prefix = :UTILS286:
[4] Prefix = :SYSTEM:
[5] Prefix = :LANG:
[6] Prefix = :ICU:
[7] Prefix = :$:
[8] Prefix =

Enter Changes [Number = new value / ^D Number / ? / H ]
:
```

Insert Command

`^I [<number> =];`

Enables you to insert an additional repetitive-fixed screen in front of the current screen, or add a new line to a repetitive screen.

Suppose you want to insert a new prefix on line 8 of the PREF screen. Enter:

```
8 = :WORK1: <CR>
```

or

```
^I 8 = :WORK1: <CR>
```

When the new line number is inserted, the ICU renumbers the remaining lines and displays the screen again. If the number you enter is larger than the actual number of lines in the screen, the ICU inserts the new line as the last line.

```
(PREF)      Prefixes

            Prefix = 1-45 characters
[1] Prefix = :PROG:
[2] Prefix = :UTILS:
[3] Prefix = :UTILS286:
[4] Prefix = :SYSTEM:
[5] Prefix = :LANG:
[6] Prefix = :ICU:
[7] Prefix = :$:
[8] Prefix = :WORK1:
[9] Prefix =

Enter Changes [Number = new value / ^D Number / ? / H ]
:
```

If you are entering numerical data on a repetitive screen, you can enter the data in any order. However, the ICU automatically arranges your data in proper order and displays it on the screen. For example, if you enter these three insert commands on the MEMS screen:

```
^I 1 = 2000H, 4000H
```

```
^I 2 = 80000H, 90000H
```

```
^I 3 = 10000H, 12000H
```

The ICU sorts the data in ascending order and redisplay the lines:

```
1 = 2000H, 4000H
2 = 10000H, 12000H
3 = 80000H, 90000H
```

Copy Command

^CO Enables you to insert an identical copy of the current repetitive-fixed screen in front of the current screen.

Generating and Testing the New System

Generating and testing the new system is the final phase of the ICU process. This is when you build a new iRMX system as described by your definition file, and give it a test run. The ICU generates all of the necessary source files; your development system provides the compilers and other utilities that create a bootable iRMX system. This process consists of these steps:

- Creating generation files
- Executing the generation submit file that compiles, assembles, binds, and builds the new bootable system
- Testing the system by booting or loading the iRMX OS image created by the submit file, and verifying proper operation

Creating Generation Files

When you have finished editing your definition file, return to command mode to generate the system, using commands from the ICU main menu screen:

1. Use the **save** command to preserve any changes made to your definition file. You may also use the **list** command to create an ASCII record of this system configuration for later reference.
2. Use the **generate** command to create generation files.
3. Use the **exit** command to save your changes and leave the ICU.

The generation process creates ASM386 source files (*.a38) and binder/builder configuration files (*.cf) for each configured subsystem, a build file (*def_file.bl*) and a submit file (*def_file.csd*) for the entire system. The number of files produced is configuration-dependent.

Generating with Prefix Option

Use the prefix option with the **generate** command to distinguish different versions of your generation files from each other. When you enter the command, the ICU displays a prompt asking you for the prefix letter you wish to assign to the files created by the ICU:

```
ENTER a letter to be used as prefix :
```

Suppose you want to generate 3 different versions of the system, versions X, Y, and Z; use the letter X as your prefix option on version X, Y on version Y, and Z on version Z. An X, Y, and Z will precede all the generation files associated with each version generated when you enter the **generate** command on the menu screen. In this case, the Nucleus files generated for version X will be:

```
XNTABL.A38  
XNUCDA.A38  
XNJOB.C.A38
```

The files created for all other subsystems will also be preceded by the letter X.

It is also a good idea to have your input definition file start with the same prefix letter you assign to the generated files. This enables you to determine which definition file goes with each set of prefixed output files, and which generation submit file produces that system. If you create a set of files with a prefix that already exists, the ICU overwrites the previous versions.

If you do not want to use the prefix option, enter a <CR> when you are prompted for the prefix. This causes the ICU to generate the output files without a prefix.

Executing the Generation Submit File

Among the generation files is a submit file with the same filename as your definition file, but with a *.csd* extension. For example, if the definition file used is called *48612s.def*, the submit file is *48612s.csd*. After you leave the ICU, execute the submit file using the **submit** command and wait for your system to be generated.

See also: **submit** command, *Command Reference*

The generation submit file does this:

- Compiles source modules for each subsystem
- Binds object files together with any libraries needed by each subsystem
- Binds the subsystems and builds a bootable system

While the submit file executes, there should be no errors. If there are, they indicate serious problems that will prevent the successful generation of your system, even though the submit file keeps executing. Errors and warnings appear in the log file created by the **submit** command. Your compiler documentation contains explanations of error conditions.

See also: System generation error and warning messages, in Appendix A

To successfully execute statements within the generation submit file, you must have a development environment that has the appropriate compilers and utilities installed.

See also: Development environment, Appendix B

Testing the System

The normal development cycle is to load your system, test it, correct any errors, and reassemble or recompile any faulty program code. Then, run the ICU to regenerate your system, and load the system again. Continue until you have created a functional target system.

You can load and debug in several ways. You may use the bootstrap loader or **loadrmx** to load the new system from secondary storage, use a debugger program, or use an in-circuit emulator.

Once you have created your final system, you can fine-tune the system through repeated generation cycles. For example, you can minimize the memory locations allocated for the system by editing the MEMS and MEMF screens. You can also copy your final system to PROM.

See also: *Programming Techniques and AEDIT Text Editor*, for more information on developing iRMX OS applications, and placing the OS into PROM

Restoring a Definition File

If you create a backup file using the **backup** command from the main menu, the restore process can create a definition file from that backup file. To start the restore process, invoke the ICU with the backup file as the input file, specifying an output file that is a new or an existing definition file that you want to restore to. Here is an example invocation line:

```
ICU386 48612S.BCK to 48612S.DEF
```

The ICU signs on and responds with this:

```
Do you want to restore from the file? [y]/n
```

If you respond No, the ICU stops executing. Yes continues the restore process. If the output filename already exists, the ICU displays this message:

```
File <output_file> exists.  OVERWRITE?  [y]/n:
```

In this example you specified an output file. If you did not specify an output filename on invocation you are also prompted:

```
Do you want to OVERWRITE the input file <in_file> ? [y/n]
```

If you enter No, the ICU prompts:

```
Enter new output filename:
```

You may want to use the *.def* suffix for the output file. Once the restore process is running, the ICU displays a series of * (asterisks) on the screen. If the process reaches completion with no problems, the ICU displays the main menu and you proceed as usual.

However, if an error is encountered, the ICU displays this message:

```
*** ERROR while restoring
The Definition File has been restored to the file: <fname.BCK>
To see the RESTORE messages, inspect the Log File: <fname.LOG>

*** NOTICE: The Definition File may have been modified while
restoring.
You may either continue to the ICU Main Menu, or
you may exit the ICU to inspect the Log File.

Continue to the ICU Main Menu? [y/n]
```

Log File

The restore log file lists each screen name followed by any errors or warnings that occurred while restoring that screen. It also lists abbreviations of fields that were not restored. The log file has the same name as the output file but with a *.log* extension. The log file enables you to compare the backup definition file and the restored file to see which values were not restored. Run the ICU again to correct the fields in error. After that, proceed as usual.

Suppose that while restoring from file *up0.bck*, the ICU could not restore the CF parameter on the HARD screen. The log file entry would look like this:

```
ICU386 <version> Restoring from file : up0.bck <date> <time>

----          Screen : HARD          ----
*** ERROR - number expected
              In field : CF

----          Screen : INT           ----
```

This example shows only a portion of the log file. The actual file will list all the screen names in the definition file. The values for <version>, <date>, and <time> in the heading are variables.

The error messages in the log file are the same as the ICU interactive error messages.

See also: Interactive error and warning messages, in Appendix A



Example ICU Session 3

This chapter is a tutorial ICU session. In it, you will learn some useful screen editing techniques while you follow these steps in the ICU process:

1. Get started
2. Enter screen-editing mode
3. Edit the definition file
4. Save and list the edited definition file
5. Generate the system

This ICU session uses a Multibus (MB) I system as the hardware platform, with an SBC 486/12S CPU board and an SBC 221 disk controller board. You will configure the system to boot from the 221 controller rather than the 486/12S onboard SCSI controller. The offboard disk controller uses the MSC device driver; the SCSI controller uses the Intel Peripheral Controller Interface driver.

Sample command lines that you enter are shown **in this type**. You may enter all commands in upper or lower case.

See also: Screen display/editing commands, in this manual

Step 1: Getting Started

Invoke the ICU, creating a new definition file from an existing one, with this command line:

```
icu386 48612s.bck to newsys.def
```

The main menu screen appears:

```
For general help in any screen enter H <CR>.

The following commands are available

Change
Generate
List
Save
Quit
Exit
Replace
Detail-Level
Backup

ENTER COMMAND :
```

You are now in command mode. You can enter any one of the commands listed on the screen. Typically, you spend very little time in command mode while using the ICU; most of the time you work in screen-editing mode.

Step 2: Entering Screen-editing Mode

Enter screen-editing mode with the **change** command:

```
c <CR>
```

You are now editing the definition file *newsys.def*. This definition file is a working copy of *48612s.bck*. The first screen appears:

```
(HARD)      Hardware

(BUS) System Bus Type [ 1=MBI / 2 = MBII ]
(TP) 8254 Timer Port [0-0FFFFH]
(CIL) Clock Interrupt Level [0-7]
(CN) Timer Counter Number [0,1,2]
(CIN) Clock Interval [0-65535 msec]
(KTR) Kernel Tick Ratio [1-65535]
(CF) Clock Frequency [0-65535 khz]
(TPS) Timer Port Separation [0-0FFH]
(EMU) Emulate Numeric Processor [Yes/No]
(IF) Initialize On-board Functions [0=No / 1-0FFH]]
(BIP) Board Initialization Procedure [1-45 Chars]

Enter [Abbreviation = new value / Abbreviation ? / H]
:
```

This is the HARD screen, one of many that are part of this definition file. The screen contains a number of parameter lines followed by a prompt line. Each parameter line includes a parameter abbreviation and a range of legal values that you may enter. The prompt line appears at the bottom of the screen, ready for you to enter a new value for a specific parameter:

```
Enter [Abbreviation = new value / Abbreviation ? / H]
```

This is what you enter at the prompt line to change the CIL (Clock Interrupt Level) parameter from 0 to 1:

```
cil=1
```

How To Access ICU Help Screens

While in screen-editing mode, you can access online help information for the whole screen or each parameter in it. If you enter ? only, a help message appears for the current screen. If you enter the parameter abbreviation followed by ?, help information about that parameter appears.

For example, if you enter the abbreviation for the CIL parameter with a question mark:

```
cil? <CR>
```

The ICU displays this:

```
*****
```

```
You must specify the interrupt line on the master PIC to
which the timer is connected. Note that this is not an
encoded interrupt level. The default value for this
parameter is set at 0.
```

Step 3: Editing the Definition File

Pressing <CR> repeatedly pages through all screens in a definition file. The screens appear in order from hardware, to OS layers and associated jobs, to drivers. To move to another screen, use the **find** command and supply the abbreviation. To see all the screen abbreviations, use the find command incorrectly (with no argument) as follows:

f <CR>

(HARD)	Hardware	(MBII)	MBII Hardware	(INT)	Interrupts
(SLAVE)	Slave Interrupt	(SUB)	Sub-systems	(MEMS)	Memory for System
(MEMF)	Memory for FSM	(PIMM)	Identity Mapped	(OSEXT)	OS Extensions
(HI)	Human Interface	(HIJOB)	HI Jobs	(RES)	Res./Recovery User
(PREF)	Prefixes	(HILOG)	HI Logical	(APPL)	Application Loader
(EIOS)	EIOS	(ABDR)	Auto Boot Dev	(LOGN)	Logical Names
(IOUS)	I/O Users	(IOJOB)	I/O Jobs	(BIOS)	BIOS
(BIOFD)	Bios File Drvrs	(PHYFD)	Phys File Drvr	(STRFD)	Stream File Drvr
(DOSFD)	DOS File Drvr	(NAMFD)	Named File Drvr	(EDSFD)	EDOS File Drvr
(IDEVS)	Intel Devices	(PCDEV)	PC Bus Devices	(USERD)	User Devices
(UDDM)	UDS Driver Mods	(SDB)	System Debugger	(SDM)	Sys. Monitor
(NUC)	Nucleus	(NCOM)	Comm. Service	(DISPJ)	DOS Dispatcher
(DRINT)	DOS Resrvd Ints	(SYSJ)	System Jobs	(PCIJ)	PCI Job
(PCISC)	PCI Server	(DLJ)	Downloader Job	(ATCJ)	ATCS/279/ARC Srv
(ATC50)	ATCS/450 Server	(BSJ)	BootServer Job	(FPIJ)	FPI Server Job
(SSKJ)	SoftScope Kernel	(USERJ)	User Jobs	(USERM)	User Modules
(TPUJ)	3rd Party UserJ	(ROM)	ROM Code	(NET)	Network Subsys
(ICMPJ)	iNA960 COMMputer	(NS)	Name Server Cfg	(NSDOM)	NS Srch. Domains
(IMIPJ)	iNA960 MIP Dvr	(MIP1)	MBI MIP Cfg.		

ENTER screen abbreviation or <CR> for next list:

Press <CR> to see the rest of the screen abbreviations including abbreviations for device driver screens:

<CR>

```
(MIP2) MBII MIP Cfg. (CEBI) CE Board Inst. (CEBN) CE Board Names
(MIPAT) PC Bus MIP Cfg. (LYR) irMX-NET Layers (RCJ) irMXNET Client
(FC) File Consumer (REM) Remote file Acc (RSJ) irMXNET Server
(FS) File Server (AFA) Apex File Acc (AFAU) Afa User
(PDEV) Public Devices (PDIR) Public Dir. (UDF) User Def. File
(CDF) Client Def. (TCPJ) TCP/IP Job (IFSP) Network xfaces
(ROUTE) IP Router Info. (TCPx) TCP/IP Params x (STRM) TCP/IP Str Info
(X25SJ) X.25 Server Job (X25CJ) X.25 Client Job (INCL) Includes & Libs
(CLIB) Shared C Libry. (GEN) Gen. File Name (COMNT) Comments screen
----- DEVICE DRIVER SCREENS -----
(D8274) 8274 (D8251) 8251A (D2530) 82530 (D350) iSBX350 LP
(DRAM) RAM Disk (D279) iSBX279A (DPCI) PCI Driver
-- MULTIBUS I DEVICE DRIVERS:
(D534) iSBC534 (D544) iSBC544A (D208) iSBC208
(D386) iSBCn86/12 LP (D220) iSBC220 (D8848) TCC (D214) Mass Storage
-- MULTIBUS II DEVICE DRIVERS:
(D224A) iSBC186/224A (D410) ATCS (DCOMM) iSBC486SX25/486DX33
-- AT BUS DEVICE DRIVERS:
(D120S) PC Serial (D120C) PC Console (D120P) PC Printer
(D120F) PC Floppy (D120W) PC Wini (D120A) PC SCSI
(D120H) H550 Serial (DDOSW) Dos Wini (DDOSF) Dos Floppy
(DOSCN) Dos Console
ENTER screen abbreviation:
```

These two lists contain all possible screens that a definition file may use. Your definition file will not use all of the screens listed here. If you get lost inside the ICU, the **find** command gets you back to this screen.

See also: Screens in Chapter 4 for screen displays

If you enter a screen abbreviation, you jump to that screen. For example, enter:

```
idevs <CR>
```

This screen appears:

```
(IDEVS)      Intel Device Drivers

(T74) 8274 Driver   [Yes/No]      (T51) 8251A Driver   [Yes/No]
(T30) 82530 Driver [Yes/No]      (RAM) RAM Disk      [Yes/No]
(L50) iSBX 350 LP  [Yes/No]      (G79) iSBX 279A     [Yes/No]
(PCI) PCI Driver   [Yes/No]

-- MULTIBUS I DEVICE DRIVERS:

(S14) MSC Driver   [Yes/No]      (TCC) TCC Driver    [Yes/No]
(L86) iSBC n86/12 LP [Yes/No]      (S20) iSBC 220      [Yes/No]
(S08) iSBC 208     [Yes/No]      (T34) iSBC 534      [Yes/No]
(T44) iSBC 544A    [Yes/No]

-- MULTIBUS II DEVICE DRIVERS:

(S24) iSBC 186/224A [Yes/No]      (S10) ATCS Driver   [Yes/No]
(ENC) COM1 and COM2 Serial Port Driver [Yes/No]
```

Use this screen to select which Intel device drivers the configuration uses. For example, you may omit the PCI driver by entering `pci=no`.

Check edits to a screen by entering the **redisplay** command:

```
r <CR>
```

The screen reappears with the change you just made.

Exploring Device Driver Screens

To look at a specific hard disk device driver and its parameters, display the help lists for screen abbreviations by entering the **find** command with no arguments (**£** <CR>). If you press <CR> instead of entering a screen abbreviation, the second of two lists appears:

```
(MIP2) MBII MIP Cfg. (CEBI) CE Board Inst. (CEBN) CE Board Names
(MIPAT) PC Bus MIP Cfg. (LYR) iRMX-NET Layers (RCJ) iRMXNET Client
(FC) File Consumer (REM) Remote file Acc (RSJ) iRMXNET Server
(FS) File Server (AFA) Apex File Acc (AFAU) Afa User
(PDEV) Public Devices (PDIR) Public Dir. (UDF) User Def. File
(CDF) Client Def. (TCPJ) TCP/IP Job (IFSP) Network xfaces
(ROUTE) IP Router Info. (TCPx) TCP/IP Params x (STRM) TCP/IP Str Info
(X25SJ) X.25 Server Job (X25CJ) X.25 Client Job (INCL) Includes & Libs
(CLIB) Shared C Libry. (GEN) Gen. File Name (COMNT) Comments screen
----- DEVICE DRIVER SCREENS -----
(D8274) 8274 (D8251) 8251A (D2530) 82530 (D350) iSBX350 LP
(DRAM) RAM Disk (D279) iSBX279A (DPCI) PCI Driver
-- MULTIBUS I DEVICE DRIVERS:
(D534) iSBC534 (D544) iSBC544A (D208) iSBC208
(D386) iSBCn86/12 LP (D220) iSBC220 (D8848) TCC (D214) Mass Storage
-- MULTIBUS II DEVICE DRIVERS:
(D224A) iSBC186/224A (D410) ATCS (DCOMM) iSBC486SX25/486DX33
-- AT BUS DEVICE DRIVERS:
(D120S) PC Serial (D120C) PC Console (D120P) PC Printer
(D120F) PC Floppy (D120W) PC Wini (D120A) PC SCSI
(D120H) H550 Serial (DDOSW) Dos Wini (DDOSF) Dos Floppy
(DOSCN) Dos Console
ENTER screen abbreviation:
```


Abbreviations for device driver screens appear at the bottom of the list. Each of these driver screens represents a DINFO (Device Information) table entry for supported Intel device drivers. For example, enter the MSC device driver screen abbreviation:

d214 <CR>

You see this DINFO screen:

```
(D214)          Mass Storage Controller Driver
(DEV) Device Name           [1-16 Chars]
(IL)  Interrupt Level      [Encoded Level]H
(ITP) Interrupt Task Priority [0-255]
(WIP) Wakeup I/O Port      [0-0FFFFH]
(IPA) I/O Processor Block Address [0-0FFFFFFH]
(SB)  Size of Buffer        [082C4H-0FFFFFFH]
```

Press <CR> and you are asked:

```
Do you want any/more Mass Storage Controller DEVICES ?
```

Press <CR> again to see the first of many UINFO (Unit Information) screens already created for this device. The ICU can provide a screen for every UINFO table and DUIB (Device-unit Information Block) associated with a particular DINFO screen.

See also: Chapter 4 for composites of device driver screens;
Online help descriptions of screen parameters;
Driver Programming Concepts, for definitions of DINFO, UINFO,
and DUIB data structures

If you enter Yes, the ICU creates a fresh UINFO screen for you and inserts it in front of the default UINFO screens, ready for you to enter values for parameters.

However, you should edit existing UINFO and DUIB screens whenever possible, especially if you are using an Intel device driver. The OS-supplied UINFO and DUIB screens may already provide support for your device, requiring only that you edit specific parameters. Here is the first UINFO screen for the MSC device driver:

```
(U214)   Mass Storage Controller Unit Information

(DEV) Device Name                [1-16 Chars]
(NAM) Unit Info Name             [1-16 Chars]
(MR) Maximum Retries             [0-0FFFFH]
(CS) Cylinder Size               [0-0FFFFH]
(NC) Number of Cylinders         [0-0FFFFH]
(NFH) Number of Heads/Fixed Disk [0-0FFH]
(NRH) Number of Heads/Removable Disk [0-0FFH]
(NS) Number of Sectors/Track     [0-0FFFFH]
(NAC) Number of Alternate Cylinders [0-0FFH]
(SSN) Starting Sector Number     [0-0FFFFFFFFH]
(BTI) Bad Track Information       [Yes/No]
(HLT) Head Load Time            [0-0FFH]
(SR) Step Rate                   [0-0FFH]
```

You can get from the UINFO screens to the DUIB screens in two ways: page through all of the UINFO screens by pressing <CR> repeatedly, or use the **next** command, which is quicker. Enter:

```
n <CR>
```

You are shown the first of many DUIB screens:

```
(I214) Mass Storage Controller Device-Unit Information

(DEV) Device Name                [1-16 Chars]
(NAM) Device-Unit Name          [1-14 Chars]
(PFD) Physical File Driver Required [Yes/No]
(NFD) Named File Driver Required  [Yes/No]
(DOS) DOS File Driver Required   [Yes/No]
(SDD) Single or Double Density Disks [Single/Double]
(SDS) Single or Double Sided Disks [Single/Double]
(EFI) Quad or Double Density Disks [8/5]
(SUF) Standard or Uniform Format   [Standard/Uniform]
(GRA) Granularity                [0-0FFFFH]
(DSZ) Device Size                [0-0FFFFFFFH]
(UN) Unit Number on this Device   [0-0FFH]
(UIN) Unit Info Name             [1-16 Chars]
(RUT) Request Update Timeout      [0-0FFFFH]
(NB) Number of Buffers            [0 or 1-0FFFFH]
(CUP) Common Update              [Yes/No]
(MB) Max Buffers                  [0-0FFH]
```

Setting Automatic Boot Device Recognition

To configure an MSC device, such as an SBC 221 controller, to boot the system, use the **find** command and supply the abbreviation for the Automatic Boot Device Recognition screen as the argument. (If you don't know the screen abbreviation, enter the **find** command with no arguments to get help on screen abbreviations.)

```
f abdr <CR>
```

```
(ABDR) Automatic Boot Device Recognition

(DLN) Default System Device Logical Name [1-12 Chars]
(DPN) Default System Device Physical Name [1-12 Chars]
(DFD) Default Sys Dev File Driver       [P/S/N/R/E/D]
(DO) Default System Device Owner's ID   [0-0FFFFH]
```

Change the name of the physical device that the system will boot from to (DPN). Then use the **redisplay** command to verify the change you just made:

```
r <CR>
```

When you are finished, leave screen-editing mode by using the **cancel** command:

```
c <CR>
```

Step 4: Saving and Listing the Edited Definition File

Now that you are back in command mode (indicated by the presence of the main menu screen), use the **save** command to store your screen edits to the definition file specified when you invoked the ICU:

```
s <CR>
```

When the save operation is complete, the ICU responds with this message:

```
The definition file has been written to file: NEWSYS.BCK
```

For future reference, list all of the parameters contained in the definition file to an ASCII file with the **list** command:

```
l newsys.lst <CR>
```

When the listing operation is complete, the ICU responds with this message:

```
The definition file has been listed to file: NEWSYS.LST
```

Step 5: Generating the System

Create all of the files to build the new system with the **generate** command:

```
g <CR>
```

You are prompted to:

```
ENTER a letter to be used as prefix :
```

Press <CR> for this example, to proceed with system generation.

See also: Generating with prefix option, in this manual

During generation, the ICU indicates the beginning and ending of each subsystem generation, and the completion of the generation process:

```
Beginning NUCLEUS File Generation
.....DONE
Beginning BIOS File Generation
.....DONE
Beginning EIOS File Generation
.....DONE
Beginning LOADER File Generation
.....DONE
Beginning HI File Generation
.....DONE
Beginning UDI File Generation
.....DONE
Beginning SDM III File Generation
.....DONE
Beginning SDB File Generation
.....DONE
Beginning CONTROL Files Generation
.....DONE
Beginning SUBMIT File Generation
.....DONE
Beginning BUILD File Generation
.....DONE

To create your application system, type:
SUBMIT NEWSYS.CSD <CR>
```

Use the **quit** command to leave the ICU:

q <CR>

Then execute the generation submit file and the new system is created for you.



Screen Names and Acronyms

Since the actual default values seen on the screens depend on the system bus type and definition file, this manual does not show default values on the sample screens. In general, use the defaults in the definition files unless you have a specific need to change them.

⇒ **Note**

This manual does not list all possible ICU screens, and it contains only a brief summary for each parameter. For more specific details, see the ICU help messages for each screen and parameter.

Acronym	Screen Name	Page
(ABDR)	Automatic Boot Device Recognition	67
(AFA)	Apex File Access.....	98
(AFAU)	AFA User	99
(APPL)	Application Loader.....	58
(ATC50)	MBII ATCS/450 Server Job	129
(ATCJ)	MBII ATCS/279/ARC Server Job	128
(BIOFD)	BIOS File Drivers	71
(BIOS)	BIOS.....	58
(BSJ)	MSA BootServer Job.....	130
(CDF)	Client Definition File.....	103
(CEBI)	COMMengine Board Instance Load Method for MBII .	90
(CEBN)	COMMengine Board Names.....	90
(CLIB)	Shared C Library.....	120
(COMNT)	Comments for Build File	60
(DISPJ)	DOS Extender Dispatcher Job.....	65
(DLJ)	MBII Downloader Job	128
(DOSFD)	DOS File Driver.....	71
(DRINT)	DOS Extender Reserved Interrupts	66
(EDSFD)	EDOS File Driver.....	71
(EIOS)	EIOS.....	67
(FC)	File Consumer Configuration	94
(FPIJ)	FPI Server Job.....	131
(FS)	File Server Configuration.....	97

(continued next page)

Acronym	Screen Name	Page
(GEN)	Generate File Names	72
(HARD)	Hardware	73
(HI)	Human Interface.....	76
(HIJOB)	HI Jobs	77
(HILOG)	HI Logical Names	79
(ICMPJ)	iNA 960 COMMputer Job.....	92
(IDEVS)	Intel Device Drivers	61
(IFSP)	TCP/IP Network Interfaces	107
(IMIPJ)	iNA MIP Driver Job.....	86
(INCL)	Includes and Libraries	80
(INT)	Interrupts	82
(IOJOB)	I/O Jobs	69
(IOUS)	I/O Users	69
(LOGN)	Logical Names	68
(MBII)	Multibus II Hardware	74
(MEMF)	Memory for Free Space Manager	84
(MEMS)	Memory for System.....	84
(MIP1)	MIP Configuration for MBI	87
(MIP2)	MIP Configuration for MBII	89
(MIPAT)	MIP Configuration for PC Bus.....	91
(NAMFD)	Named File Driver	71
(NCOM)	Nucleus Communication Service	117
(NET)	Network Subsystem	85
(NS)	Name Server Configuration	104
(NSDOM)	Name Server Search Domains	105
(NUC)	Nucleus.....	115
(OEXT)	O.S. Extension	118
(PCDEV)	PC Bus Device Drivers	61
(PCIJ)	PCI Server Job.....	126
(PCISC)	PCI Server Controller Configuration.....	127
(PDEV)	Public Devices.....	100
(PDIR)	Public Directory	101
(PHYFD)	Physical File Driver.....	71
(PIMM)	Paging Identity Mapped Memory	84
(PREF)	Prefixes.....	79
(QUERY)	Query screen	118
(RCJ)	iRMX-Net Client.....	93
(REM)	Remote File Access.....	95
(RES)	Resident/Recovery User.....	78
(ROM)	ROM Code.....	119
(ROUTE)	IP Router Information	107
(RSJ)	iRMX-NET Server.....	96
(SDB)	System Debugger.....	123
(SDM)	System Debug Console	123
(SLAVE)	Slave Interrupt Levels.....	82

(continued next page)

Acronym	Screen Name	Page
(SSKJ)	SoftScope Kernel Job.....	132
(STRFD)	Stream File Driver	71
(STRM)	Streams Parameters	112
(SUB)	Sub-systems	122
(SYSJ)	System Jobs	125
(TCP1)	TCP/IP Tunable Parameters 1.....	108
(TCP2)	TCP/IP Tunable Parameters 2.....	109
(TCP3)	TCP/IP Tunable Parameters 3.....	111
(TCPJ)	iRMX TCP/IP.....	106
(TPUJ)	Third Party Tools Generated User Job.....	133
(UDDM)	UDS Device Driver Modules	65
(UDF)	User Definition File	102
(USERD)	User Devices.....	64
(USERJ)	User Jobs	135
(USERM)	User Modules.....	136
(X25CJ)	X.25 Client Job.....	114
(X25SJ)	X.25 Server Job	113

Application Loader Screen

```
(APPL)           Application Loader

(ASC)  All System Calls [Yes/No=only RQ$A$LOAD]
(DMP)  Default Memory Pool Size [0-0FFFFFFFH]
(RBS)  Read Buffer Size [0-0FFFFH]
(VSG)  Virtual Segment Size [0-0FFFFFFFH]
```

- (ASC) **Yes** Includes all the AL system calls and requires the EIOS.
No Includes only the **rq_a_load** system call and does not require the EIOS.
- (DMP) Specifies in 16-byte paragraphs the minimum memory pool size allocated to I/O jobs created by the AL.
- (RBS) Specifies in bytes the buffer size the AL uses when loading data from secondary storage.
- (VSG) Specifies in bytes the virtual segment size created for each flat model program that the AL loads.

BIOS Screen

```
(BIOS)           BIOS

(ADP)  Attach Device Task Priority [1-255]
(TF)   Timing Facilities Required [Yes/No]
(TTP)  Timer Task Priority [0-255]
(CON)  Connection Job Delete Priority [0-255]
(ACE)  Ability to Create Existing Files [Yes/No]
(SMI)  System Manager ID [Yes/No]
(CUT)  Common Update Timeout [0-65535]
(TSC)  Terminal Support Code [Yes/No]
(CST)  Control-Sequence Translation [Yes/No]
(OSC)  Terminal OSC Controls [Yes/No]
(TSP)  Tape Support [Yes/No]
(PMI)  BIOS Pool Minimum [0-0FFFFFFFH]
(PMA)  BIOS Pool Maximum [0-0FFFFFFFH]
(LDV)  Number of loadable Devices [0-0FFFFH]
(LDU)  Number of loadable Device Units [0-0FFFFH]
(GC)   Global Clock [ATRT/546/CSM/None]
(GCN)  Global Clock Name [1-13 Chars]
```

- (ADP) Specifies in decimal the priority of the attach device task.
- (TF) Specifies whether the I/O system includes timing facilities. See also CUT.
- (TTP) Specifies in decimal the timer task priority if you included timing facilities.
- (CON) Specifies in decimal the priority of the BIOS task that deletes file and device connections.
- (ACE) Specifies whether the OS returns an error if you attempt duplicate file creation.
- (SMI) Specifies whether user ID 0 has system manager attributes.
- (CUT) Specifies in decimal units representing 10 millisecond Nucleus clock ticks how long the BIOS waits before updating all designated devices with any data buffered in memory, regardless of the activity on an individual device-unit.
- (TSC) Specifies whether to include the TSC; required by the HI and any terminal driver.
- (CST) Specifies if the translation capabilities are supported by the TSC.
- (OSC) Specifies whether to include OSC controls for terminals.
- (TSP) Specifies tape support. Use the default.
- (PMI) , (PMA)
Specifies in 16-byte paragraphs the minimum and maximum sizes of the BIOS memory pool.
- (LDV) Specifies in decimal how many loadable devices you want.
- (LDU) Specifies in decimal how many loadable device units you want. LDU must exceed LDV.
- (GC) Specifies whether there is an on-board, hardware time-of-day clock that maintains the current date and time for the entire system, even when it is turned off.
- (GCN) Specifies the global clock name.

Comments for Build File Screen

This optional screen enables you to add comments, such as the ones illustrated below, to the build file and embed them in the boot image. You can display these comments using the **sysinfo** command. These comments are not recognized or used by the ICU.

```
(COMNT)      Comments for Build file

      each line = 1-55 characters - IN QUOTES

[ 1] = 'PC-hosted iRMX Operating System           '
[ 2] = '                                           '
[ 3] = 'Company Name- Intel Demonstration Software w/ TIMEOUT '
[ 4] = 'Software Serial Number-                 '
[ 5] = '                                           '

```

[1] - [n]

Specifies build file comments such as the name of the build file, important variable locations, and the release supported. Each comment must fit on a single line.

Device Driver Screens

Intel Device Drivers Screen

This screen displays the Intel-supplied drivers for Multibus systems.

```
(IDEVS)          Intel Device Drivers

(T74) 8274 Driver  [Yes/No]      (T51) 8251A Driver  [Yes/No]
(T30) 82530 Driver [Yes/No]      (RAM) RAM Disk     [Yes/No]
(L50) iSBX 350 LP  [Yes/No]      (PCI) PCI Driver   [Yes/No]
(G79) iSBX 279A   [Yes/No]

-- MULTIBUS I DEVICE DRIVERS:
(S14) MSC Driver  [Yes/No]      (TCC) TCC Driver   [Yes/No]
(L86) iSBC n86/12 LP [Yes/No]    (S20) iSBC 220     [Yes/No]
(S08) iSBC 208    [Yes/No]      (T34) iSBC 534     [Yes/No]
(T44) iSBC 544A   [Yes/No]

-- MULTIBUS II DEVICE DRIVERS:
(S24) iSBC 186/224A [Yes/No]    (S10) ATCS Driver  [Yes/No]
(ENC) COM1 and COM2 Serial Port Driver [Yes/No]
```

PC Bus Device Drivers Screen

This screen displays the Intel-supplied drivers for PCs.

```
(PCDEV)          PC Bus Device Drivers

(S1S) PC Bus Serial [Yes/No]    (S1C) PC Bus Console [Yes/No]
(S1P) PC Bus Printer [Yes/No]   (S1F) PC Bus Floppy  [Yes/No]
(S1W) PC Bus Wini   [Yes/No]    (S1A) PC Bus SCSI    [Yes/No]
(S1H) H 550 Serial [Yes/No]    (DHD) DOS Wini       [Yes/No]
(DFD) Dos Floppy   [Yes/No]    (DCN) Dos Console    [Yes/No]
```

For each driver you specify, two or three additional screens appear: the Driver screen, the Unit Information screen, and/or the Device-Unit Information screen.

The ICU stores all the device driver screens by type, not by device. This means that all Device Driver screens, all Unit Information screens and all Device-Unit Information screens are stored together. The ICU associates Unit Information and Device-Unit Information screens to a particular device driver by the device name you enter in the DEV parameter on the particular Device Driver screen.

Parameters on the screens vary, depending on the device type. Only the common parameters are illustrated on the composite screens. For information on individual driver screens and driver options, use the ICU help screens.

Driver Screen Composite

A screen like this appears for every driver you specify on the IDEVS or PCDEV screen.

