OS-9 Device Descriptor and Configuration Module Reference

Version 3.1

www.radisys.com

World Headquarters
5445 NE Dawson Creek Drive • Hillsboro, OR
97124 USA
Phone: 503-615-1100 • Fax: 503-615-1121
Toll-Free: 800-950-0044

International Headquarters
Gebouw Flevopoort • Televisieweg 1A
NL-1322 AC • Almere, The Netherlands
Phone: 31 36 5365595 • Fax: 31 36 5365620

RadiSys Microware Communications Software Division, Inc.
1500 N.W. 118th Street
Des Moines, Iowa 50325
515-223-8000

Revision A
November 2000

Copyright and publication information

This manual reflects version 3.1 of OS-9.

Disclaimer

The information contained herein is believed to be accurate as of the date of publication. However, RadiSys Corporation will not be liable for any damages including indirect or consequential, from use of the OS-9 operating system, Microware-provided software, or reliance on the accuracy of this documentation. The information contained herein is subject to change without notice.

Reproduction notice

The software described in this document is intended to be used on a single computer system. RadiSys Corporation expressly prohibits any reproduction of the software on tape, disk, or any other medium except for backup purposes. Distribution of this software, in part or whole, to any other party or on any other system may constitute copyright infringements and misappropriation of trade secrets and confidential processes which are the property of RadiSys Corporation and/or other parties. Unauthorized distribution of software may cause damages far in excess of the value of the copies involved.

November 2000

Table of Contents

Chapter 1: Low-Level System Configuration Module (cnfgdata)7

Chapter 2: OS-9 Configuration Module (init)99

Chapter 3: SCF Device Descriptors169

Chapter 4: SBF Device Descriptors399

Chapter 1: Low-Level System Configuration Module (cnfgdata)

This chapter includes the following topics:

Overview

The cnfgdata module contains configuration data used by the low-level system modules. The following subsystems are configured in the cnfgdata module:

- Low-level system console
- Low-level auxiliary communication
- Debugger
- Low-level protocol manager and interface data
- Booters and boot services
- Notification services

The next section in this chapter provides a detailed example of the configuration options you can use to change configuration values for this module.

The rest of this chapter provides a detailed list of all available cnfgdata module fields, including a field description and available values.

cnfgdata Module Field Configuration Options

There are two methods you can use to change a `cnfgdata` module configuration field:

1. Use the `EditMod` utility to directly modify existing `cnfgdata` modules either as a stand-alone module or as part of a merged module group (such as a boot image).
2. Modify the description file for the `cnfgdata` module and rebuild it using the makefile provided.

Direct Modification Advantages

The direct modification method has the following advantages:

Fast	No source configuration file rebuilds are necessary.
Temporary	The original module or merged-module group configuration can be easily restored via the appropriate rebuild.
Contained	Changes are limited to the individual boot image modified (merged-module option).

Description File/Rebuild Advantages

The advantage of the description file/rebuild method is the changes are permanent and reproducible. Modifications apply to all subsequent module rebuilds and to all merged-module groups built containing the updated module.

Both methods are documented in this section. These procedures are used with the field descriptions starting with the **Module Header Fields**. For direct modification, use the `EditMod` LABELS data to navigate the

EditMod menus. The DESCRIPTION FILE MACRO data identifies the macro you need to define/modify in the configuration sources to rebuild the cnfgdata module.

Direct Modification

Use the Editmod utility and the following procedures to directly modify fields in the existing cnfgdata module. The module can stand-alone or it can be part of a merged-module group. A boot image, for example, contains multiple modules. Both situations are covered in this section. The field references later in this chapter contain a description of each configurable field, its supported values, and the sequence of menu options required by EditMod to modify that field.

For More Information

Refer to the ***Utilities Reference*** for a full description of EditMod's capabilities.

Figure 1-1 Directory Location for Modifying the cnfgdata Module as a Stand-alone Module

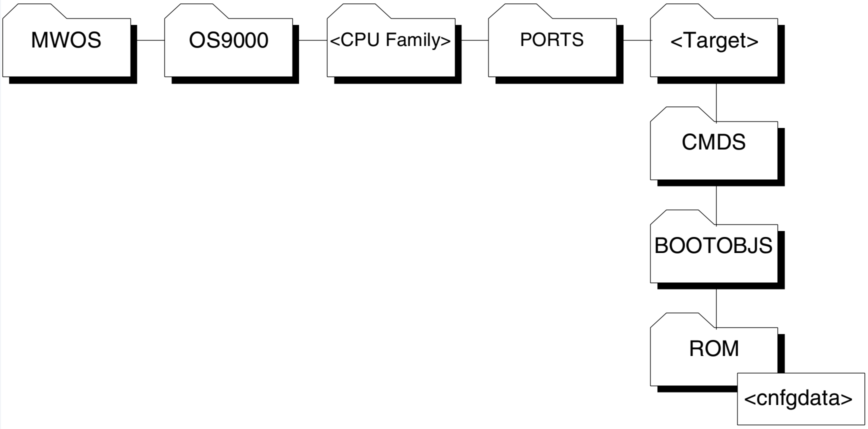
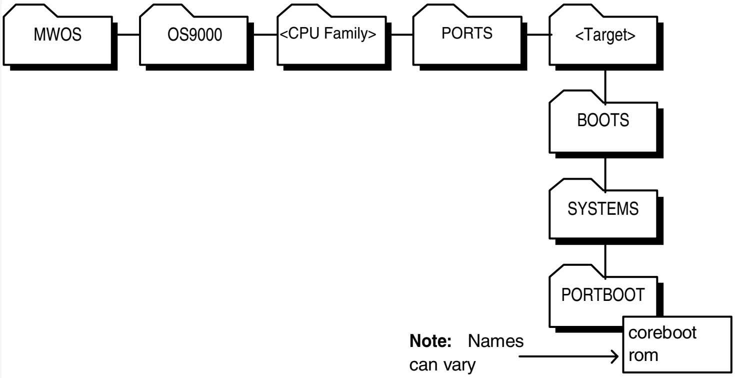


Figure 1-2 Directory Location for Modifying the cnfgdata Module as Part of a Boot Image



Refer to your board guide for information about how to modify the module lists and remake the boot images, and for specific boot image names.

Direct Modification Procedures

To modify the stand-alone module, complete the following steps:

Step 1. Change to the `CMDS/BOOTOBJS/ROM` directory (see **Figure 1-1**).

Step 2. Use EditMod to edit the module:

```
$EditMod -e -dc_all cnfgdata
```

To modify the module as part of a merged module group, complete the following steps:

Step 1. Change to the `BOOTS/SYSTEMS/PORTBOOT` directory (see **Figure 1-2**).

Step 2. Use EditMod to edit the module:

```
$EditMod -e -dc_all cnfgdata -f=<boot image name>
```

Step 3. Use the menu selections provided in the `EditMod` LABELS section of the field reference later in this chapter to locate the fields you want to edit.

Step 4. Select a new value for the field from the AVAILABLE VALUES section of the field reference. Enter that value at the EditMod prompt to modify the field.

Step 5. If you want to make additional modifications, use the `p` command (previous) to step backward through the EditMod menus. Repeat Steps 3 and 4 until you have made all desired modifications to the `cnfgdata` module.

Step 6. Select the `w` command (write) to save the changes.

Step 7. Select the `q` command (quit) to exit `EditMod`.

Note

Unless you modified the `cnfgdata` module in your boot image, you should rebuild your boot image to include the new `cnfgdata` module.

Example EditMod Session

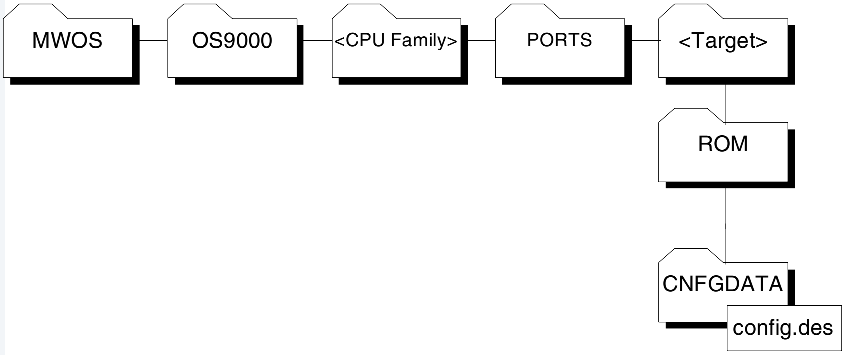
This example modifies `cnfgdata` as part of the boot image `rom`.

```
$ EditMod -e -dc_all cnfgdata -f=rom

 1. Module header
 2. Configuration data

$Which? [?/1-2/p/t/a/w/q] 2

 1. Console port data structure
 2. Communication port data structure
 3. Debugger data structure
 4. Low level protocol manager data structure
 5. Boot services data structure
 6. Notification services data structure

$Which? [?/1-6/p/t/a/w/q]
.
.  (desired modifications)
.
Which? [?/1-19/p/t/a/w/q] w

Which? [?/1-19/p/t/a/w/q] q
```

Description File Modification

You can use these procedures to modify the `cnfgdata` description file and rebuild the `cnfgdata` modules for your port directory. The DESCRIPTION FILE MACROS section of the field reference specifies the name of the macro you modify/define in the description file to configure the field. The value used in the define is chosen from the AVAILABLE VALUES specified for the field.