```
(xxx)          xxx Driver

(DEV) Device Name [1-16 Chars]
(IL)  Interrupt Level [Encoded Level]
(ITP) Interrupt Task Priority [0-255]
```

- (DEV) Specifies a name for a particular device and associates that device with all the units and DUIBs that belong to it. This parameter appears on every device driver screen.
- (IL) Specifies the encoded interrupt level for the driver. The interrupt task uses this value to associate the interrupt task with the correct interrupt level. This parameter appears on most device driver screens.
- (ITP) Specifies in decimal the initial priority of the device's interrupt task. The actual priority of the interrupt task changes because the Nucleus adjusts an interrupt task's priority according to the interrupt level that it services. This parameter appears on many device driver screens.

Unit Information Screen Composite

This screen appears for most device drivers. The ICU uses this information to create a unit information table for the driver.

```
(xxx)          xxx Driver Unit Information

(DEV) Device Name [1-16 Chars]
(NAM) Unit Info Name [1-16 Chars]
```

- (DEV) Specifies the logical ICU connection between a driver and all of its units. Enter the name you entered in the DEV parameter on the Driver screen. This parameter appears on every Unit Information screen.
- (NAM) Specifies a unique name for this unit information table. This parameter appears on every Unit Information screen.

Device-Unit Information Screen Composite

This screen appears for most device drivers. The ICU uses this information to create a DUIB for the driver.

(xxx)	xxx Driver Device-Unit Information
(DEV)	Device Name [1-16 Chars]
(NAM)	Device-Unit Name [1-14 Chars]
(UN)	Unit Number on this Device [0-0FFH]
(UIN)	Unit Info Name [1-16 Chars]

- (DEV) Specifies the logical ICU connection between a driver and all of its DUIBs, enabling you to delete all the related units and DUIBs at the same time. Enter the name you entered in the DEV parameter on the Driver screen. This parameter appears on every Device-Unit screen.
- (NAM) Specifies a name that uniquely identifies the device-unit for the I/O System. Use this physical name when you invoke the **attachdevice** command, the **a_physical_attach_device** system call, or the **logical_attach_device** system call. This parameter appears on every Device-Unit screen.
- (UN) Specifies the unit number of this device-unit, beginning with 0. This parameter appears on many Device-Unit screens.
- (UIN) Specifies the name of the unit information table for this DUIB. This parameter appears on many Device-Unit screens.

See also: *Driver Programming Concepts*, particularly the DUIB and UDS chapters, for more information on configuring device drivers

User Devices Screen

This screen enables you to add your own devices to the system. It appears if you answer Yes to the Query screen.

(USERD)	User Devices	
(OPN)	Random Access Object Code Path Name [1-45 Chars/NONE]	
(TOP)	Terminal Object Code Path Name [1-45 Chars/NONE]	
(DPN)	Duib Source Code Path Name [1-45 Chars/NONE]	
(DUP)	Random Access Device and Unit Source Code Path Name [1-45 Chars/NONE]	
(TUP)	Terminal Device and Unit Source Code Path Name [1-45 Chars/NONE]	
(ND)	Number of User Defined Devices [0-0FFH]	
(NDU)	Number of User Defined Device-Units [0-0FFH]	
	Terminal Device and Unit Information Names [1-16 Chars]	
(N01)	(N02)	(N03)
(N04)	(N05)	(N06)
(N07)	(N08)	(N09)
(N10)	(N11)	(N12)
(N13)	(N14)	(N15)
(N16)	(N17)	(N18)

- (OPN) Specifies the object code pathname for your custom random access device drivers.
- (TOP) Specifies the object code pathname for your custom terminal device drivers.
- (DPN) Specifies the include file pathname for your custom DUIBS.
- (DUP) Specifies the include file pathname for your device or unit information tables for custom random access device drivers.
- (TUP) Specifies the include file pathname for your device or unit information tables for custom Terminal device drivers.
- (ND) Specifies how many different device numbers you used in the DUIBs in the DUIB include file.
- (NDU) Specifies how many different device-unit numbers you used in the DUIBs in the DUIB include file.

Terminal Device and Unit Information Names
Specifies the terminal device-unit name.

UDS Device Driver Modules Screen

This screen enables you to specify object code pathnames for your custom devices.

```
(UDDM)      UDS Device Driver Modules

Module= Driver type      , Object code pathname
              [T/C]      , [1-55 Chars]

[ 1] Module=
```

Driver type

Specifies whether the device is a Terminal or Common/Named.

Object code pathname

Specifies the object code pathname for this device.

DOS Extender Dispatcher Screens

```
(DISPJ)      DOS Extender Dispatcher Job

(ODS) Object Directory Size [0-3840]
(PMI) Pool Minimum [60H-0FFFFFFFH]
(PMA) Pool Maximum [60H-0FFFFFFFH]
(MOB) Maximum Objects [1-0FFFFFH]
(MTK) Maximum Tasks [1-0FFFFFH]
(MPR) Maximum Priority [0-255]
(PV) Parameter Validation [Yes/No]
(TP) Task Priority [0-255]
(SS) Stack Size [0-0FFFFFH]
(VIE) Enable interrupt virtualization [Yes/No]
```

(ODS) Specifies in decimal how many entries can be in this job's object directory. 0 means create no directory.

(PMI) , (PMA)

Specifies in 16-byte paragraphs the minimum and maximum sizes of this job's memory pool.

(MOB) Specifies how many objects can exist simultaneously in this job.

(MTK) Specifies how many tasks can exist simultaneously in this job.

(MPR) Specifies in decimal the highest priority of tasks in this job.

- (PV) Specifies whether to include parameter validation for Nucleus system calls made by tasks in this job, if you included system-level parameter validation.
- (TP) Specifies in decimal the priority of this job's initialization task.
- (SSI) Specifies in bytes the size of the initialization task's stack segment.
- (VIE) Specifies whether to enable Interrupt Virtualization.

DOS Extender Reserved Interrupts Screen

This screen appears only if you enable VIE on the DISPJ screen.

(DRINT) DOS Extender Reserved Interrupts			
(ML0)	Master Level 0	[Yes/No]	(SL0) Slave Level 0 [Yes/No]
(ML1)	Master Level 1	[Yes/No]	(SL1) Slave Level 1 [Yes/No]
(ML2)	Master Level 2	[Yes/No]	(SL2) Slave Level 2 [Yes/No]
(ML3)	Master Level 3	[Yes/No]	(SL3) Slave Level 3 [Yes/No]
(ML4)	Master Level 4	[Yes/No]	(SL4) Slave Level 4 [Yes/No]
(ML5)	Master Level 5	[Yes/No]	(SL5) Slave Level 5 [Yes/No]
(ML6)	Master Level 6	[Yes/No]	(SL6) Slave Level 6 [Yes/No]
(ML7)	Master Level 7	[Yes/No]	(SL7) Slave Level 7 [Yes/No]

(ML0) – (ML7)

ML0 is the system clock; ML2 is the slave PIC. Others specify interrupt levels used by devices.

(SL0) – (SL7)

Specify interrupt levels used by devices.

EIOS Screens

EIOS Screen

```
(EIOS)          EIOS

(ABR)  Automatic Boot Device Recognition [Yes/No]
(DDS)  Default IO Job Directory Size [5-3840]
(ITP)  Internal EIOS Tasks' Priorities [0-255]
(PMI)  EIOS Pool Minimum [0-0FFFFFFFH]
(PMA)  EIOS Pool Maximum [0-0FFFFFFFH]
(CD)   Configuration Directory [1-45 Chars]

(RPA)  Retries on Physical Attachdevice [0-0FFFFH]
```

- (ABR) Specifies whether to include ABDR, which assigns the bootstrap device as your system device.
- (DDS) Specifies in decimal how many object directory entries can exist for the EIOS job and all I/O jobs.
- (ITP) Specifies in decimal the priority of EIOS internal housekeeping tasks.
- (PMI) , (PMA)
Specifies in 16-byte paragraphs the minimum and maximum sizes of the EIOS memory pool.
- (CD) Specifies the pathname for the directory that contains the non-resident user configuration information used to configure and verify terminals and non-resident users.
- (RPA) Specifies how many times the EIOS should automatically retry to perform a physical attachdevice when an error occurs.

Automatic Boot Device Recognition Screen

This screen appears if you respond Yes to the ABR parameter on the EIOS screen.

```
(ABDR)          Automatic Boot Device Recognition

(DLN)  Default System Device Logical Name [1-12 Chars]
(DPN)  Default System Device Physical Name [1-12 Chars]
(DFD)  Default Sys Dev File Driver [P/S/N/R/E/D]
(DO)   Default System Device Owner's ID [0-0FFFFH]
```

- (DLN) Specifies the logical name of the system device.
- (DPN) Specifies the physical device unit name; do not use a name you specified in the LOGN screen.
- (DFD) Specifies the file driver type for the system device.
- (DO) Specifies the owner ID of the system device; the default is the system manager.

Logical Names Screen

This is a repetitive screen. Specify up to 20 logical names, including the two defaults.

```

(LOGN)          Logical Names

Logical Name = logical_name, device_name, file_driver, owners-id
                [1-12 Chars], [1-14 Chars], [P/S/N/R/E/D], [0-0FFFFH]
[1] Logical Name = BB          ,    BB          ,    PHYSICAL ,    0H
[2] Logical Name = STREAM     ,    STREAM     ,    STREAM   ,    0H
[3] Logical Name =             ,             ,             ,
[4]

```

`Logical_name`

Specifies the logical name of the device.

`device_name`

Specifies a name that uniquely identifies the device-unit for the I/O System. Use this name when you invoke commands or system calls requiring a device name.

`file_driver`

Specifies the file driver type for the device.

`owners-id`

Specifies the owner ID of the device; this does not need to be defined on the IOUS screen.

I/O Users Screen

This is a repetitive screen. Specify up to 20 user names.

```
(IOUS)          I/O Users

      I/O User = user name  , Owner-ID[ ,ID,ID,ID,ID]
                  [1-12 Chars], [0-0FFFFH]
[1] I/O User = WORLD      , 0FFFFH, 0FFFFH, 0FFFFH, 0FFFFH, 0FFFFH
[2] I/O User =
```

user name

Specifies the name of the user object.

Owner-ID

The first ID is the Owner ID; you can specify up to 4 additional User IDs.

I/O Jobs Screen

This is a repetitive screen. Specify I/O jobs in the order you want them created and initialized.

```
(IOJOB)          I/O Jobs

(IJD) I/O Job Default Prefix [1-12 Chars]
(DU)  Default User [1-12 Chars]
(PMI) Pool Minimum [20H-0FFFFFFFH]
(PMA) Pool Maximum [20H-0FFFFFFFH]
(EHS) Exception Handler Entry Point [1-31 Chars]

(EM)  Exception Mode [Never/Prog/Environ/All]
(PV)  Parameter Validation [Yes/No]
(TP)  Task Priority [0-255]
(TSA) Task Entry Point [1-31 Chars]

(VAR) Public Variable Name [1-31 Chars]

(SSA) Stack Segment Address [SS:SP]
(SSS) Stack Size [0-0FFFFH]
(NPX) Numeric Processor Extension Used [Yes/No]
```

- (IJD) Specifies the logical name for this I/O job.
- (DU) Specifies the default user object for this I/O job.
- (PMI) , (PMA)
Specifies in 16-byte paragraphs the minimum and maximum sizes of this I/O job's memory pool.
- (EHS) Specifies the exception handler for this I/O Job.
- (EM) Specifies the exception mode of the exception handler listed in EHS.
- (PV) Specifies whether the Nucleus performs parameter validation for all Nucleus system calls made in this I/O job.
- (TP) Specifies in decimal the priority of this job's initialization task.
- (TSA) Specifies the PUBLIC name of the I/O job's entry procedure.
- (VAR) Specifies the PUBLIC name of any of the I/O job's public variables.
- (SSA) Specifies the address of the initialization task's stack.
- (SSI) Specifies in bytes the size of the initialization task's stack segment.
- (NPX) Specifies whether the I/O job's initial task contains floating-point instructions.

File Drivers Screens

BIOS File Drivers Screen

```
(BIOFD)          BIOS File Drivers

(NFD)  Named File Driver [Yes/No]
(DFD)  DOS File Driver [Yes/No]
(EFD)  EDOS File Driver [Yes/No]
(MFD)  Max. Number of Loadable File Drivers [0-0FFH]
```

(NFD), (DFD), (EFD)
Specifies which file drivers to include in the system.

(MFD) Specifies how many loadable drivers to include in the system.

File Driver Screen Composites

The file driver screens are: Physical (PHYFD), Stream (STRFD), DOS (DOSFD), Named (NAMFD), and EDOS (EDSFD). This EDOS File Driver screen represents all file driver screens. The DOS and EDOS screens have an LCN parameter and the Physical and Named file driver screens have an EBS parameter.

```
(EDSFD)          EDOS File Driver

(COS)  Connection Object Size [0-255]
(ISS)  I/O Task Stack Size [0-65534]
(ITP)  I/O Task Priority [0-255]
(DDS)  Device Descriptor Size [0-255]
(DRQ)  DUIBs Required [Yes/No]
(LCN)  Force DOS Filenames to Lower Case [Yes/No]
```

(COS) Specifies the size in bytes of the file driver's connection object.

(ISS) Specifies in bytes the size of the I/O task's stack.

(ITP) Specifies in decimal the priority of the I/O task.

(DDS) Specifies the size of the device descriptor for the file driver.

(DRQ) Specifies whether the file driver requires DUIBs.

(LCN) (DOS and EDOS file drivers only) Specifies whether to force DOS filenames to lower case.

(EBS) (Physical and Named file drivers only) Specifies the size in bytes of the EIOS buffer used to perform I/O operations.

Generate File Names Screen

```
(GEN)    Generate File Names

(RMB)    Remote Boot Translation [Yes/No]

(RBF)    Remote Boot File Name [1-55 Chars]

(ROF)    ROM Code File Name [1-55 Chars]

(RAF)    RAM Code File Name [1-55 Chars]
```

- (RMB) Specifies whether to boot the system remotely.
- (RBF) Specifies the pathname of the application system's REMOTE bootloadable file.
- (ROF) Specifies the pathname of your system's ROM code.
- (RAF) Specifies the pathname of the system's local bootloadable file.

Hardware Screens

Hardware Screen

(HARD)	Hardware
(BUS)	System Bus Type [1=MBI/2=MBII/3=AT]
(TP)	8254 Timer Port [0-0FFFFH]
(CIL)	Clock Interrupt Level [0-7]
(CN)	Timer Counter Number [0,1,2]
(CIN)	Clock Interval [0-65535 msec]
(KTR)	Kernel Tick Ratio [1-65535]
(CF)	Clock Frequency [0-65535 khz]
(TPS)	Timer Port Separation [0-0FFH]
(EMU)	Emulate Numeric Processor [Yes/No]
(IF)	Initialize On-board Functions [0=No/1-0FFH]
(BIP)	Board Initialization Procedure [1-45 Chars]

- (BUS) Specifies the system bus type.
- (TP) Specifies the base (lowest valued) port address of the 8254 PIT timer, which provides OS timing.
- (CIL) Specifies the master PIC interrupt line (not an encoded level) that the timer is connected to.
- (CN) Specifies the PIT counter used by the Nucleus, usually 0; use the default for Intel boards.
- (CIN) Specifies in milliseconds the standard Nucleus clock tick; for portability, use the default.
- (KTR) Specifies the tick interval in iRMK system calls; the Nucleus tick interval remains unchanged. Use the default Nucleus clock tick unless you need a smaller iRMK Kernel tick interval.
- (CF) Specifies in kilohertz the frequency of the clock input to the timer; use the default for Intel boards.
- (TPS) Specifies the 8254 PIT timer port separation; use the default for Intel boards.
- (EMU) Specifies whether the Numeric Emulator software is to be included in the system configuration.
- (IF) Specifies for which board in the system the Nucleus is initialized.
- (BIP) Specifies the custom initialization routine if you responded No to the IF parameter.

Multibus II Hardware Screen

This screen appears only if you specify 2 for the BUS parameter of the HARD screen. If you are using MB II message passing, and require >16Mbyte support, set the DIB and DOB parameters as described.

(MBII)	Multibus II Hardware
(MDP)	Message Device Base Port Address [0-0FFFFH]
(MDS)	Message Device Port Separation [0-0FFH]
(MDL)	Message Interrupt Level [Encoded Level]
(MCO)	Message Device Duty Cycle for One Cycle DMA [0-0FFH]
(MCT)	Message Device Duty Cycle for Two Cycle DMA [0-0FFH]
(MDC)	Message Device Duty Cycle for Burst DMA [0-0FFH]
(DDP)	Message Device ADMA Data Port [0-0FFFFH]
(GBR)	ADMA Burst Register [0-0FFFFH]
(GDR)	ADMA Delay Register [0-0FFFFH]
(AIB)	ADMA Base Port Address [0-0FFFFH]
(ACI)	ADMA Channel for Input [0-0FFFFH]
(ACO)	ADMA Channel for Output [0-0FFFFH]
(DIB)	DMA Input Buffer Size [0-0FFFFFFFFH]
(DOB)	DMA Output Buffer Size [0-0FFFFFFFFH]
(DDA)	DAG Device Used [Yes/No]
(WDP)	Watchdog Present [Yes/No]
(WDI)	Watchdog Transmission Interval [1-0FFFFFFFFH]
(WDT)	Watchdog Timeout [1-0FFFFFFFFH]

- (MDP) Specifies the MPC's lowest I/O port address.
- (MDS) Specifies the separation between port addresses of the MPC's registers.
- (MDL) Specifies the encoded interrupt level that the MPC is attached to on the master PIC.
- (MCO) Specifies the MPC duty cycle for one-cycle DMA on SBC 486/150 boards.
- (MCT) Specifies the MPC duty cycle for two-cycle DMA on all Intel boards with MPC support.
- (MDC) Specifies the MPC duty cycle for burst DMA on all Intel boards with DAG and MPC support.
- (DDP) Specifies the DMA data port address; use the default for MB II boards.
- (GBR) Specifies how many contiguous bus cycles the ADMA can request.
- (GDR) Specifies the minimum number of Nucleus clock ticks between ADMA accesses.
- (AIB) Specifies the I/O base address of the message passing DMA device.
- (ACI) Specifies the ADMA channel used for input.

- (ACO) Specifies the ADMA channel used for message-passing output.
- (DIB) Specifies the size in bytes of the buffer that holds message-passing input transfers to unaligned user data buffers; set to the size of the largest message you expect.
- (DOB) Specifies the size in bytes of the buffer that holds message-passing output transfers from/to unaligned user data buffers; set to the size of the largest message you expect.
- (DDA) Specifies whether you are using the DAG.
- (WDP) Specifies whether you are using the Watchdog Timer.
- (WDI) Specifies the transmission interval (in Kernel ticks) between broadcasting of an existence message by this board.
- (WDT) Specifies the timeout value (in Kernel ticks) this board waits between existence messages from other boards.

Human Interface Screens

If you include the HI, the ICU automatically includes all HI, AL, EIOS, BIOS, and Nucleus system calls.

Human Interface Screen

```
(HI)                Human Interface

(ICL)  Initial Command Line Size [0-65535]
(CNM)  Command Name Length [1-255]
(SYS)  System Directory [1-45 Chars]

(RIP)  Resident Initial Program [IntelCLI/1-45 Chars]

(UXC)  User Extension for Intel CLI [1-45 Chars]

(SS)   Initial Program Stack Size [0-0FFFFH]
(PMI)  Human Interface Pool Minimum [0-0FFFFFFFH]
(PMA)  Human Interface Pool Maximum [0-0FFFFFFFH]
(JST)  Loadable Job Sync.Timeout [0-065535]
(DTN)  Default Terminal Name [1-6 Chars]
(RU)   Resident User [Yes/Recovery/None]
(SCF)  System Command File Name [1-14 Chars]
```

- (ICL) Specifies in bytes the initial size of the command line buffer.
- (CNM) Specifies how many characters can exist in the longest command pathname that the HI CLI handles.
- (SYS) Specifies the pathname of the system directory on the system device. See also *:sd:*, ABDR screen.
- (RIP) Specifies the resident initial program; the default is the HI CLI.
- (UXC) Specifies the pathname of the user extension module to bind with the HI CLI.
- (SS) Specifies in bytes the size of the initial program's stack; use the default for the HI CLI, adding the user extension requirements, if any.
- (PMI) , (PMA) Specifies in 16-byte paragraphs the initial, minimum and maximum sizes of the HI memory pool.

- (JST) Specifies the maximum number of Nucleus clock ticks that the HI waits for a loadable job to initialize.
- (DTN) Provides a default terminal name for the entire system if there is no terminal entry in the `:config:terminals` file.
- (RU) Defines the resident user. See also RIP.
- (SCF) Specifies the name of the system command file.

Human Interface Jobs Screen

```

(HIJOB)           HI Jobs

(MIN)  Jobs Minimum Memory [0-0FFFFFFFH]
(MAX)  Jobs Maximum Memory [0-0FFFFFFFH]
(NPX)  Numeric Processor Extension Used [Yes/No]
```

- (MIN) , (MAX) Specifies the minimum and maximum memory pool size for HI-created I/O jobs.
- (NPX) Specifies whether the HI jobs contain floating-point instructions and use a math coprocessor or NPX.

Resident/Recovery User Screen

(RES)	Resident/Recovery User
(TN)	Terminal Name [1-6 Chars]
(TDN)	Terminal Device Name [1-14 Chars]
(MTP)	Maximum Task Priority [0-255]
(UID)	User ID Number [0-0FFFFH]
(MIN)	Minimum Memory Required [0-0FFFFFFFH]
(MAX)	Maximum Memory Required [0-0FFFFFFFH]
(IPP)	Initial-Program Pathname [RESIDENT/1-45 Chars]
(DEF)	Default Directory [1-45 Chars]

- (TN) Specifies the name of the Resident/Recovery user terminal.
See also: *:config:terminals* file, terminal names, *System Configuration and Administration*
- (TDN) Specifies the physical device name of the resident user's terminal.
- (MTP) Specifies the highest priority for resident user tasks.
- (UID) Specifies the user ID of the Resident/Recovery user so a user object can be created.
See also SMI parameter, BIOS screen.
See also: Users, user IDs, World, *System Concepts*
- (MIN), (MAX) Specifies the amount of memory for the I/O job of the resident user's terminal.
- (IPP) Specifies the initial program for the resident user.
- (DEF) Specifies the pathname of the default directory for the Resident/Recovery user, which will be assigned the logical name *:home:*.

Prefixes Screen

The Prefixes screen is similar to the DOS **path** command; it enables you to specify which directories are searched for HI commands. You can delete the defaults or move them below your own directories.

```
(PREF)          Prefixes

      Prefix = 1-45 Chars
[ 1] Prefix = :PROG:
[ 2] Prefix = :UTILS:
[ 3] Prefix = :UTIL286:
[ 4] Prefix = :SYSTEM:
[ 5] Prefix = :LANG:
[ 6] Prefix = :ICU:
[ 7] Prefix = :$:
[ 8] Prefix =
```

Prefix

Specifies up to 20 prefixes. Define prefixes you list here on the HILOG screen or in the SYS parameter on the HI screen.

HI Logical Names Screen

:config:, *:prog:*, and *:system:* are automatically assigned as logical names. Do not assign them again using this screen.

```
(HILOG)          HI Logical Names

      logical_name , path_name
      [1-12 Chars] , [1-45 Chars]
[ 1] Name = WORK           , :SD:WORK
[ 2] Name = UTILS          , :SD:UTIL386
[ 3] Name = UTIL286        , :SD:UTIL286
[ 4] Name = LANG           , :SD:LANG286
[ 5] Name = RMX            , :SD:RMX386
[ 6] Name = ICU            , :SD:RMX386/ICU
[ 6] Name = INCLUDE        , :SD:INTEL/INC
[ 7] Name =
```

logical_name

Specifies the logical name for the path_name parameter.

path_name

Specifies the pathname.

Includes and Libraries Screen

This screen enables you to specify the directory pathnames for libraries, INCLUDE files and tools that the ICU references while generating a submit file that assembles, binds, and builds your application system. The names you enter on this screen must be logical names surrounded by colons or pathnames ending in a slash.

(INCL)	Includes and Libraries [1-30 Chars]
(UDF)	UDI Includes and Libs
(HIF)	Human Interface Includes and Libs
(EIF)	Extended I/O System Includes and Libs
(ALF)	Application Loader Includes and Libs
(BIF)	Basic I/O System Includes and Libs
(MNF)	Intel Monitor Includes and Libs
(SDF)	System Debugger Includes and Libs
(NUF)	Nucleus Includes and Libs
(ILF)	Interface Libraries
(DTF)	Development Tools Path Name
(VMF)	Virtual 8086 Mode includes and libs
(NET)	iRMX-NET Files
(CLF)	Shared C Libraries
(ISL)	Intel Support Libraries
(SJM)	System Jobs Object Modules

- (UDF) Specifies the pathname of the directory that contains the UDI files.
- (HIF) Specifies the pathname of the directory that contains the HI files.
- (EIF) Specifies the pathname of the directory that contains the EIOS files.
- (ALF) Specifies the pathname of the directory that contains the AL files.
- (BIF) Specifies the pathname of the directory that contains the BIOS files.
- (MNF) Specifies the pathname of the directory that contains the Intel Monitor files.
- (SDF) Specifies the pathname of the directory that contains the System Debugger files.
- (NUF) Specifies the pathname of the directory that contains the Nucleus job files.
- (ILF) Specifies the pathname of the directory that contains the Interface library files.
- (DTF) Specifies the pathname of the directory that contains the Development Tools.
- (VMF) Specifies the pathname of the directory that contains the Virtual Mode includes and libs .
- (NET) Specifies the pathname of the directory that contains the iRMX-NET files.
- (CLF) Specifies the pathname of the directory that contains the C library files.
- (ISL) Specifies the pathname of the directory that contains the floating point support libraries.
- (SJM) Specifies the pathname of the directory that contains the object modules for the system jobs specified in the System Jobs screen (SYSJ).

Interrupts Screens

Interrupts Screen

```
(INT)           Interrupts

(MP)   8259A Master Port [0-0FFFFH]
(MPS)  Master PIC Port Separation [0-0FFH]
(IS)   Interrupt Slaves [Yes/No]
```

- (MP) Specifies the base (lowest valued) port address of the 8259A Master PIC; use the default for Intel boards.
- (MPS) Specifies the interval between each 8259A PIC port; use the default for Intel boards.
- (IS) Specifies whether there are slave 8259A PICs connected to the master 8259A PIC.

Slave Interrupt Levels Screen

This screen appears only if you responded Yes to the IS parameter.

```
(SLAVE)           Slave Interrupt Levels

Slave = Slave_number, Level_Sensitive, Port,      Separation
          [0-7],           [Yes/No] [0-0FFFFH]   [0-0FFH]
[1] Slave =
```

Slave_number

Specifies the master interrupt the slave is tied to.

Level_Sensitive

Specifies whether the 8259A is in level-sensitive or edge-sensitive mode.

Port

Specifies the base (lowest valued) port address of the 8259A.

Separation

Specifies the separation between the 8259A's control and mask port.

Memory Screens

Once you have a final system, you can minimize the system memory by changing the values on these screens.

See also: Generating and testing the new system in this manual,
 Adjusting system memory, *Programming Techniques and AEDIT Text Editor*

Set the lower limit to at least this so the system will boot properly:

Value	For System Type
6000H or greater	MB I (to avoid 3rd Stage BSL)
B8000H to C0000H	If your boot image has grown, keep this area free for the MB I 2nd Stage BSL stack (to avoid 1st/3rd stage stacks)
10000H or greater	MB II (to avoid 2nd Stage MSA)
0F000H or greater	PC Bus Platforms (to avoid the PC 3rd stage)

In addition, do not overlap System or Free Space memory with memory for these entities. In general, allow the minimum memory you think you will need, then tune the parameters using output from the builder *.mp2* file.

Memory For	Address
MSC Device Driver Wakeup Addresses and Wakeup Blocks	1000H to 1200H
ROM-based systems	Address specified in RIA
RAM driver	Address specified in BMA
I/O Controller	Address specified in IPA, MA, BBA, etc. Start MEMS Start MEMF

When calculating the system memory requirements, also consider the memory required by the subsystems and your embedded applications.

See also SUB screen description for memory required for subsystems

Memory for System Screen

```
(MEMS)           Memory for System

                SYS      = low [0-0FFFFFFFFH], high [0-0FFFFFFFFH]
[ 1 ] SYS       =
```

low, high

Specify the start and end addresses of each contiguous memory block (32-bit physical memory location) to reserve for code and data segments.

Memory for Free Space Manager Screen

```
(MEMF)           Memory for Free Space Manager

                FSM      = low [0-0FFFFFFFFH], high [0-0FFFFFFFFH]
[ 1 ] FSM       =
```

low, high

Specify the start and end addresses of each contiguous block (32-bit physical memory location) of memory to reserve for dynamic allocation.

Do not overlap memory you reserved in the MEMS screen.

Paging Identity Mapped Memory Screen

```
(PIMM)           Paging Identity Mapped Memory

                IMM      = low [0-0FFFFFFFFH], high [0-0FFFFFFFFH]
[ 1 ] IMM       =
```

low, high

Specify the start and end addresses of each contiguous block (32-bit physical memory location) of memory you want identified with the Paging Subsystem.

Do not enter any memory areas that have been previously reserved for use by the FSM or the system.

Network Screens

Network Subsystem Screen

Use this screen to configure the parameters for the Networking Subsystem.

```
(NET)          Network Subsystem

(MIP) iNA 960 MIP Driver Job [Yes/No]
(CMP) iNA 960 COMMputer Job [Yes/No]
(RCJ) iRMXNET Client Job [Yes/No]
(RSJ) iRMXNET Server Job [Yes/No]
(TCP) iRMX TCP/IP Job [Yes/No]
(XSJ) X25 Server Job [Yes/No]
(XCJ) X25 Client Job [Yes/No]
```

- (MIP) Specifies whether to configure the iNA 960 MIP Driver Job into the OS.
- (CMP) Specifies whether to configure the iNA 960 COMMputer Job into the OS.
- (RCJ) Specifies whether to configure the iRMX-NET client job into the OS.
- (RSJ) Specifies whether to configure the iRMX-NET server job into the OS.
- (TCP) Specifies whether to configure the TCP/IP job into the OS.
- (XSJ) Specifies whether to configure the X.25 server job into the OS.
- (XCJ) Specifies whether to configure the X.25 client job into the OS.

iNA MIP Driver Job Screen

This screen appears if you specify MIP on the NET screen.

```
(IMIPJ)          iNA MIP Driver Job

(ODS)  Object Directory Size [0-3840]
(PMI)  Pool Minimum [60H-0FFFFFFFH]
(PMA)  Pool Maximum [60H-0FFFFFFFH]
(MOB)  Maximum Objects [1-0FFFFFFH]
(MTK)  Maximum Tasks [1-0FFFFFFH]
(MPR)  Maximum Priority [0-255]
(PV)   Parameter Validation [Yes/No]
(TP)   Task Priority [0-255]
(SSJ)  Stack Size [0-0FFFFFFH]
(NPX)  Numeric Processor Extension Used [Yes/No]
(MJN)  Name of MIP Job to be generated [1-45 Chars]

(CLJ)  Create Loadable Job [Yes/No]
(NL)   iNA Network Layer [0/1/2/3]
(CET)  COMMengine Type [1=MBI/2=MBII/3=AT]
```

- (ODS) Specifies in decimal the maximum number of entries in this job's object directory; 0 means create no directory.
- (PMI), (PMA) Specifies in 16-byte paragraphs the minimum and maximum sizes of this job's memory pool.
- (MOB) Specifies how many objects can exist simultaneously in the job; 0FFFFFFH means no limit to the number of created objects.
- (MTK) Specifies how many tasks can exist simultaneously in this job.
- (MPR) Specifies in decimal the highest priority of tasks in this job.
- (PV) Specifies whether to include parameter validation for Nucleus system calls made by tasks in this job, if you included system-level parameter validation.
- (TP) Specifies in decimal the priority of this job's initialization task.
- (SSJ) Specifies in bytes the size of the initialization task's stack segment, according to your segmentation model.
- (NPX) Specifies if the job's initial task contains floating-point instructions.
- (MJN) Specifies the name of the generated MIP job.

- (CLJ) Specifies whether to generate a separate job that can be loaded by **sysload** or bind the job with the boot image.
- (NL) Specifies the type of network layer configured in the iNA Transport Software.
- (CET) Specifies the type of COMMengine.

MIP Configuration for Multibus I Screen

Use this screen to configure the MIP job in a COMMengine hardware environment on a Multibus I system.

```

(MIP1)          MIP Configuration for MBI

(CBN)  Communication Board Name [1-20 Chars]
(NEM)  Number of External Mailboxes [1-0FFFFH]
(RQB)  Request Queue Addr Base [0-0FFFFH]
(RQO)  Request Queue Addr Offset [0-0FFFFH]
(WP)   Wakeup Port [0-0FFFFH]
(IL)   Interrupt Level [0-0FFH]
(LD)   Load [L/R/P/N]
(DN)   Device Name [1-14 Chars]
(FN)   File Name [1-30 Chars]
(CC)   Class Code [0-0FFFFH]
(HBA)  Host Buffer Address [0-0FFFFFFH]
(BAB)  Boot Address Base [0-0FFFFH]
(BAO)  Boot Address Offset [0-0FFFFH]
(CSB)  Comm Start Address Base [0-0FFFFH]
(CSO)  Comm Start Address Offset [0-0FFFFH]
(CD)   COMMengine Delay [0-0FFFFH]
```

- (CBN) Specifies the name of the NIC used as a communication board by the MIP job.
- (NEM) Specifies how many unique response mailboxes applications can have in request blocks at any given time.
- (RQB) Specifies the address base for the start of communication request buffers.
- (RQO) Specifies the address offset for the start of communication request buffers.
- (WP) Specifies the I/O port that the COMMengine recognizes.
- (IL) Specifies which one of the eight MB encoded interrupt levels that the COMMengine board uses to signal messages to the iRMX system.
- (LD) Specifies how the COMMengine is loaded with the iNA Transport Software.
- (DN) Specifies the device name where the iNA software resides.

- (FN) Specifies the pathname of the load file.
- (CC) Specifies the class code that the COMMengine uses when requesting a boot service.
- (HBA) Specifies the absolute address of the 180DH byte host buffer where the COMM software will be located for the firmware to load.
- (BAB) Specifies the base portion of the address of one of eight addresses from which the COMMengine gets its boot commands.
- (BAO) Specifies the offset portion of the address of one of eight addresses from which the COMMengine gets its boot commands.
- (CSB) Specifies the base portion of the start address of the communication software on the COMMengine, if iNA is in PROM.
- (CSO) Specifies the offset portion of the start address of the communication software on the COMMengine, if iNA is in PROM.
- (CD) Specifies how many Nucleus clock ticks the software executing on the host processor waits so that the iNA Transport Software running on the communications board can complete initialization.

MIP Configuration for Multibus II Screen

Use this screen to configure the MIP job in a COMMengine hardware environment on a Multibus II system.

```
(MIP2)          MIP Configuration for MBII

(PID)  COMM Port ID [0-0FFFFH]
(NEM)  Number of External Mailboxes [10-65535]
(RDT)  Response Delay Time [1-0FFFFH]
(NLB)  Number of Large Buffers [2-65535]
(LBS)  Large Buffer Size [2048-65535]
(NSB)  Number of Small Buffers [10-65535]
(SBS)  Small Buffer Size [256-2048]
(CD)   COMMengine Delay [0-0FFFFH]
(QS)   Queue Size [1-65535]
```

(PID) Specifies the MB II Port ID at which the iNA 960 Transport Software is listening for requests.

See also: *Network User's Guide and Reference*

(NEM) Specifies how many unique response mailboxes applications can have in request blocks at any given time.

(RDT) Specifies how many Nucleus clock ticks to wait for a response from the software on the communication board.

(NLB) Specifies how many large buffers MIP will create and release to the buffer pool.

See also: *Buffer pools, System Concepts*

(LBS) Specifies in bytes the size of each large buffer created by an **rq_create_segment** system call.

(NSB) Specifies how many small buffers MIP will create and release to the buffer pool.

(SBS) Specifies in bytes the size of each small buffer created by an **rq_create_segment** system call.

(CD) Specifies how many Nucleus clock ticks the software executing on the host processor waits so that the iNA Transport Software running on the communications board can complete initialization.

(QS) Specifies how many MB II messages the OS will queue at the receive port.

COMMengine Board Instance Load Method for MBII Screen

Use this screen to specify the iNA 960 file to load on a particular COMMengine NIC and to specify the load method.

```
(CEBI)  COMMengine Board Instance Load Method for MBII

(CBI)  Comm. Board Instance [0-19]
(LD)   Load [L/P/N]
(DN)   Device Name [1-14 Chars]
(FN)   File Name [1-30 Chars]
(CC)   Class Code [0-0FFFFH]
```

- (CBI) Specifies the Ethernet board instance to which the other parameters refer.
- (LD) Specifies how the COMMengine is loaded with the iNA Transport Software.
- (DN) Specifies the device name where the iNA Transport Software resides.
- (FN) Specifies the pathname of the load file.
- (CC) Specifies the class code the COMMengine uses when requesting a boot service.

COMMengine Board Names Screen

Use this screen to specify all possible boards used as COMMengine NICs on the Multibus II backplane.

```
(CEBN)  COMMengine Board Names

CEBN = 1-20 characters
[ ]CEBN =
```

- (CEBN) Specifies the boards used as COMMengine NICs.

MIP Configuration for PC Bus Screen

Use this screen to configure the MIP job in a COMMengine hardware environment on a PC.

```
(MIPAT)          MIP Configuration for PC Bus

(CBN)  Communication Board Name [1-20 Chars]
(NEM)  Number of External Mailboxes [0-0FFFFH]
(WP)   Wakeup Port [0-0FFFFH]
(IL)   Interrupt Level [0-0FFH]
(LD)   Load [L/P/N]
(DN)   Device Name [1-14 Chars]
(FN)   File Name [1-30 Chars]
(CC)   Class Code [0-0FFFFH]
(BAB)  Boot Address Base [0-0FFFFH]
(BAO)  Boot Address Offset [0-0FFFFH]
(CD)   COMMengine Delay [0H-0FFFFH]
```

- (CBN) Specifies the name of the Ethernet board used by the MIP job.
- (NEM) Specifies the maximum number of unique response mailboxes that iRMX applications can have in request blocks at any given time.
- (WP) Specifies the I/O port that the PCL2 COMMengine recognizes.
- (IL) Specifies the encoded interrupt level that the iRMX system receives from the PCL2 COMMengine board.
- (LD) Specifies how the COMMengine is loaded with the iNA Transport Software.
- (DN) Specifies the device name where the iNA software resides.
- (FN) Specifies the pathname of the load file.
- (CC) Specifies the class code that the COMMengine uses when requesting a boot service.
- (BAB) Specifies the base portion of the address of one of eight addresses from which the COMMengine gets its boot commands.
See also: *Network User's Guide and Reference* for further information about configuring the COMMengine boot address
- (BAO) Specifies the offset portion of the address of one of eight addresses from which the COMMengine gets its boot commands.
- (CD) Specifies the number of Nucleus clock ticks that the software executing on the host processor waits so that the iNA Transport Software running on the communications board can complete initialization.

iNA 960 COMMputer Job Screen

This screen appears if you specify CMP on the NET screen.

```
(ICMPJ)      iNA 960 COMMputer Job

(ODS) Object Directory Size [0-3840]
(PMI) Pool Minimum [60H-0FFFFFFFH]
(PMA) Pool Maximum [60H-0FFFFFFFH]
(MOB) Maximum Objects [1-0FFFFFFH]
(MTK) Maximum Tasks [1-0FFFFFFH]
(MPR) Maximum Priority [0-255]
(PV)  Parameter Validation [Yes/No]
(TP)  Task Priority [0-255]D
(SS1) Stack Size [0-0FFFFFFH]
(NPX) Numeric Processor Extension Used [Yes/No]
(OFN) Object Module File Name [1-45 Chars]

(CLJ) Create Loadable Job [Yes/No]
(NL)  iNA Network Layer [0/1/2/3]
(SN1) iNA Subnet 1 ID [0-0FFFFFFH]
(SN2) iNA Subnet 2 ID [0-0FFFFFFH]
(SN3) iNA Subnet 3 ID [0-0FFFFFFH]
(SN4) iNA Subnet 4 ID [0-0FFFFFFH]
```

- (ODS) Specifies in decimal the maximum number of entries in this job's object directory; 0 means create no directory.
- (PMI), (PMA) Specifies in 16-byte paragraphs the minimum and maximum sizes of this job's memory pool.
- (MOB) Specifies how many objects can exist simultaneously in the job; 0FFFFFFH means no limit to the number of created objects.
- (MTK) Specifies how many tasks can exist simultaneously in this job.
- (MPR) Specifies in decimal the highest priority of tasks in this job.
- (PV) Specifies whether to include parameter validation for Nucleus system calls made by tasks in this job, if you included system-level parameter validation.
- (TP) Specifies in decimal the priority of this job's initialization task.
- (SSI) Specifies in bytes the size of the initialization task's stack segment, according to your segmentation model.
- (NPX) Specifies if the job's initial task contains floating-point instructions.

- (OFN) Specifies the pathname of the object module for the job.
- (CLJ) Specifies whether to generate a separate job loadable by sysload or generate and bind the job with the system boot image.
- (NL) Specifies the type of network layer configured in the iNA Transport Software.
- (SN1) Specifies the subnet ID of the first subnet configured in the job.
- (SN2) Specifies the subnet ID of the second subnet configured in the job.
- (SN3) Specifies the subnet ID of the third subnet configured in the job.
- (SN4) Specifies the subnet ID of the fourth subnet configured in the job.

iRMX-NET Client Job Screen

Use this screen to configure the parameters for the iRMX-NET client job and remote file driver.

```

(RCJ)                iRMX-NET Client Job

(ODS) Object Directory Size [0-3840]
(PMI) Pool Minimum [60H-0FFFFFFFH]
(PMA) Pool Maximum [60H-0FFFFFFFH]
(MOB) Maximum Objects [1-0FFFFFFH]
(MTK) Maximum Tasks [1-0FFFFFFH]
(MPR) Maximum Priority [0-255]
(PV)  Parameter Validation [Yes/No]
(TP)  Task Priority [0-255]
(SSJ) Stack Size [0-0FFFFFFH]H
(NPX) Numeric Processor Ext. Used [Yes/No]
(CLJ) Create Loadable Job [Yes/No]
(CBN) COMM Board Name [1-20 Chars]
(CBI) COMM Board Instance [0-20]
(COS) Connection Object Size [0-255]
(ISS) I/O Task Stack Size [0-65534]
(ITP) I/O Task Priority [0-255]
(DDS) Device Descriptor Size [0-255]
(EBS) EIOS Buffer Size [0-0FFFFFFH]
(DRQ) DUIBs Required [Yes/No]

```

- (ODS) Specifies in decimal how many entries can be in this job's object directory.
- (PMI), (PMA) Specifies in 16-byte paragraphs the minimum and maximum size of this job's memory pool.

- (MOB) Specifies how many objects can exist simultaneously in this job.
- (MTK) Specifies how many tasks can exist simultaneously in this job.
- (MPR) Specifies in decimal the highest priority of tasks in this job.
- (PV) Specifies whether to include parameter validation for Nucleus system calls made by tasks in this job, if you included system-level parameter validation.
- (TP) Specifies in decimal the priority of this job's initialization task.
- (SSI) Specifies in bytes the size of the initialization task's stack segment.
- (NPX) Use the default setting for this parameter.
- (CLJ) Specifies whether to generate a separate job loadable by **sysload** or to generate and bind the job with the system boot image.
- (CBN) Specifies the name of the board in a Multibus II system that this job uses as a COMMengine from which it takes iNA 960 transport services.
- (CBI) Specifies the instance of the COMMengine whose name is specified in the CBN parameter.
- (COS) Specifies in bytes the size of this file driver's connection object.
- (ISS) Specifies the file driver I/O task stack size.
- (ITP) Specifies the file driver I/O task priority.
- (DDS) Specifies the size of the device descriptor for the file driver
- (EBS) Specifies in bytes the size of the buffers the EIOS uses when performing I/O operations
- (DRQ) Specifies whether or not the file driver requires DUIBs

File Consumer Configuration Screen

```

(FC)                File Consumer Configuration

(MRR) Maximum RFD Request [1-255]
(MSC) Maximum Server Connections [1-60]
(DDS) Datalink Data Size [2-65535]
(CTA) Connect Time Alarm [1-255]
(STA) Send Time Alarm [1-255]
```

- (MRR) Specifies in decimal how many outstanding RFD requests can exist simultaneously.
- (MSC) Specifies in decimal how many servers can connect as remote devices to this consumer at any given time.

- (DDS) Specifies in decimal how many bytes to deliver in one full data link packet.
- (CTA) Specifies how many seconds the File Consumer waits for a server connection.
- (STA) Specifies how many seconds the File Consumer waits for a server to respond to a request.

Remote File Access Screen

Use this screen to configure the iRMX-NET client job and remote file driver (RFD).

```

(REM)           Remote File Access

(ITP)  I/O Task Priority [0-255]
(NOR)  Number of Outstanding RFD system calls [0-255]
(NOS)  Number of Outstanding RFD status calls [0-255]
(JEI)  Job Exit Interval [0-0FFFFFFFFH]
(LI)   Logoff Interval [0-0FFFFFFFFH]
(CBT)  Configuration Base Time [0-0FFFFFFFFH]
```

- (ITP) Specifies in decimal the priority of the two RFD service tasks created.
- (NOR) Specifies in decimal the average number of simultaneous outstanding RFD system calls.
- (NOS) Specifies the average number of simultaneous outstanding RFD **a_get_connection_status**, **a_get_file_status**, and **a_get_extension_data** system calls.
- (JEI) Specifies, in units of Nucleus clock ticks multiplied by 1024, how long the RFD waits to delete resources after a job detaches its last connection to files residing on a remote file server.
- (LI) Specifies the logout interval for the Remote File Driver.
- (CBT) Specifies how many seconds from midnight January 1, 1978 (the RFD fixed point) to the fixed point used by your system.

iRMX-NET Server Job Screen

Use this screen to configure the parameters for the iRMX-NET server job and remote file driver.

```
(RSJ)                iRMX-NET Server Job

(ODS) Object Directory Size [0-3840]
(PMI) Pool Minimum [60H-0FFFFFFFH]
(PMA) Pool Maximum [60H-0FFFFFFFH]
(MOB) Maximum Objects [1-0FFFFH]
(MTK) Maximum Tasks [1-0FFFFH]
(MPR) Maximum Priority [0-255]
(PV)  Parameter Validation [Yes/No]
(TP)  Task Priority [0-255]
(SSI) Stack Size [0-0FFFFH]H
(NPX) Numeric Processor Ext. Used [Yes/No]
(CLJ) Create Loadable Job [Yes/No]
(CBN) COMM Board Name [1-20 Chars]
(CBI) COMM Board Instance [0-20]
```

- (ODS) Specifies in decimal how many entries can be in this job's object directory.
- (PMI) , (PMA)
Specifies in 16-byte paragraphs the minimum and maximum size of this job's memory pool.
- (MOB) Specifies how many objects can exist simultaneously in this job.
- (MTK) Specifies how many tasks can exist simultaneously in this job.
- (MPR) Specifies in decimal the highest priority of tasks in this job.
- (PV) Specifies whether to include parameter validation for Nucleus system calls made by tasks in this job, if you included system-level parameter validation.
- (TP) Specifies in decimal the priority of this job's initialization task.
- (SSI) Specifies in bytes the size of the initialization task's stack segment.
- (NPX) Use the default setting for this parameter.

- (CLJ) Specifies whether to generate a separate job loadable by **sysload** or to generate and bind the job with the system boot image.
- (CBN) Specifies the name of the board in a Multibus II system that this job uses as a COMMengine from which it takes iNA 960 transport services.
- (CBI) Specifies the instance of the COMMengine whose name is specified in the CBN parameter.

File Server Configuration Screen

Use this screen to configure the iRMX-NET File Server job.

(FS)	File Server Configuration
(VC)	Number of Virtual Circuits [1-60]
(UVC)	Number of Users per VC [1-255]
(PVC)	Number of Processes per VC [1-255]
(BVC)	Number of Binds per VC [1-255]
(OFV)	Number of Open Files per VC [1-255]
(OFF)	Number of Open Files per process [1-255]
(SC)	Server Concurrency [1-80]
(FSN)	File Server Name [NONE/1-16 Chars]

- (VC) Specifies in decimal how many virtual circuits can be used by the File Server, which determines how many client systems can concurrently access the File Server.
- (UVC) Specifies in decimal how many distinct users can access a server from the same client system at the same time, as limited by the space in the File Server's data segment.
- (PVC) Specifies in decimal how many jobs (client processes) can concurrently access a virtual circuit, as limited by the space in the File Server's data segment.
- (BVC) Specifies in decimal how many file attaches a client system has on the server at any given time, as limited by the space in the File Server's data segment.
- (OFV) Specifies in decimal how many opened server files a client can have at any given time.
- (OFF) Specifies in decimal how many files a job (client process) can have open.
- (SC) Specifies in decimal how many outstanding requests the File Server can process at any given time.
- (FSN) Specifies the name of the File Server.

Apex File Access Screen

(AFA)	Apex File Access
(MDV)	Max number of Public Devices [1-20]
(MPD)	Max number of Public Dirs [1-64]
(NAT)	Number of AFA tasks [1-60]
(ATT)	Attributes [Yes/No]
(DPD)	Dynamic Public dirs [Yes/No]
(WD)	Wildcard Delete [Yes/No]
(WFS)	Wildcard File Search [1-255]
(WTI)	Wildcard Wait Interval [0-65535]
(PSC)	Print Spooler Commands [Yes/No]

- (MDV) Specifies in decimal how many devices may contain public directories.
- (MPD) Specifies in decimal how many public directories on the server can be accessed by remote systems.
- (NAT) Specifies in decimal how many I/O requests the File Server can service at any given time.
- (ATT) Specifies if special UNIX or DOS file attributes are required.
- (DPD) Specifies whether to add new public directories when the iRMX-NET system is running.
- (WD) Specifies whether to allow wildcard notation in deleting remote files.
- (WFS) Specifies in decimal how many file searches iRMX-NET can concurrently support while performing searches.
- (WTI) Specifies in decimal units of Nucleus clock ticks how long iRMX-NET holds a file connection while performing searches.
- (PSC) Specifies whether to include the Print Spooler Commands in the File Server.

AFA User Screen

This screen appears only if you enable the iRMX-NET server; use it to specify the characteristics of the server's internal AFA User.

(AFAU)	AFA User
(USS)	User Subsystem [0-0FFH]
(NRB)	Number of RBs [1-0FFH]
(EDF)	Extension Data Field [0-255]
(NSB)	Number of Small Buffers [2-255]
(SBS)	Small Buffer Size [2-65535]
(NBB)	Number of Large Buffers [1-255]
(BBS)	Large Buffer Size [2-65535]
(NCB)	Number of Command Buffers [1-255]
(CBS)	Command Buffer Size [2-65535]

- (USS) Specifies the AFA subsystem.
- (NRB) Specifies how many requests an AFA user can make to AFA at the same time.
- (EDF) Set to 0.
- (NSB) Specifies in decimal how many small data buffers to place into one iRMX segment.
- (SBS) Specifies in bytes the size of the small data buffers that hold the data sent between systems during read and write operations.
- (NBB) Specifies in decimal how many large data buffers to place into one iRMX segment.
- (BBS) Specifies in bytes the size of the large buffers used to hold the data that is sent between systems during read and write operations.
- (NCB) Specifies in decimal how many command buffers to use.
- (CBS) Specifies in bytes the size of the command buffers.

Public Devices Screen

This screen specifies which devices in a local system can contain directories available for public use. Entries vary depending on the bus type.

```
(PDEV)          Public Devices

PDEV = Logical Name, Device Name , File Driver, Who Attaches
          [1-12 Chars], [1-14 Chars] , [N/P/S/E/D], [Rmxnet/Eios]

[1] PDEV = SD , , NAMED , EIOS
[2] PDEV = BB , BB , PHYSICAL , EIOS
[3] PDEV =
```

Logical Name

Specifies the logical name of the device that contains public directories.

Device Name

Specifies the name of the particular physical device that contains public directories.

File Driver

Specifies the file driver associated with the device.

Who Attaches

Specifies whether iRMX-NET or EIOS attaches the physical device; use the EIOS to attach public devices so you can delete them with the **detachdevice** command or the **logical_detach_device system** call.

Public Directory Screen

The maximum number of entries in this screen is determined by the maximum number of public directories specified in the AFA screen. Entries vary depending on the bus type.

(PDIR)	Public Directory		
PDIR =	Logical name	Actual name	Public name
	[1-12 Chars]	[1-33 Chars]	[1-14 Chars]
[1]	PDIR = SD	, WORK	, WORK
[2]	PDIR = SD	,	, SD
[3]	PDIR = BB	,	, BB
[4]	PDIR = SD	, LANG286	, LANG286
[5]	PDIR = SD	, SYS386	, SYS386
[6]	PDIR = SD	, UTIL386	, UTIL386
[7]	PDIR = SD	, UTIL286	, UTIL286
[8]	PDIR = SD	, USER	, USER
[9]	PDIR = SD	, RMX286	, RMX286
[10]	PDIR = SD	, RMX386	, RMX386
[11]	PDIR = SD	, BOOT32	, BOOT32
[12]	PDIR = SD	, NET	, NET
[13]	PDIR = SD	, INC	, INC
[14]	PDIR = SD	, LIB	, LIB
[15]	PDIR = SD	, RBOOT32	, RBOOT32
[16]	PDIR = SD	, MSA	, MSA
[17]	PDIR = SD	, MSA32	, MSA32
[18]	PDIR = SD	, BSL	, BSL
[19]	PDIR = SD	, HELPS	, HELPS
[20]	PDIR =		
[20]	PDIR =		

Logical name

Specifies the logical name of the device as cataloged in the root directory that contains the public directory and as defined in a PDEV screen.

Actual name

Specifies the pathname of the directory, starting from the root of the public device.

Public name

Specifies the public directory name that a client sees when accessing this server.

User Definition File Screen

(UDF)	User Definition File
(MUL)	Master UDF Location [Named/Remote]
(MLN)	Master UDF Logical Name [1-12 Chars]
(MPN)	Master UDF Path Name [1-45 Chars]
(MD)	Master UDF Device [1-14 Chars]
(LUL)	Local UDF Location [Named/Edos/Dos/Unused]
(LLN)	Local UDF Logical Name [1-12 Chars]
(LPN)	Local UDF Path Name [1-45 Chars]
(LD)	Local UDF Device [1-14 Chars]
(UPD)	Update UDF [0-0FFFFH]
(WV)	Who Verifies [Eios/Rmxnet]
(WA)	Who Attaches [Eios/Rmxnet]
(ALN)	Automatic Loadname [Yes/No]

- (MUL) Specifies the location of the Master UDF.
- (MLN) Specifies the logical name assigned to the device where the Master UDF resides.
- (MPN) Specifies the pathname of the Master UDF.
- (MD) Specifies the device name according to the location of the Master UDF that the User Administration module must physically attach.
- (LUL) Specifies the physical location of the local UDF.
- (LLN) Specifies the logical name of the device where the local UDF resides.
- (LPN) Specifies the pathname of the local UDF.
- (LD) Specifies the name of the device where the local UDF resides.
- (UPD) Specifies how many seconds the User Administration module waits between updates of the local UDF from the Master UDF.
- (WV) Specifies whether iRMX-NET verifies users or uses the **rq_verify_user** system call.
- (WA) Specifies whether iRMX-NET or the EIOS attaches the system device.
- (ALN) Specifies whether iRMX-NET should initialize the Name Server Database.

Client Definition File Screen

(CDF)	Client Definition File
(CLC)	CDF Location [Named/Remote]
(CLN)	CDF Logical Name [1-12 Chars]
(CPN)	CDF Path Name [1-45 Chars]
(CDD)	CDF Device [1-14 Chars]
(CNN)	Client Name [1-8 Chars]
(CNP)	Client Password [1-8 Chars]
(CIB)	Client Info Addr Base [0-0FFFFH]
(CIO)	Client Info Addr Offset [0-0FFFFH]

- (CLC) Specifies the location of the CDF file.
- (CLN) Specifies the logical name assigned to the device where the CDF resides; use all UPPERCASE letters.
- (CPN) Specifies the pathname of the CDF file.
- (CDD) Specifies the name of the device where the CDF resides.
- (CNN) Specifies the name of the client system.
- (CNP) Specifies the password of the client system.
- (CIB) Specifies the base portion of the absolute memory address where the client system's name and password are placed.
- (CIO) Specifies the offset portion of the absolute memory address where the client system's name and password are placed.

Name Server Configuration Screen

This screen appears only if you specify a COMMputer job on the NET screen.

(NS)	Name Server Configuration
(NO)	Maximum Number of Objects [2-246]
(PVL)	Maximum Property Value Length [20-255]
(MC1)	Multicast Address 1 [0-0FFH]
(MC2)	Multicast Address 2 [0-0FFH]
(MC3)	Multicast Address 3 [0-0FFH]
(MC4)	Multicast Address 4 [0-0FFH]
(MC5)	Multicast Address 5 [0-0FFH]
(MC6)	Multicast Address 6 [0-0FFH]
(ITI)	Initiator TSAP ID [0-0FFFFH]
(RTI)	Responder TSAP ID [0-0FFFFH]
(FST)	File Server TSAP ID [0-0FFH]
(FCT)	File Consumer TSAP ID [0-0FFH]
(RET)	Retry Timeout [0-0FFFFH]
(NR)	Maximum Number of Retries [0-255]
(RSP)	Maximum Number of Responses [1-255]
(LO)	Name Server Local Only [Yes/No]

- (NO) Specifies in decimal how many objects can be entered into the local object table, not including objects that iRMX-NET loads into the local object table during initialization.
- (PVL) Specifies in decimal the maximum length of the property value.
- (MC1) – (MC6)
Specifies the first through the sixth bytes of the multicast address that the Name Server uses to transmit Name Server queries over the network.
- (ITI) Specifies the Transport Service Access Point Identification (TSAP-ID) the initiator of Name Server queries uses.
- (RTI) Specifies the TSAP-ID for the Name Server Responder.
- (FST) Specifies the TSAP-ID for the File Server.
- (FCT) Specifies the TSAP-ID for the File Consumer.

- (RET) Specifies in 100 microsecond increments the time interval for the Name Server to wait before trying a request to a remote network system.
- (NR) Specifies how many times a request is retransmitted by the Name Server before being discontinued.
- (RSP) Specifies how many responses can be handled by the Name Server for a request.
- (LO) Specifies whether the Name Server supports local operations only and cannot access remote object tables to retrieve property values or execute other functions.

Name Server Search Domains Screen

Use this screen to configure subnet IDs to be searched by the Name Server in a system where you use an iNA 960 job with an ES-IS network layer.

```
(NSDOM)      Name Server Search Domains

DOM = Subnet I.D.
      [ 0-0FFFFH ]
[ ] DOM =
```

- (DOM) Specifies up to twenty subnet IDs to be searched by the Name Server.

TCP/IP Job Screen

Use this screen to configure the parameters for the TCP/IP job.

```
(TCPJ)                iRMX TCP/IP Job

(ODS) Object Directory Size [0-3840]
(PMI) Pool Minimum [60H-0FFFFFFFH]
(PMA) Pool Maximum [60H-0FFFFFFFH]
(MOB) Maximum Objects [1-0FFFFFFH]
(MTK) Maximum Tasks [1-0FFFFFFH]
(MPR) Maximum Priority [0-255]
(PV)  Parameter Validation [Yes/No]
(TP)  Task Priority [0-255]
(SSJ) Stack Size [0-0FFFFFFH]H
(NPX) Numeric Processor Ext. Used [Yes/No]
```

- (ODS) Specifies in decimal how many entries can be in this job's object directory.
- (PMI) , (PMA)
Specifies in 16-byte paragraphs the minimum and maximum size of this job's memory pool.
- (MOB) Specifies how many objects can exist simultaneously in this job.
- (MTK) Specifies how many tasks can exist simultaneously in this job.
- (MPR) Specifies in decimal the highest priority of tasks in this job.
- (PV) Specifies whether to include parameter validation for Nucleus system calls made by tasks in this job, if you included system-level parameter validation.
- (TP) Specifies in decimal the priority of this job's initialization task.
- (SSJ) Specifies in bytes the size of the initialization task's stack segment.
- (NPX) Specify 'No' for this parameter.

Parameters for TCP/IP Network Interfaces

Use this screen to configure TCP/IP interfaces such as Ethernet or SLIP.

```
(IFSP)      Parameters for TCP/IP Network Interfaces

(IN)  Interface Name [0-10 Chars]
(HNM)  HOSTNAME [1-45 Chars]
(HID)  HOSTID IP Address [0-15 Chars]
(ISN)  iNA Subsystem Number or [20H-2FH]
       Serial Line Internet Protocol [SLIP = 0H]
(DST)  Destination IP Address for SLIP [0-15 Chars]
(NM)   Netmask [0-10 Chars]
```

- (IN) Specifies the name of the network interface.
- (HNM) Specifies the ASCII domain name of this network interface.
- (HID) Specifies the IP address of the network interface.
- (ISN) Specifies the iNA subsystem number (20H through 2FH) for the Ethernet network interface. Specify a 0 for SLIP.
- (DST) Specifies the destination IP address for SLIP.
- (NM) Specifies the netmask of the network interface.

IP Router Information Screen

Use this screen to configure the Internet Protocol Router.

```
(ROUTE)     IP Router Information

(DT)  Destination Type [net/host]
(DST)  Destination IP Address [0-15 Chars]
(GTW)  Gateway IP Address [0-15 Chars]
(MTR)  Metric [0-255]
```

- (DT) Specifies the type of destination.
- (DST) Specifies the IP address of the host or network to which the route leads.
- (GTW) Specifies the IP address of the gateway through which packets for this destination should be routed.
- (MTR) Specifies the number of hops from the local hosts to the destination.

TCP/IP Tunable Parameters 1 Screen

Use this screen to set some of the tunable parameters for TCP/IP.

See also: Tunable parameters, *TCP/IP and NFS for the iRMX Operating System*

```
(TCP1)          TCP/IP Tunable Parameters 1

(PE)  PRIO EDLINA [0-255]
(PT)  PRIO TIMEOUT [0-255]
(PP)  PRIO PTMX [0-255]
(PQ)  PRIO QSCHED [0-255]
(PD)  PRIO DELETION [0-255]
(AND) ARP NDEV [1-255]
(ANT) ARP NTABSIZE [0-255]
(ATI) ARP TIMEOUT [0-255]
(ILS) IP LOCALSUBNETS [0-1]
(IMA) IP MYAUTHORITY [0-255]
(IMC) IP MYCLASS [0-255]
(IND) IP NDEV [1-255]
(INL) IP NLINK [0-255]
(INR) IP NROUTES [0-255]
(ISE) IP SECURITY [0-1]
(ITO) IP TOS [0-255]
(ITT) IP TTL [0-255]
```

- (PE) Specifies the priority of the EDLINA task.
- (PT) Specifies the priority of the Timer task.
- (PP) Specifies the priority of the Receiver task.
- (PQ) Specifies the priority of the Queue Scheduler Task.
- (PD) Specifies the priority of the Deletion task.
- (AND) Specifies the number of minor devices for the ARP module.
- (ANT) Specifies the maximum number of entries in the ARP table.
- (ATI) Specifies in minutes how long the host waits for ARP packets before deleting an ARP table entry.
- (ILS) Set this parameter to 1 if your system is directly connected to a network that is divided into subnets. If not, set this parameter to 0.
- (IMA) Specifies the value of the protection authority flag used for the Internet Protocol Security Option.

- (IMC) Specifies the value of the classification protection level used for the Internet Protocol Security Option.
- (IND) Specifies the number of minor devices for the IP module.
- (INL) Specifies the maximum number of network interfaces you can configure under the IP module.
- (INR) Specifies the maximum number of entries in the IP routing tables.
- (ISE) Controls the processing of incoming messages that have the IP options for security.
- (ITO) Specifies the Internet Protocol encoding for precedence and type of service.
- (ITT) Specifies the maximum time-to-live (TTL) for an outgoing datagram.

TCP/IP Tunable Parameters 2 Screen

Use this screen to set more tunable parameters for TCP/IP.

See also: Tunable parameters, *TCP/IP and NFS for the iRMX Operating System*

```
(TCP2)          TCP/IP Tunable Parameters 2

(NTI) NUMTIM [0-255]
(NTR) NUMTRW [0-255]
(RND) RAW NDEV [1-255]
(SND) SLIP NDEV [1-255]
(SON) SOMOD NDEV [1-255]
(TBI) TCP BSD42 ISS [0-1]
(TBU) TCP BSD42 URGENT [0-1]
(HDP) TCP HEADERPRED [0-1]
(LIN) TCP LINGER [0-255]
(MQL) TCP MAXQLEN [0-255]
(MSE) TCP MAXSEG [1024-32768]
(MWI) TCP MAXWINDOW [1024-32768]
(MIS) TCP MINSEG [128-32768]
(MIW) TCP MINWINDOW [128-32768]
(NDV) TCP NDEV [1-255]
(NDA) TCP NODELAYACK [0-1]
(TVM) TCP TVMSL [0-255]
(WIN) TCP WINDOW [128-32768]
```

- (NTI) Specifies the maximum number of instances of a timod module.
- (NTR) Specifies the maximum number of instances of a tirdwr module.
- (RND) Specifies the number of minor devices for the raw module.

- (SND) Specifies the number of minor devices for the slip module.
- (SON) Specifies the number of minor devices for the somod module.
- (TBI) Specifies the sequence number passed in TCP segments.
- (TBU) Requests urgent handling of TCP messages (setting of 1).
- (HDP) Increases TCP performance (setting of 1).
- (LIN) Controls how long TCP waits for queued data to be flushed before it honors a request to close the stream.
- (MQL) Specifies the maximum number of pending connections that can be queued for a single TCP minor device.
- (MSE) , (MIS)
Specifies the maximum and minimum values an application can set the per-connection maximum segment size using the **w_setopt** and **setsockopt()** function calls.
- (MWI) , (MIW)
Specifies the maximum and minimum values an application can set the per-connection receive window using the **w_setopt** and **setsockopt()** functions calls.
- (NDV) Specifies the number of minor devices for the TCP module.
- (NDA) Controls how acknowledgments are set upon receiving a segment.
- (TVM) Controls (in half-second units) how long TCP waits for protocol state transition messages.
- (WIN) Specifies the default size of the TCP receive window.

TCP/IP Tunable Parameters 3 Screen

Use this screen to set more tunable parameters for TCP/IP.

See also: Tunable parameters, *TCP/IP and NFS for the iRMX Operating System*

```
(TCP3)          TCP/IP Tunable Parameters 3

(TND) TELNET NDEV [1-255]
(UCS) UDP CKSUM [0-1]
(UHW) UDP HIWATER [128-32768]
(ULW) UDP LOWATER [128-32768]
(UMH) UDP MAXHIWATER [128-32768]
(UML) UDP MAXLOWATER [128-32768]
(UND) UDP NDEV [1-255]
(EBR) EDL BOARDS [0-16]
(EBP) EDL BUF5 POSTED [1-255]
(EEP) EDL ENDPOINTS [1-101]
```

- (TND) Specifies the number of minor devices for the kernel telnet module.
- (UCS) Specifies whether a checksum verification is performed by default on incoming and outgoing UDP packets.
- (UHW), (ULW)
Specifies in bytes the Stream's high and low water marks for the UDP driver.
- (UMH), (UML)
Specifies the maximum and minimum that UDP_HIWATER and UDP_LOWATER may be programmatically set through UO_WR_HIWAT and UO_WR_LOWAT ioctl requests.
- (UND) Specifies the number of minor devices for the udd module.
- (EBR) Specifies the maximum number of Data Link interfaces allowed.
- (EBP) Specifies the number of RawEDL receive buffers automatically posted by TCP/IP to iNA.
- (EEP) Specifies the number of Data Link service access points (LSAPs) that can be active at the same time.

Streams Parameters Screen

Use this screen to set more tunable parameters for TCP/IP.

See also: Tunable parameters, *TCP/IP and NFS for the iRMX Operating System*

(STRM)	Streams Parameters
(NC)	NCALL [0-255]
(NM)	NMUXLINK [0-255]
(NQ)	NQUEUE [0-4096]
(NS)	NSTREAM [0-255]
(NT)	NTIMEOUT [0-255]
(NW)	NWAKEUP [0-255]
(MS)	MSGSEG [0-4096]
(N4)	NBLK4 [0-4096]
(N16)	NBLK16 [0-4096]
(N64)	NBLK64 [0-4096]
(N28)	NBLK128 [0-4096]
(N56)	NBLK256 [0-4096]
(N12)	NBLK512 [0-4096]
(N24)	NBLK1024 [0-4096]
(N48)	NBLK2048 [0-4096]
(N96)	NBLK4096 [0-255]
(N92)	NBLK8192 [0-255]
(N32)	NBLK32K [0-255]

- (NC) Specifies how many call-out table entries to allocate.
- (NM) Specifies how many links can exist between multiplexed Streams drivers.
- (NQ) Reflects adjustments made to the NSTREAM parameter. It should be about four times the number of Streams configured into the TCP/IP kernel.
- (NS) Specifies the maximum number of simultaneous active open Streams.
- (NT) Specifies the number of timeout table entries to allocate.
- (NW) Specifies the number of sleep/wakeup table entries to allocate.
- (MS) Specifies the number of command/message segments for each user-level stream. There should be at least two.
- (N4) Specifies the number of four-byte Streams message blocks.
- (N16) Specifies the number of 16-byte Streams message blocks.
- (N64) Specifies the number of 64-byte Streams message blocks.

- (N128) Specifies the number of 128-byte Streams message blocks.
- (N56) Specifies the number of 256-byte Streams message blocks.
- (N12) Specifies the number of 512-byte Streams message blocks.
- (N24) Specifies the number of 1024-byte Streams message blocks.
- (N48) Specifies the number of 2048-byte Streams message blocks.
- (N96) Specifies the number of 4096-byte Streams message blocks.
- (N92) Specifies the number of 8192-byte Streams message blocks.
- (N32) Specifies the number of 32K byte Streams message blocks.

X.25 Server Job Screen

Use this screen to configure the parameters for the X.25 Server job.