Figure 1-3 Directory Location for Modifying the cnfgdata Description Files



Description File Modification Procedures

Step 1. Change to the `ROM/CNFGDATA` directory (see **Figure 1-3**).

Step 2. Edit the file `config.des` and read the included comments for more information on using the specific description file provided in your software distribution. The `config.des` file contains a list of macro names which can be defined to override the global default values for the configuration fields.

Step 3. Refer to the DESCRIPTION FILE MACRO section in the field reference later in this chapter to determine the macro name you define to configure the target field.

Step 4. Read the comments in `config.des` to determine where to place the define for this macro.

Step 5. Select the value you want to use to configure the field. See the AVAILABLE VALUES section of the field reference data for values or macros that can be used for the definition. Define the macro by entering a definition in the appropriate description files as follows:

```
#define <macro> <value>
```

Step 6. Save the changes and rebuild the module by entering the following command from the `ROM/CNFGDATA` directory:

```
os9make
```

Step 7. Rebuild your boot image to include the new `cnfgdata` module.

Low-Level Configuration Module Field Reference

This section contains a list of all configurable fields in the cnfgdata module. Each field entry contains the following information:

- <Field name> - The call name for each field that can be reconfigured in the module.
- EditMod LABELS - EditMod menu selections for navigating to the proper field in an EditMod session.
- DESCRIPTION FILE MACRO - The macro name you modify/define in the description file.
- DESCRIPTION - A brief description of the field's purpose and use.
- EXAMPLE - An optional example of the description file entry showing how to change the value of this field.
- PORT GENERIC DEFAULT VALUE - The value set in the port generic description file for this field. This is the value the field is assigned when the module is built, unless the appropriate macro has been defined in the port specific description file to override this default value.
- PORT SPECIFIC OVERRIDE VALUE - The value set in the port specific description file for this field. If defined, this is the value the field is assigned when the module is built, overriding the port generic default value.
- AVAILABLE VALUES - Values to which the field can be set through EditMod or the description files. In many cases, this data is presented in a table that maps a description of the value to a numeric value appropriate for entry in EditMod, and to a pre-defined macro available for use in the description file.

The cnfgdata module consists of a module header and six distinct sections of configuration data. Each section is used by a specific low-level sub-system. The reference data in this chapter is divided into sections based on sub-system.

Module Header Fields

The following section contains the module header fields in the order they appear during an interactive EditMod session. Defined fields can appear in a different order in config.des.

Table 1-1 Module Header Fields

Field	Description File Macro
_m_group	MH_GROUP
_m_user	MH_USER
mod_name	MH_NAME
m_access	MH_ACCESS
m_tylan	MH_TYLAN
m_attrev	MH_ATTREV
m_edit	MH_EDIT

EditMod Labels

```
1-module header
1-module owner’s group number
```

Description

Group ID of the module’s owner. The group number allows people working in the same department or on the same project to share a common identification number.

Port Generic Default Value

0 (zero)

Port Specific Override Value

Refer to ROM/CNFGDATA/config.des (**Figure 1-3**).

Available Values

0 to 65535

_m_user

MH_USER

EditMod Labels

```
1-module header
2-module owner’s user number
```

Description

User ID of the module’s owner. The user number identifies a specific user.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `ROM/CNFGDATA/config.des` (**Figure 1-3**).

Available Values

0 to 65535

mod_name

MH_NAME

EditMod Labels

```
1-module header
3-module name
```

Description

Contains the module name string.

Port Generic Default Value

```
NULL
```

Port Specific Override Value

Refer to `ROM/CNFGDATA/config.des` (**Figure 1-3**).

Available Values

Any ASCII character string. The string may contain C-style character escapes (such as `\n` and `\012`).

m_access

MH_ACCESS

EditMod Labels

```
1-module header
4-access permissions
```

Description

Defines the permissible module access by its owner or by other users.

Port Generic Default Value

Macro

```
MP_OWNER_READ | MP_OWNER_EXEC | MP_GROUP_READ |
MP_GROUP_EXEC | MP_WORLD_READ | MP_WORLD_EXEC
```

EditMod

```
0x555
```

Port Specific Override Value

Refer to `ROM/CNFGDATA/config.des` (**Figure 1-3**).

Available Values

Module access permission values are located in the header file, `module.h`, and are listed in **Table 1-2**.

Table 1-2 m_access Available Values

Description	Macro	EditMod
Read permission by owner	`MP_OWNER_READ`	`0x0001`
Write permission by owner	`MP_OWNER_WRITE`	`0x0002`
Execute permission by owner	`MP_OWNER_EXEC`	`0x0004`

Table 1-2 m_access Available Values (continued)

Description	Macro	EditMod
Owner permission mask	`MP_OWNER_MASK`	`0x000f`
Read permission by group	`MP_GROUP_READ`	`0x0010`
Write permission by group	`MP_GROUP_WRITE`	`0x0020`
Execute permission by group	`MP_GROUP_EXEC`	`0x0040`
Group permission mask	`MP_GROUP_MASK`	`0x00f0`
Read permission by world	`MP_WORLD_READ`	`0x0100`
Write permission by world	`MP_WORLD_WRITE`	`0x0200`
Execute permission by world	`MP_WORLD_EXEC`	`0x0400`
World permission mask	`MP_WORLD_MASK`	`0x0f00`
All permissions for owner, group, and world	`MP_WORLD_ACCESS`	`0x0777`
System permission mask	`MP_SYSTM_MASK`	`0xf000`

m_tylan

MH_TYLAN

EditMod Labels

```
1-module header
5-type/language
```

Description

Contains the module's type (first byte) and language (second byte). The language codes indicate if the module is executable and which language the run-time system requires for execution, if any.

Port Generic Default Value

Macro

```
(MT_DATA<<8) + ML_OBJECT
```

EditMod

```
0x401
```

Port Specific Override Value

Refer to ROM/CNFGDATA/config.des (Figure 1-3).

Available Values

Module type values and language codes are located in the header file, module.h, and are listed in Table 1-3 and Table 1-4.

Table 1-3 m_tylan Available Module Type Values

Description	Macro	EditMod
Not used (wildcard value in system calls)	MT_ANY	0x0000
Program module	MT_PROGRAM	0x0001

Table 1-3 m_tylan Available Module Type Values (continued)

Description	Macro	EditMod
Subroutine module	MT_SUBROUT	0x0002
Multi-module (reserved for future use)	MT_MULTI	0x0003
Data module	MT_DATA	0x0004
Configuration data block data module	MT_CDBDATA	0x0005
Reserved for future use	0xb-0xa	0xb-0xa
User trap library	MT_TRAPLIB	0x000b
System module	MT_SYSTEM	0x000c
File manager module	MT_FILEMAN	0x000d
Physical device driver	MT_DEVDRVR	0x000e
Device descriptor module	MT_DEVDESC	0x000f
User definable	0x10-0xfe	0x10-0xfe
Module type mask	MT_MASK	0xff00

Table 1-4 m_tylan Available Language Code Values

Description	Macro	EditMod
Unspecified language (wildcard in system calls)	ML_ANY	0x0
Machine language	ML_OBJECT	0x1
Basic I-code (reserved for future use)	ML_ICODE	0x2
Pascal P-code (reserved for future use)	ML_PCODE	0x3
C I-code (reserved for future use)	ML_CCODE	0x4
Cobol I-code (reserved for future use)	ML_CBLCODE	0x5
Fortran	ML_FRTNCODE	0x6
Reserved for future use	0x7-0xf	0x7-0xf
User-definable	0x10-0xfe	0x10-0xfe
Module language mask	ML_MASK	0x00ff

m_attrev

MH_ATTREV

EditMod Labels

```
1-module header
6-revision/attributes
```

Description

Contains the module’s attributes (first byte) and revision (second byte).

Port Generic Default Value

Macro

```
MA_REENT<<8
```

EditMod

```
0x8000
```

Port Specific Override Value

Refer to `ROM/CNFGDATA/config.des` (**Figure 1-3**).

Available Values

Module attribute and revision codes are located in the header file `module.h.`, and are listed in **Table 1-5**.

Note

If two modules with the same name are found in the memory search or are loaded into the current module directory, only the module with the highest revision level is kept. This enables easy substitution of modules for update or correction.

Table 1-5 `m_attrev` **Available Attribute and Revision Values**

Description	Macro	EditMod
The module is re-entrant (sharable by multiple tasks).	`MA_REENT` (shifted left to first byte: `MA_REENT<<8)`	`0x80` (shifted left to first byte: `0x8000`)
The module is sticky. A sticky module is not removed from memory until its link count becomes -1 or memory is required for another use.	`MA_GHOST` (shifted left to first byte: `MA_GHOST<<8)`	`0x40` (shifted left to first byte: `0x4000`)
The module is a system-state module.	`MA_SUPER` (shifted left to first byte: `MA_SUPER<<8)`	`0x20` (shifted left to first byte: `0x2000`)
User-definable revision number	`0x0-0xfe`	`0x0-0xfe`
Module attribute mask	`MA_MASK`	`0xff00`
Module revision mask	`MR_MASK`	`0x00ff`

EditMod Labels

`1-module header`
`7-edition`

Description

Indicates the software release level for maintenance. OS-9 does not use this field. Whenever a program is revised (even for a small change), increase this number. We recommend internal documentation within the source program be keyed to this system.