```

(X25SJ)      X.25 Server Job

(ODS) Object Directory Size [0-3840]
(PMI) Pool Minimum [60H-0FFFFFFFH]
(PMA) Pool Maximum [60H-0FFFFFFFH]
(MOB) Maximum Objects [1-0FFFFFFH]
(MTK) Maximum Tasks [1-0FFFFFFH]
(MPR) Maximum Priority [0-255]
(PV)  Parameter Validation [Yes/No]
(TP)  Task Priority [0-255]
(SSJ) Stack Size [0-0FFFFFFH]
(NPX) Numeric Processor Extension Used [Yes/No]
```

- (ODS) Specifies the maximum allowable number of entries in this job's object directory.
- (PMI), (PMA) Specifies in 16-byte paragraphs the minimum and maximum sizes of this job's memory pool.
- (MOB) Specifies how many objects can exist simultaneously in the job; 0FFFFFFH means no limit to the number of created objects.
- (MTK) Specifies how many tasks can exist simultaneously in this job.
- (MPR) Specifies in decimal the highest priority of tasks in this job.
- (PV) Specifies whether to include parameter validation for Nucleus system calls made by tasks in this job, if you included system-level parameter validation.
- (TP) Specifies in decimal the priority of this job's initialization task.

- (SSI) Specifies in bytes the size of the initialization task's stack segment, according to your segmentation model.
- (NPX) Specifies if the job's initial task contains floating-point instructions.

X.25 Client Job Screen

Use this screen to configure the parameters for the X.25 Client job.

```

(X25CJ)      X.25 Client Job

(ODS) Object Directory Size [0-3840]
(PMI) Pool Minimum [60H-0FFFFFFFH]
(PMA) Pool Maximum [60H-0FFFFFFFH]
(MOB) Maximum Objects [1-0FFFFFFH]
(MTK) Maximum Tasks [1-0FFFFFFH]
(MPR) Maximum Priority [0-255]
(PV)  Parameter Validation [Yes/No]
(TP)  Task Priority [0-255]
(SSI) Stack Size [0-0FFFFFFH]
(NPX) Numeric Processor Extension Used [Yes/No]
```

- (ODS) Specifies the maximum allowable number of entries in this job's object directory.
- (PMI) , (PMA) Specifies in 16-byte paragraphs the minimum and maximum sizes of this job's memory pool.
- (MOB) Specifies how many objects can exist simultaneously in the job; 0FFFFFFH means no limit to the number of created objects.
- (MTK) Specifies how many tasks can exist simultaneously in this job.
- (MPR) Specifies in decimal the highest priority of tasks in this job.
- (PV) Specifies whether to include parameter validation for Nucleus system calls made by tasks in this job, if you included system-level parameter validation.
- (TP) Specifies in decimal the priority of this job's initialization task.
- (SSI) Specifies in bytes the size of the initialization task's stack segment, according to your segmentation model.
- (NPX) Specifies if the job's initial task contains floating-point instructions.

Nucleus Screens

Nucleus Screen

(NUC)	Nucleus
(NGE)	Number of GDT Entries [440-8190]
(NIE)	Number of IDT Entries [0-256]
(PV)	Parameter Validation [Yes/No]
(ROD)	Root Object Directory Size [0-3840]
(DSH)	Default Software Exception Handler [Job/Task/STask/User]
(EM)	Exception Mode [Never/Program/Environ/All]
(NEH)	Name of Ex Handler Object Module [1-55 Chars]
(DHH)	Default Hardware Exception Handler [Job/Task/STask/Monitor]
(NMI)	NMI Exception Handler [Yes/No/Ignore/Job/SDM/User]
(NEB)	NMI Enable Byte [0-255]
(LSE)	Low GDT/LDT Slot Excluded from FSM [440-8189/NONE=0]
(HSE)	High GDT/LDT Slot Excluded from FSM [440-8189/NONE=0]
(RRP)	Round Robin Priority Threshold [0-255]
(RRT)	Round Robin Time Quota [0-255]
(RIE)	Report Initialization Errors [Yes/No]
(MCE)	Maximum Data Chain Elements [0-0FFFFH]
(CS)	Nucleus Communication Service [Yes/No]

⇒ **Note**

If you use an iNA 960 MIP job in a Multibus II system, you must exclude GDT slots 4096 through 4767. Set LSE to 4096 and HSE to 4767.

- (NGE) Specifies in decimal the number of objects in your system, since each object must have a descriptor in the GDT. Add 16 bytes for each entry to required memory on the MEMS screen.
- (NIE) Specifies in decimal the number of entries in the IDT.
- (PV) Specifies whether system-level parameter validation occurs.
See also: Parameter validation, *System Call Reference*
- (ROD) Specifies in decimal how many entries can exist in the root job's object directory.
- (DSH) Specifies the system's default exception handler.
See also: Exception handlers, *System Concepts*

- (EM) Specifies the exception mode of the default system exception handler.
- (NEH) Specifies the pathname of the exception handler object module output by BND386, if you specified User for the DSH parameter.
- (DHH) Specifies the system's default hardware exception handler.
- (NMI) Specifies how an NMI exception is handled.
- (NEB) (MBII only) Specifies the NMI interrupt source(s) to be enabled.
- (LSE) , (HSE)
Specifies in decimal the range additional GDT/LDT slots to reserve for first-level user jobs and I/O jobs, in addition to the default of approximately 350 slots.
- (RRP) Specifies in decimal the priority below which tasks will be assigned round-robin scheduling.

See also: Round-robin scheduling, *Introducing the iRMX Operating Systems*
- (RRT) Specifies in decimal the time, in Nucleus clock ticks, each task is allowed to run before the Nucleus transfers control to a waiting task with the same priority. See also CIN.
- (RIE) Specifies whether initialization errors for all iRMX layers are displayed at the monitor console as a hexadecimal code and a mnemonic.
- (MCE) Specifies how many elements can exist in a single DMA data chain.

See also: Data chains, *System Concepts*
- (CS) Specifies whether to include the Nucleus Communication Service.

Nucleus Communication Service Screen

(NCOM)	Nucleus Communication Service
(PMT)	Message Task Priority [0-255]
(PDT)	Deletion Task Priority [0-255]
(DPT)	Default Number of Port Transactions [0-255]
(DHI)	Default Host ID [0=None/1-254]
(VBP)	Validate Buffer Parameters [Yes/No]
(MST)	Max. No. of Simultaneous Transactions [0-0FFFFH]
(MSM)	Max. No. of Simultaneous Messages [0-0FFFFH]
(RFT)	Receive Fragment Failsafe Timeout [0-0FFFFH]
(NTM)	Number of Trace Messages [0-255]

- (PMT) Specifies in decimal the priority of the Nucleus Communications Service message task, which runs when a solicited message is received.
- (PDT) Specifies in decimal the Port Deletion Task priority, typically a higher priority than your application.
- (DPT) Specifies in decimal a default number of simultaneous port transactions.
- (DHI) Specifies in decimal the message address of each board in a MB II system that needs to pass messages.
- (VBP) Specifies whether buffer parameter validation occurs, if you enabled PV on the NUC screen.
- (MST) Specifies how many simultaneous transactions can exist in the system.
- (MSM) Specifies how many simultaneous messages can exist in the system.
- (RFT) Specifies in Nucleus clock ticks how long the **receive_fragment** system call will wait for a message fragment.
- (NTM) Specifies in decimal the number of trace messages permitted on this board for debugging purposes.

OS Extension Screen

This is a repetitive screen. Do not enter names or numbers you have already used on another OS Extension screen.

See also: OS Extensions, *System Concepts*

(OSEXT)	O.S. Extension
(GSN)	GDT Slot Number [440-8189]
(EPN)	Entry Point Name [1-45 Chars]

- (GSN) Specifies in decimal the actual GDT slot to reserve for a call gate.
- (EPN) Specifies the PUBLIC name of the entry procedure that the call gate enters the OS extension from.

Query Screen

The Query screen appears after you complete the Device Driver screens, the Unit Information screens, the Device-Unit Information screens, the I/O Jobs screen, the AFAU screen, the OSEXT screen, and the User Jobs screen.

Do you need any/more [screen name] ?

Respond Yes if you need more screens; otherwise, respond No or <CR>.

ROM Code Screen

Use this screen to place your system in (EP)ROM; otherwise respond No to the first parameter.

See also: *Developing applications for ROM, Programming Techniques and AEDIT Text Editor*

```
(ROM)          ROM Code

(SYR) System in ROM [Yes/No]
(CPI) Copy ROM Initialization Code to RAM [Yes/No]
(EOR) Execute System Out of ROM/Flash [Yes/No]
(VSS) Volatile System Memory Starting Address [0-0FFFFFFFH]
(VSE) Volatile System Memory Ending Address [0-0FFFFFFFH]
(RBA) Base Address of ROM Init code at reset [0-0FFFFFFFH]
(RDA) RAM Destination Address of ROM Init code [0-0FFFFFFFH]
(SRC) Size of ROM Initialization Code [0-0FFFFFFFH]
(CRS) Custom ROM Initialization Source File [1-45 Chars]
(CRO) Custom ROM Initialization Object File [1-45 Chars]
```

- (SYR) Specifies whether the application will reside in ROM.
- (CPI) Specifies whether ROM initialization code is copied to RAM.
- (EOR) Specifies whether to execute the first-level jobs out of ROM or FLASH.
- (VSS), (VSE)
Specifies the first and last addresses for volatile memory not dedicated to any hardware function.
- (RBA) Specifies the 32-bit physical base address of the ROM Initialization Code at system reset.
- (RDA) Specifies the 32-bit physical address in RAM where the ROM Initialization Code is copied if the CPI parameter is YES.
- (SRC) Specifies the size of the ROM Initialization Code.
- (CRS) Specifies a source file with customized ROM Initialization Code.
- (CRO) Specifies the object code containing external procedures used by your custom ROM Initialization Code.

Shared C Library Screen

(CLIB)	Shared C Library
(NEB)	Number of EIOS buffers per connection [0-225]
(MBS)	Malloc block size [16-0FFFFFFFFH]
(NL)	Numeric Library [Yes/No]
(CIO)	Console I/O [EIOS/iSDM/User/None]
(UCP)	User Console I/O Procedure [1-45 Chars]

- (NEB) Specifies how many buffers to allocate per connection.
- (MBS) Specifies the size in bytes of the memory block to allocate for each **malloc** call.
- (NL) Specifies whether to include the library that provides floating-point functions.
- (CIO) Specifies which sub-system provides I/O for *stdin* and *stdout*.
- (UCP) Specifies the pathname of the object module containing the PUBLIC names for *:ci:* and *:co:*, if you selected other than None for (CIO).

Subsystems Screen

The Nucleus is always included; the subsystems are optional. Each optional subsystem requires the Nucleus and may require at least one other subsystem; the ICU adds the required subsystems as shown below.

See also: Using segments and subsystems, *Programming Techniques and AEDIT Text Editor*

Subsystem	Supporting Subsystems					
	HI	AL	EIO	BIO	SDM	SDB
UDI	Req	Req	Req	Req		
HI		Req	Req	Req		
AL			Req*	Req		
iRMX-NET				Req		
EIOS				Req		
BIOS						
SDB						Req
SDM						
CLIB (for full functionality)	Req	Req	Req	Req	Req	

* See also ASC on the APPL screen

Adding a subsystem increases both capabilities and memory requirements. The memory requirements for each of the subsystems are

Sub-system	Approximate Memory Required
Nucleus	105K
BIOS	95K*
EIOS	19K
AL	12K
HI	36K
CLI	56K (Release 2 CLI)
UDI	11K
SDB	57K
CLIB (for full functionality)	85K

* This is the BIOS size in the Intel-supplied start-up system. The BIOS varies depending on how many file and device drivers you included in the system.

See also: Individual subsystems, *Introducing the iRMX Operating Systems*

(SUB)	Sub-systems
(UDI)	Universal Development Interface [Yes/No]
(CLB)	Shared C Library [Yes/No]
(HI)	Human Interface [Yes/No]
(AL)	Application Loader [Yes/No]
(NET)	Networking [Yes/No]
(EIO)	Extended I/O System [Yes/No]
(BIO)	Basic I/O System [Yes/No]
(PGS)	Paging Subsystem [Yes/No]
(VMD)	VM86 Dispatcher [Yes/No]
(SDM)	System Debug Monitor [Yes/No]
(SDB)	System Debugger [Yes/No]
(OE)	OS Extension [Yes/No]

- (UDI) Provides a standard set of system calls that enables applications to run on any OS supporting the UDI.
- (CLB) Provides ANSI-compatible C functions as a single, multitasking subsystem.
- (HI) Provides an interactive interface between users and applications.
- (AL) Loads object files into memory from secondary storage: absolute code into fixed locations, relocatable code into dynamically allocated memory locations, and files containing overlays.
- (NET) Specifies whether you want to include any of the networking systems, including iNA 960, iRMX-NET, TCP/IP, or X.25.
- (EIO) Provides high-level, synchronous file access and device independence for applications.
- (BIO) Provides asynchronous file access and device independence for applications.
- (PGS) Specifies whether or not the Paging Subsystem is included to support your flat-model applications.
- (VMD) (iRMX for PCs and iRMX for Windows) Enables DOS and ROM/BIOS software to run in VM86 mode.
- (SDM) Provides the SDM monitor to help debug applications.
- (SDB) Extends the use of SDM so that you can interactively examine data structures.
See also: *System Debugger Reference*
- (OE) Enables you to modify the OS to add OS extensions.
See also: OS extensions, *System Concepts*

System Debugger Screens

System Debugger Screen

```
(SDB) System Debugger

(SLV) SDB Interrupt Level [Encoded Level/NONE=0FFH]

(ESC) Enable Screen Scrolling [Yes/No]
```

- (SLV) Specifies the encoded interrupt level, if any, that your system uses to invoke the SDB.
- (ESC) Specifies whether or not to enable screen display scroll control.

System Debug Console Screen for Multibus I/II

```
(SDM) System Debug Console

Multibus Drivers

(D51) 8251 Console Controller Driver [Primary/Secondary/No]
(A54) 354 Port A Console Controller Driver [Primary/Secondary/No]
(B54) 354 Port B Console Controller Driver [Primary/Secondary/No]
(A74) 8274 Port A Console Controller Driver [Primary/Secondary/No]
(B74) 8274 Port B Console Controller Driver [Primary/Secondary/No]
(G79) SBX 279 Console Controller Driver [Primary/Secondary/No]
(A30) 82530 Port A Console Controller Driver [Primary/Secondary/No]
(B30) 82530 Port B Console Controller Driver [Primary/Secondary/No]
(RCI) Remote Console Interface Driver [Primary/Secondary/No]

PC Drivers

(SR1) Serial Port One [Primary/Secondary/No]
(BP1) Serial Port One Base Address [0-0FFFFH]
(SR2) Serial Port Two [Primary/Secondary/No]
(BP2) Serial Port Two Base Address [0-0FFFFH]
(CON) Console Port [Primary/Secondary/No]
```

- (D51), (A54), (B54), (A74), (B74), (G79), (A30), (B30), (RCI) Specifies which I/O device SDM uses. Choose at least one primary channel and optionally one secondary channel.

(SR1), (SR2), (CON)

Specifies which I/O device SDM uses on a PC. Choose at least one primary channel and optionally one secondary channel.

(BP1), (BP2)

Specifies the base IO Address of Serial Ports One and Two (COM1 and COM2).

System Jobs Screens

System Jobs Screen

System Jobs are I/O jobs and first-level jobs included in the iRMX OS.

See also: *System Configuration and Administration* for more information on System Jobs

(SYSJ)	System Jobs
(PCI)	PCI Server Job [Yes/No]
(DL)	MBII Downloader Job [Yes/No]
(ATC)	ATCS/279/ARC Server Job [Yes/No]
(A50)	ATCS/450 Server Job [Yes/No]
(BS)	MSA BootServer Job [Yes/No]
(FPI)	FPI Server Job [Yes/No]
(SSK)	Soft-Scope Kernel Job [Yes/No]

- (PCI) Specifies whether to include the PCI Server Job and use the SCSI interface on the SBC 386/258, 386/258D, 486/133SE, 386/12S or 486/12S board.
- (DL) Specifies whether to include the MB II Downloader Job, if your system contains MB II boards that need to be downloaded at initialization time.
- (ATC) Specifies whether to include the ATCS/279/ARC Server so other MB II hosts can access an iSBX 279 module mounted on a CPU board or an on-board serial channel, including iSBX modules.
- (A50) Specifies whether to include an ATCS/450 Server, which provides access to serial channels on MIX/450 modules mounted on a MIX base board or MPI/450 boards on the MB II PSB.
- (BS) Specifies whether to include the MSA BootServer Job, which provides MSA booting services to other MB II boards in the system.
- (FPI) Specifies whether to include the MB II Front Panel Interrupt (FPI) Server Job.
- (SSK) Specifies whether to include the Soft-Scope Kernel in the system.

PCI Server Job Screen

(PCIJ)	PCI Server Job
(ODS)	Object Directory Size [0-3840]
(PMI)	Pool Minimum [60H-0FFFFFFFH]
(PMA)	Pool Maximum [60H-0FFFFFFFH]
(MOB)	Maximum Objects [1-0FFFFFFH]
(MTK)	Maximum Tasks [1-0FFFFFFH]
(MPR)	Maximum Priority [0-255]
(PV)	Parameter Validation [Yes/No]
(TP)	Task Priority [0-255]
(SSI)	Stack Size [0-0FFFFFFH]
(MI)	Multibus I SPC RMX Level [0-0FFH]
(AC)	SCSI Auto Configuration [Yes/No]
(NS)	Number of PCI Server SCSI controllers [1-10]

- (ODS) Specifies in decimal the maximum number of entries in this job's object directory.
- (PMI), (PMA) Specifies in 16-byte paragraphs the minimum and maximum sizes of this job's memory pool.
- (MOB) Specifies how many objects can exist simultaneously in this job.
- (MTK) Specifies how many tasks can exist simultaneously in this job.
- (MPR) Specifies in decimal the highest priority of tasks in this job.
- (PV) Specifies whether to include parameter validation for Nucleus system calls made by tasks in this job, if you included system-level parameter validation.
- (TP) Specifies the priority of this job's initialization task.
- (SSI) Specifies in bytes the size of the initialization task's stack segment, according to your segmentation model.
- (MI) Specifies the RMX interrupt level for the SCSI Protocol Chip as defined jumper.
- (AC) Specifies whether a custom or default PCI Server is required to control the SCSI chip on a Multibus board.
- (NS) Specifies the number of SCSI controllers or devices being supported.

PCI Server Controller Configuration Screen

```
(PCISC)      PCI Server Controller Configuration

(SCT) SCSI Controller Type [0-0FFH]
(BA)  Base Address [0-0FFFFH]
(STO) Select Timeout [0-0FFH]
(BON) Bus On Time [0-0FFH]
(BOF) Bus Off Time [0-0FFH]
(XS)  Xfer Speed [0-0FFH]
(L00) Scan Lun 0 Only [Yes/No]
(INT) RMX Interrupt Level [0-0FFH]
(SI)  SCSI Id [0-7]
(RES) Reset SCSI Bus [Yes/No]
(DMA) DMA Channel [5-7]
(NC)  Number of CCBs [1-8]
```

- (SCT) Specifies the SCSI controller device.
- (BA) Specifies the base address of the SCSI controller.
- (STO) Specifies how long the SCSI controller will wait before timing out on a select.
- (BON), (BOF)
Specifies in microseconds the burst-on and burst-off times during DMA transfers across the bus.
- (XS) Specifies the SCSI controller's maximum transfer rate during data transfers.
- (L00) Specifies whether the PCI Server scans all logical devices or just logical device 0 during the SCSI bus scan.
- (INT) Specifies the iRMX encoded value used by the SCSI controller for the interrupt level.
- (SI) Specifies the SCSI identification for the device.
- (RES) Specifies whether the SCSI bus will reset at initialization.
- (DMA) Specifies the DMA channel used by the SCSI controller.
- (NC) Allocates space to interface with the Adaptec controller.



Note

For more specific details, see the ICU help messages for this screen and for each parameter.

Multibus II Downloader Job Screen

This screen applies to Multibus II systems.

```
(DLJ)          MBII Downloader Job

(SD)  Storage Device Name [1-14 Chars]
(FD)  File Driver [Named/Remote]
(ER)  Error Reporting [Yes/No]
```

- (SD) Specifies the physical device name of the storage device containing the configuration file for the Downloader Job.
- (FD) Specifies the file driver for the storage device specified by the SD parameter.
- (ER) Specifies whether you want the downloader to log any errors encountered during the download process.

ATCS/279/ARC Server Job Screen

This screen applies to Multibus II systems.

```
(ATCJ)          MBII ATCS/279/ARC Server Job

(SDN)  Serial Device Name [0-14 Chars]
(ODS)  Object Directory Size [0-3840]
(PMI)  Pool Minimum [60H-0FFFFFFFH]
(PMA)  Pool Maximum [60H-0FFFFFFFH]
(MOB)  Maximum Objects [1-0FFFFFFH]
(MTK)  Maximum Tasks [1-0FFFFFFH]
(MPR)  Maximum Priority [0-255]
(PV)   Parameter Validation [Yes/No]
(TP)   Task Priority [0-255]
(SSJ)  Stack Size [0-0FFFFFFH]

(MSW)  Maximum System Windows [0-0FFH]
(MDW)  Maximum Debug Windows [0-0FFH]
```

- (SDN) Specifies the name of the serial device used by the ARC Server for communication.
- (ODS) Specifies how many entries can exist in this job's object directory, according to the total number of windows supported.
- (PMI) , (PMA)
Specifies in 16-byte paragraphs the initial, minimum and maximum sizes of this job's memory pool.

- (MOB) Specifies how many objects can exist simultaneously in the job.
- (MTK) Specifies how many tasks can exist simultaneously in this job.
- (MPR) Specifies in decimal the highest priority of tasks in this job.
- (PV) Specifies whether to include parameter validation for Nucleus system calls made by tasks in this job, if you included system-level parameter validation.
- (TP) Specifies in decimal the priority of this job's initialization task.
- (SSI) Specifies in bytes the size of the initialization task's stack segment, according to your segmentation model.
- (MSW) Specifies how many windows the 279 portion of the ATCS/279/ARC Server Jobs can provide to other hosts in the MB II system.
- (MDW) Specifies how many windows the 279 portion of the ATCS/279/ARC Server can provide for debug monitors running on other MB II processors in the system.

ATCS/450 Server Job Screen

This screen applies to Multibus II systems.

(ATC50)	MBII ATCS/450 Server Job
(ODS)	Object Directory Size [0-3840]
(PMI)	Pool Minimum [60H-0FFFFFFFH]
(PMA)	Pool Maximum [60H-0FFFFFFFH]
(MOB)	Maximum Objects [1-0FFFFFFH]
(MTK)	Maximum Tasks [1-0FFFFFFH]
(MPR)	Maximum Priority [0-255]
(PV)	Parameter Validation [Yes/No]
(TP)	Task Priority [0-255]
(SSI)	Stack Size [0-0FFFFFFH]
(TPA)	8254 Timer Port [0-0FFFFFFH]
(CIL)	Clock Interrupt Level [Encoded Level]
(CN)	Timer Counter Number [0,1,2]
(CF)	Clock Frequency [0-65535 khz]
(TPS)	Timer Port Separation [0-0FFH]

- (ODS) Specifies in decimal how many entries can exist in this job's object directory.
- (PMI) , (PMA)
Specifies in 16-byte paragraphs the initial, minimum and maximum sizes of this job's memory pool, according to the total number of windows supported.

- (MOB) Specifies how many objects can exist simultaneously in this job.
- (MTK) Specifies how many tasks can exist simultaneously in this job.
- (MPR) Specifies in decimal the highest priority of tasks in this job.
- (PV) Specifies whether to include parameter validation for Nucleus system calls made by tasks in this job, if you included system-level parameter validation.
- (TP) Specifies in decimal the priority of this job's initialization task.
- (SSI) Specifies in bytes the size of the initialization task's stack segment, according to your segmentation model.
- (TPA) Specifies the base port (lowest valued) address of the 8254 PIT used by the ATCS/450 Server.
- (CIL) Specifies the encoded interrupt level of the timer use by the ATCS/450 Server.
- (CN) Specifies the number of the 8254 PIT used by the ATCS/450 Server.
- (CF) Specifies the frequency in kilohertz of the clock input to the timer used by the ATCS/450 Server.
- (TPS) Specifies the timer port separation of the 8254 PIT used by the ATCS/450 Server.

MSA BootServer Job Screen

This screen applies to Multibus II systems.

```

(BSJ)          MSA BootServer Job

(IJD)  I/O Job Default Prefix [1-12 Chars]
(DU)   Default User [1-12 Chars]
(PMI)  Pool Minimum [60-0FFFFFFFH]
(PMA)  Pool Maximum [1000H-0FFFFFFFH]
(PV)   Parameter Validation [Yes/No]
(TP)   Task Priority [0-255]
```

- (IJD) Specifies the logical name for the default prefix for the Bootserver Job.
- (DU) Specifies the default user object for the Bootserver Job.
- (PMI) , (PMA) Specifies in 16-byte paragraphs the initial, minimum and maximum sizes of this job's memory pool.
- (PV) Specifies whether to include parameter validation for Nucleus system calls made by tasks in this job, if you included system-level parameter validation.
- (TP) Specifies in decimal the priority of the Bootserver task.

FPI Server Job Screen

This screen applies to Multibus II systems.

(FPIJ)	FPI Server Job
(STP)	Server Task Priority [0-255]
(ODS)	Object Directory Size [0-3840]
(PMI)	Pool Minimum [60H-0FFFFFFFH]
(PMA)	Pool Maximum [60H-0FFFFFFFH]
(MOB)	Maximum Objects [1-0FFFFFH]
(MTK)	Maximum Tasks [1-0FFFFFH]
(MPR)	Maximum Priority [0-255]
(PV)	Parameter Validation [Yes/No]
(SSI)	Stack Size [0-0FFFFFH]
(FIL)	Front Panel Interrupt Level [Encoded Level]

- (STP) Specifies in decimal the Front Panel Interrupt Server's message task priority.
- (ODS) Specifies in decimal the maximum number of entries in this job's object directory.
- (PMI) , (PMA)
Specifies in 16-byte paragraphs the initial, minimum and maximum size of the FPI Job's memory pool.
- (MOB) Specifies how many objects can exist simultaneously in this job.
- (MTK) Specifies how many tasks can exist simultaneously in this job.
- (MPR) Specifies in decimal the highest priority of tasks in this job.
- (PV) Specifies whether to include parameter validation for Nucleus system calls made by tasks in this job, if you included system-level parameter validation.
- (SSI) Specifies in bytes the size of the initialization task's stack segment, according to your segmentation model.
- (FIL) Specifies the encoded hardware interrupt level for the FPI.

Soft-Scope Kernel Job Screen

```
(SSKJ)      Soft-Scope Kernel Job

(CMM)      Communication Method [Local/Serial/Network]
(SCP)      Serial Communication Port [1=Com1 2=Com2]
(SBR)      Serial Baud Rate [400-57600]
(NCP)      Network Communication Port in hex [0-0FFFF no H]
(WDN)      Pathname To Working Directory [1-45 Chars]
```

- (CMM) Specifies the communication method used to interface to the kernel.
- (SCP) Specifies the serial communications port used to communicate with the Soft-Scope kernel.
- (SBR) Specifies the Soft-Scope serial communication port's baud rate.
- (NCP) Specifies the TCP/IP port number for the UDP protocol used to listen for messages.
- (WDN) Specifies the target pathname into which Soft-Scope loads an application.

Third Party Tools-Generated Job Screen

Use this screen to define the environment of a first-level job that you create with non-Intel tools. To create a first-level job, you must use segmented tools, not flat model. This screen allocates memory and GDT slots to the job and creates an *.icu* file containing the address used by the **rmxloc** utility.

See also: C Compiler-specific Information, *Programming Techniques and Aedit Text Editor* manual

```
(TPUJ)      Third Party Tools-Generated User Job

(SEQ) Job Sequence [0-3]
(ODS) Object Directory Size [10-3840]
(PMI) Pool Minimum [100H-0FFFFFFFH]
(PMA) Pool Maximum [400H-0FFFFFFFH]
(MOB) Maximum Objects [3-0FFFFH]
(MTK) Maximum Tasks [2-0FFFFH]
(MPR) Maximum Priority [0-255]
(TPR) Task Priority [0-255]
(SSS) Stack Size [800H-0FFFFH]
(SGS) Static GDT Slots for Application [3-49]
(BAA) Base Address for Application Image [0-0FFFFFFFH]
(SIA) Size of Application Image [0-0FFFFFFFH]
```

- (SEQ) Specifies the order in which the Application Starter Job (ASJ) is initialized.
 - 0 Initialize ASJ before the BIOS initializes.
Specify a 0 if the application uses only Nucleus/Kernel calls.
 - 1 Initialize ASJ before the EIOS initializes.
Specify a 1 if the application uses only BIOS and/or asynchronous Loader calls.
 - 2 Initialize ASJ before the Human Interface initializes.
Specify a 2 if the application uses EIOS and/or synchronous Application Loader calls.
 - 3 Initialize ASJ after the Human Interface.
Specify a 3 if the application requires loadable jobs to have finished initialization before the application starts to run.
- (ODS) Specifies the maximum number of entries in the object directory for your application job and the Application Starter Job.
- (PMI) Specifies the initial amount of memory (in 16-byte paragraphs) the Application Starter/User Job will be allocated.

- (PMA) Specifies the maximum amount of memory (in 16-byte paragraphs) the Application Starter/User Job can have. If the amounts specified for PMI and PMA are the same, then borrowing is not allowed.
- (MOB) Specifies the maximum number of objects that can exist simultaneously in the Application Starter/User Job.
- (MTK) Specifies the maximum number of tasks that can exist simultaneously in the Application Starter/User Job.
- (MPR) Specifies the maximum allowable priority of tasks created in the Application Starter/User Job.
- (TPR) Specifies the priority of the initial task of both the Application Starter Job and the User Job.
- (SSI) Specifies the stack size used to create the initial task of the User Job.
- (SGS) Specifies the number of static GDT slots required by the User Job.
- (BAA) Specifies the base address in physical memory at which the application image will be located.
- (SIA) Specifies the size of the application image.

User Jobs Screens

User Jobs Screen

Each User Jobs screen represents a different user job. The Nucleus can create up to 32 user jobs. User jobs are created after all other first-level jobs and all I/O jobs, but before the HI and any of its jobs. The order in which you define your user jobs is the order in which the root task initializes them.

If you do not configure the HI, define at least one user job or one I/O job to be created during initialization.

The parameters on this screen are the same as those in the **create_job** system call.

(USERJ)	User Jobs
(NAM)	Job Name [0-14 Chars]
(SEQ)	Job Sequence [Before/After]
(ODS)	Object Directory Size [0-3840]
(PMI)	Pool Minimum [20H-0FFFFFFFH]
(PMA)	Pool Maximum [20H-0FFFFFFFH]
(MOB)	Maximum Objects [1-0FFFFFFH]
(MTK)	Maximum Tasks [1-0FFFFFFH]
(MPR)	Maximum Priority [0-255]
(EHS)	Exception Handler Entry Point [1-31 Chars]
(EM)	Exception Mode [Never/Prog/Environ/All]
(PV)	Parameter Validation [Yes/No]
(TP)	Task Priority [0-255]
(TSA)	Task Entry Point [1-31 Chars]
(VAR)	Public Variable Name [0-31 Chars]
(SSA)	Stack Segment Address [SS:SP]
(SSI)	Stack Size [0-0FFFFFFH]
(NPX)	Numeric Processor Ext. Used [Yes/No]

- (NAM) Specifies a name that will identify a particular user job screen.
- (SEQ) Specifies the whether this user job must start before or after EIOS initialization.
- (ODS) Specifies how many entries can exist in this user job's object directory.
- (PMI), (PMA)
Specifies in 16-byte paragraphs the minimum and maximum size of the user job's memory pool.

- (MOB) Specifies how many objects can exist simultaneously in this user job.
- (MTK) Specifies how many tasks can exist simultaneously in this user job.
- (MPR) Specifies in decimal the highest priority of tasks in this user job.
- (EHS) Specifies the PUBLIC name of an exception handler procedure, other than the default system exception handler specified for the Nucleus.
- (EM) Specifies the exception mode of the exception handler for this user job.
- (PV) Specifies whether the Nucleus performs parameter validation for all Nucleus system calls made in this user job.
- (TP) Specifies in decimal the priority of this job's initialization task.
- (TSA) Specifies the PUBLIC name of the user job's entry procedure.
- (VAR) Specifies the PUBLIC name of any of the user job's public variables.
- (SSA) Specifies the address of the initialization task's stack.
- (SSI) Specifies in bytes the size of the initialization task's stack segment; set according to your segmentation model.
- (NPX) Specifies whether the I/O job's initial task contains floating-point instructions.

User Modules Screen

Use this screen to specify the user modules included by the builder in the application system.

<pre>(USERM) User Modules Module = 1-55 characters [1] Module =</pre>

Module

Specifies the pathname for the object code that has been bound for the application jobs defined on the IOJOB screen, the USERJ screen or for the OS extensions defined on the OSEXT screen.

The order of the user module pathnames is not significant.



Error Messages

A

This appendix contains descriptions of all ICU-generated error and warning messages. There are five categories of messages:

- Invocation messages
- Interactive errors and warnings
- ICU internal errors
- System generation errors and warnings
- iRMX OS internal errors

Invocation messages appear instead of the menu when errors occur while starting the ICU.

Interactive error messages indicate conditions that are not allowed by the ICU. These errors occur during the ICU session, in command or screen-editing modes. All error messages are preceded by the characters `*** ERROR`.

Interactive warning messages indicate conditions that are allowed during the ICU session, but may cause problems later. All warning messages are preceded by the characters `*** WARNING`.

ICU internal errors usually occur if the screen master or template files are corrupt.

System generation errors and warnings occur during the execution of the generation submit file. These can be compiler, binder, or builder errors and warnings.

While running the ICU, if you encounter any error messages not shown in this appendix, you are probably experiencing iRMX OS internal error conditions. These are indicated by a 4-digit condition code followed by a short error message such as:

```
0026: E_FILE_ACCESS
```

See also: [Condition code master list](#), [System Call Reference](#)

Invocation Messages

If the ICU does not present the main menu, various messages can appear. If you invoke the ICU with incorrect command syntax, this message appears:

```
*** INVALID INVOCATION ***
USAGE:  ICU386  [input-file TO] output-file
```

This means you tried to invoke the ICU without an input definition file or with invalid filename(s):

```
*** ERROR - FILE: <file_name> IS NOT VALID
```

This means you tried to invoke the ICU with a corrupted definition file, or a file that is not a definition file.

Version Control Messages

On invocation the ICU validates file version numbers for *icu386.scm*, *icu386.tpl*, and each definition file. Each of these files has Intel, Update, and User version numbers. The Intel version number changes whenever Intel upgrades the ICU to support new releases. The Update version number changes when an OS update modifies the ICU. The User version number changes whenever you add a device driver using the UDS and ICUMRG utilities.

This means there is an inconsistency in the version numbers of either *icu386.scm* or *icu386.tpl*:

```
***ERROR-INCONSISTENCY IN THE VERSION OF THE INTERNAL ICU FILES
Versions:                Intel    Update    User
    ICU386.SCM           <Intel> + <Update> <User version>
    ICU386.TPL           <Intel> + <Update> <User version>
```

This means there is an inconsistency in the definition file:

```
*** WARNING - DEFINITION FILE VERSION IS NOT CORRECT.
                                           Intel    Update    User
ITS VERSION IS:                          <the inconsistency>
VERSION EXPECTED:                         <correct version>
```

Automatic Restore and Update Messages

If the ICU needs to restore from the file in order to use it, you will be prompted:

```
Do you want to restore from the file? y/[n]
```

If you answer No, the ICU stops executing. If you answer Yes, the ICU will start a restore process.

See also: Restoring a definition file, in this manual

If the definition file version number is higher than the ICU version number, you are probably using the wrong version of the ICU. In this case, the ICU displays this warning before the restore prompt:

```
*** WARNING - The Definition File version is NEWER
```

If the ICU is able to use the file without restoring, it prompts with this message:

```
Do you want to update the file? y/[n]
```

If you answer No, the ICU stops executing. If you answer Yes, the ICU will update the file. This is a simple process that requires no further input.

Interactive Error and Warning Messages

These are the possible interactive error and warning messages. Some of the messages have descriptions; the others are self-explanatory when viewed in context with the error condition. Values inside of < > are variables filled in by the ICU.

Interactive Error Messages

A filename is expected

You entered something other than a valid filename.

A prefix of a legal string expected

Address expected

You entered something other than a valid address.

Allowed only in MBII systems

The screen specified is not allowed unless the BUS parameter indicates MB II architecture.

BACKUP FILE <filename> IS NOT VALID

A restore was attempted from an invalid backup file or a file that is not an ICU backup file.

Cannot insert new instance; limit has been reached

You have exceeded the maximum allowable number of entries in a repetitive screen.

Data of the screen <abbreviation> was not restored.

This message appears in the restore log file.

Device name: <abbreviation>, is not unique.

The device abbreviation you specified already exists in a driver screen.

Erroneous delimiter

You used an invalid delimiter in a repetitive screen entry (other than a comma).

Illegal field name: <abbreviation>

The parameter abbreviation is invalid; it is not defined in the screen master file.

Illegal input, number expected

You entered something other than a valid number.

Illegal interrupt level

You entered an interrupt level that is not supported by your hardware or you specified an encoded interrupt level (as a system call would expect).

Illegal line number

In a repetitive screen, you specified a line number that is out of range.

In screen: <abbreviation>, DINFO : <device>, is missing.
In screen: <abbreviation>, two DINFOs have the same name.
In screen: <abbreviation>, two UINFOs have the same name.
Invalid Bits per Pixel, BPP Reset to Default Value.

The graphics display hardware does not support the value specified.

Invalid Command. To Delete enter: <abbreviation>

You entered a line number with the **delete** command on a repetitive-fixed screen.

Invalid Command. To Delete enter: <number>

You did not enter a line number with the **delete** command on a repetitive screen.

Invalid input

You entered a command or value at the screen prompt that does not apply to this screen.

Invalid input_or: Invalid Command. To Insert enter: number= values

You tried to use the **insert** command on a repetitive screen without specifying any parameters.

iRMX-NET also configured into the system.

Line number expected.

Memory areas for SYS and FSM overlap.

You specified values that cause system and free space memory spaces to overlap.

Min memory greater than max memory. Is it OK ?

A starting address is higher than an ending address.

Missing quote.

No user modules defined for I/O jobs.

No user modules defined for User jobs.

Number expected or number too large.

Number is not within its range.

Offset in address is not a number.

One and Only One Primary Must Be Specified. Enter <CR> to Continue.

OS Extension slot number is bigger than number of slots.

You specified a GDT slot for an OS extension that is outside the allowable limits.

OS Extension slot number is contained in the reserved area.

You specified a GDT slot for an OS extension that collides with a reserved GDT slot.

(SRV) or (CON) must be YES. Abort iRMX-NET configuration?

String too long.

The control character cannot be a UDI delimiter.

The field is Req; cannot be changed.

The first number must be SMALLER than the second.

The line entered overlaps.

The selector is not within its range.

To Insert enter: line_number= new_values.

Two OS Extensions have the same slot number.

Upper limit is smaller than the lower. Is it OK?

(WV) indicates EIOS, EIOS is not configured. Is it OK ?

Yes or No expected.

Interactive Warning Messages

Background Color Exceeds Bit Depth.

The graphics display hardware does not support the color value specified.

BIOS not configured into the system.

Change GCN to T546_CC.

DEFINITION FILE USER VERSION IS NOT CORRECT.

Do not include this logical name, it is already included.

DOS Extender Sub-system not chosen.

EIOS not configured into the system.

Foreground Color Exceeds Bit Depth.

The graphics display hardware does not support the color value specified.

HI needs at least 25 directory entries.

If serial drivers are configured system generation will fail.

If terminal I/O required, (BIOS) parameter (TSC) must be "Yes".

iNA MIP Driver Job also configured into the system.

IP is not supported.

IRMX-NET also configured into the system.

No AFA User (AFAU) configured with IRMXNET Server.

No user modules defined for OS Extensions.

NOGEN causes loss of NET configuration information.

NULL1 is not supported.

Problem accessing <filename> This warning does NOT affect ICU operation.

Remote File Access was not chosen.

SDB not configured into the system.

The excluded GDT slots are out of GDT limit. Is it OK?

The screen requested cannot be displayed.

Valid only for MBII Systems.

You should change NPX parameter in HARDWARE screen.

Internal ICU Errors

If during execution the ICU encounters an internal error such as the screen master file or the template file being corrupted, it displays this message:

```
*** ICU Internal Error - <number[,s]>
```

Where:

```
<number[,s]>
```

Can be either one number or two numbers separated by a comma.

The numbers represent an internal code for the ICU and are not meaningful for you. Internal ICU errors rarely occur, but if you should receive this error message, follow these guidelines.

1. First, assume your definition file has become corrupted, and try running the ICU again with a new definition file.
2. If Step 1 is not the solution, try running the ICU with a new screen master file and a new template file. Original versions of these files are kept in the directory *:config:default*.
3. If neither of these solves the problem, contact your Intel sales office.

System Generation Error and Warning Messages

No errors should occur during the generation of your system. However, the Builder (*bld386*) can generate warnings or errors when you execute the generation submit file.

See also: *ASM386 Assembly Language Reference, Intel386 Family Utilities User's Guide* for other error descriptions.

Builder Warnings

You can expect these Builder warnings during generation. Warning 119 always occurs when you generate an iRMX for Windows or iRMX for PCs system. Ignore either warning; they do not interfere with a successful generation.

```
WARNING 269: <line-number>, NEAR '<identifier>', SEGMENT SIZE REDUCED
```

```
WARNING 119: OVERLAP BETWEEN SEGMENTS OR TABLES
```

```
LOW ADDRESS: <absolute hex address>
```

```
HIGH ADDRESS: <absolute hex address>
```

Builder Errors

In addition to warning messages, the Builder can return error messages. Do not ignore error messages. They indicate serious problems which prevent the successful generation of your system. These errors can occur:

```
ERROR 117: INVALID OBJECT FILE
```

```
FILE: file name
```

```
MODULE: module name
```

Possible causes for this error include:

- The file is corrupt.
- Another part of the OS received errors when it generated. Check the generation log file for this possibility.
- Your application object module specified in the error has an invalid format. This condition can occur because of a translator or linker error, or by using loadable or bootloadable modules as input. Try re-translating the source files with 80386 translators to create linkable input modules for *bld386*, relinking with *bnd386* if necessary.

ERROR 118: INPUT SEGMENTS EXCEED TARGET MEMORY

There is not enough system memory for the OS image, as opposed to Free Space memory. Adjust the MEMS screen to include more memory and reduce the Free Space memory on the MEMF screen by the same amount. Then regenerate the system.

WARNING 128: SPECIFIED MODULE NOT FOUND IN INPUT FILE

FILE: *file name*

MODULE: *module name*

The specified module was on the input list but was not found in the associated file. This can occur when you define a first-level job or I/O job but do not specify an object module on the user module (USERM) screen.

ERROR 131: REFERENCE TO UNRESOLVED EXTERNAL SYMBOL

FILE: *file name*

MODULE: *module name*

REFERRING LOCATION: *location*

REFERENCED LOCATION: *target*

Input modules do not contain a public definition identified by *target*, which is referred to as an external symbol in the segment named by *location*. The external declaration occurs in *module name* of *file name*. Check the *bnd386* output of the linker phase for unresolved symbols in your application module(s).

ERROR 137: BAD SELF RELATIVE REFERENCE

FILE: *file name*

MODULE: *module name*

REFERRING LOCATION: *location*

REFERENCED LOCATION: *target*

The specified *target* and *location* are not in the same segment. The indicated *location* is in *module name* in *file name*. This error can be caused by a translator error or by using erroneous ASM386 code.

ERROR 158: REFERENCE TO AN EMPTY SEGMENT

FILE: *file name*

MODULE: *module name*

SEGMENT: *segment name*

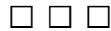
An application module contains a reference to an empty segment identified by *segment name*. Ensure that the reference is correct and reinvoke *bld386* with different input modules if necessary.

ERROR 164: INPUT HAS TWO MODULES WITH SAME NAME
FILE: *file name*
MODULE: *module name*

An application module has the same name as an internal OS module. Rename your module and regenerate the system.

ERROR 192: NO SPACE FOR SEGMENT - BASE NOT SET
SEGMENT: *??_CODE*

Not enough space has been allocated for system memory. The problem must be corrected. *??_CODE* is the name of the segment that did not get added to the bootfile because of a lack of system memory. You can add up the size of each segment that was excluded and determine how much additional system memory to allocate and how much Free Space Memory to allocate. Adjust the MEMS screen to include more memory and adjust the MEMF screen downward by the same amount.



Development Environment

B

This appendix outlines the required development environments for ICU-configurable systems. This includes recommended hardware platform and software requirements.

Hardware Platforms

This version of the ICU operates on the iRMX target system or a PC development system running iRMX for Windows or iRMX for PCs.

iRMX Target System

The iRMX target system can be a Multibus I or II system consisting of:

- CPU board and memory (e.g. SBC 486/12S)
- System chassis and cardcage
- Mass storage devices (hard disk or tape drive)
- Video display terminal

The iRMX target system has the iRMX III OS installed on it, as preconfigured for your basic hardware. You then run the ICU to create a bootable version of the iRMX OS that includes your application jobs and device support.

PC Target System

The PC must have iRMX For Windows or iRMX for PCs installed on it. In addition, you must install the PC-hosted ICU Development Environment.

To generate a Multibus system on a PC, create the new version of the OS, specifying the hardware parameters and networking environment of the Multibus target system. Generate the target definition files on the development system. Use your development network or flexible diskettes to install the generated system on the target system. Test the target system and regenerate it on the development system if necessary.

If the target system is a PC system, use the **loadrmx** command to boot the new system, then test and regenerate if necessary.

Software Requirements

To execute the generation submit file, you must have these Intel compilers and utilities installed on your development system:

- ASM386 and ASM286
- BND386 and BND286
- BLD386

The iRMX installation process places these in their proper directories for the generation process to execute without problems.

Testing and Debugging Tools

For debugging and testing, you can use the Soft-Scope Debugger, and/or iRMX System Debuggers (SDB and/or SDM) running on your target system. For loading and debugging of the bootable object using another system, you can use the I²ICE In-Circuit Emulator.

See also: *Programming Techniques and AEDIT Text Editor*



Using the ICU to Configure Custom Device Drivers

C

For your custom driver to work in an ICU-configurable system, you must define the device-specific procedures as reentrant, public procedures, and compile them using the `rom` and `compact` controls. Assembly language routines must follow the conditions and conventions used by the `compact` control. In particular, the procedures must function in the same way as high-level language procedures.

See also: *ASM386 Macro Assembler User's Guide*;
iC-386 Compiler User's Guide;
PL/M-386 Programmer's Guide

This chapter explains how to use the OS-supplied tools UDS and ICUMRG that support adding your custom device drivers to the ICU. Before using these tools, you must:

- Assemble or compile the code for each driver you have written.
- Put the resulting object modules for terminal drivers in a single library, such as *terminal.lib*.
- Put the resulting object modules for random/common/custom drivers in a single module, such as *driver.lib*.

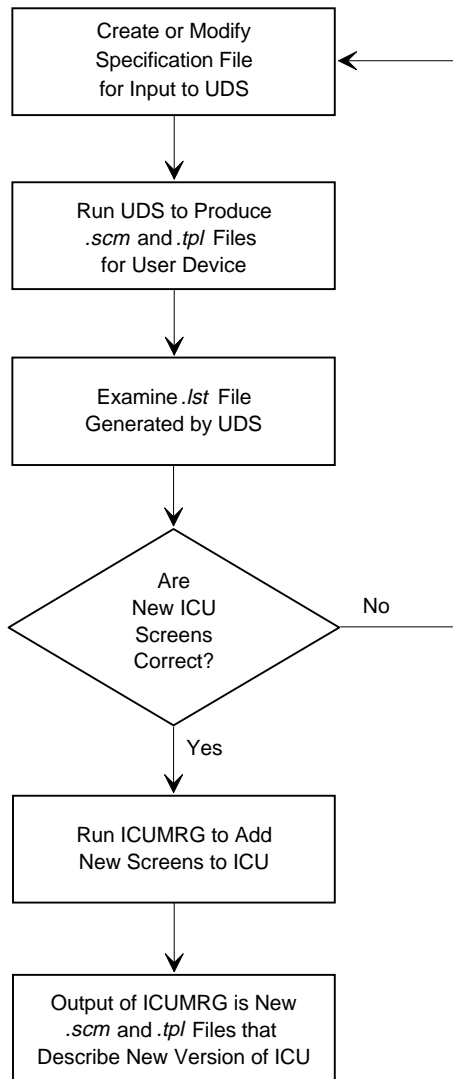
Adding Drivers with the UDS and ICUMRG Utilities

The two utilities are UDS (User Device Support) and ICUMRG (ICU Merge).

- UDS transforms files of screen specifications into files that are compatible with the ICU.
- ICUMRG merges the new files into the ICU.

With these utilities, you can add screens so that configuring your driver involves just running the ICU and answering the appropriate questions. You can add information about devices, units, and device-unit screens for as many of your custom device drivers as you wish. Then the ICU builds the proper DUIB, DINFO, and UINFO structures.

Figure C-1 shows a flowchart overview of using these utilities. The next sections describe the utilities in detail.



W-2774

Figure C-1. Adding Drivers with UDS and ICUMRG

UDS Utility

UDS lets you set up a device information screen, a unit information screen, and a device-unit information screen for your custom driver. The steps are:

1. Set up the screens by placing information in a file that the UDS reads.
2. Choose from a set of standard screens. For example, when describing a device information screen, you can choose from three terminal support screens, two random access support screens, and a general screen.
3. Add auxiliary lines to the device information and unit information screens. This allows your device-specific information to be entered during configuration.

By choosing the appropriate screens and adding the correct number of auxiliary lines, you can set up the ICU to configure almost any device driver. Depending on the number of auxiliary fields defined, you can provide the new auxiliary fields with descriptive names.

After you create the input file, the utilities do these steps:

1. Using the input file you provide, the UDS creates two files that define the new screens. These files have extensions *.scm* and *.tpl*.
2. The ICU Merge Utility can merge these new files with the ICU.
3. The UDS also produces a listing file that has a *.lst* extension; the list file shows how the screens will look when added the ICU.

Creating the Input File for UDS

The UDS includes two input file templates you can modify to suit your application needs:

- The file *templ_1.uds* is an example of a basic user input file; it contains no auxiliary help fields.
- The file *templ_2.uds* is a complete input file and contains examples of most auxiliary fields.

You must add each user device driver separately, because a UDS input file can add only one driver.

Before invoking UDS, you must create an input file that defines how the ICU screens for your driver should look. Figure C-2 shows the format of that input file. The information in brackets ([]) is not part of the input file; it simply describes the lines of the file. The *xxxx* characters mean you must fill in a value. The paragraphs after the figure describe the individual lines of the input file.

```

#version = xxxx          [1-4 character version number]
#name = xxxx             [1-25 character name]
#abbr = xxx              [1-3 character abbreviation]
#driver = x               [driver type value, from 1 to 7]
#device                  [start of device information]
#dev_aux = xx            [number of auxiliaries, from 0 to 20]
#d01 = 'parameter name' [1-41 character parameter name, in quotes]
d01 help information     [0-1024 character help information]
                        [Max of 1024 chars, help msgs are required]
.
.                        [names and help information for other]
.                        [auxiliary parameters]

#end                     [end of device information]
#unit                    [start of unit information]
#unit_aux = xx           [number of auxiliaries, from 0 to 20]
#u01 = 'parameter name' [1-41 character parameter name, in quotes]
u01 help information     [0-1024 character help information]
                        [Max of 1024 chars, help msgs are required]
.
.                        [names and help information for other]
.                        [auxiliary parameters]

#end                     [end of unit information]
#duib
#duib_aux = 0
#end                     [end of device unit information]

```

Figure C-2. Syntax of UDS Input File

#version Specify a new 1- to 4-character user version number used to determine whether the definition files are current. Use consistent version numbers to keep track of the latest version of the ICU.

When you invoke the ICU using an existing definition file, the ICU checks the version number of the definition file against the version number of the master *.scm* and *.tpl* files. If they are inconsistent, the ICU displays the differing version numbers and asks if you want to update the file. The version number that the ICU displays is built from the value you specify here, plus the date and time on which you run the ICUMRG utility.

#name Specify a 1- to 25-character name of the new driver.

#abbr Specify the 1- to 3-character abbreviation used to form screen names and abbreviations for all three driver screens:

Screen Abbreviation	Screen Name
D_<abbr>	<name> Driver
U_<abbr>	<name> Unit Information
I_<abbr>	<name> Device-unit Information

If you enter `abbr ABC` and `name High Speed ABC`, your screen abbreviations would be `D ABC`, `U ABC`, and `I ABC`. The screen names would be `High Speed ABC Driver`, `High Speed ABC Unit Information`, and `High Speed ABC Device-unit Information`.

#driver Specify the kind of driver this is and the kind of screens to display:

Value	Driver
1	Terminal Support driver with one interrupt level
2	Terminal Support driver with two interrupt levels
3	Interrupt-less MB I and MB II Full Message-based Terminal support driver
4	Interrupt-driven Random Access Support and common drivers
5	Multibus II Full Message-based Random Access devices
6	Reserved
7	General driver

#device Specify the start of the information that applies to the device information screen. The information continues until an `#end` field appears.

`#dev aux` Specify the number of auxiliary parameters on the device information screen; range is 0 to 20. If the value is 4 or less for terminal support or random access devices, or 14 or less for general devices, each auxiliary parameter is displayed on a separate line, and the parameter names you specify in the `#d` fields are also displayed. If more auxiliary parameters are specified, the parameters are displayed on the device information screen in rows of five parameters each. Then there is no room for the parameter names, and if any are entered, the UDS ignores them.

When the ICU generates a system, it gets auxiliary parameters from the device information screen. The ICU places random/common device parameters in `?icdev.a38` files and places terminal parameters in `?itdev.a38` files, immediately after the Device Information structure. The `?` means the character can vary.

`#d01` Each field (fixed `#d01` through `#d20`) identifies auxiliary parameters in the device information table. If a parameter fits on a single line, the 1- to 41-character 'parameter name' you specify (surrounded by quotes) will be included on the menu.

Even if your table contains too many auxiliary parameters to include a parameter name for each, you must specify the `#d` field for a parameter if you plan to add help information for that field. Then, you can specify the `#d` field without a parameter name:

```
#d03 =
```

You can also modify the parameter names and help information for the standard parameters that normally appear on the device information screen you selected. For example, if you are setting up a random access device and you want to modify the parameter name and help information for the `DS` field, you could include this information in the input file:

```
#ds = 'Size of Device Local Data [0-0FFFFH]'
```

`d01 help` Specify up to 1024 characters of help information; help information is required. The UDS assumes that the help information ends when a `#` appears at the start of a subsequent line or when the maximum character count is reached.

`#end` Designates the end of the device, unit, or device-unit information.

`#unit` Specify the start of the information that applies to the unit information screen. The information continues until an `#end` field appears.

`#unit aux` Specify the number of auxiliary parameters on the unit information screen; range is 0 to 20. If 10 or less, each auxiliary parameter is displayed on a separate line with the parameter names you specify. With more than 10, the parameters are displayed two to a row, with no room for parameter names.

When the ICU generates a system, it places the auxiliary parameters from the unit information screen in the `?itdev.a38` or `?icdev.a38` files it creates, immediately after the Unit Information structure. The file that is actually altered depends on the type of device: `?icdev.a38` for common and random devices, `?itdev.a38` for terminal devices.

`#u01` Each of fields (`fixed #u01` through `#u20`) identifies auxiliary parameters in the unit information screen. If each of the auxiliary parameters fits on a single line, the 1- to 41-character parameter name you specify here as 'parameter name' (surrounded by quotes) will be included on the menu to describe the auxiliary parameter.

You can also use similar fields to change the parameter names and help information for any of the standard parameters of the unit information screen.

`u01 help` Specify up to 1024 characters of help information for the parameters; help information is required for all parameters. The UDS assumes that the help information ends when a `#` appears at the start of a subsequent line.

`#duib` Specify the start of the information that applies to the device-unit screen. The device-unit information continues until a `#end` field is encountered.

`#duib aux` The UDS does not support any auxiliary parameters; set this field to:

```
#duib_aux=0
```

⇒ **Note**

All auxiliary parameter fields (`#dev_aux`, `#unit_aux`, `#duib_aux`) must be WORD values. The UDS will not accept DWORDs and will write BYTE values as WORDs.

Device Information Screens

This section lists the different Device Information Screens that the UDS can generate. When adding support for your own driver, choose the screen that matches the way the driver expects the DINFO table to look. All screens in this group can also contain auxiliary parameter lines. Set up auxiliary parameter lines if none of the Device Information Screens listed contain enough fields to support the needs of your driver.

The meanings of fields in these screens are the same as the fields in the DINFO table.

See also: DINFO table structure in *Driver Programming Concepts*

- One-Interrupt Terminal Device Information
- Two-Interrupt Terminal Device Information
- Interrupt-less Multibus I and Multibus II Full Message-based Terminal Device Information
- Multibus I Random Access Device Information
- Multibus II Random Access Device Information
- General Device Information

Unit Information Screens

This section lists the Unit Information Screens that the UDS can generate. You define these screens by placing information into an input file, which the UDS reads. By choosing the appropriate driver type and adding the correct number of auxiliary lines to the driver's screens, you can set up the ICU to handle the configuration of virtually any driver. All screens in this group can contain auxiliary parameter lines. If none of the Unit Information Screens listed contain enough fields to support your driver, set up auxiliary parameter lines.

The meanings of the individual fields in these screens are the same as the fields in the UINFO table.

See also: UINFO table structure, in *Driver Programming Concepts*

- Terminal Support Unit Information
- Random Access Support Unit Information
- General Device Unit Information

Device-Unit Information Screens

This section lists the Device-Unit Information Screens that the UDS generates. When adding support for your own driver, choose the screen that matches the way the driver expects the DUIB to look. None of the screens in this group currently allow auxiliary parameter lines.

The meanings of the individual fields in these screens are the same as the fields in the DUIB.

See also: DUIB in *Driver Programming Concepts*

- Terminal Support Device-Unit Information
- Random Access Device-Unit Information
- General Device-Unit Information

Invoking the UDS Utility

Once you have created an input file that specifies how the screens for your device driver should appear, you are ready to invoke the UDS utility. To do this, ensure that the directory containing the UDS program also contains the UDS database file named *uds.scm*. Then invoke the utility by typing:

```
UDS input_file TO output_file
```

Where:

input_file

The name of the file that contains the information used as input to the UDS utility.

See also: UDS input file in this section

output_file

The name portion of the output files generated by UDS. UDS adds three-character extensions to this name when generating its output files. The two primary output files are *output-file.scm* and *output-file.tpl*. You will use these output files as input to the ICUMRG utility. The other output file is *output-file.lst*, a listing file that shows exactly how the screens will appear when added to the ICU.

Do not name your UDS output files *icu386.scm* or *icu386.tpl*.

For example, suppose you created an input file called *newdriver.txt* and wanted the UDS utility to generate output files called *special.scm* and *special.tpl*. To do this, you would enter:

```
uds newdriver.txt to special
```

Part of the output of the UDS utility are two files with extensions *.scm* and *.tpl* (in the example, *special.scm* and *special.tpl*). These files contain the definitions of the ICU screens for your driver. After running the UDS utility, you will use the ICUMRG utility to add these files to the ICU.

However, before running ICUMRG, examine the listing file (in the example, *special.lst*). This file shows how the Device Information screen, the Unit Information screen, and the Device-unit Information screen will look when added to the ICU. If there is a problem with the appearance of any of these files, you can catch the problem early and rerun UDS, instead of adding incorrect screens to the ICU.

UDS Error Messages

If you make a mistake when creating the files to use as input to UDS, the UDS utility will display an error message.

External File and Memory Type Errors.

The detailed message will be preceded by:

```
*** Error in UDS
*** Cannot Attach Input File
You did not have the proper permission to access the file containing the UDS
instructions.
*** Not enough memory for buffers
Your memory partition is not large enough to permit the UDS utility to run.
*** Cannot Attach UDS SCM File
UDS needs to access a file called uds.scm, but you do not have read access to that
file.
*** Invalid UDS.SCM File
The UDS file uds.scm has been corrupted.
*** Cannot Create New SCM File
UDS cannot create the output file (output_file.scm).
*** Cannot Create New TPL File
UDS cannot create the output file (output_file.tpl).
*** Cannot Create LST File
UDS cannot create the listing file (output_file.lst).
*** I/O Error in File [file-name]
The specified file or directory lacks read or creation permission.
```


UDS Input File Errors.

The detailed message will be preceded by:

```
*** Error in UDS Input File on line <line-number>
```

where <line-number> is where the error occurred in the user input file.

```
*** Missing User Version
```

The required #version statement is missing.

```
*** Illegal Version
```

The #version number in the input file is outside the legal range of 1 to 4 characters.

```
*** Missing User Device Name
```

The required #name field is missing.

```
*** Illegal Device Name
```

The #name identifier is 0 length or is greater than 25 characters in length.

```
*** Missing User Device Abbr
```

The required #abbr identifier is missing.

```
*** Illegal Device Abbr
```

The #abbr value in the user input file is outside the legal range of 1 to 3 characters.

```
*** Missing User Driver Type
```

The required #driver identifier is missing.

```
*** Illegal Driver Type
```

The #driver value is outside the legal range of 1 to 7.

```
*** Missing User Device
```

The required #device identifier is missing.

```
*** Missing Number of Device Auxiliaries
```

The required #dev_aux identifier is missing.

```
*** Missing User Unit
```

The required #unit identifier is missing.

```
*** Missing Number of Unit Auxiliaries
```

The #unit_aux identifier is missing.

```
*** Missing User Duib
```

The #duib identifier is missing.

```
*** Missing Number of DUIB auxiliaries
```

The required #duib_aux identifier is missing.

```
*** DUIB Screen Can Not Have Auxiliary Fields
```

The #duib_aux value in the user input file is set to other than 0.

*** Missing Equal Sign

The equal sign is missing from an identifier that requires one.

*** Line Too Long

A line in the user input file is longer than the allowable 132 characters.

*** Missing Auxiliary Help Message

An auxiliary parameter line was added without its required help message.

*** Auxiliary Line Out of Sequence

Auxiliary parameter lines must be listed sequentially, beginning with line 01.

*** Less Auxiliary Lines than Expected

The number of auxiliary lines is less than the xxx_aux value of the user input file.

*** More Auxiliary Lines than Expected

The number of auxiliary lines is more than the xxx_aux value of the input file.

*** Illegal Input

Extra characters were entered on a line after the valid input.

*** Invalid Abbreviation

The abbreviation for an auxiliary field is outside the legal range of 1 to 3 characters.

*** Abbreviation Not Found

When a standard parameter line or its help message was changed, the abbreviation was entered incorrectly.

*** Number Exceeds Maximum

Dev_aux or unit_aux is greater than 20.

*** Number Expected

A nonnumeric value was entered.

*** Syntax Error

The opening quote on a parameter name line is missing.

*** Do Not Use # Sign in Text

A parameter name contains a pound symbol (#).

*** Do Not Use (Sign in Text

A parameter name contains a left parenthesis "(".

*** Missing End of Text Sign

The closing quote on a parameter name line is missing.

*** Text Line Too Long

A parameter name exceeds 41 characters.

*** Help Message is too Long

The Help message you entered exceeds 1024 characters in length.

*** Field Name Expected

A blank line was detected in the device, unit, or DUIB information.

*** Unexpected eof

The user input file is incomplete.

ICUMRG Utility

After using UDS to generate *.scm* and *.tpl* files for your new driver, use the ICUMRG utility to combine the information in these files with the definitions of all other ICU screens (in the *icu386.scm* and *icu386.tpl* files). Before running ICUMRG, make sure these *icu386.** files reside in the same directory as the ICUMRG command. Then, invoke the ICUMRG utility:

```
ICUMRG input_file TO output_file
```

Where:

`input_file`

The name (minus the extension part) of the *.scm* and *.tpl* files generated by the UDS. For example, if the UDS utility created files called *special.scm* and *special.tpl*, you would specify the name *special* here.

`output_file`

The name (minus the extension part) of new ICU files that ICUMRG will create. For example, if you specified the name *icunew*, the ICUMRG utility will create files called *icunew.scm* and *icunew.tpl*. These new files will contain the complete definition of the ICU, including the screens you just defined for your new driver. By naming the files something other than *icu386*, you can save the previous version of the ICU files. For testing, you can change the name of the ICU executable file to match the base name of the new file (e.g., *icunew*). Then, when you are satisfied with the updated ICU, rename your *icunew*, *icunew.scm*, and *icunew.tpl* test files to their *icu386* counterparts so they match the standard user documentation.

After adding driver support to the ICU, you can configure the drivers almost as you would any OS-supplied drivers:

1. Invoke the ICU and go to the (UDDM) UDS Device Drivers Module screen.
2. Enter the appropriate driver type, (T)erminal or (C)ommon, and the full pathname for the location of the object code for your device driver.
3. After entering the correct value, choose the device you want to configure.
4. Fill in the appropriate values when the ICU displays the Device Information, Unit Information, and Device-Unit Information screens.

UDS Modules Screen in the ICU

```
(UDDM)          UDS   Device Driver Modules
                Module= Driver type , Object code pathname
                   [T/C]      , [1-55 Characters]
[ 1 ] Module=
```

Specify C for common/random/custom drivers and T for terminal drivers.

Place the modules according to type, with all of your terminal modules in one module, and all your common/random/custom drivers in a separate module. For example, 1 = T, terminal.lib, and 2 = C, driver.lib.

⇒ **Note**

Before changing the name of any ICUMRG output files to *icu386.scm* and *icu386.tpl*, save the original files by copying them to other files (such as *icu_old.scm* and *icu_old.tpl*). Although ICUMRG lets you add support for new drivers, once you add that support, there is no way to remove it. If you decide you don't want the ICU to display information about one of your drivers, or you made a mistake when adding information about your driver, you must revert to the original files you saved (or an intermediate version that doesn't contain support for that driver).

Adding Your Driver Without the UDS and ICUMRG

If you don't want to modify the ICU, you can add your custom device driver by doing this:

1. Get the device numbers and device-unit numbers to use in the DUIBs for your devices:
 - a. Use the ICU to configure a system containing all the OS-supplied and ICU-supported user drivers you require.
 - b. Use the **G** command to generate that system.
 - c. Use a text editor to examine the file *?icdev.a38* (the ? means the first letter can vary). This file contains DUIBs for all the device-units defined in your configuration.
 - d. Look for the %DEVICETABLES macro that appears after all the `define_duib` structures. The second and third parameters in that macro list are the next available device-unit number and the device number, respectively. For example, suppose the %DEVICETABLES macro appears:

```
%DEVICETABLES(NUMDUIB,0000CH,005H,003E8H)
```

The next available device-unit number is 0CH and the next available device number is 05H.

- e. Use the next available device number and device-unit number in your DUIBs.
2. Create these:
 - a. A file containing the DUIBs for all device-units you are adding. Use the `define_duib` structures, and place all the structures in the same file. The ICU will include this file when assembling the *?icdev.a38* file.
 - b. A file containing all the device information tables of the random/common/custom type that you are adding. Use the `radev_dev_info` structures for any random access drivers you add. Later, the ICU includes this file when assembling the *?icdev.a38* file.
 - c. If applicable, any random access or common unit information table(s). Use the `radev_unit_info` structures for any random access drivers you add. Add these tables to the file created in step 2b.
 - d. A file containing all the device information tables of the terminal type you are adding. Use a structure similar to the `terminal_device_information` structure for terminal drivers. The ICU will include this file when assembling the *?itdev.a38* file.

- e. If applicable, any terminal unit information table(s). Use a structure similar to `terminal_unit_information` for terminal drivers. Add these tables to the file created in step 2b.
 - f. External declarations for any procedures you write. The procedure names appear in either the DUIB or the DINFO table associated with this device driver. Add these declarations to the file created in steps 2b and 2d.
3. Use the ICU to configure your final system. When doing so:
 - a. Answer yes when asked if you have any device drivers not supported by the ICU.
 - b. As input to the Custom User Devices screen, enter the pathname of your random/common/custom device driver library. This refers to the library built earlier; for example, `:fl:driver.lib`.
 - c. As input to the Custom User Devices screen, enter the pathname of your terminal device driver library. This refers to the library built earlier; for example, `:fl:terminal.lib`.
 - d. Enter these:
 - DUIB source code pathname (the file created in step 2a).
 - Device and Unit source code pathnames (the files created in steps 2b through 2f).
 - Number of user-defined devices.
 - Number of user-defined device-units.

The ICU does the rest.

Figure C-3 contains an example of the Custom User Devices screen. The bold text represents user input to the ICU. In this example:

- `:fl:driver.lib` contains the object code for the random/common/custom drivers
- `:fl:terminal.lib` contains the object code for the terminal driver
- `:fl:duib` contains the source code for the DUIBs
- `:fl:rinfo.inc` contains the source code for the Device and UINFO tables along with the necessary external procedure declarations for the random/common/custom drivers
- `tinfo.inc` contains the source code for the Device and UINFO tables and the necessary external procedure declarations for the terminal driver

The code in the *driver.lib* file supports 1 device with 2 units. The code in *terminal.lib* supports 1 device with 2 units; therefore, the (ND) Number of User Defined Devices [0-0FFH] field equals 2, and the (NDU) Number of User Defined Device-Units [0-0FFH] field equals 4.