Port Generic Default Value

`1`

Port Specific Override Value

Refer to `ROM/CNFGDATA/config.des` (**Figure 1-3**).

Available Values

`0` to `65535`

Console Device Fields

The console device fields are in the order they appear during an interactive EditMod session. Defined fields may appear in a different order in config.des. The field values can be changed using the EditMod utility or by modifying the config.des description file. See **cnfgdata Module Field Configuration Options** for detailed instructions on changing these fields.

Table 1-6 Console Device Fields

Field	Description File Macro
console_name	CONS_NAME
cons_vector	CONS_VECTOR
cons_priority	CONS_PRIORITY
cons_level	CONS_LEVEL
cons_timeout	CONS_TIMEOUT
cons_parity	CONS_PARITY
cons_baudrate	CONS_BAUDRATE
cons_wordsize	CONS_WORDSIZE
cons_stopbits	CONS_STOPBITS
cons_flow	CONS_FLOW

console_name

CONS_NAME

EditMod Labels

```
2-configuration data
1-console port data structure
1-console port name
```

Description

Contains the console device name string.

Macro Example

```
#define CONS_NAME “iovcons”
```

Port Generic Default Value

```
NULL
```

Port Specific Override Value

Refer to `ROM/CNFGDATA/config.des` (**Figure 1-3**).

Available Values

Any ASCII character string. The string may contain C-style character escapes (such as `\n` and `\012`).

cons_vector

CONS_VECTOR

EditMod Labels

```
2-configuration data
1-console port data structure
2-interrupt vector number
```

Description

This is the vector number of the console device passed to the processor at interrupt time.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `ROM/CNFGDATA/config.des` (**Figure 1-3**).

Available Values

`0` to `4294967295`

cons_priority

CONS_PRIORITY

EditMod Labels

```
2-configuration data
1-console port data structure
3-interrupt priority
```

Description

This is the software (polling) priority for the console device on the IRQ polling table.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `ROM/CNFGDATA/config.des` (**Figure 1-3**).

Available Values

The interrupt priority value range is 0-65534 (65535 is reserved). A non-zero priority determines the position of the device within the vector. Lower values are polled first.

Some considerations to keep in mind when selecting an interrupt priority:

- A priority of 0 indicates the device desires exclusive use of the vector.
- If the priority is 1, it is polled first and no other device can have a priority of 1 on the vector. For all other priority values, more than one device can share the same priority on a vector. In this case, first-in, first-out (FIFO) scheduling determines the order of precedence in the polling table for the devices.
- OS-9 does not allow a device to claim exclusive use of a vector if another device has already been installed on the vector. Additionally, it does not allow another device to use the vector once the vector has been claimed for exclusive use.

- This value is software dependent.

See Also

`F_IRQ` system call entry in the ***OS-9 Technical Manual***.

cons_level

CONS_LEVEL

EditMod Labels

```
2-configuration data
1-console port data structure
4-interrupt level
```

Description

This is the hardware priority of the console device interrupt. When a device interrupts the processor, the level of the interrupt is used to mask lower priority interrupts.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `ROM/CNFGDATA/config.des` (**Figure 1-3**).

Available Values

`0` to `4294967295`. The number of supported interrupt levels is dependent on the processor being used (for example, 1-7 on 680x0 type CPUs).

See Also

The ***OS-9 Input/Output System*** section of the ***OS-9 Technical Manual***.

cons_timeout

CONS_TIMEOUT

EditMod Labels

```
2-configuration data
1-console port data structure
5-polling timeout
```

Description

Polling time-out value for the console device.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `ROM/CNFGDATA/config.des` (**Figure 1-3**).

Available Values

`0` to `4294967295`

EditMod Labels

```
2-configuration data
1-console port data structure
6-parity
```

Description

Parity mode to be used by the console device.

Port Generic Default Value

0 (zero)

Port Specific Override Value

Refer to ROM/CNFGDATA/config.des (**Figure 1-3**).

Available Values

The configuration modules parity values are located in the header file, rom.h, and are listed in **Table 1-7**.

Table 1-7 cons_parity Available Values

Description	Macro	EditMod
No parity	CONS_NOPARITY	0x00
Odd parity	CONS_ODDPARITY	0x01
Even parity	CONS_EVENPARITY	0x02
Mark parity	CONS_MARKPARITY	0x03
Space parity	CONS_SPACEPARITY	0x04

Table 1-7 `cons_parity` Available Values (continued)

Description	Macro	EditMod
Parity mask	`CONS_PARITY_MASK`	`0x0F`
Parity shift	`CONS_PARITY_SHIFT`	`0`

EditMod Labels

```
2-configuration data
1-console port data structure
7-baud rate
```

Description

Baud rate to be used by the console device.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `ROM/CNFGDATA/config.des` (**Figure 1-3**).

Available Values

The configuration modules baud rate values are located in the header file, `rom.h`, and are listed in **Table 1-8**.

Table 1-8 `cons_baudrate` **Available Values**

Description	Macro	EditMod
Hardwire baud rate	CONS_BAUDRATE_HARDWIRE	0x00
50 bits per second (bps)	CONS_BAUDRATE_50	0x01
75 bps	CONS_BAUDRATE_75	0x02
110 bps	CONS_BAUDRATE_110	0x03
134.5 bps	CONS_BAUDRATE_134P5	0x04

Table 1-8 cons_baudrate Available Values (continued)

Description	Macro	EditMod
150 bps	CONS_BAUDRATE_150	0x05
300 bps	CONS_BAUDRATE_300	0x06
600 pbs	CONS_BAUDRATE_600	0x07
1200 bps	CONS_BAUDRATE_1200	0x08
1800 bps	CONS_BAUDRATE_1800	0x09
2000 bps	CONS_BAUDRATE_2000	0x0A
2400 bps	CONS_BAUDRATE_2400	0x0B
3600 bps	CONS_BAUDRATE_3600	0x0C
4800 bps	CONS_BAUDRATE_4800	0x0D
7200 bps	CONS_BAUDRATE_7200	0x0E
9600 bps	CONS_BAUDRATE_9600	0x0F
19,200 bps	CONS_BAUDRATE_19200	0x10
31,250 bps	CONS_BAUDRATE_31250	0x11
38,400 bps	CONS_BAUDRATE_38400	0x12
56,000 bps	CONS_BAUDRATE_56000	0x13
57,600 bps	CONS_BAUDRATE_57600	0x14
64,000 bps	CONS_BAUDRATE_64000	0x15

Table 1-8 cons_baudrate Available Values (continued)

Description	Macro	EditMod
115,200 bps	CONS_BAUDRATE_115200	0x16
No echo	CONS_NOECHO	0x80
Baud rate mask	CONS_BAUDRATE_MASK	0x3F

cons_wordsize

CONS_WORDSIZE

EditMod Labels

```
2-configuration data
1-console port data structure
8-character size
```

Description

Bits-per-byte to be used by the console device.

Port Generic Default Value

0 (zero)

Port Specific Override Value

Refer to ROM/CNFGDATA/config.des (**Figure 1-3**).

Available Values

The configuration modules word size values are located in the header file, rom.h, and are listed in **Table 1-9**.

Table 1-9 cons_wordsize Available Values

Description	Macro	EditMod
8 bit word size	CONS_8BITS	0x00
7 bit word size	CONS_7BITS	0x40
6 bit word size	CONS_6BITS	0x80
5 bit word size	CONS_5BITS	0xC0

Table 1-9 cons_wordsize Available Values (continued)

Description	Macro	EditMod
Word size mask	CONS_DBITS_MASK	0xC0
Word size shift	CONS_DBITS_SHIFT	6

cons_stopbits

CONS_STOPBITS

EditMod Labels

```
2-configuration data
1-console port data structure
9-stop bit
```

Description

Number of stop bits to be used by the console device.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `ROM/CNFGDATA/config.des` (**Figure 1-3**).

Available Values

The configuration modules stop bit values are located in the header file `rom.h`, and are listed in **Table 1-10**.

Table 1-10 `cons_stopbits` **Available Values**

Description	Macro	EditMod
Stop bit length of 1	`CONS_1STOP`	`0x00`
Stop bit length of 1.5	`CONS_1P5STOP`	`0x10`
Stop bit length of 2	`CONS_2STOP`	`0x20`
Stop bit mask	`CONS_STOP_MASK`	`0x30`

Table 1-10 `cons_stopbits` Available Values (continued)

Description	Macro	EditMod
Stop bit shift	`CONS_STOP_SHIFT`	`0x40`
Stop data bit shift	`CONS_DBITS_SHIFT`	`0x60`

cons_flow

CONS_FLOW

EditMod Labels

```
2-configuration data
1-console port data structure
10-flow control
```

Description

Flow control mode of the console device.

Port Generic Default Value

0 (zero)

Port Specific Override Value

Refer to ROM/CNFGDATA/config.des (**Figure 1-3**).

Available Values

The configuration modules flow control values are located in the header file, rom.h, and are listed in **Table 1-11**.