```

(USERD)    User Devices
(OPN)     Random Access Object Code Path Name [1-45 Chars/NONE]
                                                NONE
(TOP)     Terminal Object Code Path Name [1-45 Chars/NONE]
                                                NONE
(DPN)     DUIB Source Code Path Name [1-45 Chars/NONE]
                                                NONE
(DUP)     Random Access Device and Unit Source Code Path Name
          [1-4 Chars/NONE]
                                                NONE
(TUP)     Terminal Device and Unit Source Code Path Name
          [1-45 Chars/NONE]
                                                NONE
(ND)      Number of User Defined Devices [0-0FFH] 0H
(NDU)     Number of User Defined Device-Units [0-0FFH] 0H

Terminal Device and Unit Information Names [1-16 Chars]

(N01) NONE      (N02) NONE      (N03) NONE
(N04) NONE      (N05) NONE      (N06) NONE
(N07) NONE      (N08) NONE      (N09) NONE
(N10) NONE      (N11) NONE      (N12) NONE
(N13) NONE      (N14) NONE      (N15) NONE
(N16) NONE      (N17) NONE      (N18) NONE

: OPN = :F1:DRIVER.LIB <CR>
: TOP = :F1:TERMINAL.LIB <CR>
: DPN = :F1:DUIB.INC <CR>
: DUP = :F1:RINFO.INC <CR>
: TUP = :F1:TINFO.INC <CR>
: ND = 2 <CR>
: NDU = 4 <CR>

```

Figure C-3. Example User Devices Screen

Example of Adding an Existing Driver as a Custom Driver

This section illustrates how to create the screens needed for adding the 544A device to your system using the UDS. Because device configuration is complex, the example covers the process in detail.

While reading this example, keep in mind that the code for terminal drivers is in a different segment than the code for random or common drivers. Because of this split in the segments, you must be careful to properly provide the correct `publics`, `extrns`, and `nopublics` except, and also to properly bind the code segments together.

```
(USERD)      User Devices
(OPN) Random Access Object Code Path Name [1-45 Chars/NONE]
                                                    NONE
(TOP) Terminal Object Code Path Name [1-45 Chars/NONE]
                                                    NONE
(DPN) Duib Source Code Path Name [1-45 Chars/NONE]
                                                    DUIB.INC
(DUP) Random Access Device and Unit Source Code Path Name
           [1-45 Chars/NONE]
                                                    NONE
(TUP) Terminal Device and Unit Source Code Path Name
           [1-45 Chars/NONE]
                                                    TINFO.INC
(ND)  Number of User Defined Devices [0-0FFH]          01H
(NDU) Number of User Defined Device-Units [0-0FFH]    04H

      Terminal Device and Unit Information Names [1-16 Chars]

      (N01) DINFO_544A (N02) UINFO_544A   (N03) NONE
      (N04) NONE      (N05) NONE         (N06) NONE
      (N07) NONE      (N08) NONE         (N09) NONE
      (N10) NONE      (N11) NONE         (N12) NONE
      (N13) NONE      (N14) NONE         (N15) NONE
      (N16) NONE      (N17) NONE         (N18) NONE
```

The TOP option was left at NONE in this example because the 544A driver code is already in the driver library *xcmdrv.lib*. If you were adding another module, you would enter the location of the file as a full pathname.

The OPN and DUP options were left at NONE because the driver being configured is a terminal driver, not a random access, common, or custom driver.

You can add combinations of up to 18 Terminal DINFO and UINFO public names in this screen.

Contents of the Duib.inc File Specified in the (DPN) Parameter

Figure C-4 shows the contents of the file whose pathname you supplied in the (DPN) DUIB Source Code Pathname parameter of the User Devices Screen. This assembly-language file provides the information to define how the operating system should interface with the device.

Note the lines with arrows pointing to them. These are the device number and device-unit number for this device, and the numbers were taken from the *?icdev.a38* file:

1. Make sure that the files you start with contain all of the OS-supplied and ICU-supported drivers you require. If you haven't generated such a system, use the ICU to do so before continuing.
2. Use a text editor to examine the file *?icdev.a38* (the ? means that the first letter can vary). You will find all of the DUIBs for your entire system in this file. Scan this file for a line that starts with %DEVICETABLE.
3. %DEVICETABLE is a macro that appears below all of the systems' `define_duib` structures. The second and third parameters in that macro are the next available device-unit and device number, respectively. For example, suppose the %DEVICETABLE macro appears:

```
%DEVICETABLE (NUMDUIB, 0002EH, 008H, 003E8H)
```

In this case, the next available device-unit number is 2EH and the next available device number is 08H.

4. Use these numbers to fill in the two lines of the file identified by the arrows.

At the end of this file are several more lines that should be noted. Be sure to examine the last part of this figure and read the text that goes with it.

```

DEFINE_DUIB <
& 'T2',
& 00001H,
& 0FBH,
& 00,
& 00,
& 00,
& 00,
& 08H, ...<----- Put next available DEVICE NUMBER here
& 0H,
& 2EH, ...<----- Put next available DEVICE-UNIT NUMBER here
& TSINITIO,
& TSFINISHIO,
& TSQUEUEIO,
& TSCANCELIO,
& DINFO_544A,
& UINFO_544A,
& 0FFFFH,
& 0,
& 130,
& FALSE,
& 0H,
& 0
&>
DEFINE_DUIB <
& 'T3',
& 00001H,
& 0FBH,
& 00,
& 00,
& 00,
& 00,
& 08H, ...<----- The DEVICE NUMBER is the same
& 0H,
& 2FH, ...<----- The DEVICE-UNIT number (T3) is equal to the
& TSINITIO,                               DEVICE-UNIT number of 'T2' plus one.
& TSFINISHIO,
& TSQUEUEIO,
& TSCANCELIO,
& DINFO_544A,
& UINFO_544A,
& 0FFFFH,
& 0,
& 130,
& FALSE,
& 0H,
& 0
&>

```

Figure C-4. Computing Device and Device-Unit Numbers

```

DEFINE_DUIB <
& 'T4',
& 00001H,
& 0FBH,
& 00,
& 00,
& 00,
& 00,
& 08H, ...<----- The DEVICE NUMBER is the same
& 0H,
& 30H, ...<----- The DEVICE-UNIT number (T4) is equal to the
& TSINITIO,                               DEVICE-UNIT number of 'T3' plus one.
& TSFINISHIO,
& TSQUEUEIO,
& TSCANCELIO,
& DINFO_544A,
& UINFO_544A,
& 0FFFFH,
& 0,
& 130,
& FALSE,
& 0H,
& 0
&>
DEFINE_DUIB <
& 'T5',
& 00001H,
& 0FBH,
& 00,
& 00,
& 00,
& 00,
& 08H, ...<----- The DEVICE NUMBER is the same
& 0H,
& 31H, ...<----- The DEVICE-UNIT number (T5) is equal to the
& TSINITIO,                               DEVICE-UNIT number of 'T4' plus one.
& TSFINISHIO,
& TSQUEUEIO,
& TSCANCELIO,
& DINFO_544A,
& UINFO_544A,
& 0FFFFH,
& 0,
& 130,
& FALSE,
& 0H,
& 0
&>

```

Figure C-4. Computing Device and Device-Unit Numbers (continued)

```

BIOS_CODE ENDS      ...<-----
TSC_CODE SEGMENT ER PUBLIC
    extrn DINFO_544A : far
    extrn UINFO_544A : far
TSC_CODE ENDS
BIOS_CODE SEGMENT  ...<-----

```

|----- NEW PORTION OF FILE
|----- TO ACCOUNT FOR NEW SEGMENT

Figure C-4. Computing Device and Device-Unit Numbers (continued)

The lines starting with `BIOS_CODE ENDS` through `BIOS_CODE SEGMENT` must be added to the end of the file. They provide BND386 with information on the location of your information tables. You must provide an `extrn <MODULE_NAME>: far` declaration for each `DINFO` and `UINFO` public name specified here; these names must be supplied as parameters `N01` through `N18` above in the `USERD` screen. This declaration is required because all terminal information is stored in a different physical segment than other driver information, and a far call is required to access it.

Contents of the File Specified in the (TUP) Parameter

Figure C-5 shows the contents of the file whose pathname you supplied in the (TUP) Terminal Device and Unit Source Code Path Name parameter of the User Devices Screen. This assembly-language file provides the information to define how the operating system should interface with this device.

```

extrn I544INIT : near
    extrn I544FINISH : near
    extrn I544SETUP : near
    extrn I544CHECK : near
    extrn I544ANSWER : near
    extrn I544HANGUP : near
    extrn I544UTILITY : near
;

    PUBLIC DINFO_544A <-----
DINFO_544A DW 04H
DW 9
%DW 300
%DW I544INIT
%DW I544FINISH
%DW I544SETUP
%DW TERMNULL
%DW I544ANSWER
%DW I544HANGUP
%DW I544UTILITY
DW 1
DW 071H
%DW I544CHECK
DD 0FE0000H
DW 04000H
DB 01H
    PUBLIC UINFO_544A <-----
UINFO_544A DW 01AH
DW 0109H
%DW 02580H
%DW 00000H
DW 012H
    PUBLIC
DECLARATIONS

```

Figure C-5. Public Declarations Needed for the DINFO and UINFO Tables

Provide the normal `extrn <MODULE_NAME>: near` declarations for I544INIT, ..., I544FINISH procedures. You must also provide a `PUBLIC <table name>` label before each DINFO and UINFO table specified.

Portion of System Generation Submit File as Changed by this Process

After completing the changes outlined above, you must generate a new system using the ICU. During the generation process, information is sent to the screen. Figure C-6 presents those portions of system generation that are changed by the previous steps.

```
;      BIOS

.
ASM386 ICDEV.A38
ASM386 ITDEV.A38

.
BND386 &          ←————— SEPARATE BIND OF TSC CODE
SEGMENT
ITDEV.OBJ, &
/RMX386/IOS/XDRMB1.LIB, &
/RMX386/IOS/XCMDRV.LIB(XTSIF), &
/RMX386/IOS/XCMDRV.LIB(XTSIO), &
/RMX386/IOS/XCMDRV.LIB, &
/INTEL/LIB/PLM386.LIB, &
/RMX386/LIB/RMXIFC32.LIB &
RENAMESEG(CODE32 TO TSC_CODE, TSC_CODE32 TO TSC_CODE, &
CODE TO TSC_CODE, DATA TO TSC_DATA) &
OBJECT (TSC.LNK) NODEBUG NOTYPE SEGSIZE(STACK(0)) &
NOLOAD NOPUBLICS EXCEPT( TSCINITIO, &
    TSCFINISHIO, &
    DINFO_02H, &
    UINFO_8251, &
    DINFO_03H, &
    UINFO_18848, &
    DINFO_04H, &
    UINFO_546, &
    UINFO_546CC, &
    DINFO_05H, &
    UINFO_547A, &
    DINFO_06H, &
    UINFO_547B, &
    DINFO_07H, &
    UINFO_547C, &
DINFO_544A, & ←————— USER SPECIFIED
```

Figure C-6. Portion of the Modified Submit File

```

BND386 &
IOS1.LNK, &
TSC.LNK, & ←———— INCLUSION OF TSC SUBSYSTEM IN IOS
                                SYSTEM BIND

ICDEV.OBJ, &
/RMX386/IOS/XDRMB1.LIB, &
/RMX386/IOS/XCMDRV.LIB, &
/INTEL/LIB/PLM386.LIB, &
/RMX386/LIB/RMXIFC32.LIB &
RENAMESEG(DRV_CODE TO CODE, CODE32 TO CODE, TSC_DATA TO DATA) &
OBJECT (IOS2.LNK) NODEBUG NOTYPE SEGSIZE(STACK(0)) &
NOLOAD NOPUBLICS EXCEPT (rqaiosinittask , &
    RqAttachDevice , &
.
.
.

```

Figure C-6. Portion of the Modified Submit File (continued)



Standard Definition Files for the iRMX III OS **D**

This appendix describes the definition files that you receive with the OS. It includes files for Multibus I and II systems as well as PC systems. Use the ICU to create or edit a definition file. A definition file contains values for the ICU-configurable parameters of the operating system. Create different definitions to create different systems.

For example, you can add or delete devices from your target system using the ICU. When you change a parameter using the ICU, save the change when you exit the ICU. Use the updated file to generate a custom version of the operating system. This appendix lists the attributes of the systems defined by the standard definition files.

See also: ICU Operations , Chapter 2;
Programming Techniques and Aedit Text Editor

⇒ **Note**

In the `/rmx386/icu` directory you can find an example definition file named `433expl.bck`. This file contains definitions for DUIBs no longer documented. These include all PCI_B devices and all devices with SCSI ID 4 and 5 (e.g. GSCW5_4 and M4380_5). If you need these DUIBs defined, you can get their definitions from this example file.

See also: `/rmx386/update/readme.txt`

Standard Definition File Names

The Standard ICU Definition Files provided for CPU boards use a naming convention that includes the board name followed by a suffix that indicates the nature of the application created by the definition file. These are the suffixes and their meanings:

- s Supports PCI server
- cp Supports PCI client and iRMXNET COMMputer
- scp CP configuration with PCI server
- net Supports PCI client and iRMXNET COMMengine
- snet NET configuration with PCI server
- rsd Diskless operation (LCI client)
- io I/O application; PCI server, LCI server, and no Human Interface. This is the only configuration where CLIB and SDB are not first-level jobs.
- msd diskless operation iRMXNET COMMputer using the Multibus II Subnet
- cp2 Multibus II multiple subnet support, TCP/IP
- cpp Supports PCI client, iRMXNET COMMputer, and iRMX partitioning

For example, *38612net.bck* definition file builds a COMMengine networking configuration for the SBC 386/12 board.

Multibus I Standard Definition Files

This section describes information about the standard definition files for Multibus I systems.

File Applications

Most boards are supported with several definition files: one for local applications, one or two for networking applications, and one for diskless networking applications. This list describes the different types of applications:

- | | |
|-------------------|--|
| Local | Configurations of the operating system that use only the resources in the chassis containing the CPU board. A local hard disk for the system device <code>:sd:</code> is required. These definition filenames map to the name of the CPU board, for example, <i>38612.bck</i> . |
| COMMengine | Networking configurations of the operating system that use a CPU board and a separate Ethernet controller board. The OS accesses resources both inside and outside the chassis that contains the CPU board. The iRMX-NET software is configured to be both a file server and client to provide access to the Local Area Network (LAN). A local hard disk for the system device <code>:sd:</code> is required. The definition files for COMMengine networking applications consist of the name of the CPU board with <code>net</code> appended. |
| COMMputer | Networking configurations of the operating system that use a CPU board that includes an Ethernet controller. These boards require an SBX 586 module for the Ethernet controller. The OS accesses resources both inside and outside the chassis that contains the CPU board. The iRMX-NET software is configured to be both a file server and client to provide access to the LAN. The SBX 586 module is controlled by the iRMX-NET job which is selected in these definition files. A local hard disk for the system device <code>:sd:</code> is required. The definition file names for COMMputer networking applications consist of the name of the board with <code>cp</code> appended. |

Diskless

Networking configurations of the OS that use a CPU board and a separate Ethernet controller board. The OS accesses resources both inside and outside the chassis that contains the CPU board. The iRMX-NET software is configured to be a file client only to provide access to the LAN. A hard disk located in another chassis is required for the system device `:sd:`; a Remote System Device (RSD). The definition files for diskless networking applications consist of the name of the CPU board with `rsd` appended.

Table D-1 shows which definition files can be used to create which types of applications.

Table D-1. Multibus I Application Types and Definition Files

Application	Definition File	CPU Board
Local	38612.bck 38612s.bck 38620.bck 48612.bck 48612s.bck pcp4s.bck	SBC 386/12 SBC 386/12S SBC 386/2X and SBC 386/3X SBC 486/12 SBC 486/12S SBC PCP4DX, PCP4SX, and PCP4DX2
COMMEngine	38612net.bck 38612snet.bck 38620net.bck 48612net.bck 48612snet.bck	SBC 386/12 SBC 386/12S SBC 386/2X and SBC 386/3X SBC 486/12 SBC 486/12S
COMMputer	38612cp.bck 38612scp.bck 48612cp.bck 48612scp.bck	SBC 386/12 SBC 386/12S SBC 486/12 SBC 486/12S
Diskless	38612rsd.bck 38620rsd.bck 48612rsd.bck	386/12 SBC 386/2X SBC 486/12

Default OS Layer and Jobs for Multibus I Definition Files

Table D-2 on page 182 shows the default OS layers and first-level jobs configured into each Multibus I definition file. The columns in Table D-2 indicate which ICU screens the OS layers and jobs are configured on: SUB, SYSJ, and NET. Each of the columns within these three columns represents a layer of the iRMX OS or a first-level job. A shaded cell means the layer or job is configured in the definition file, while a clear cell means the layer or job is not configured. The following list provides a key for Table D-2.

Column	Layer or Job
A50	ATCS/450 Server Job
AL	Application Loader
ATC	ATCS/279/ARC Server Job
BIO	Basic I/O System
BS	Multibus II Bootserver Job
CLB	Shared C Library
CMJ	iNA 960 COMMengine
DL	Multibus II Downloader Job
EIO	Extended I/O System
FPI	Front-Panel Interrupt Server Job
HI	Human Interface
MIP	iNA 960 MIP
OE	OS Extensions
PCI	PCI Server Job
PGS	Paging Subsystem
RCJ	iRMXNET Client
RSJ	iRMXNET Server
SDM	System Debug Monitor and System Debugger
SSK	Soft-Scope Kernel Job
SUB	Multibus II Multiple Subnet Support
TCP	TCP/IP
UDI	Universal Development Interface
VMD	VM86 Dispatcher
XCJ	X.25 Client
XSJ	X.25 Server

Table D-2. OS Layer and Jobs in Multibus I Definition Files

File	SUB Screen										SYSJ Screen										NET Screen					
	U D I	C L B	H I	A L	E I O	B I O	P G S	V M D	S D M	O E	P C I	D L	A T C	A 5 0	B S	F P I	S S K	M I P	C M J	S U B	R C J	R S J	T C P	X S J	X C J	
38612																										
38612net																										
38612cp																										
38612rsd																										
38612s																										
38612snet																										
38612scp																										
38620																										
38620net																										
38620rsd																										
48612																										
48612net																										
48612cp																										
48612rsd																										
48612s																										
48612snet																										
48612scp																										
pcp4s																										

Multibus I Device Driver Support

Table D-3 outlines the device drivers that specific definition files support. The *.bck* extension has been omitted from the file names.

Table D-3. Multibus I Device Driver Support

Definition File	MSC Driver	8251 Driver	8274 Driver	544A Driver	TCC Driver	Comm Service	PCI Server	PCI Driver
SBC 386/12 and SBC 386/12S Boards								
38612	Yes	No	Yes	No	Yes ¹	No	No	No
38612net	Yes	No	Yes	Yes	Yes ¹	No	No	No
38612cp	Yes	No	Yes	Yes	Yes ¹	No	No	No
38612rsd	No	No	Yes	Yes	Yes ¹	No	No	No
38612s	Yes	No	Yes	No	Yes ¹	Yes	Yes	Yes
38612snet	Yes	No	Yes	Yes	Yes ¹	Yes	Yes	Yes
38612scp	Yes	No	Yes	Yes	Yes ¹	Yes	Yes	Yes
SBC 386/2X and SBC 386/3X Boards								
38620	Yes	Yes	No	No	Yes ²	No	No	No
38620net	Yes	Yes	No	No	Yes ²	No	No	No
38620rsd	No	Yes	No	Yes	Yes ²	No	No	No

¹ Supports combinations of the SBC 188/48 or 188/56 boards and the SBC 548 board. See Table D-9 on page 194.

² Supports combinations of the SBC 188/48 or 188/56 boards and the SBC 546/547 boards. See Table D-8 on page 194.

Table D-3. Multibus I Device Driver Support (continued)

Definition File	MSC Driver	8251 Driver	8274 Driver	544A Driver	TCC Driver	Comm Service	PCI Server	PCI Driver
SBC 486/12 and SBC 486/12S Boards								
48612	Yes	No	Yes	No	Yes	No	No	No
48612net	Yes	No	Yes	Yes ³	Yes	No	No	No
48612cp	Yes	No	Yes	Yes ³	Yes	No	No	No
48612rsd	No	No	Yes	Yes ³	Yes	No	No	No
48612s	Yes	No	Yes	No	Yes	Yes	Yes	Yes
48612snet	Yes	No	Yes	Yes ³	Yes	Yes	Yes	Yes
48612scp	Yes	No	Yes	Yes ³	Yes	Yes	Yes	Yes
SBC PCP4DX, SBC PCP4SX, and SBC PCP4DX2 Boards								
pcp4s	N/A ⁴	N/A ⁴	N/A ⁴	N/A ⁴	N/A ⁴	Yes	Yes	Yes

³ Supports combinations of the SBC 188/48 or 188/56 boards and the SBC 548 board. See Table D-9 on page 194.

⁴ Because the SBC PCP4DX, PCP4SX, and PCP4DX2 boards are PC boards with embedded 486 architecture, driver support is limited to that found on standard PC boards. These boards have the COM1/COM2 serial driver enabled by default.

Additional Notes

1. The definition files for the SBC 386/12 and the SBC 486/12 use the peripherals standard in the Intel System 310AP microcomputer.
2. The definition files for the SBC 386/2X and SBC 386/3X use the peripherals standard in the Intel System 320 microcomputer.
3. With the exception of *pcp4s.bck*, all of the Multibus I definition files use identical configuration parameters for those devices that they support in common. For example, all the networking definition files use the same interrupt level and port address for the SBC 552A Ethernet Controller and hard disk controller (SBC 215G, 214, 221, and 221S). Therefore, if you have a standard System 310AP with an SBC 286/12 in it and you want to upgrade to an SBC 386/12 or 486/12 board, use the operating system running the SBC 286/12 to generate a system for the new board, power down the chassis, swap the boards and re-boot the new board.
4. You will need to change the interrupt levels, port addresses and buffer addresses of terminal controllers if you replace an SBC 386/2X/3X board in a standard System 320 with an SBC 386/12 or 486/12 board. You will also need to add support for the global clock if you use one. Networking and peripheral controllers are compatible between Intel microcomputers.

5. The Mass Storage Controller (MSC) Driver supports the SBC 214, SBC 215G, SBC 221, and SBC 221S controllers.
6. The Terminal Communications Controller (TCC) Driver supports the SBC 188/48, 188/56, 546, 547, 548, and 549 controllers.
7. Because the *pcp4s.bck* definition file is for a PC-type board device support is limited to standard PC board peripherals. This means configuration parameters will not be common with other iRMX boards.

SCSI Support for SBC 386/12S and 486/12S Boards

1. All the *38612s*.bck*, *48612s*.bck*, and *pcp4s.bck* definition files for Multibus I boards include the PCI Server Job, the PCI Device Driver, and the Communication Service to provide SCSI support. In Multibus I systems, only short-circuit message passing is possible. The definition files also include the Mass Storage Controller (MSC) Device Driver for ST506 and ESDI peripherals.
2. The definition files for the SBC 386/12S board, the SBC 486/12S board, and the SBC PCP4DX, PCP4SX, and PCP4DX2 boards use a SCSI hard disk as the system device. This allows you to boot from a SCSI hard disk and also allows you to use a non-SCSI hard disk in your application. If you do not need the ST506 or ESDI support, delete the MSC Device Driver. See Table D-3 on page 183 for more information on device driver support for the SBC 386/12S and SBC 486/12S definition files.

Networking Definition Files

1. The networking definition files for the SBC 386/2X/3X load the iNA 960 file, */net/ina552a.32l*, that works on the SBC 552A board. For Multibus I systems, the default is the 552A board. This board has more RAM on it than the earlier SBC 552 board and can host more functions such as the Boot Server. Due to timing constraints, the SBC 386/12(S) and 486/12(S) boards can only be used with the SBC 552A board.
2. There are no specific diskless networking definition files supplied for the SBC 386/12S, 486/12S, PCP4DX, PCP4SX, or PCP4DX2 boards. Because of this, delete the parameter *fileserver* from the iRMX-NET configuration in the *38612net.bck*, *48612net.bck*, *38612cp.bck*, or *48612cp.bck*.

Interrupt Levels Used in the Standard Definition Files

All of the Intel386 and Intel486-based CPU boards that run the iRMX III OS contain a master and a slave 8259A programmable interrupt controller (PIC). This section describes the interrupt level assignments (both Multibus and PIC) in the standard definition files. In order to accommodate a large number of configurations, the definition files may assign a single Multibus or PIC interrupt level to two or more different functions. You may use only one of these functions at a time without changing the configuration. If you need to use more than one at a time, you must run the ICU and generate a version of the OS that reflects your needs.

In these discussions, the term *dedicated* means the function is not selectable with a jumper; the board fixes the input and output of the signal. Also, the term *available* means the function is not assigned any function in the standard definition files and will not conflict with your application. Some of the definition files enable you to use multiple SBC 547 boards. These boards are indicated: SBC 547 number 1, SBC 547 number 2, and SBC 547 number 3.

The SBC 552A Ethernet Communication Controller board is listed in the following sections. If you are not using iRMX-NET, the interrupt level assigned to the SBC 552A board is available for another use.

Table D-4 shows the Multibus and the Master and Slave PIC interrupt level assignments for SBC 386/12(S) and 486/12(S) Boards.

Table D-4. Interrupt Level Assignments for SBC 386/12(S) and SBC 486/12(S)

Multibus Interrupt	Master PIC Interrupt	Function or Connection
None	0	System clock (8254 timer 0)
INT0	NMI	Memory parity errors
INT1	1	System debugger (SDB)
INT2	2	SCSI on SBC 386/12S and 486/12S, otherwise available
INT3	3	SBC 188/48/56 or SBC 548 or SBC 544A
INT4	4	SBC 552A
INT5	5	SBC 214 or SBC 215G/SBX 217C/218A or SBC 221 or 221S
None	6	On-board 8274 Channels A and B
None	7	On-board Slave PIC (dedicated)
Multibus Interrupt	Slave PIC Interrupt	Function or Connection
INT6	0	Available
INT7	1	Available
None	2	Available
None	3	SBX interrupt (dedicated)
None	4	SBX interrupt (dedicated)
None	5	SBX interrupt (dedicated)
None	6	SBX interrupt (dedicated)
None	7	On-board line printer (dedicated)

Table D-5 shows the Multibus and the Master and Slave PIC interrupt level assignments for SBC 386/2X and 386/3X Boards.

Table D-5. Interrupt Level Assignments for SBC 386/2X and SBC 386/3X

Multibus Interrupt	Master PIC Interrupt	Function or Connection
None	0	System clock (8254 timer 0)
INT0	NMI	Memory parity errors
INT1	1	System debugger (SDB)
INT2	2	SBC 547 number 1
INT3	3	SBC 546 or SBC 188/48/56
INT4	4	SBC 552A
INT5	5	SBC 214 or SBC 215G/SBX 218A or SBC 221 or 221S
None	6	Available
None	7	On-board slave PIC (dedicated)
Multibus Interrupt	Slave PIC Interrupt	Function or Connection
INT6	0	SBC 547 number 2
INT7	1	SBC 547 number 3
None	2	Available
None	3	SBX interrupt (dedicated)
None	4	SBX interrupt (dedicated)
None	5	8254 timer 1 (dedicated)
None	6	8251A USART receive interrupt
None	7	8251A USART transmit interrupt

Table D-6 shows the Multibus and the Master and Slave PIC interrupt level assignments for the SBC PCP4DX, SBC PCP4SX, and SBC PCP4DX2 Boards. This table includes optional sources through jumpers and default assignments.

Table D-6. Interrupt Level Assignments for the SBC PCP4DX(2) and PCP4SX Boards

Multibus Interrupt	Master PIC Interrupt	Optional Sources Using Jumpers	Connection	Default Assignment
INT0	0	NONE	NONE	82378ZB: TIMER 1, COUNTER 0
INT1	1	SBXBOPT0 E50-E49	SBXB	KEYBOARD CONTROLLER
INT2	2			
INT3	3	MINT0 E26-E27 MINT3 E26-E36	AVAILABLE SBC546 or 188/48/56	AIP: COM2
INT4	4	MINT4 E35-E34 MINT3 E35-E36 MINT6 E35-E45	SBC552A SBC546 or 188/48/56 AVAILABLE	AIP: COM1
INT5	5	MINT0 E37-E27 MINT3 E37-E36 MINT2 E37-E38 MINT5 E37-E47	AVAILABLE SBC546 or 188/48/56 SBC547 NUMBER 1 SBC221S	NONE
INT6	6	NONE	NONE	AIP: FLEXIBLE DISK
INT7	7	MINT6 E60-E45 MINT1 E60-E59 MINT7 E60-E61	AVAILABLE AVAILABLE AVAILABLE	AIP: LPT1

continued

**Table D-6. Interrupt Level Assignments for the SBC PCP4DX(2) and PCP4SX Boards
(continued)**

Multibus Interrupt	Slave PIC Interrupt	Optional Sources Using Jumpers	Connection	Default Assignment
INT8	0	NONE	NONE	REALTIME CLOCK
INT9	1	SBXAIRQ0 E24-E23 MINT4 E24-E34	SBXA SBC552A	82595TX ETHERNET
INT10	2	MB_ERRINT E42-E43 SBXBIRQ1 E42-E57	MB_ERRINT SBXB	82378TX SIO (PIRQ0); SCSI-2
INT11	3	MINT4 E44-E34 MB_ERRINT E44-E43 MINT1 E44-E59 MINT6 E44-E45	SBC552A MB_ERRINT AVAILABLE AVAILABLE	MB_ERRINT
INT12	4	SBXAIRQ0 E33-E23 MINT4 E33-E34 MB_ERRINT E33-E43	NONE SBC552A MB_ERRINT	MOUSE
INT13	5	NONE	NONE	CPU: FLOATING POINT ERROR
INT14	6	MINT2 E48-E38 MINT5 E48-E47 SBXBOPT0 E48-E49 SBXAOPT0 E48-E63	SBC547 NUMBER 1 SBC221S SBXB SBXA	IDE
INT15	7	MINT3 E46-E36 MINT6 E46-E45 MINT5 E46-E47 MINT7 E46-E61	SBC546 or 188/48/56 SBC547 NUMBER 2 SBC552A SBC547 NUMBER 3	NONE

I/O Controller Board Support

Table D-7 shows the I/O controller support for the standard definition files. Whenever more than one I/O board is listed, you may use only one of the I/O controllers in the system because they use a common system resource (such as an interrupt level or a port address) that cannot be shared. If you need to use a different combination of I/O controllers, you can either create a new definition file or modify one of the standard definition files. Depending on what changes you make to the device configuration screens in the definition files, you may need to jumper the I/O controllers differently than described in this manual.

See also: *Command Reference*, for the default device names supported in the Standard Definition Files

Table D-7. I/O Controller Support for Multibus I Definition Files

CPU Board	I/O Controller Support
SBC 386/12, 386/12S, 486/12, and 486/12S	8274 Channels A and B SBC 544A or 548 or SBC 188/48 or 188/56 SBC 214 or SBC 215G/SBX 217C/218A SBC 221 or 221S On-board SCSI (SBC 386/12S and 486/12S only)
SBC 386/2X and 3863X in a System 320 Microcomputer	8251A SBC 546 (includes line printer and global clock) SBC 547 number 1 SBC 547 number 2 SBC 547 number 3 SBC 548, 188/48 or 188/56 SBC 214 or SBC 215G/SBX 217C/218A or SBC 221 or 221S
SBC PCP4DX(2) and PCP4SX	None

Multibus I Operating System Layer Configuration

Depending on the OS layer, configuration aspects for the definition files differ. This list points out the differences for OS layers:

Nucleus	All definition files define the Nucleus identically except for NPX definition.
SDM	The iRMX III OS uses the SDM monitor for its low level debug support. All standard definition files except <i>pcp4s.bck</i> select the serial channel on the CPU board as the primary SDM serial channel. The file <i>pcp4s.bck</i> uses the COM1 channel as the primary SDM serial channel.
System Debugger	The iRMX III SDB is defined identically in the standard definition files for all Multibus I systems. Interrupt level 1 (encoded level 18H) on the master PIC is used to enter the SDB. To use the SDB, your system must include the SDM monitor.
BIOS	<p>The iRMX III BIOS is defined identically, except for the Global Clock parameter, in all but the <i>pcp4s.bck</i> standard definition files.</p> <p>In the <i>pcp4s.bck</i> definition file the Tape Support, BIOS Pool Minimum, Number of Loadable Devices, and Number of Loadable Device Units as well as the Global Clock are defined differently than on other Multibus I standard definition files.</p>

EIOS The iRMX III EIOS is defined identically, except for the line printer physical name, in all but the *pcp4s.bck* standard definition files. The line printer physical device name does not apply in the *pcp4s.bck* definition file.

Physical Device Name	Logical Name	Definition Files For
bb (Byte Bucket)	:bb:	All boards
stream	:stream:	All boards
w0 or Bootstrap Device	:sd:	All boards
lp	:lp:	SBC 386/12, 386/12S, 486/12, 486/12S
t546_lp	:lp:	SBC 386/2X/3X boards

Application Loader The iRMX III AL is defined identically in the standard definition files.

UDI The iRMX III UDI is defined identically in the standard definition files. There are no configurable parameters to the UDI.

Human Interface The iRMX III HI is defined identically in the standard definition files.

Initially, one terminal is configured for use by HI users. You can add more terminals to all boards by modifying the file *:config:terminals*. Tables D-8 and D-9 on page 194 list the maximum number of terminals that can be supported by the systems defined by the standard definition files. The different combinations indicate the different types of terminal controllers which may be used at the same time. If a different combination is needed, you must run the ICU to change the interrupt level, flag byte address, and the dual-port buffer addresses of the necessary terminal controller boards.

For the *pcp4s.bck* definition file, the default supported terminal device is the console (physical device *d_cons*). In addition to this device, the SBC PCP4DX, PCP4SX, and PCP4DX2 boards can support an additional user on both the COM1 and COM2 serial devices. Thus, the *pcp4s.bck* definition file can support up to three users.

Table D-8. Terminal Support for the SBC 386/2X/3X Boards

Terminal Device	Comb. #1	Comb. #2	Comb. #3	Comb. #4
CPU board	1 user	1 user	1 user	1 user
SBC 546	4 users	4 users	4 users	
SBC 547 #1	8 users	8 users	8 users	
SBC 547 #2		8 users	8 users	
SBC 547 #3			8 users	
SBC 188/48 or SBC 188/56				8 users
TOTAL USERS	13 users	21 users	29 users	9 users

Table D-9. Terminal Support for the SBC 386/12(S) and 486/12(S) Boards

Terminal Device	Comb. #1	Comb. #2	Comb. #3
CPU board	2 users	2 users	2 users
SBC 544A	4 users		
SBC 188/48 or SBC 188/56		8 users	
SBC 548			8 users
TOTAL USERS	6 users	10 users	10 users

Multibus II Standard Definition Files

This section describes information about the standard definition files for Multibus II systems.

Intel provides definition files that support Multibus II systems, such as the System 520. Because the System 520 can contain various combinations of boards, you must choose between various combinations of boot files generated from the definition files. Different combinations of definition files are used as the base for different development directions.

Most boards are supported with at least three definition files: one for local applications, one for networking applications, and one for diskless networking applications. The boards described in this section are assumed to use MSA firmware, which is supplied on the boards.

The Standard ICU Definition Files provided with the iRMX III OS for Multibus II CPU boards follow the naming convention described at the beginning of this appendix. However, some exceptions exist:

CPU Board	Definition File
MIX 486DX33	mix4
MIX 486DX66	mix4
MIX 486SX33	mix4
SBC 386/258	258
SBC 386/258D	258
SBC 486/150	486125
SBC 486/133SE	433
SBC 486/166SE	433
SBC 486DX66	486dx33
SBC P5090	p90
SBC P5090ISE	p90
SBC P5090CPU	p90
SBC P5120ISE	p90
SBC P5120CPU	p90

Multibus II File Applications

Most boards are supported with several definition files: one for local applications, one or two for networking applications, one for diskless networking applications, and one for I/O applications. This list describes the different types of applications:

- Local** Configurations of the iRMX III OS which use only the resources of the chassis in which the CPU board is located.
- The definition files for local applications consist only of the CPU board name. The target files they create include no networking software. A dedicated hard disk for the system device *:sd:* is required.
- COMMengine** Network configurations of the iRMX III OS which use a CPU board and a separate Ethernet controller board. This is true even for boards that have an on-board Ethernet controller. The OS can access resources both inside and outside the chassis that contains the CPU board. The iRMX-NET software is configured both as a file server and consumer to provide LAN access. A dedicated hard disk for the system device *:sd:* is required. The definition files for COMMengine networking applications consist of the name of the CPU board with *net* appended.
- COMMputer** Network configurations of the iRMX III OS that use a CPU board that also provides an Ethernet controller. These CPU boards require an on-board Ethernet controller. The OS can access resources both inside and outside the chassis that contains the CPU board. The iRMX-NET software is configured to be both a file server and client to provide access to the LAN. The on-board Ethernet controller is controlled by the iRMX-NET job which is selected in these definition files. The appropriate iTP4 module is combined with iRMX-NET automatically. A dedicated hard disk for the system device *:sd:* is required. The definition file names for COMMputer networking applications consist of the name of the board with *cp* appended.

Diskless

Network configurations of the iRMX III OS that use a CPU board and a separate Ethernet controller board. The OS can access resources both inside and outside the chassis that contains the CPU board. The iRMX-NET software is a file client only providing LAN access. A hard disk controlled by another CPU board, either in the same chassis or in a different chassis, is required for the system device `:sd:`, a Remote System Device (RSD). This disk can be shared with other CPU boards. The definition files for diskless networking applications consist of the name of the CPU board with `rsd` or `msd` appended. I/O server boards that have an on-board SCSI subsystem do not have corresponding definition files for diskless applications.

I/O

Configurations of the iRMX III OS that provide Multibus II clients access to the resources on an I/O board such as SCSI, networking, and serial controllers. The target files defined by these definition files are used on the associated I/O board in conjunction with other boards in a Multibus II system. For example, a CPU board running either the iRMX or the UNIX operating system could access disks controlled by an iRMX I/O configuration running on an I/O server board.

These applications do not include an HI or AL and do not require a system device. They do not require any system resources beyond those located on their local board. The application defined by each definition file fully supports all of the I/O capabilities of the associated board. These applications are designed to be boot loaded with the MSA Bootstrap Loader or PROMmed.

Table D-10 shows which definition files can be used to create which types of applications.

Table D-10. Multibus II Application Types and Definition Files

Application	Definition File	CPU Board
Local	386020.bck ¹ 386020a.bck 486020a.bck mix4.bck 386120.bck 386133.bck 258s.bck 486125.bck 433.bck 433s.bck 486dx33.bck 486sx25.bck p90s.bck	MIX 386/020 MIX 386/020A MIX 486/020A MIX 486DX33, 486DX66 and 486SX33 SBC 386/116 and 386/120 SBC 386/133 SBC 386/258 and 386/258D SBC 486/125 and 486/150 SBC 486/133SE and 486/166SE SBC 486/133SE and 486/166SE SBC 486DX266 and 486DX33 SBC 486SX25 SBC P5090ISE, P5090, and P5120ISE
COMMEngine	386020net.bck 386020anet.bck 486020anet.bck mix4net.bck 386120net.bck 386133net.bck 258snet.bck 486125net.bck 433net.bck ³ 486dx33net.bck 486sx25net.bck p90net.bck	MIX 386/020 MIX 386/020A MIX 486/020A MIX 486DX33, 486DX66 and 486SX33 SBC 386/116 and 386/120 SBC 386/133 SBC 386/258 and 386/258D SBC 486/125 and 486/150 SBC 486/133SE and 486/166E SBC 486DX266 and 486DX33 SBC 486SX25 SBC P5090ISE, P5090CPU, P5090, P5120ISE, and P5120CPU

continued

Table D-10. Multibus II Application Types and Definition Files (continued)

Application	Definition File	CPU Board
COMMputer	386020cp.bck 386020acp.bck 486020acp.bck mix4cp.bck 433cp.bck ⁴ 433scp.bck ² 433scpp.bck ² 486dx33cp.bck 486sx25cp.bck p90cp.bck p90scp.bck p90scpp.bck	MIX 386/020 MIX 386/020A MIX 486/020A MIX 486DX33, 486DX66 and 486SX33 SBC 486/133SE ⁴ and 486/166SE SBC 486/133SE ⁴ and 486/166SE SBC 486/133SE ⁴ and 486/166SE SBC 486DX266 and 486DX33 SBC 486SX25 SBC P5090ISE, P5090 and P5120ISE SBC P5090ISE, P5090 and P5120ISE SBC P5090ISE, P5090 and P5120ISE
Diskless	386020rsd.bck ⁵ 386020arsd.bck ⁵ 486020arsd.bck ⁵ mix4rsd.bck ⁵ 386120rsd.bck 386133rsd.bck 486125rsd.bck 433rsd.bck 433msd.bck 486dx33rsd.bck 486sx25rsd.bck p90rsd.bck	MIX 386/020 MIX 386/020A MIX 486/020A MIX 486DX33, 486DX66 and 486SX33 SBC 386/116 and 386/120 SBC 386/133 SBC 486/125 and 486/150 SBC 486/133SE and 486/166SE SBC 486/133SE and 486/166SE SBC 486DX266 and 486DX33 SBC 486SX25 SBC P5090ISE, P5090CPU, P5090, P5120ISE, and P5120CPU
I/O	386020io.bck ⁷ 386020aio.bck 486020aio.bck ⁵ mix4io.bck 258io.bck ⁸ 433io.bck ^{7,8} p90io.bck	MIX 386/020 ⁶ MIX 386/020A ⁶ MIX 486/020A ⁶ MIX 486DX33, 486DX66 and 486SX33 SBC 386/258 and 386/258D SBC 486/133SE and 486/166SE SBC P5090ISE, P5090 and P5120ISE

- ¹ The *386020.bck* file defines a minimal target file with no networking. It treats the MIX baseboard as a CPU board.
- ² The target files created by the *433scp.bck* and *433scpp.bck* files treat an SBC 486/133SE or 486/166SE board as a single board system. They allow the board to act as both an I/O server board and a CPU board. They contain iRMX-NET, the PCI Server, and the *atcs/279/arc* Server. They can be booted when the board is in slot 0. The definition files select iRMX-NET in a COMMMputer configuration. If iRMX-NET is not required, use *433s.bck*.
- ³ The target file created by the *433net.bck* file treats an SBC 486/133SE or 486/166SE board as a CPU board, not as an I/O Server board. The target file does not contain a PCI Server or *atcs/279/arc* Server. Because of this, it cannot automatically be booted on a 486/133SE or 486/166SE acting as an I/O Server board in slot 0. Boards in slot 0 are assumed to provide access to SCSI peripherals through the PCI Server, and access to some sort of system console through the *atcs/279/arc* Server.
- ⁴ The target file created by the *433cp.bck* file treats a 486/133SE or 486/166SE board as a CPU board, not as an I/O server. The target file does not contain a PCI Server or *atcs/279/arc* Server. Because of this, it cannot automatically be booted on a 486/133SE or 486/166SE acting as an I/O Server in slot 0. Boards in slot 0 are assumed to provide access to SCSI peripherals through the PCI Server, and access to some sort of system console through the *atcs/279/arc* Server. You can also use *433scp.bck* or *433scpp.bck* as another option for a COMMMputer configuration definition file for the 486/133SE or 486/166SE board.
- ⁵ The MIX remote system device definition files (*386020rsd.bck*, *386020arsd.bck*, *486020arsd.bck*, *mix4rsd.bck*) create target files that require a separate Ethernet controller, even though the MIX 560 board could provide that service.
- ⁶ Up to three MIX 450 modules and one MIX 560 module in any combination can be mounted on a MIX baseboard.
- ⁷ The *386020io.bck*, *486020io.bck*, *433io.bck*, and *mix4io.bck* files can be used to create target files that when booted on the associated board provide an Ethernet controller which other CPU boards can access as a COMMEngine.
- ⁸ The *258io.bck* and *433io.bck* files create target files that can be booted on the associated I/O Server board in slot 0.

Default OS Layer and Jobs for Multibus II Definition Files

Table D-11 on page 202 shows the default OS layers and first-level jobs configured into each Multibus II definition file. The columns in Table D-11 indicate which ICU screens the OS layers and jobs are configured on: SUB, SYSJ, and NET. Each of the columns within these three columns represents a layer of the iRMX OS or a first-level job. A shaded cell means the layer or job is configured in the definition file, while a clear cell means the layer or job is not configured. The following list provides a key for Table D-11.

Column	Layer or Job
A50	ATCS/450 Server Job
AL	Application Loader
ATC	ATCS/279/ARC Server Job
BIO	Basic I/O System
BS	Multibus II Bootserver Job
CLB	Shared C Library
CMJ	iNA 960 COMMengine
DL	Multibus II Downloader Job
EIO	Extended I/O System
FPI	Front-Panel Interrupt Server Job
HI	Human Interface
MIP	iNA 960 MIP
OE	OS Extensions
PCI	PCI Server Job
PGS	Paging Subsystem
RCJ	iRMXNET Client
RSJ	iRMXNET Server
SDM	System Debug Monitor and System Debugger
SSK	Soft-Scope Kernel Job
SUB	Multibus II Multiple Subnet Support
TCP	TCP/IP
UDI	Universal Development Interface
VMD	VM86 Dispatcher
XCJ	X.25 Client
XSJ	X.25 Server

Table D-11. OS Layer and Jobs in Multibus II Definition Files

File	SUB Screen										SYSJ Screen										NET Screen					
	U D I	C L B	H I	A L	E I O	B I O	P G S	V M D	S D M	O E	P C I	D L	A T C	A 5 0	B S	F P I	S S K	M I P	C M J	S U B	R C J	R S J	T C P	X S J	X C J	
258s																										
258snet																										
258io																										
386020																										
386020cp																										
386020net																										
386020rsd																										
386020io																										
386020a																										
386020acp																										
386020anet																										
386020arsd																										
386020aio																										
386120																										
386120net																										
386120rsd																										
386133																										
386133net																										
386133rsd																										
433																										
433cp																										
433net																										

continued

Table D-11. OS Layer and Jobs in Multibus II Definition Files (continued)

File	SUB Screen										SYSJ Screen							NET Screen							
	U D I	C L B	H I	A L	E I O	B I O	P G S	V M D	S D M	O E	P C I	D L	A T C	A 5 0	B S	F P I	S S K	M I P	C M J	S U B	R C J	R S J	T C P	X S J	X C J
433rsd																									
433msd																									
433io																									
433s																									
433scp																									
433scpp																									
433scp2																									
486020a																									
486020acp																									
486020anet																									
486020arsd																									
486020aio																									
mix4																									
mix4cp																									
mix4cp2																									
mix4net																									
mix4rsd																									
mix4io																									
mixx25																									
p90																									
p90cp																									
p90net																									
p90rsd																									
p90msd																									
p90io																									
p90s																									
p90scp																									
p90scpp																									
p90scp2																									

continued

Table D-11. OS Layer and Jobs in Multibus II Definition Files (continued)

File	SUB Screen										SYSJ Screen										NET Screen									
	U D I	C L B	H I	A L	E I O	B I O	P I G	V M	S D	O E	P C	D L	A T C	A 5 0	B S	F P	S S	M I P	C M	S U B	R C J	R S J	T C P	X S J	X C J					
486dx33																														
486dx33cp																														
486dx33net																														
486dx33rsd																														
486125																														
486125net																														
486125rsd																														

Memory Addresses Used in Multibus II Standard Definition Files

The default memory addresses used by the iRMX III OS, defined in the standard definition files, are listed below. If you develop a custom application which does not use the hardware supported in the standard definition files, you may use the memory addresses for other functions. You may also assign other memory addresses to the function if required.

Start	End	Function
0H	0FFFFH	Firmware Data
10000H	0FFFEFFFFH	OS and Application Code and data
0FFF00000H	0FFFFFFFHH	EPROM space (SDM, IDX, MSA Bootstrap Loader)

Multibus II Operating System Layer Configuration

Depending on the OS layer, configuration aspects for the Multibus II definition files differ. This list points out the differences for OS layers:

Nucleus The iRMX III Nucleus is defined identically in the standard Multibus II definition files except for RAM, MPC, ADMA, and DAG configuration which is different for each file.

- MIX n86/020A, MIX 486DX33, and MIX 486SX33 use enhanced implicit bus arbitration mode.

System Debugger The iRMX III SDB is defined identically in the standard definition files for all Multibus II systems. To use the SDB, your system must include the SDM monitor.

BIOS The iRMX III BIOS is defined identically, except for the Global Clock parameter, in the standard definition files.

EIOS The iRMX III EIOS is defined identically, except for the line printer physical name, in the standard definition files.

These logical devices are attached by the EIOS when it is initialized:

Physical Device Name	Logical Name	Definition File
bb (byte bucket)	:bb:	All
stream	:stream:	All
scw	:sd:	All
No line printer configured	---	386120.bck

Application Loader The iRMX III AL is defined identically in the standard definition files.

Human Interface The iRMX III HI is defined identically in the standard definition files.

The ATCS driver can be used to configure terminal devices.

See also: *ATCS Driver, System Configuration and Administration*

UDI The iRMX III UDI is defined identically in the standard definition files. There are no configurable parameters to the UDI.

PC Standard Definition Files

This section describes information about the standard definition files for PC systems.

Intel provides definition files that support iRMX for Windows and PCs that run on a PC motherboard. You can choose between various combinations of boot files generated from the definition files for your particular system. Different combinations of definition files are used as the base for different development directions.

The Standard ICU Definition Files provided with the iRMX III OS for Windows and PCs are:

Definition File	System
pc.bck	PC Platforms (including SBC PCP4DX, PCP4SX, PCP4DX2, SBC 486SX25, 486DX33, SBC P5090, P5090ISE, and P5090CPU with BIOS boot enabled)
pccp.bck	EtherExpress™ Class Networking
pccprsd.bck	EtherExpress Class Networking
pcnet.bck	PCL2 and PCL2A Networking
pcp90.bck	SBC P5090, P5090ISE, P5090CPU, SBC P5120ISE, and P5120CPU

PC Applications

PC definition files support several applications: one for local applications, one or two for networking applications, and one for diskless networking applications. This list describes the different types of applications:

Local	Configurations of the iRMX III OS which use only the resources of the chassis in which the CPU board is located. The definition files for local applications include <i>pc.bck</i> and <i>pcp90.bck</i> . The target files they create include no networking software. A dedicated hard disk for the system device <i>:sd:</i> is required.
--------------	--

- COMMengine** Network configurations of the iRMX III OS which use a CPU board and a separate Ethernet controller board. This is true even for boards that have an on-board Ethernet controller. The OS can access resources both inside and outside the chassis that contains the CPU board. The iRMX-NET software is configured both as a file server and consumer to provide LAN access. A dedicated hard disk for the system device *:sd:* is required. The definition file for COMMengine networking applications is *pcnet.bck*.
- COMMputer** Network configurations of the iRMX III OS that use a CPU board that also provides an Ethernet controller. These CPU boards require an on-board Ethernet controller. The OS can access resources both inside and outside the chassis that contains the CPU board. The iRMX-NET software is configured to be both a file server and client to provide access to the LAN. The on-board Ethernet controller is controlled by the iRMX-NET job which is selected in these definition files. The appropriate iTP4 module is combined with iRMX-NET automatically. A dedicated hard disk for the system device *:sd:* is required. The definition file for COMMputer networking applications is *pccp.bck*.
- Diskless** Network configurations of the iRMX III OS that use a CPU board and a separate Ethernet EtherExpress class controller board. The OS can access resources both inside and outside the chassis that contains the CPU board. The iRMX-NET software is a file client only providing LAN access. A hard disk controlled by another CPU board, either in the same chassis or in a different chassis, is required for the system device *:sd:*, a Remote System Device (RSD). This disk can be shared with other CPU boards. The definition file for diskless networking applications is *pccprsd.bck*.