Table 1-11 cons_flow **Available Values**

Description	Macro	EditMod
No handshaking	CONS_NOSHAKE	0x00
XOFF, any character on	CONS_SWSHAKE	0x01
Hardware handshaking	CONS_HWSHAKE	0x02
Strictly XON-XOFF	CONS_SWSTRICT	0x03

Communication Device Fields

The communication device fields are in the order they appear during an interactive EditMod session. Defined fields may appear in a different order in config.des. The fields can be changed using the EditMod utility or by modifying the description files. See **cnfgdata Module Field Configuration Options** for detailed instructions on changing these fields.

Table 1-12 Communication Device Fields

Field	Description File Macro
comm_name	COMM_NAME
cons_vector	COMM_VECTOR
cons_priority	COMM_PRIORITY
cons_level	COMM_LEVEL
cons_timeout	COMM_TIMEOUT
cons_parity	COMM_PARITY
cons_baudrate	COMM_BAUDRATE
cons_wordsize	COMM_WORDSIZE
cons_stopbits	COMM_STOPBITS
cons_flow	COMM_FLOW

comm_name

COMM_NAME

EditMod Labels

```
2-configuration data
2-communication port data structure
1-communication port name
```

Description

Contains the communication device name string.

Port Generic Default Value

```
NULL
```

Port Specific Override Value

Refer to `ROM/CNFGDATA/config.des` (**Figure 1-3**).

Available Values

Any ASCII character string. The string may contain C-style character escapes (such as `\n` and `\012`).

cons_vector

COMM_VECTOR

EditMod Labels

```
2-configuration data
1-console port data structure
2-interrupt vector number
```

Description

This is the vector number of the console device passed to the processor at interrupt time.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `ROM/CNFGDATA/config.des` (**Figure 1-3**).

Available Values

`0` to `4294967295`

cons_priority

COMM_PRIORITY

EditMod Labels

```
2-configuration data
1-console port data structure
3-interrupt priority
```

Description

This is the software (polling) priority for the console device on the IRQ polling table.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `ROM/CNFGDATA/config.des` (**Figure 1-3**).

Available Values

The interrupt priority value range is 0-65534 (65535 is reserved). A non-zero priority determines the position of the device within the vector. Lower values are polled first.

Some considerations to keep in mind when selecting an interrupt priority:

- A priority of 0 indicates the device desires exclusive use of the vector.
- If the priority is 1, it is polled first and no other device can have a priority of 1 on the vector. For all other priority values, more than one device may share the same priority on a vector. In this case, first-in, first-out (FIFO) scheduling determines the order of precedence in the polling table for the devices.
- OS-9 does not allow a device to claim exclusive use of a vector if another device has already been installed on the vector. Additionally, it does not allow another device to use the vector once the vector has been claimed for exclusive use.

- This value is software dependent.

See Also

F_IRQ system call entry in the ***OS-9 Technical Manual***.

cons_level

COMM_LEVEL

EditMod Labels

```
2-configuration data
1-console port data structure
4-interrupt level
```

Description

This is the hardware priority of the console device interrupt. When a device interrupts the processor, the level of the interrupt is used to mask lower priority interrupts.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `ROM/CNFGDATA/config.des` (**Figure 1-3**).

Available Values

`0` to `4294967295`. The number of supported interrupt levels is dependent on the processor being used (for example, 1-7 on 680x0 type CPUs).

See Also

The ***OS-9 Input/Output System*** section of the ***OS-9 Technical Manual***.

cons_timeout

COMM_TIMEOUT

EditMod Labels

```
2-configuration data
1-console port data structure
5-polling timeout
```

Description

Polling time-out value for the console device.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `ROM/CNFGDATA/config.des` (**Figure 1-3**).

Available Values

`0` to `4294967295`

cons_parity

COMM_PARITY

EditMod Labels

```
2-configuration data
1-console port data structure
6-parity
```

Description

Parity mode to be used by the console device.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `ROM/CNFGDATA/config.des` (**Figure 1-3**).

Available Values

The configuration modules parity values are located in the header file, `rom.h`, and are listed in **Table 1-13**.

Table 1-13 `cons_parity` **Available Values**

Description	Macro	EditMod
No parity	`CONS_NOPARITY`	`0x00`
Odd parity	`CONS_ODDPARITY`	`0x01`
Even parity	`CONS_EVENPARITY`	`0x02`
Mark parity	`CONS_MARKPARITY`	`0x03`
Space parity	`CONS_SPACEPARITY`	`0x04`

Table 1-13 cons_parity Available Values (continued)

Description	Macro	EditMod
Parity mask	CONS_PARITY_MASK	0x0F
Parity shift	CONS_PARITY_SHIFT	0

cons_baudrate

COMM_BAUDRATE

EditMod Labels

```
2-configuration data
1-console port data structure
7-baud rate
```

Description

Baud rate to be used by the console device.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `ROM/CNFGDATA/config.des` (**Figure 1-3**).

Available Values

The configuration modules baud rate values are located in the header file, `rom.h`, and are listed in **Table 1-14**.

Table 1-14 `cons_baudrate` **Available Values**

Description	Macro	EditMod
Hardwire baud rate	`CONS_BAUDRATE_HARDWIRE`	`0x00`
50 bits per second (bps)	`CONS_BAUDRATE_50`	`0x01`
75 bps	`CONS_BAUDRATE_75`	`0x02`
110 bps	`CONS_BAUDRATE_110`	`0x03`
134.5 bps	`CONS_BAUDRATE_134P5`	`0x04`

Table 1-14 `cons_baudrate` Available Values (continued)

Description	Macro	EditMod
150 bps	CONS_BAUDRATE_150	0x05
300 bps	CONS_BAUDRATE_300	0x06
600 pbs	CONS_BAUDRATE_600	0x07
1200 bps	CONS_BAUDRATE_1200	0x08
1800 bps	CONS_BAUDRATE_1800	0x09
2000 bps	CONS_BAUDRATE_2000	0x0A
2400 bps	CONS_BAUDRATE_2400	0x0B
3600 bps	CONS_BAUDRATE_3600	0x0C
4800 bps	CONS_BAUDRATE_4800	0x0D
7200 bps	CONS_BAUDRATE_7200	0x0E
9600 bps	CONS_BAUDRATE_9600	0x0F
19,200 bps	CONS_BAUDRATE_19200	0x10
31,250 bps	CONS_BAUDRATE_31250	0x11
38,400 bps	CONS_BAUDRATE_38400	0x12
56,000 bps	CONS_BAUDRATE_56000	0x13
57,600 bps	CONS_BAUDRATE_57600	0x14
64,000 bps	CONS_BAUDRATE_64000	0x15

Table 1-14 `cons_baudrate` Available Values (continued)

Description	Macro	EditMod
115,200 bps	`CONS_BAUDRATE_115200`	`0x16`
No echo	`CONS_NOECHO`	`0x80`
Baud rate mask	`CONS_BAUDRATE_MASK`	`0x3F`

EditMod Labels

```
2-configuration data
1-console port data structure
8-character size
```

Description

Bits-per-byte to be used by the console device.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `ROM/CNFGDATA/config.des` (**Figure 1-3**).

Available Values

The configuration module word size values are located in the header file, `rom.h`, and are listed in **Table 1-15**.

Table 1-15 `cons_wordsize` **Available Values**

Description	Macro	EditMod
8 bit word size	`CONS_8BITS`	`0x00`
7 bit word size	`CONS_7BITS`	`0x40`
6 bit word size	`CONS_6BITS`	`0x80`
5 bit word size	`CONS_5BITS`	`0xC0`

Table 1-15 cons_wordsize Available Values (continued)

Description	Macro	EditMod
Word size mask	CONS_DBITS_MASK	0xC0
Word size shift	CONS_DBITS_SHIFT	6

EditMod Labels

```
2-configuration data
1-console port data structure
9-stop bit
```

Description

Number of stop bits to be used by the console device.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `ROM/CNFGDATA/config.des` (**Figure 1-3**).

Available Values

The configuration modules stop bit values are located in the header file `rom.h`, and are listed in **Table 1-16**.

Table 1-16 `cons_stopbits` Available Values

Description	Macro	EditMod
Stop bit length of 1	`CONS_1STOP`	`0x00`
Stop bit length of 1.5	`CONS_1P5STOP`	`0x10`
Stop bit length of 2	`CONS_2STOP`	`0x20`
Stop bit mask	`CONS_STOP_MASK`	`0x30`

Table 1-16 `cons_stopbits` Available Values (continued)

Description	Macro	EditMod
Stop bit shift	`CONS_STOP_SHIFT`	`0x40`
Stop data bit shift	`CONS_DBITS_SHIFT`	`0x60`

EditMod Labels

```
2-configuration data
1-console port data structure
10-flow control
```

Description

Flow control mode of the console device.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `ROM/CNFGDATA/config.des` (**Figure 1-3**).

Available Values

The configuration module flow control values are located in the header file, `rom.h`, and are listed in **Table 1-17**.

Table 1-17 `cons_flow` **Available Values**

Description	Macro	EditMod
No handshaking	`CONS_NOSHAKE`	`0x00`
XOFF, any character on	`CONS_SWSHAKE`	`0x01`
Hardware handshaking	`CONS_HWSHAKE`	`0x02`
Strictly XON-XOFF	`CONS_SWSTRICT`	`0x03`

Debugger Fields

The debugger fields are in the order they appear during an interactive `EditMod` session. Defined fields may appear in a different order in `config.des`. The fields can be changed using the `EditMod` utility or by modifying the description files. See **cnfgdata Module Field Configuration Options** for detailed instructions on how to change these fields.

Table 1-18 Debugger Fields

Field	Description File Macro
`debug_name`	`DEBUGGER_NAME`
`debug_call_at_cold`	`DEBUGGER_COLD_FLAG`

debug_name

DEBUGGER_NAME

EditMod Labels

```
2-configuration data
3-debugger data structure
1-debugger name
```

Description

Contains the name string of the debugger module used as the low-level debugger.

Port Generic Default Value

`NULL`

Port Specific Override Value

Refer to `ROM/CNFGDATA/config.des` (**Figure 1-3**).

Available Values

Any ASCII character string. The string may contain C-style character escapes (such as `\n` and `\012`).

debug_call_at_cold

DEBUGGER_COLD_FLAG

EditMod Labels

```
2-configuration data
3-debugger data structure
2-cold start flag
```

Description

Cold start flag.

Port Generic Default Value

0 (zero)

Port Specific Override Value

Refer to ROM/CNFGDATA/config.des (Figure 1-3).

Available Values

The configuration modules debug_call_at_cold values are located in the header file, rom.h, and are listed in Table 1-19.

Table 1-19 debug_call_at_cold Available Values

Description	Macro	EditMod
Bypass calling debugger during boot sequence	DEBUG_BYPASS	0x0
Call debugger during boot sequence	DEBUG_CALL	0x1

Low-Level Protocol Manager Fields

The low-level protocol manager fields are in the order they appear during an interactive EditMod session. Defined fields may appear in a different order in config.des. The fields can be changed using the EditMod utility or by modifying the config.des description file. See **cnfgdata Module Field Configuration Options** for detailed instructions on changing these fields.

Table 1-20 Low-Level Protocol Manager Fields

Field	Description File Macro
maxllpmprotos	LLPM_MAXPROTOS
maxrcvmbufs	LLPM_MAXRCVMBUFS
maxllpmconns	LLPM_MAXCONNS
llpm_count	LLPM_COUNT

maxllpmprotos

LLPM_MAXPROTOS

EditMod Labels

```
2-configuration data
4-low level protocol manager data structure
1-maximum number of protocols
```

Description

Maximum number of protocol modules allowed on the protocol stack.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `ROM/CNFGDATA/config.des` (**Figure 1-3**).

Available Values

`0` to `65535`

maxrcvmbufs

LLPM_MAXRCVMBUFS

EditMod Labels

```
2-configuration data
4-low level protocol manager data structure
2-maximum number of receive mbufs
```

Description

Maximum number of memory buffers available for receiving packets. The size of each memory buffer varies depending on the driver used. (For example, llslip: 1024, ll21040: 1520).

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `ROM/CNFGDATA/config.des` (**Figure 1-3**).

Available Values

`0` to `65535`

maxllpmconns

LLPM_MAXCONNS

EditMod Labels

```
2-configuration data
4-low level protocol manager data structure
3-maximum number of connections
```

Description

Maximum number of low-level protoman connections allowed.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `ROM/CNFGDATA/config.des` (**Figure 1-3**).

Available Values

`0` to `65535`

llpm_count

LLPM_COUNT

EditMod Labels

```
2-configuration data
4-low level protocol manager data structure
4-number of data entries
```

Description

Number of low-level interface data entries.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `ROM/CNFGDATA/config.des` (**Figure 1-3**).

Available Values

`0` to `4294967295`

Interface Data Fields

The interface data fields are in the order they appear during an interactive `EditMod` session. Defined fields can appear in a different order in `config.des`. The fields can be changed using the `EditMod` utility or by modifying the description files. See **cnfgdata Module Field Configuration Options** for detailed instructions on changing these fields.

Table 1-21 Interface Data Fields

Field	Description
`ip_address`	Low-level IP address
`subnet_mask`	Low-level subnet mask
`brdcst_address`	Low-level broadcast address
`gw_address`	Low-level gateway address
`mac_address`	Low-level MAC address
`hwtype`	Low-level interface data driver type
`if_flags`	Interface flags
`if_name`	Low-level protocol manager name
`port_address`	Low-level protocol manager physical address
`if_vector`	Low-level protocol manager vector number

Table 1-21 Interface Data Fields (continued)

Field	Description
`if_priority`	Low-level protocol manager polling priority
`if_level`	Low-level protocol manager hardware priority

ip_address

Low-level IP Address

EditMod Labels

```
2-configuration data
4-low level protocol manager data structure
5-low level protocol interface data
<n>-low level protocol interface data[<n>]
1-internet address
```

Description

Low-level internet protocol (IP) address.

Port Generic Default Value

```
0.0.0.0
```

Port Specific Override Value

Refer to `ROM/CNFGDATA/config.des` (Figure 1-3).

Available Values

Any dot(.) separated four item sequence of decimal numbers in the range of zero to 255.

subnet_mask

Low-level Subnet Mask

EditMod Labels

```
2-configuration data
4-low level protocol manager data structure
5-low level protocol interface data
<n>-low level protocol interface data[<n>]
2-subnet mask
```

Description

Low-level interface data subnet mask.

Port Generic Default Value

`0.0.0.0`

Port Specific Override Value

Refer to `ROM/CNFGDATA/config.des` (**Figure 1-3**).

Available Values

Any dot(.) separated four item sequence of decimal numbers in the range of zero to 255.

brdcst_address

Low-level Broadcast Address

EditMod Labels

```
2-configuration data
4-low level protocol manager data structure
5-low level protocol interface data
<n>-low level protocol interface data[<n>]
3-broadcast address
```

Description

Low-level interface data broadcast address.

Port Generic Default Value

```
0.0.0.0
```

Port Specific Override Value

Refer to `ROM/CNFGDATA/config.des` (**Figure 1-3**).

Available Values

Any dot(.) separated four item sequence of decimal numbers in the range of zero to 255.

gw_address

Low-level Gateway Address

EditMod Labels

```
2-configuration data
4-low level protocol manager data structure
5-low level protocol interface data
<n>-low level protocol interface data[<n>]
4-gateway address
```

Description

Low-level interface data gateway address.

Port Generic Default Value

```
0.0.0.0
```

Port Specific Override Value

Refer to `ROM/CNFGDATA/config.des` (**Figure 1-3**).

Available Values

Any dot(.) separated four item sequence of decimal numbers in the range of zero to 255.

mac_address

Low-level MAC address

EditMod Labels

```
2-configuration data
4-low level protocol manager data structure
5-low level protocol interface data
<n>-low level protocol interface data[<n>]
5-MAC (ethernet) address
```

Description

Low-level MAC (Ethernet address), machine address or hardware address.

Port Generic Default Value

```
0:0:0:0:0:0
```

Port Specific Override Value

Refer to `ROM/CNFGDATA/config.des` (**Figure 1-3**).

Available Values

Any colon(:) separated six item sequence of hexadecimal numbers in the range of zero to 255 `(0xff)`. The `0x` or $ prefix is not valid.

hwtype

Low-level Interface Data Driver Type

EditMod Labels

```
2-configuration data
4-low level protocol manager data structure
5-low level protocol interface data
<n>-low level protocol interface data[<n>]
6-driver type
```

Description

Low-level interface data driver type.

Port Generic Default Value

0 (zero)

Port Specific Override Value

Refer to ROM/CNFGDATA/config.des (**Figure 1-3**).

Available Values

The configuration modules hwtype values are located in the header file, rom.h, and are listed in **Table 1-22**.

Table 1-22 hwtype Available Values

Description	Macro	EditMod
No driver type	LLPM_NOHW	0x0
SLIP driver type	LLPM_SLIP	0x1
Ethernet driver type	LLPM_ETHER	0x2

if_flags

Interface Flags

EditMod Labels

```
2-configuration data
4-low level protocol manager data structure
5-low level protocol interface data
<n>-low level protocol interface data[<n>]
12-interface-specific flag(s)
```

Description

Interface flags.

Port Generic Default Value

0 (zero)

Port Specific Override Value

Refer to ROM/CNFGDATA/config.des (**Figure 1-3**).

Available Values

The configuration modules if_flags values are located in the header file, rom.h, and are listed in **Table 1-23**.

Table 1-23 if_flags Available Values

Description	Macro	EditMod
Applies only to SLIP array entries.	LLIF_CSLIP_ON	0x8000
Applies only to SLIP array entries.	LLIF_CSLIP_OFF	0x0000

if_name

Low-level Protocol Manager Name

EditMod Labels

```
2-configuration data
4-low level protocol manager data structure
5-low level protocol interface data
<n>-low level protocol interface data[<n>]
13-interface name
```

Description

Contains the llpm interface device name string.

Port Generic Default Value

`NULL`

Port Specific Override Value

Refer to `ROM/CNFGDATA/config.des` (**Figure 1-3**).

Available Values

Any ASCII character string. The string may contain C-style character escapes (such as `\n` and `\012`).

port_address

Low-level Protocol Manager Physical Address

EditMod Labels

```
2-configuration data
4-low level protocol manager data structure
5-low level protocol interface data
<n>-low level protocol interface data[<n>]
14-interface port address
```

Description

This is the absolute physical address of the llpm interface device.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `ROM/CNFGDATA/config.des` (**Figure 1-3**).