Table D-12 summarizes which definition files can be used to create which types of applications.

Table D-12. PC Application Types and Definition Files

Application	Definition File
Local	pc.bck
Local	pcp90.bck
COMMputer	pccp.bck
COMMputer and remote	pccprsd.bck
COMMengine	pcnet.bck

Default OS Layer and Jobs for PC Definition Files

Table D-13 on page 210 shows the default OS layers and first-level jobs configured into each PC definition file. The columns in Table D-13 indicate which ICU screens the OS layers and jobs are configured on: SUB, SYSJ, and NET. Each of the columns within these three columns represents a layer of the iRMX OS or a first-level job. A shaded cell means the layer or job is configured in the definition file, while a clear cell means the layer or job is not configured. The following list provides a key for Table D-13.

Column	Layer or Job
A50	ATCS/450 Server Job
AL	Application Loader
ATC	ATCS/279/ARC Server Job
BIO	Basic I/O System
BS	Multibus II Bootserver Job
CLB	Shared C Library
CMJ	iNA 960 COMMengine
DL	Multibus II Downloader Job
EIO	Extended I/O System
FPI	Front-Panel Interrupt Server Job
HI	Human Interface
MIP	iNA 960 MIP
OE	OS Extensions
PCI	PCI Server Job
PGS	Paging Subsystem
RCJ	iRMXNET Client
RSJ	iRMXNET Server
SDM	System Debug Monitor and System Debugger
SSK	Soft-Scope Kernel Job
SUB	Multibus II Multiple Subnet Support
TCP	TCP/IP
UDI	Universal Development Interface
VMD	VM86 Dispatcher
XCJ	X.25 Client
XSJ	X.25 Server

Table D-13. OS Layer and Jobs in PC Definition Files

File	SUB Screen										SYSJ Screen										NET Screen					
	U D I	C L B	H I	A L	E I O	B I O	P G S	V M D	S D M	O E	P C I	D L	A T C	A 5 0	B S	F P I	S S K	M I P	C M J	S U B	R C J	R S J	T C P	X S J	X C J	
pc																										
pccp																										
pccprsd																										
pcnet																										
pcp90																										

PC Operating System Layer Configuration

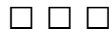
Depending on the OS layer, configuration aspects for the PC definition files differ. This list points out the differences for OS layers:

- Nucleus** The iRMX III Nucleus is defined identically in the standard PC definition files.
- System Debugger** The iRMX III SDB is defined identically in the standard definition files for PC systems. To use the SDB, your system must include the SDM monitor.
- BIOS** The iRMX III BIOS is defined identically in the standard PC definition files.
- EIOS** The iRMX III EIOS is defined identically in the standard PC definition files.

These logical devices are attached by the EIOS when it is initialized:

Physical Device Name	Logical Name	Definition File
bb (byte bucket)	:bb:	All
stream	:stream:	All

- Application Loader** The iRMX III AL is defined identically in the standard PC definition files.
- Human Interface** The iRMX HI is defined identically in the standard PC definition files.
- UDI** The iRMX III UDI is defined identically in the standard PC definition files. There are no configurable parameters to the UDI.



The Parameter Index is followed by a General Index

- (A30), 124
- (A50), 125
- (A54), 124
- (A74), 124
- (ABR), 67
- (AC), 126
- (ACE), 59
- (ACI), 75
- (ACO), 75
- (Actual name), 101
- (ADP), 59
- (AIB), 74
- (AL), 122
- (ALF), 81
- (ALN), 102
- (AND), 108
- (ANT), 108
- (ASC), 58
- (ATC), 125
- (ATI), 108
- (ATT), 98
- (B30), 124
- (B54), 124
- (B74), 124
- (BA), 127
- (BAA), 134
- (BAB), 88, 91
- (BAO), 88, 91
- (BBS), 99
- (BIF), 81
- (BIOS), 122
- (BIP), 74
- (BOF), 127
- (BON), 127
- (BS), 125
- (BUS), 73
- (BVC), 97
- (CBI), 90, 94, 97
- (CBN), 87, 91, 94, 97
- (CBS), 99
- (CBT), 95
- (CC), 88, 90, 91
- (CD), 67, 88, 89, 91, 103
- (CEBN), 90
- (CET), 87
- (CF), 73, 130
- (CIB), 103
- (CIL), 73, 130
- (CIN), 73
- (CIO), 103, 120
- (CLB), 122
- (CLC), 103
- (CLIB), 81
- (CLJ), 87, 93, 94, 97
- (CLN), 103
- (CMM), 132
- (CMP), 85
- (CN), 73, 130
- (CNM), 76
- (CNN), 103
- (CNP), 103
- (CON), 59, 124
- (COS), 71, 94
- (CPI), 119
- (CPN), 103
- (CRO), 119
- (CRS), 119
- (CS), 116
- (CSB), 88
- (CSO), 88
- (CST), 59
- (CTA), 95
- (CUT), 59
- (D51), 124
- (DDA), 75
- (DDP), 74

(DDS), 67, 71, 94, 95
 (DEF), 78
 (DEH), 115
 (DEV), 62, 63
 (Device Name), 100
 (device_name), 68
 (DFD), 68, 71
 (DHH), 116
 (DHI), 117
 (DIB), 75
 (DL), 125
 (DLN), 68
 (DMA), 127
 (DMP), 58
 (DN), 88, 90, 91
 (DO), 68
 (DOB), 75
 (DOM), 105
 (DPD), 98
 (DPN), 64, 68
 (DPT), 117
 (DRQ), 71, 94
 (DST), 107
 (DT), 107
 (DTF), 81
 (DTN), 77
 (DU), 70, 130
 (DUP), 64
 (EBP), 111
 (EBR), 111
 (EBS), 71, 94
 (EDF), 99
 (EEP), 111
 (EFD), 71
 (EHS), 70, 136
 (EIF), 81
 (EIOS), 122
 (EM), 70, 116, 136
 (EMU), 73
 (EOR), 119
 (EPN), 118
 (ER), 128
 (ESC), 123
 (FCT), 105
 (FD), 128
 (FIL), 131
 (file_driver), 68
 (FN), 88, 90, 91
 (FPI), 125
 (FSN), 97
 (FST), 104
 (G79), 124
 (GBR), 74
 (GC), 59
 (GCN), 60
 (GDR), 74
 (GSN), 118
 (GTW), 107
 (HBA), 88
 (HDP), 110
 (HI), 122
 (HID), 107
 (HIF), 81
 (high), 84
 (HNM), 107
 (HSE), 116
 (ICL), 76
 (IF), 73
 (IJD), 70, 130
 (IL), 62, 87, 91
 (ILF), 81
 (ILS), 108
 (IMA), 109
 (IMC), 109
 (IN), 107
 (IND), 109
 (INL), 109
 (INR), 109
 (INT), 127
 (IPP), 78
 (IS), 82
 (ISE), 109
 (ISL), 81
 (ISN), 107
 (ISS), 71, 94
 (ITI), 104
 (ITO), 109
 (ITP), 62, 67, 71, 94, 95
 (ITT), 109
 (JEI), 95
 (JST), 77
 (KTR), 73
 (LOO), 127
 (LBS), 89

(LCN), 71
 (LD), 87, 90, 91, 102
 (LDU), 59
 (LDV), 59
 (Level_Sensitive), 82
 (LI), 95
 (LIN), 110
 (LLN), 102
 (LO), 105
 (logical name), 68, 100, 101
 (logical_name), 79
 (low), 84
 (LPN), 102
 (LSE), 116
 (LUL), 102
 (MAX), 77, 78
 (MBS), 120
 (MC1)-(MC6), 104
 (MCE), 116
 (MCO), 74
 (MCT), 74
 (MD), 102
 (MDC), 74
 (MDL), 74
 (MDP), 74
 (MDS), 74
 (MDV), 98
 (MDW), 129
 (MFD), 71
 (MI), 126
 (MIN), 77, 78
 (MIP), 85
 (MIS), 110
 (MIW), 110
 (MJN), 87
 (MLN), 102
 (MNF), 81
 (MOB), 65, 86, 92, 94, 96, 106, 113, 114, 126,
 129, 130, 131, 134, 136
 (Module), 136
 (MP), 82
 (MPD), 98
 (MPN), 102
 (MPR), 65, 86, 92, 94, 96, 106, 113, 114, 126,
 129, 130, 131, 134, 136
 (MPS), 82
 (MQL), 110
 (MRR), 94
 (MS), 112
 (MSC), 95
 (MSE), 110
 (MSM), 117
 (MST), 117
 (MSW), 129
 (MTK), 65, 86, 92, 94, 96, 106, 113, 114, 126,
 129, 130, 131, 134, 136
 (MTP), 78
 (MTR), 108
 (MUL), 102
 (MWI), 110
 (N12), 113
 (N128), 113
 (N16), 112
 (N24), 113
 (N32), 113
 (N4), 112
 (N48), 113
 (N56), 113
 (N64), 113
 (N92), 113
 (N96), 113
 (NAM), 62, 63, 135
 (NAT), 98
 (NBB), 99
 (NC), 112, 127
 (NCB), 99
 (NCP), 132
 (ND), 64
 (NDA), 110
 (NDU), 64
 (NDV), 110
 (NEB), 116, 120
 (NEH), 116
 (NEM), 87, 89
 (NET), 81, 122
 (NFD), 71
 (NGE), 115
 (NIE), 115
 (NL), 87, 93, 120
 (NLB), 89
 (NM), 107, 112
 (NMI), 116
 (NO), 104
 (NOR), 95

(NOS), 95
 (NPX), 70, 77, 86, 93, 94, 97, 106, 114, 136
 (NQ), 112
 (NR), 105
 (NRB), 99
 (NS), 112, 126
 (NSB), 89, 99
 (NT), 112
 (NTI), 109
 (NTM), 117
 (NTR), 109
 (NUF), 81
 (NW), 112
 (ODS), 65, 86, 92, 93, 96, 106, 113, 114, 126,
 128, 129, 131, 133, 135
 (OE), 122
 (OFN), 93
 (OFP), 97
 (OFV), 97
 (OPN), 64
 (OSC), 59
 (Owner-ID), 69
 (owners-id), 68
 (path_name), 79
 (PCI), 125
 (PD), 108
 (PDT), 117
 (PE), 108
 (PGS), 122
 (PID), 89
 (PMA), 59, 70, 77, 86, 92, 94, 96, 106, 113,
 114, 126, 129, 130, 131, 134, 136
 (PMI), 59, 65, 67, 70, 77, 86, 92, 94, 96, 106,
 113, 114, 126, 129, 130, 131, 134, 136
 (PMT), 117
 (Port), 82
 (PP), 108
 (PQ), 108
 (Prefix), 79
 (PSC), 98
 (PT), 108
 (Public name), 101
 (PV), 66, 70, 86, 92, 94, 96, 106, 113, 114,
 115, 126, 129, 130, 131, 136
 (PVC), 97
 (PVL), 104
 (QS), 89
 (RAF), 72
 (RBA), 119
 (RBF), 72
 (RBS), 58
 (RCI), 124
 (RCJ), 85
 (RDA), 119
 (RDT), 89
 (RES), 127
 (RET), 105
 (RFT), 117
 (RIE), 116
 (RIP), 76
 (RMB), 72
 (RND), 110
 (ROD), 115
 (ROF), 72
 (RPA), 67
 (RQB), 87
 (RQO), 87
 (RRP), 116
 (RRT), 116
 (RSJ), 85
 (RSP), 105
 (RTI), 104
 (RU), 77
 (SBR), 132
 (SBS), 89, 99
 (SC), 97
 (SCF), 77
 (SCP), 132
 (SCT), 127
 (SD), 128
 (SDB), 122
 (SDF), 81
 (SDM), 122
 (SDN), 128
 (Separation), 82
 (SEQ), 133, 135
 (SGS), 134
 (SI), 127
 (SIA), 134
 (SJM), 82
 (SL0)-(SL7), 66
 (Slave_number), 82
 (SLV), 123
 (SMI), 59

(SN1), 93
 (SN2), 93
 (SN3), 93
 (SN4), 93
 (SND), 110
 (SON), 110
 (SR1)-(SR2), 124
 (SRC), 119
 (SS), 76
 (SSA), 70, 136
 (SSI), 66, 70, 86, 92, 94, 96, 106, 114, 126,
 129, 130, 131, 134, 136
 (SSK), 125
 (STA), 95
 (STO), 127
 (STP), 131
 (SYR), 119
 (SYS), 76
 (TBI), 110
 (TBU), 110
 (TCP), 85
 (TDN), 78
 (TF), 59
 (TN), 78
 (TND), 111
 (TOP), 64
 (TP), 66, 70, 73, 86, 92, 94, 96, 106, 114, 126,
 129, 130, 136
 (TPA), 130
 (TPR), 134
 (TPS), 73, 130
 (TSA), 70, 136
 (TSC), 59
 (TSP), 59
 (TTP), 59
 (TUP), 64
 (TVM), 110
 (UCS), 111
 (UDF), 81
 (UDI), 122
 (UHW), 111
 (UID), 78
 (UIN), 63
 (ULW), 111
 (UMH), 111
 (UML), 111
 (UN), 63
 (UND), 111
 (UPC), 120
 (UPD), 102
 (user name), 69
 (USS), 99
 (UVC), 97
 (UXC), 76
 (VAR), 70, 136
 (VBP), 117
 (VC), 97
 (VIE), 66
 (VSE), 119
 (VSG), 58
 (VSS), 119
 (WA), 102
 (WD), 98
 (WDI), 75
 (WDN), 132
 (WDP), 75
 (WDT), 76
 (WFS), 98
 (Who Attaches), 100
 (WIN), 110
 (WP), 87, 91
 (WTI), 98
 (WV), 102
 (XCJ), 85
 (XS), 127
 (XSJ), 85
 .a28, ASM286 source files, 16
 .a38, ASM386 source files, 16
 .cf, binder/builder configuration files, 16
 .csd, submit files, 16
 :home: logical name, 78
 <Ctrl-C> to abort commands, 19
 ?icdev.a38 file, 156
 ?itdev.a38 file, 156

A

abbr field, 155
 ABDR screen, Automatic Device Boot
 Recognition, 68
 Ability to Create Existing Files parameter, 59
 aborting commands, <Ctrl-C>, 19
 Actual name parameter, 101

- adding
 - device drivers, 151
 - example, 168
- ADMA Base Port Address parameter, 74
- ADMA Burst Register parameter, 74
- ADMA Channel for Input parameter, 75
- ADMA Channel for Output parameter, 75
- ADMA Delay Register parameter, 74
- ADMA parameter, 74
 - Data Port, 74
- AFA screen, Apex File Access, 98
- AFAU screen, Afa User, 99
- All System Calls parameter, 58
- APPL screen, Application Loader, 58
- Application Image Base Address parameter, 134
- Application Image Size parameter, 134
- Application Loader Includes and Lib parameter, 81
- Application Loader parameter, 122
- Application Starter Job Initial Memory parameter, 134
- Application Starter Job Maximum Memory parameter, 134
- Application Starter Job Objects parameter, 133
- Application Starter Job Order parameter, 133
- ARP NDEV parameter, 108
- ARP NTABSIZE parameter, 108
- ARP TIMEOUT parameter, 108
- ATC50 screen, ATCS/450 Server Job, 129
- ATCJ screen, ATCS/279/ARC Server, 128
- ATCS/279/ARC Server Job parameter, 125
- ATCS/279/ARC system job, 128
- ATCS/450 Server Job parameter, 125
- ATCS/450 system job, 129
- Attach Device Task Priority parameter, 59
- Attributes parameter, 98
- Auto Configuration parameter, 126
- Automatic Boot Device Recognition parameter, 67
- Automatic Loadname parameter, 102

B

- B command, 24, 32, 39
- back command, 32
- backup command, 24, 39

- Basic I/O System Includes and Libs parameter, 81
- BIOFD screen, BIOS File Drivers, 71
- BIOS parameter, 122
- BIOS Pool Maximum, Minimum parameter, 59
- BIOS screen, 59
- bld, builder files, 16
- Board Initialization Procedure parameter, 74
- Boot Address Base parameter, 88, 91
- Boot Address Offset parameter, 88
- Boot Address parameter, 91
- Bootsrv system job, 130, 131
- bootstrap loader, 67
- BSJ screen, MSA Bootsrv Job, 130
- buffer, command line, 76
- builder errors, 145
- builder warnings, 145
- Burst DMA parameter, 74

C

- C command, 20, 32
- C Library Files parameter, 81
- call gate, 118
- cancel command, 32
- cancelling editing session, 19
- case insensitivity, 19
- CDF Device parameter, 103
- CDF Location parameter, 103
- CDF Logical Name parameter, 103
- CDF Path Name parameter, 103
- CDF screen, Client Definition File, 103
- CEBI screen, 90
- CEBN screen, 90
- change command, 20
- changing
 - definition files, 20, 30
 - screens, description, 25
- Class Code parameter, 88, 90, 91
- CLI, 76, 78
- CLIB screen, Shared C Library, 120
- Client Info Address Base parameter, 103
- Client Info Address Offset parameter, 103
- Client Name parameter, 103
- Client Password parameter, 103
- Clock Frequency parameter, 73, 130
- Clock Interrupt Level parameter, 73, 130

Clock Interval parameter, 73
 clock tick, Nucleus, 59
 Comm Board Instance parameter, 90, 94, 97
 Comm Board Name parameter, 90, 94, 97
 Comm Start Address Base parameter, 88
 Comm Start Address Offset parameter, 88
 Command Buffer Size parameter, 99
 command mode, 17
 Command Name Length parameter, 76
 COMMengine board instance load method for MBII, 90
 COMMengine board Names, 90
 COMMengine Delay parameter, 88, 89, 91
 COMMengine Load parameter, 90
 COMMengine Type parameter, 87
 Common Update Timeout parameter, 59
 Communication Board Name parameter, 87, 91
 COMNT screen, Comments for Build File, 60
 Configuration Base Time parameter, 95
 Configuration Directory parameter, 67
 configuration files, generating, 36
 configuring custom device drivers, 151
 Connect Time Alarm parameter, 95
 Connection Job Delete Priority parameter, 59
 Connection Object Size parameter, 71, 94
 Console I/O parameter, 120
 Console Port parameter, 124
 Control-Sequence Translation parameter, 59
 copy command, 36
 copying current screen, 36
 Create Loadable Job parameter, 87, 93, 94, 97
 creating files for UDS, 153
 custom board initialization, 74
 Custom ROM External Procedures parameter, 119
 Custom ROM Init File parameter, 119

D

D command, 23, 33
 d01 field, 156
 d01 help field, 156
 DAG device used parameter, 75
 data chains, 116
 Datalink Data Size parameter, 95
 debugger, 122
 Default Directory parameter, 78

Default Exception Handler parameter, 115
 Default Host ID parameter, 117
 Default IO Job Directory Size parameter, 67
 Default Memory Pool Size parameter, 58
 Default Number of Port Transactions parameter, 117
 default prefix, 70
 Default System Device File Driver parameter, 68
 Default System Device Logical Name parameter, 68
 Default System Device Owner's ID parameter, 68
 Default System Device Physical Name parameter, 68
 Default Terminal Name parameter, 77
 Default User parameter, 70, 130
 definition files
 changing, 30
 editing, 20
 listing, example, 52
 restoring, 39
 saving, example, 52
 definition, ICU files, 16
 delete command, 33
 deleting screens, 33
 Deletion Task Priority parameter, 117
 Destination IP Address for SLIP parameter, 107
 Destination IP Address parameter, 107
 Destination Type parameter, 107
 detail-level command, 23
 dev aux field, 156
 development environment required, 149
 Development Tools Path Name parameter, 81
 Device Descriptor Size parameter, 71, 94
 device drivers
 adding without modifying the ICU, 165
 adding, example, 168
 configuring custom, 151
 device field, 155
 Device Name parameter, 62, 63, 88, 90, 91, 100
 device_name parameter, 68
 Device-Unit Information composite screen, 63
 Device-Unit Name parameter, 63
 DHH Exception Handler parameter, 116

- directories, :config:default, 144
- displaying
 - list of screens, 32
 - next screen, 32
 - previous screen, 32
- DLJ screen, Multibus II Downloader Job, 128
- DMA, 74
- DMA Input Buffer Size parameter, 75
- DMA Output Buffer Size parameter, 75
- DMA Transfer Burst Off parameter, 127
- DMA Transfer Burst On parameter, 127
- DOS File Driver parameter, 71
- DOSFD screen, DOS File Driver, 71
- Driver composite screen, 62
- driver field, 155
- Driver type parameter, 65
- driver.lib file, 151
- duib aux field, 157
- duib field, 157
- Duib Source Code Path Name parameter, 64
- DUIBs Required parameter, 71, 94
- Dynamic Public dirs parameter, 98

E

- E command, 22, 36
- editing
 - definition files, 20
 - definition files, example, 45, 46
 - screens, 31
 - screens, description, 25
- editing session, cancelling, 19
- EDL BOARDS parameter, 111
- EDL BUFS POSTED parameter, 111
- EDL ENDPOINTS parameter, 111
- EDOS File Driver parameter, 71
- EDSFD screen, EDOS File Driver, 71
- EIOS Buffer Size parameter, 71, 94
- EIOS parameter, 122
- EIOS Pool Minimum, Maximum parameter, 67
- EIOS screen, 67
- EMU, 73
- Enable interrupt virtualization parameter, 66
- Enable Screen Scrolling parameter, 123
- end field, 156
- Entry Point Name parameter, 118

- error messages, 137, 160
 - returned by BLD386, 145
- Error Reporting parameter, 128
- examples
 - adding device drivers, 168
 - deleting data on screen, 34
 - editing definition file, 45, 46
 - entering screen-editing mode, 43
 - generating a system, 53
 - help command, 44
 - ICU session, 41
 - inserting data on screen, 35
 - invocation, 42
 - restoring definition file, 39
- Exception Handler Entry Point parameter, 70, 136
- Exception Mode parameter, 70, 116, 136
- exit command, 22, 36
- Extended I/O System Includes and Libs parameter, 81
- Extension Data Field parameter, 99

F

- F 2- command, 32
- FC screen, File Consumer Configuration, 94
- File Consumer TSAP ID parameter, 105
- file driver, 68
- File Driver parameter, 128
- File Name parameter, 88, 90, 91
- File Server Name parameter, 97
- File Server TSAP ID parameter, 104
- file_driver parameter, 68
- find command, 32
- finding a screen, 32
- fixed screen format, 26
- Force DOS Filenames to Lower Case parameter, 71
- FPI Server Job parameter, 125
- FPIJ screen, FPI Server Job, 131
- Free Space Manager memory, 84, 116
- Front Panel Interrupt Level parameter, 131
- FS screen, File Server Configuration, 97
- FSM parameter, 84

G

G command, 20, 36, 53
Gateway IP Address parameter, 107
GDT entries, 115, 116
GDT Slot Number parameter, 118
GEN screen, Generate File Names, 72
generate command, 20, 36, 53
generate command, prefix option, 37
generating
 configuration files, 36
 ICU files, 20
 new system, 36
 system, example, 53
generating ICU files, 20
Global Clock Name parameter, 60
Global Clock parameter, 59

H

H command, 32
HARD screen, Hardware, 73
hardware required, 149
help command, 32
HI screen, Human Interface, 76
High GDT/LDT Slot Excluded from FSM
 parameter, 116
High RAM Address parameter, 119
HIJOB screen, Human Interface Jobs, 77
HILOG screen, Logical Names, 79
Host Buffer Address parameter, 88
HOSTID IP Address parameter, 107
HOSTNAME parameter, 107
Human Interface Includes and Libs
 parameter, 81
Human Interface parameter, 122
Human Interface Pool Minimum, Maximum
 parameter, 77

I

I command, 35
I/O Job Default Prefix parameter, 70, 130
I/O job memory parameters, 77
I/O Task Priority parameter, 71, 94, 95
I/O Task Size parameter, 94
I/O Task Stack Size parameter, 71

ICMPJ screen, iNA 960 COMMputer Job, 92
ICU

 invoking or starting, 17
 quitting or exiting, 22
ICU generation files, 16
icu386.scm file, 15
icu386.tpl file, 15
ICUMRG (ICU Merge) utility, 152
IDEVS screen, Intel Device Drivers, 61
IDT entries, 115
IFSP screen, Parameters for TCP/IP Network
 Interfaces, 107
IMIPJ screen, iNA MIP Driver Job, 86
iNA 960 COMMputer job, 92
iNA 960 COMMputer parameter, 85
iNA 960 MIP Driver parameter, 85
iNA COMM Port ID parameter, 89
iNA Network Layer parameter, 87, 93
iNA Subnet 1 parameter, 93
iNA Subnet 2 parameter, 93
iNA Subnet 3 parameter, 93
iNA Subnet 4 parameter, 93
iNA Subsystem Number parameter, 107
iNA system job, 86
INCL screen, Includes and Libraries, 81
Includes and Libs parameter, 81
including custom device drivers, 151
Initial Command Line Size parameter, 76
Initial Program Stack Size parameter, 76
Initial Task Priority parameter, 134
Initial Task Stack Size parameter, 134
initialization error reporting, 116
Initialize On-board Functions parameter, 73
Initial-Program Pathname parameter, 78
Initiator TSAP ID parameter, 104
insert command, 35
inserting
 data on screen, 35
 new line on screen, 35
 screen, 35
INT screen, Interrupts, 82
Intel Monitor Includes and Libs parameter, 81
Intel Support Libraries parameter, 81
interconnect register, NMI Enable, 116
Interface Libraries parameter, 81
Interface Name parameter, 107
Internal EIOS Tasks Priorities parameter, 67

- Interrupt Level parameter, 62, 87, 91
- Interrupt Slaves parameter, 82
- Interrupt Task Priority parameter, 62
- invocation
 - error messages, 138
 - example, 42
- invoking
 - ICU, 17
 - example, 42
- IOJOB screen, I/O Jobs, 70
- IOUS screen, I/O Users, 69
- IP LOCALSUBNETS parameter, 108
- IP MYAUTHORITY parameter, 109
- IP MYCLASS parameter, 109
- IP NDEV parameter, 109
- IP NLINK parameter, 109
- IP NROUTES parameter, 109
- IP SECURITY parameter, 109
- IP TOS parameter, 109
- IP TTL parameter, 109
- iRMX TCP/IP Job parameter, 85
- iRMXNET Client JOB parameter, 85
- iRMX-NET Files parameter, 81
- iRMX-NET Server Job parameter, 85

J

- Job Name parameter, 135
- Job Sequence parameter, 135
- Jobs Memory Minimum, Maximum parameter, 77

K

- Kernel Tick Ratio parameter, 73

L

- L command, 21
- Large Buffer Size parameter, 89, 99
- LDT entries, 116
- Level_Sensitive parameter, 82
- list command, 21
- listing
 - definition file, example, 52
 - screen contents, 21
- Load parameter, 87, 91

- Loadable Job Sync. Timeout parameter, 77
- Local UDF Device parameter, 102
- Local UDF Location parameter, 102
- Local UDF Logical Name parameter, 102
- Local UDF Path Name parameter, 102
- log files, 40
- Logical Name parameter, 100, 101
- logical_name parameter, 68, 79
- LOGN screen, Logical Names, 68
- Logoff Interval parameter, 95
- Low GDT/LDT Slot Excluded from FSM parameter, 116
- Low RAM Address parameter, 119

M

- main menu commands, 19
- Malloc block size parameter, 120
- Master Level 0-7 parameter, 66
- Master PIC port Separation parameter, 82
- Master UDF Device parameter, 102
- Master UDF Location parameter, 102
- Master UDF Logical Name parameter, 102
- Master UDF Path Name parameter, 102
- math coprocessor, 70, 77, 86, 93, 94, 97, 106, 114
- Max No. of Simultaneous Messages parameter, 117
- Max number of Public Devices parameter, 98
- Max number of Public Dirs parameter, 98
- Max. No. of Simultaneous Transactions parameter, 117
- Maximum Data Chain Elements parameter, 116
- Maximum Debug Windows parameter, 129
- Maximum Memory Required parameter, 78
- Maximum Number of Objects parameter, 104, 134
- Maximum Number of Responses parameter, 105
- Maximum Number of Retries parameter, 105
- Maximum Number of Tasks parameter, 134
- Maximum Objects parameter, 65, 86, 92, 94, 96, 106, 113, 114, 126, 129, 130, 131, 136
- Maximum Priority parameter, 65, 86, 92, 94, 96, 106, 113, 114, 126, 129, 130, 131, 136

Maximum Property Value Length parameter, 104
 Maximum RFD Request parameter, 94
 Maximum Server Connections parameter, 95
 Maximum System Windows parameter, 129
 Maximum Task Priority parameter, 78
 Maximum Tasks parameter, 65, 86, 92, 94, 96, 106, 113, 114, 126, 129, 130, 131, 136
 MBII Downloader Job parameter, 125
 MBII Downloader system job, 128
 MBII screen, Multibus II Hardware, 74
 MEMF screen, Memory for Free Space Manager, 84
 memory restrictions, 83
 MEMS screen, Memory for System, 84
 message
 address, 117
 fragment, 117
 simultaneous, 117
 solicited, 117
 trace, 117
 Message Device Base Port Address parameter, 74
 Message Device Port Separation parameter, 74
 Message Interrupt Level parameter, 74
 Message Task Priority parameter, 117
 Metric parameter, 108
 Minimum Memory Required parameter, 78
 MIP Configuration
 for MBI, 87
 for MBII, 89
 for PCs, 91
 MIP1 screen, 87
 MIP2 screen, 89
 modifying definition files, 20
 Module parameter, 136
 MPC, 74
 MSA BootServer Job parameter, 125
 MSGSEG parameter, 112
 Multibus I SPC RMX Level parameter, 126
 Multicast Address 1-6 parameter, 104

N

N command, 32
 name field, 155

Name of EX Handler Object Module parameter, 116
 Name of Generated MIP Job parameter, 87
 Name Server Local Only parameter, 105
 Named File Driver parameter, 71
 NAMFD screen, Named File Driver, 71
 naming generation files, 37
 NBLK1024 parameter, 113
 NBLK128 parameter, 113
 NBLK16 parameter, 112
 NBLK2048 parameter, 113
 NBLK32K parameter, 113
 NBLK4 parameter, 112
 NBLK4096 parameter, 113
 NBLK512 parameter, 113
 NBLK56 parameter, 113
 NBLK64 parameter, 113
 NBLK8192 parameter, 113
 NCALL parameter, 112
 NCOM screen, Nucleus Communication Service, 117
 NET screen, Network Subsystem, 85
 Netmask parameter, 107
 Network Access parameter, 122
 next command, 32
 NMI (Non-maskable interrupt), 123
 NMI Enable Byte parameter, 116
 NMI Exception Handler parameter, 116
 NMUXLINK parameter, 112
 non-resident user, 67
 NPX, 77, 86, 93, 94, 97, 106, 114
 NQUEUE parameter, 112
 NS, Name Server Configuration screen, 104
 NSDOM, Name Server Search Domains screen, 105
 NSTREAM parameter, 112
 NTIMEOUT parameter, 112
 NUC screen, Nucleus, 115
 Nucleus and Root Job Includes and Libs parameter, 81
 Nucleus clock tick, 59
 Nucleus Communication Service parameter, 116
 Number of AFA tasks parameter, 98
 Number of Binds per VC parameter, 97
 Number of Command Buffers parameter, 99
 Number of EIOS buffers parameter, 120

- Number of External Mailboxes parameter, 87, 89
- Number of GDT Entries parameter, 115
- Number of GDT Slots parameter, 134
- Number of IDT Entries parameter, 115
- Number of Large Buffers parameter, 89, 99
- Number of loadable Device Inits parameter, 59
- Number of loadable Devices parameter, 59
- Number of Open Files per process parameter, 97
- Number of Open Files per VC parameter, 97
- Number of Outstanding RFD status calls parameter, 95
- Number of Outstanding RFD system calls parameter, 95
- Number of PCI Server SCSI Controllers parameter, 126
- Number of Processes per VC parameter, 97
- Number of Rbs parameter, 99
- Number of Small Buffers parameter, 89, 99
- Number of Trace Messages parameter, 117
- Number of User Defined Devices parameter, 64
- Number of User Defined Device-Units parameter, 64
- Number of Users per VC parameter, 97
- Number of Virtual Circuits parameter, 97
- Numeric Emulator Processor parameter, 73
- Numeric Library parameter, 120
- Numeric Processor Extension Used parameter, 70, 77, 86, 93, 94, 97, 106, 114, 136
- NUMTIM parameter, 109
- NUMTRW parameter, 109
- NWAKEUP parameter, 112

O

- Object code pathname parameter, 65
- object directories, 67
- Object Directory Size parameter, 65, 86, 92, 93, 96, 106, 113, 114, 126, 128, 129, 131, 135
- One Cycle DMA parameter, 74
- OS Extension parameter, 122
- OSC controls, 59
- OSEXT screen, OS Extension, 118

- Owner-ID parameter, 69
- owners-id parameter, 68

P

- paging identity mapped memory, 84
- Paging Subsystem parameter, 122
- parameter, Job Exit Interval, 95
- Parameter Validation parameter, 66, 70, 86, 92, 94, 96, 106, 113, 114, 115, 126, 129, 130, 131, 136
- path_name parameter, 79
- Pathname for Object Modules parameter, 93
- PCDEV screen, PC Bus Drivers, 61
- PCI Server Job parameter, 125
- PCI system job, 126, 127
- PCIJ screen, PCI Server Job, 126
- PCISC screen, PCI Server Job, 127
- PDEV screen, Public Devices, 100
- PDIR screen, Public Directory, 101
- physical device, 68
- PIC, 73, 74
 - port, 82
 - slave, 82
- PIMM screen, Paging Identity Mapped Memory, 84
- PIT timer, 73
- Pool Maximum parameter, 65, 70, 86, 92, 94, 96, 106, 113, 114, 129, 130, 131, 136
- Pool Minimum parameter, 65, 70, 86, 92, 94, 96, 106, 113, 114, 129, 130, 131, 136
- Pool Minimum, Maximum parameter, 126
- Port parameter, 82
- port transaction, 117
- PREF screen, Prefixes, 79
- Prefix parameter, 79
- Print Spooler Commands parameter, 98
- PRI0 DELETION parameter, 108
- PRI0 EDLINA parameter, 108
- PRI0 PTMX parameter, 108
- PRI0 QSCHED parameter, 108
- PRI0 TIMEOUT parameter, 108
- priority
 - EIOS task, 67
 - task, 70
- Public name parameter, 101
- Public Variable Name parameter, 70, 136

Q

Q command, 22
Queue Size parameter, 89
quit command, 22

R

R command, 23, 33
RAM Code File Name parameter, 72
Random Access Device and Unit Source Code Path Name parameter, 64
Random Access Object Code Path Name parameter, 64
RAW NDEV parameter, 110
RCJ screen, iRMX-Net Client Job, 93
Read Buffer Size parameter, 58
Receive Fragment Failsafe Timeout parameter, 117
recovery user, 77, 78
redisplay command, 33
REM screen, Remote File Access, 95
REMFDF screen, Remote File Driver, 71
Remote Boot File Name parameter, 72
Remote Boot Translation parameter, 72
Remote Console Interface Driver parameter, 124
repetitive screen format, 27
repetitive-fixed screen format, 28
replace command, 23
Report Initialization Errors parameter, 116
Request Queue Addr Base parameter, 87
Request Queue Addr Offset parameter, 87
requirements
 hardware, 149
 software, 150
 subsystem, 121
RES screen, Resident/Recovery User, 78
Resident Initial Program parameter, 76
Resident User parameter, 77, 78
Responder TSAP ID parameter, 104
Response Delay Time parameter, 89
restoring, definition file, 39, 139
Retry Timeout parameter, 105
Return on Physical Attachdevice parameter, 67
returning to command mode, 32
ROM Code File Name parameter, 72

ROM Copied To RAM parameter, 119
ROM Init Code Base Address parameter, 119
ROM Init Code RAM Destination Address parameter, 119
ROM Init Code Size parameter, 119
ROM or FLASH Execution parameter, 119
ROM screen, ROM Code, 119
ROM-based system memory restrictions, 84
Root Object Directory parameter, 115
Round Robin Priority Threshold parameter, 116
Round Robin Time Quota parameter, 116
ROUTE screen, IP Router Information, 107
RSJ screen, iRMX-NET Server Job, 96

S

S command, 21, 33
save command, 21
saving definition file, example, 52
SBX 279 Console Controller Driver parameter, 124
screen
 displaying list of, 32
 editing commands, 31
 editing, description, 25
 elements, 29
 formats, 25
 PHYFD, Physical File Driver, 71
 STRFD, Stream File Driver, 71
screen-editing commands, 19
screen-editing mode, 19
 example, 43
screens, TUPJ, TUPJ Job, 133
SCSI Bus Reset parameter, 127
SCSI Bus Scan parameter, 127
SCSI Controller Base Address parameter, 127
SCSI Controller Interrupt parameter, 127
SCSI Controller Time Out parameter, 127
SCSI Controller Type parameter, 127
SCSI DMA Channel parameter, 127
SCSI Identification parameter, 127
SCSI Maximum Xfer Rate parameter, 127
SCSI Number of CCBs parameter, 127
SDB Interrupt Level parameter, 123
SDB screen, System Debugger, 123
SDM screen, System Debug Console, 123

- search command, 33
- searching for string in screen, 33
- Send Time Alarm parameter, 95
- Separation parameter, 82
- Serial Device Name parameter, 128
- Serial Port parameter, 124
- Server Concurrency parameter, 97
- Server Task Priority parameter, 131
- Shared C Library parameter, 122
- Slave Level 0-7 parameter, 66
- SLAVE screen, Slave Interrupt Levels, 82
- Slave_number parameter, 82
- SLIP NDEV parameter, 110
- Small Buffer Size parameter, 89, 99
- SoftScope Application Load Path parameter, 132
- SoftScope Communication parameter, 132
- SoftScope Kernel job, 132
- SoftScope Kernel parameter, 125
- SoftScope Serial Baud Rate parameter, 132
- SoftScope Serial Port parameter, 132
- SoftScope UDP Port parameter, 132
- software required, 150
- SOMOD NDEV parameter, 110
- SSKJ screen, SoftScope Kernel Job, 132
- Stack Segment Address parameter, 70, 136
- Stack Size parameter, 66, 70, 86, 92, 94, 96, 106, 114, 126, 129, 130, 131, 136
- starting the ICU, 17
- Storage Device Name parameter, 128
- STRM screen, Streams Parameters, 112
- SUB screen, Sub-systems, 122
- Subnet ID parameter, 105
- subsystem dependencies, 121
- SYS parameter, 84
- SYSJ screen, System Jobs, 125
- System Bus Type parameter, 73
- System Command File Name parameter, 77
- System Debug Monitor parameter, 122
- System Debugger Includes and Libs parameter, 81
- System Debugger parameter, 122
- system device, 68, 76
- System Directory parameter, 76
- system generation error messages, 145
- System in ROM parameter, 119
- System Jobs

- ATCS/279/ARC, 128
- ATCS/450, 129
- Bootserver, 130, 131
- iNA, 86, 92
- MBII Downloader, 128
- PCI, 126
- PCISC, 127
- SSKJ, 132
- TPUJ, 133
- System Jobs Object Modules parameter, 82
- system manager, 68
- System Manager ID parameter, 59
- system memory, 84

T

- Tape Support parameter, 59
- Task Entry Point parameter, 70, 136
- Task Priority ATCS/279/ARC Server job parameter, 129
- Task Priority ATCS/450 Server Job parameter, 130
- Task Priority I/O job parameter, 70
- Task Priority iNA MIP job parameter, 86, 92, 114
- Task Priority iRMX-NET job parameter, 94, 96, 106
- Task Priority MSA Bootserver job parameter, 130
- Task Priority parameter, 66
- Task Priority PCI Server job parameter, 126
- Task Priority User job parameter, 136
- task, EIOS priority, 67
- TCP BSD42 ISS parameter, 110
- TCP BSD42 URGENT parameter, 110
- TCP HEADERPRED parameter, 110
- TCP LINGER parameter, 110
- TCP MAXQLEN parameter, 110
- TCP MAXSEG parameter, 110
- TCP MAXWINDOW parameter, 110
- TCP MINSEG parameter, 110
- TCP MINWINDOW parameter, 110
- TCP NDEV parameter, 110
- TCP NODELAYACK parameter, 110
- TCP TVMSL parameter, 110
- TCP WINDOW parameter, 110

TCP1 screen, TCP/IP Tunable Parameters 1, 108

TCP2 screen, TCP/IP Tunable Parameters 2, 109

TCP3 screen, TCP/IP Tunable Parameters 2, 111

TCPJ screen, iRMX TCP/IP Job, 106

TELNET NDEV parameter, 111

templ_1.uds file, 153

templ_2.uds file, 153

Terminal Device and Unit Information Names parameter, 64

Terminal Device and Unit Source Code Path Name parameter, 64

Terminal Device Name parameter, 78

Terminal Name parameter, 78

Terminal Object Code Path Name parameter, 64

Terminal OSC Controls parameter, 59

Terminal Support Code parameter, 59

terminal.lib file, 151

testing system after using ICU, 38

timeout value parameter, 76

Timer Counter Number parameter, 73, 130

Timer Port Separation parameter, 73, 130

Timer Task Priority parameter, 59

Timing Facilities Required parameter, 59

transmission interval parameter, 75

TSC, 59

Two Cycle DMA parameter, 74

U

u01 field, 157

u01 help field, 157

UDDM screen, UDS Device Driver Modules, 65

UDF screen, User Definition File, 102

UDI parameter, 122

UDP CKSUM parameter, 111

UDP HIWATER parameter, 111

UDP LOWATER parameter, 111

UDP MAXHIWATER parameter, 111

UDP MAXLOWATER parameter, 111

UDP NDEV parameter, 111

UDS (User Device Support) utility, 152

uds.scn file, 159

unit field, 156

Unit Info Name parameter, 62, 63

Unit Information Screen composite screen, 62

Unit Number on this Device parameter, 63

Update UDF parameter, 102

User Console I/O Procedure parameter, 120

User Device Support utility, see UDS

User Extension for Intel parameter, 76

User ID Number parameter, 78

user name parameter, 69

User Subsystem parameter, 99

USERD screen, User Devices, 64

USERJ screen, User Jobs, 135

USERM screen, User Modules, 136

V

Validate Buffer Parameters parameter, 117

validation of parameter, enabling, 115

version control messages, 138

version number

- files, 138
- specifying, 155

version numbers, files, 138

Virtual Segment Size parameter, 58

W

Wakeup Port parameter, 87, 91

warning messages, 137

Watchdog Timer used parameter, 75

Who Attaches parameter, 100, 102

Who Verifies parameter, 102

Wildcard Delete parameter, 98

Wildcard File Search parameter, 98

Wildcard Wait Interval parameter, 98

X

X25 Client Job parameter, 85

X25 Server Job parameter, 85

X25CJ screen, X.25 Client Job, 114

X25SJ screen, X.25 Server Job, 113

xcmdrv.lib file, 168

WE'D LIKE YOUR OPINION

Please rate the following:	Excellent	Good	Fair	Poor
■ Manual organization	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
■ Technical accuracy	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
■ Completeness	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
■ Clarity of concepts and wording	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
■ Quality of examples and illustrations	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
■ Overall ease of use	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

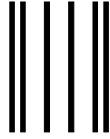
Comments: _____

Please list any errors you found (include page number): _____

Name _____
Company Name _____
Address _____

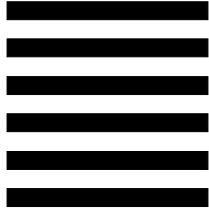
May we contact you? _____ Phone _____

Thank you for taking the time to fill out this form.



**NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES**

BUSINESS REPLY MAIL
FIRST CLASS PERMIT NO. 79 HILLSBORO, OR



POSTAGE WILL BE PAID BY ADDRESSEE

**OPD Technical Publications, HF2-72
Intel Corporation
5200 NE Elam Young Parkway
Hillsboro, OR 97124-9978**



Please fold here and close the card with tape. Do not staple.

WE'D LIKE YOUR COMMENTS....

This document is one of a series describing Intel products. Your comments on the other side of this form will help us produce better manuals. Each reply will be reviewed. All comments and suggestions become the property of Intel Corporation.

If you are in the United States and are sending only this card, postage is prepaid.

If you are sending additional material or if you are outside the United States, please insert this card and any enclosures in an envelope. Send the envelope to the above address, adding "United States of America" if you are outside the United States.

Thanks for your comments.

International Sales Offices

AUSTRALIA

Intel Australia Pty. Ltd.
Unit 1A
2 Aquatic Drive
Frenchs Forest, NSW, 2086
Sydney

Intel Australia Pty. Ltd.
711 High Street
1st Floor
East Kw. Vic., 3102
Melbourne

BRAZIL

Intel Semicondutores do Brazil LTDA
Avenida Paulista, 1159-CJS 404/405
CEP 01311-Sao Paulo - S.P.

CANADA

Intel Semiconductor of Canada, Ltd.
999 Canada Place
Suite 404, #11
Vancouver V6C 3E2
British Columbia

Intel Semiconductor of Canada, Ltd.
2650 Queensview Drive
Suite 250
Ottawa K2B 8H6
Ontario

Intel Semiconductor of Canada, Ltd.
190 Attwell Drive
Suite 500
Rexdale M9W 6H8
Ontario

Intel Semiconductor of Canada, Ltd.
1 Rue Holiday
Suite 115
Tour East
Pt. Claire H9R 5N3
Quebec

CHINA/HONG KONG

Intel PRC Corporation
China World Tower, Room 517-518
1 Jian Guo Men Wai Avenue
Beijing, 100004
Republic of China

Intel Semiconductor Ltd.
32/F Two Pacific Place
88 Queensway
Central
Hong Kong

FINLAND

Intel Finland OY
Ruusilantie 2
00390 Helsinki

FRANCE

Intel Corporation S.A.R.L.
1, Rue Edison-BP 303
78054 St. Quentin-en-Yvelines
Cedex

GERMANY

Intel GmbH
Dornacher Strasse 1
85622 Feldkirchen bei Muenchen
Germany

INDIA

Intel Asia Electronics, Inc.
4/2, Samrah Plaza
St. Mark's Road
Bangalore 560001

ISRAEL

Intel Semiconductor Ltd.
Atidim Industrial Park-Neve Sharet
P.O. Box 43202
Tel-Aviv 61430

ITALY

Intel Corporation Italia S.p.A.
Milanofiori Palazzo E
20094 Assago
Milano

JAPAN

Intel Japan K.K.
5-6 Tokodai, Tsukuba-shi
Ibaraki, 300-26

Intel Japan K.K.
Hachioji ON Bldg.
4-7-14 Myojin-machi
Hachioji-shi, Tokyo 192

Intel Japan K.K.
Bldg. Kumagaya
2-69 Hon-cho
Kumagaya-shi, Saitama 360

Intel Japan K.K.
Kawa-asa Bldg.
2-11-5 Shin-Yokohama
Kohoku-ku, Yokohama-shi
Kanagawa, 222

Intel Japan K.K.
Ryokuchi-Eki Bldg.
2-4-1 Terauchi
Toyonaka-shi, Osaka 560

Intel Japan K.K.
Shinmaru Bldg.
1-5-1 Marunouchi
Chiyoda-ku, Tokyo 100

Intel Japan K.K.
Green Bldg.
1-16-20 Nishiki
Naka-ku, Nagoya-shi
Aichi 460

KOREA

Intel Korea, Ltd.
16th Floor, Life Bldg.
61 Yoido-dong, Youngdeungpo-
Ku
Seoul 150-010

MEXICO

Intel Technologica de Mexico
S.A. de C.V.
Av. Mexico No. 2798-9B, S.H.
44620 Guadalajara, Jal.,

NETHERLANDS

Intel Semiconductor B.V.
Postbus 84130
3009 CC Rotterdam

RUSSIA

Intel Technologies, Inc.
Kremenchugskaya 6/7
121357 Moscow

SINGAPORE

Intel Singapore Technology, Ltd.
101 Thomson Road #08-03/06
United Square
Singapore 1130

SPAIN

Intel Iberia S.A.
Zurbaran, 28
28010 Madrid

SWEDEN

Intel Sweden A.B.
Dalvagen 24
171 36 Solna

TAIWAN

Intel Technology Far East Ltd.
Taiwan Branch Office
8th Floor, No. 205
Bank Tower Bldg.
Tung Hua N. Road
Taipei

UNITED KINGDOM

Intel Corporation (U.K.) Ltd.
Pipers Way
Swindon, Wiltshire SN3 1RJ

If you need to contact Intel Customer Support

Contacting us is easy. Be sure that you have the following information available:

- Your phone and FAX numbers ready
- Complete description of your hardware or software configuration(s)
- Your product's product code
- Current version of all software you use
- Complete problem description

Type of Service		How to contact us
FaxBACK* fax-on-demand system 24 hrs a day, 7 days a week	Using any touch-tone phone, have technical documents sent to your fax machine. Know your fax number before calling.	U.S. and Canada: (800) 628-2283 (916) 356-3105 Europe: +44-1793-496646
Intel PC and LAN Enhancement Support BBS 24 hrs a day, 7 days a week	Information on products, documentation, software drivers, firmware upgrades, tools, presentations, troubleshooting.	U.S and Canada: (503) 264-7999 Europe: +44-1793-432955 Autobaud detect 8 data bits, no parity, 1 stop
CompuServe* Information Service 24 hrs a day, 7 days a week	Worldwide customer support: information and technical support for designers, engineers, and users of 32-bit iRMX® OS and Multibus product families.	Worldwide Locations: (check your local listing) Type: GO INTEL C once online.
Customer Support	Intel Multibus Support engineers offering technical advice and troubleshooting information on the latest Multibus products.	U.S. and Canada: (800) 257-5404 (503) 696-5025 FAX: (503) 681-8497 Hrs: M-F; 8-5 PST Europe: +44-1793-641469 FAX: +44-1793-496385 Hrs: M-F; 9-5:30 GMT
Hardware Repair	Multibus board and system repair.	U.S. and Canada: (800) 628-8686 (602) 554-4904 FAX: (602) 554-6653 Hrs: M-F; 7-5 PST Europe: +44-1793-403520 FAX: +44-1793-496156 Hrs: M-F; 9-5:30 GMT
Sales	Intel Sales engineers offering information on the latest iRMX and Multibus products and their availability.	Worldwide: Contact your local Intel office or distributor U.S. and Canada: (800) 438-4769 (503) 696-5025 FAX: (503) 681-8497 Hrs: M-F; 8-5 PST
Correspondence Mail letters to:	Worldwide: Intel Customer Support Mailstop HF3-55 5200 NE Elam Young Parkway Hillsboro, Oregon 97124-6497	Europe: European Application Support Intel Corporation (UK) Ltd. Pipers Way Swindon, Wiltshire England SN3 1RJ

* Third-party trademarks are the property of their respective owners.