Available Values

`0` to `4294967295`

if_vector

Low-level Protocol Manager Vector Number

EditMod Labels

```
2-configuration data
4-low level protocol manager data structure
5-low level protocol interface data
<n>-low level protocol interface data[<n>]
15-interrupt vector
```

Description

This is the vector number of the llpm interface device passed to the processor at interrupt time.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `ROM/CNFGDATA/config.des` (**Figure 1-3**).

Available Values

`0` to `4294967295`

Note

Value range is hardware/software dependent and determined at the OS level (OS-9 vs. OS-9 for 68K).

if_priority

Low-level Protocol Manager Polling Priority

EditMod Labels

```
2-configuration data
4-low level protocol manager data structure
5-low level protocol interface data
<n>-low level protocol interface data[<n>]
16-interrupt priority
```

Description

This is the software (polling) priority for the llpm interface device on the IRQ polling table.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `ROM/CNFGDATA/config.des` (**Figure 1-3**).

Available Values

The interrupt priority value range is 0-65534 (65535 is reserved). A non-zero priority determines the position of the device within the vector. Lower values are polled first. Some considerations to keep in mind when selecting an interrupt priority:

- A priority of 0 indicates the device desires exclusive use of the vector.
- If the priority is 1, it is polled first and no other device can have a priority of 1 on the vector. For all other priority values, more than one device may share the same priority on a vector. In this case, first-in, first-out (FIFO) scheduling determines the order of precedence in the polling table for the devices.

- OS-9 does not allow a device to claim exclusive use of a vector if another device has already been installed on the vector. Additionally, it does not allow another device to use the vector once the vector has been claimed for exclusive use.
- This value is software dependent.

See Also

`F_IRQ` system call entry in the ***OS-9 Technical Manual***.

if_level

Low-level Protocol Manager Hardware Priority

EditMod Labels

```
2-configuration data
4-low level protocol manager data structure
5-low level protocol interface data
<n>-low level protocol interface data[<n>]
17-interrupt level
```

Description

This is the hardware priority of the llpm interface device interrupt. When a device interrupts the processor, the level of the interrupt is used to mask out lower priority devices.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `ROM/CNFGDATA/config.des` (**Figure 1-3**).

Available Values

`0` to `65535`. The number of supported interrupt levels is dependent on the processor being used (for example, 1-7 on 680x0 type CPUs).

See Also

The ***OS-9 Input/Output System*** section of the ***OS-9 Technical Manual***.

Configuration Boot Data Fields

The configuration boot data fields are in the order they appear during an interactive EditMod session. Defined fields can appear in a different order in config.des. The fields can be changed using the EditMod utility or by modifying the description files. See **cnfgdata Module Field Configuration Options** for detailed instructions on how to change these fields.

Table 1-24 Configuration Boot Data Fields

Field	Description File Macro
boot_count	BOOT_COUNT
boot_cmdsize	BOOT_CMDSIZE

boot_count

BOOT_COUNT

EditMod Labels

```
2-configuration data
5-boot services data structure
1-number of boot system entries
```

Description

Number of boot system configuration entries.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `ROM/CNFGDATA/config.des` (**Figure 1-3**).

Available Values

`0` to `4294967295`

EditMod Labels

```
2-configuration data
5-boot services data structure
3-maximum size of user input string
```

Description

This field defines the maximum size of user input string during boot menu selection.

Port Generic Default Value

32 characters

Port Specific Override Value

Refer to `ROM/CNFGDATA/config.des` (**Figure 1-3**).

Available Values

`0` to `4294967295`

Boot Data Fields

The boot data fields are in the order they appear during an interactive `EditMod` session. Defined fields can appear in a different order in `config.des`. The fields can be changed using the `EditMod` utility or by modifying the description files. See **cnfgdata Module Field Configuration Options** for detailed instructions on how to change these fields.

Table 1-25 Boot Data Fields

Field	Description
`boot_abname`	Abbreviated booter name
`boot_newab`	New abbreviated booter name
`boot_newname`	Optional replacement full name
`boot_automenu`	Booter types for registration
`boot_params`	Optional parameter string
`autoboot_delay`	Autoboot delay value

boot_abname

Abbreviated Booter Name

EditMod Labels

```
2-configuration data
5-boot services data structure
2-boot data
<n>-boot data[<n>]
1-abbreviated booter name
```

Description

Abbreviated booter name.

Port Generic Default Value

`NULL`

Port Specific Override Value

Refer to `ROM/CNFGDATA/config.des` (**Figure 1-3**).

Available Values

Any ASCII character string. The string may contain C-style character escapes (such as `\n` and `\012`).

boot_newab

New Abbreviated Booter Name

EditMod Labels

```
2-configuration data
5-boot services data structure
2-boot data
<n>-boot data[<n>]
2-optional replacement abname
```

Description

New abbreviated booter name.

Port Generic Default Value

```
NULL
```

Port Specific Override Value

Refer to `ROM/CNFGDATA/config.des` (**Figure 1-3**).

Available Values

Any ASCII character string. The string may contain C-style character escapes (such as `\n` and `\012`).

boot_newname

Optional Replacement Full Name

EditMod Labels

```
2-configuration data
5-boot services data structure
2-boot data
<n>-boot data[<n>]
3-optional replacement full name
```

Description

Optional replacement full name.

Port Generic Default Value

`NULL`

Port Specific Override Value

Refer to `ROM/CNFGDATA/config.des` (**Figure 1-3**).

Available Values

Any ASCII character string. The string may contain C-style character escapes (such as `\n` and `\012`).

boot_automenu

Booter Types For Registration

EditMod Labels

```
2-configuration data
5-boot services data structure
2-boot data
<n>-boot data[<n>]
4-auto/menu flag
```

Description

Booter types for registration.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `ROM/CNFGDATA/config.des` (**Figure 1-3**).

Available Values

The configuration modules `boot_automenu` values are located in the header file, `rom.h`, and are listed in **Table 1-26**.

Table 1-26 `boot_automenu` **Available Values**

Description	config.des Macro	EditMod Hex
Auto booter	`BT_AUTO`	`0x1`
Menu booter	`BT_MENU`	`0x2`

EditMod Labels

```
2-configuration data
5-boot services data structure
2-boot data
<n>-boot data[<n>]
5-optional parameter string
```

Description

Optional parameter string.

Port Generic Default Value

`NULL`

Port Specific Override Value

Refer to `ROM/CNFGDATA/config.des` (**Figure 1-3**).

Available Values

Any ASCII character string. The string may contain C-style character escapes (such as `\n` and `\012`).

autoboot_delay

Autoboot Delay Value

EditMod Labels

```
2-configuration data
5-boot services data structure
2-boot data
<n>-boot data[<n>]
6-autoboot delay in microseconds
```

Description

Handled in the `bootsys` module, this is the delay value in microseconds prior to proceeding with an autoboot entry.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `ROM/CNFGDATA/config.des` (**Figure 1-3**).

Available Values

`0` to `4294967295`

Notification Services Field

The notification field can be changed using the EditMod utility or by modifying the description files. See **cnfgdata Module Field Configuration Options** for detailed instructions on changing this field.

Table 1-27 Notification Services Fields

Field	Description File Macro
max_notifiers	MAX_NOTIFIERS

max_notifiers

MAX_NOTIFIERS

EditMod Labels

```
2-configuration data
6-notification services data structure
1-maximum number of registered notifiers
```

Description

Used by the notification services module to indicate the maximum number of notification routines that can be registered.

Port Generic Default Value

0 (zero)

Port Specific Override Value

Refer to ROM/CNFGDATA/config.des (**Figure 1-3**).

Available Values

0 to 4294967295. While the only adverse effect of defining a larger max_notifiers value than necessary is the extra memory used for the unused records, here are some considerations to help determine an acceptable value:

- Notification services are required by any module that needs to know when the systems are in transition from polled mode to interrupt mode. Essentially this means the low-level serial and ethernet drivers (including iovcons).
- A module generally only installs one notification routine, but if a single module is used for two ports (like io16550 on Powerstacks and PCs), it installs two.

See Also

The ***Low-Level System Configuration*** section and the ***Porting OS-9*** section of ***OS-9 Porting Guide***.

Chapter 2: OS-9 Configuration Module (init)

The `init` (initialization) module contains configuration data used by the kernel and other OS-9 system modules to control system bootup and execution. Values that can be configured in the `init` module include:

- Initial system data table sizes
- Memory layout and characteristics
- Names of the system ticker and other OS extensions
- Flag fields specifying various operational modes
- Process scheduling control, including first process to execute

The next section in this chapter provides a detailed example of the two reconfiguration options you can use to change configuration values for this module.

The rest of this chapter provides a detailed list of all of the `init` module fields, including field descriptions and available values.

This chapter includes the following topics:

RadiSys.
MICROWARE SOFTWARE

Init Module Field Configuration Options

To change an init module configuration field, you can use either of the following methods:

1. Use the EditMod utility to directly modify existing init modules either as a stand-alone module or as part of a merged module group (such as a boot image).
2. Modify the description file for the init module and rebuild it using the makefile provided.

Direct Modification Advantages

The direct modification method has the following advantages:

Fast	No source configuration file rebuilds are necessary.
Temporary	The original module or merged-module group configuration can be easily restored through the appropriate rebuild.
Contained	Changes are limited to the individual boot image modified (merged-module option).

Description File/Rebuild Advantages

The advantage of the description file/rebuild method is that the changes are permanent and reproducible. Modifications apply to all subsequent module rebuilds and to all merged-module groups built containing the updated module.

Both methods are documented in this section. These procedures are used with the field descriptions starting with the **Module Header Fields**. For direct modification, use the EditMod LABELS data to navigate the

EditMod menus. The DESCRIPTION FILE MACRO data identifies the macro you need to define/modify in the configuration sources to rebuild the init module.

Direct Modification

Use the Editmod utility and the following procedures to directly modify fields in the existing init module. The module can stand-alone or it can be part of a merged-module group. A boot image, for example, contains multiple modules. Both situations are covered in this section. The field references later in this chapter contain a description of each configurable field, its supported values, and the sequence of menu options required by EditMod to modify that field.

For More Information

Refer to the ***Utilities Reference*** for a full description of EditMod's capabilities.

Figure 2-1 Directory Location for Modifying the init Module as a Stand-alone Module

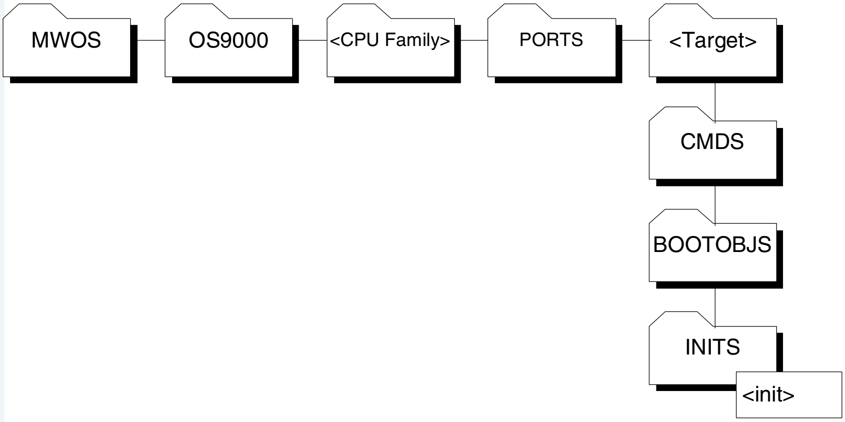
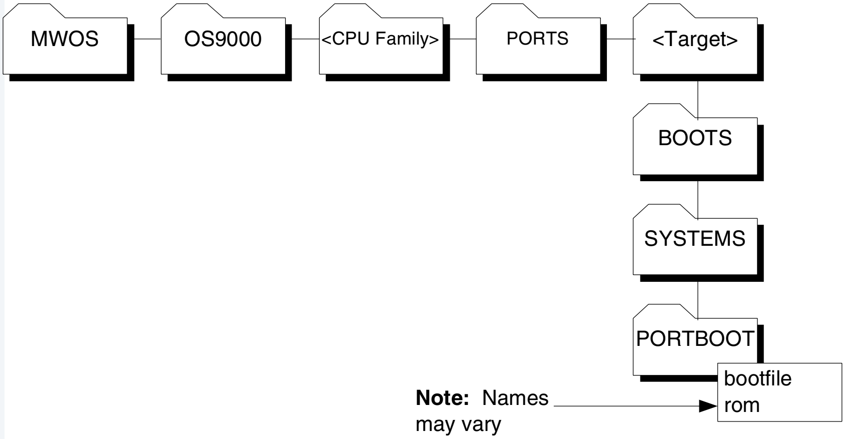


Figure 2-2 Directory Location for Modifying the init Module as Part of a Boot Image



For More Information

See your board guide for information about how to modify the module lists and remake the boot images, and for specified boot image names.

Direct Modification Procedures

To modify the stand-alone module, complete the following steps:

Step 1. Change to the `CMDS/BOOTOBJS/INITS` directory (see **Figure 2-1**).

Step 2. Use EditMod to edit the module:

```
$EditMod -e init
```

To modify the module as part of a merged module group, complete the following steps:

Step 1. Change to the `BOOTS/SYSTEMS/PORTBOOT` directory (see **Figure 2-2**).

Step 2. Use EditMod to edit the module:

```
$EditMod -e init -f=<boot image name>
```

Step 3. Use the menu selections provided in the `EditMod` LABELS section of the field reference later in this chapter to locate the fields you want to edit.

Step 4. Select a new value for the field from the AVAILABLE VALUES section of the field reference. Enter that value at the EditMod prompt to modify the field.

Step 5. If you want to make additional modifications, use the `p` command (previous) to step backward through the EditMod menus. Repeat Steps 3 and 4 until you have made all desired modifications to the `init` module.

Step 6. Select the `w` command (write) to save the changes.

Step 7. Select the `q` command (quit) to exit `EditMod`.

Note

Unless you modified the `init` module in your boot image, you should rebuild your boot image to include the new `init` module.

Example EditMod Session

This example modifies `init` as part of the boot image `rom`.

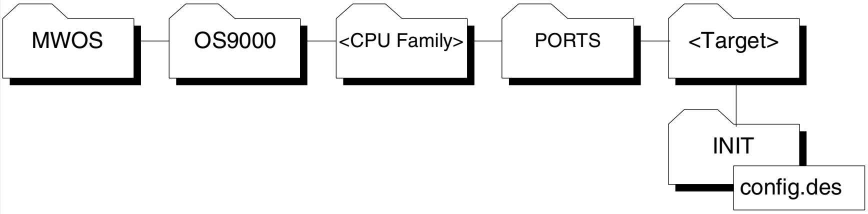
```
$ EditMod -e init -f-rom

 1. module header
 2. init module contents

Which? [?/1-2/p/t/a/w/q] 2

 1. installation site code                      : 0x0
 2. cpu class                                   : 0x1bc7
 3. installation string                         : “PS7111”
 4. OS-9000 level/revision string               : “OS-9000 for the ARM”
 5. initial module name                         : “shell”
 6. parameter list                              : “”
 7. system RBF device                           : “”
 8. system SCF device                           : “/term”
 9. customization module list                   : “OS9P2 fpu ssm”
10. ticker module name                          : “tkarm”
11. real-time clock module name                 : “rtc7110”
12. IO manager module name                      : “Ioman”
13. user accounting module name                 : “”
14. memory list
15. number of process table entries             : 0x40
16. number of path table entries                : 0x40
17. number of system event table entries        : 0x20
18. number of ticks per second                  : 0x64
19. number of clock ticks per time slice        : 0x2
20. initial system priority                     : 0x80
21. initial minimum executable priority         : 0x0
22. initial maximum natural process age         : 0x0
23. system call dispatch table entries          : 0x100
24. reserved for system specific flags          : 0x0
25. system time zone                            : 0
26. OS-9000 level                               : 1
27. OS-9000 major release number                : 2
```

```
28. OS-9000 minor release number           : 0
29. OS-9000 edition number                 : 0
30. compatibility flags                    : 0x2
31. process signal queue size              : 0x20
32. pre-IO customization module list       : “irq7110 irq7111”
33. cache list

$Which? [?/1-6/p/t/a/w/q]
.
.
.
.
Which? [?/1-19/p/t/a/w/q] w

Which? [?/1-19/p/t/a/w/q] q
```

Description File Modification

You can use these procedures to modify the `init` description file sources and rebuild the `init` module for your port directory. The DESCRIPTION FILE MACROS section of the field reference specifies the name of the macro you modify/define in the description files to configure the field. The value used in the define is chosen from the AVAILABLE VALUES specified for the field.

Figure 2-3 Directory Location for Modifying the init Description File



Description File Modification Procedures

Step 1. Change to the INIT directory. (see **Figure 2-3**).

Step 2. Edit the file config.des and read the included comments for more information on using the specific description file provided in your software distribution. The config.des file contains a list of macro names defined to override the global default values for the configuration fields.

Step 3. Refer to the DESCRIPTION FILE MACRO section in the field reference later in this chapter to determine the macro name you define to configure the target field.

Step 4. Read the comments in config.des to determine where to place the define for this macro.

Step 5. Select the value you want to use to configure the field. See the AVAILABLE VALUES section of the field reference data for values or macros that can be used for the definition. Define the macro by entering a definition in the appropriate description files as follows:

```
#define <macro> <value>
```

Step 6. Save the changes and rebuild the module, entering the following command in the INIT directory:

```
os9make
```

Step 7. Rebuild your boot image to include the new init module.

Init Module Field Reference

This section contains a list of the most commonly configured fields in the `init` module. Each field entry contains the following information:

- <Field name> - The call name for each field that can be reconfigured in the module.
- EditMod LABELS - `EditMod` menu selections for navigating to the proper field in an `EditMod` session.
- DESCRIPTION FILE MACRO - The macro name you modify/define in the description file.
- DESCRIPTION - A brief description of the field's purpose and use.
- EXAMPLE - An optional example of the description file entry showing how to change the value of this field.
- PORT GENERIC DEFAULT VALUE - The value set in the port generic description file for this field. This is the value the field is assigned when the module is built, unless the appropriate macro has been defined in the port specific description file to override this default value.
- PORT SPECIFIC OVERRIDE VALUE - The value set in the port specific description file for this field. If defined, this is the value the field is assigned when the module is built, overriding the port generic default value.
- AVAILABLE VALUES - Values to which the field can be set through `EditMod` or the description files. In many cases, this data is presented in a table that maps a description of the value to a numeric value appropriate for entry in EditMod, and to a pre-defined macro available for use in the description file.

The `init` module is sometimes referred to as the configuration module. It is located in memory in the `sysboot` file or in ROM. The `init` module is a non-executable module of type `MT_SYSTEM`. The init module contains system parameters used to configure OS-9 during start-up. The parameters set up the initial table sizes and system device names and the `init` module must always be available to determine system limits. For example, the amount of memory to allocate for internal tables, the name of the first program to run (usually either sysgo or shell), an initial directory, and other initialization settings are specified. You can examine the system limits defined in the init module at any time.

The init module begins with a standard module header. The module header's m_exec offset is a pointer to the system's constant table. The init fields are defined in the init.h header file.

Note

The init module *must* be present in the system in order for OS-9 to work.

For more information on the init module, see the ***OS-9 Technical Manual***.

Module Header Fields

The following section contains the module header fields in the order they appear during an interactive EditMod session. Defined fields can appear in a different order in config.des.

Table 2-1 Module Header Fields

Field	Description File Macro
_m_group	MH_GROUP
_m_user	MH_USER
mod_name	MH_NAME
m_access	MH_ACCESS
m_tylan	MH_TYLAN
m_attrev	MH_ATTREV
m_edit	MH_EDITION

EditMod Labels

```
1-module header
1-module owner's group number
```

Description

Group ID of the module's owner. The group number allows people working in the same department or on the same project to share a common identification number.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `INIT/config.des` (**Figure 2-3**).

Available Values

`0` to `65535`

_m_user

MH_USER

EditMod Labels

```
1-module header
2-module owner’s user number
```

Description

User ID of the module’s owner. The user number identifies a specific user.

Port Generic Default Value

0 (zero)

Port Specific Override Value

Refer to `INIT/config.des` (**Figure 2-3**).

Available Values

`0` to `65535`

mod_name

MH_NAME

EditMod Labels

```
1-module header
3-module name
```

Description

Contains the module name string.

Port Generic Default Value

```
NULL
```

Port Specific Override Value

Refer to `INIT/config.des` (**Figure 2-3**).

Available Values

Any ASCII character string. The string may contain C-style character escapes (such as `\n` and `\012`).

m_access

MH_ACCESS

EditMod Labels

```
1-module header
4-access permissions
```

Description

Defines the permissible module access by its owner or by other users.

Port Generic Default Value

Macro

```
MP_OWNER_READ | MP_OWNER_EXEC | MP_GROUP_READ |
MP_GROUP_EXEC | MP_WORLD_READ | MP_WORLD_EXEC
```

EditMod

```
0x555
```

Port Specific Override Value

Refer to `INIT/config.des` (**Figure 2-3**).

Available Values

Module access permission values are located in the header file, `module.h`, and are listed in **Table 2-2**.

Table 2-2 `m_access` Available Values

Description	Macro	EditMod
Read permission by owner	`MP_OWNER_READ`	`0x0001`
Write permission by owner	`MP_OWNER_WRITE`	`0x0002`

Table 2-2 m_access Available Values (continued)

Description	Macro	EditMod
Execute permission by owner	MP_OWNER_EXEC	0x0004
Owner permission mask	MP_OWNER_MASK	0x000f
Read permission by group	MP_GROUP_READ	0x0010
Write permission by group	MP_GROUP_WRITE	0x0020
Execute permission by group	MP_GROUP_EXEC	0x0040
Group permission mask	MP_GROUP_MASK	0x00f0
Read permission by world	MP_WORLD_READ	0x0100
Write permission by world	MP_WORLD_WRITE	0x0200
Execute permission by world	MP_WORLD_EXEC	0x0400
World permission mask	MP_WORLD_MASK	0x0f00
All permissions for owner, group, and world	MP_WORLD_ACCESS	0x0777
System permission mask	MP_SYSTM_MASK	0xf000

m_tylan

MH_TYLAN

EditMod Labels

```
1-module header
5-type/language
```

Description

Contains the module's type (first byte) and language (second byte). The language codes indicate if the module is executable and which language the run-time system requires for execution, if any.

Port Generic Default Value

Macro

```
(MT_DATA<<8) + ML_OBJECT
```

EditMod

```
0x401
```

Port Specific Override Value

Refer to `INIT/config.des` (**Figure 2-3**).

Available Values

Module type values and language codes are located in the header file, `module.h`, and are listed in **Table 2-3** and **Table 2-4**.

Table 2-3 `m_tylan` Available Module Type Values

Description	Macro	EditMod
Not used (wildcard value in system calls)	`MT_ANY`	`0x0000`
Program module	`MT_PROGRAM`	`0x0001`
Subroutine module	`MT_SUBROUT`	`0x0002`
Multi-module (reserved for future use)	`MT_MULTI`	`0x0003`
Data module	`MT_DATA`	`0x0004`
Configuration data block data module	`MT_CDBDATA`	`0x0005`
Reserved for future use	`0xb-0xa`	`0xb-0xa`
User trap library	`MT_TRAPLIB`	`0x000b`
System module	`MT_SYSTEM`	`0x000c`
File manager module	`MT_FILEMAN`	`0x000d`
Physical device driver	`MT_DEVDRVR`	`0x000e`
Device descriptor module	`MT_DEVDESC`	`0x000f`

Table 2-3 `m_tylan` Available Module Type Values (continued)

Description	Macro	EditMod
User definable	`0x10-0xfe`	`0x10-0xfe`
Module type mask	`MT_MASK`	`0xff00`

Table 2-4 `m_tylan` Available Language Code Values

Description	Macro	EditMod
Unspecified language (wildcard in system calls)	`ML_ANY`	`0x0`
Machine language	`ML_OBJECT`	`0x1`
Basic I-code (reserved for future use)	`ML_ICODE`	`0x2`
Pascal P-code (reserved for future use)	`ML_PCODE`	`0x3`
C I-code (reserved for future use)	`ML_CCODE`	`0x4`
Cobol I-code (reserved for future use)	`ML_CBLCODE`	`0x5`
Fortran	`ML_FRTNCODE`	`0x6`
Reserved for future use	`0x7-0xf`	`0x7-0xf`
User-definable	`0x10-0xfe`	`0x10-0xfe`
Module language mask	`ML_MASK`	`0x00ff`

EditMod Labels

```
1-module header
6-revision/attributes
```

Description

Contains the module's attributes (first byte) and revision (second byte).

Port Generic Default Value

Macro

```
MA_REENT<<8
```

EditMod

```
0x8000
```

Port Specific Override Value

Refer to `INIT/config.des` (**Figure 2-3**).

Available Values

Module attribute and revision codes are located in the header file `module.h`, and are listed in **Table 2-5**.

Note

If two modules with the same name are found in the memory search or are loaded into the current module directory, only the module with the highest revision level is kept. This enables easy substitution of modules for update or correction.

Table 2-5 `m_attrev` Available Attribute and Revision Values

Description	Macro	EditMod
The module is re-entrant (sharable by multiple tasks).	`MA_REENT` (shifted left to first byte: `MA_REENT<<8)`	`0x80` (shifted left to first byte: `0x8000`)
The module is sticky. A sticky module is not removed from memory until its link count becomes -1 or memory is required for another use.	`MA_GHOST` (shifted left to first byte: `MA_GHOST<<8)`	`0x40` (shifted left to first byte: `0x4000`)
The module is a system-state module.	`MA_SUPER` (shifted left to first byte: `MA_SUPER<<8)`	`0x20` (shifted left to first byte: `0x2000`)
User-definable revision number	`0x0-0xfe`	`0x0-0xfe`
Module attribute mask	`MA_MASK`	`0xff00`
Module revision mask	`MR_MASK`	`0x00ff`

EditMod Labels

```
1-module header
7-edition
```

Description

Indicates the software release level for maintenance. OS-9 does not use this field. Whenever a program is revised (even for a small change), increase this number. It is recommended that internal documentation within the source program be keyed to this system.

Port Generic Default Value

`1`

Port Specific Override Value

Refer to `INIT/config.des` (**Figure 2-3**).

Available Values

`0` to `65535`

Module Body Fields

The following section contains the module body fields in the order they appear during an interactive EditMod session. Defined fields can appear in a different order in config.des.

Table 2-6 Module Header Fields

Field	Description File Macro
m_site	SITE
m_cputyp	MPUCHIP
install_name	INSTALNAME
os9rev_name	OS9K_REVSTR
sysgo_name	SYS_START
sparam_string	SYS_PARAMS
drive_name	SYS_DEVICE
console_name	CONS_NAME
extens_list	EXTENSIONS
ticker_name	TICK_NAME
rtc_name	RTC_NAME
ioman_name	IOMAN_NAME
acct_name	USRACCT_NAME

Table 2-6 Module Header Fields (continued)

Field	Description File Macro
m_procs	PROCS
m_paths	PATHS
m_events	EVENTS
m_ticksec	TICK_SEC
m_slice	SLICE
m_syspri	SYS_PRIOR
m_minpty	MINPTY
m_maxage	MAXPTY
m_dsptbl	DSPTBLSZ
m_cpucompat	CPUCOMPAT
m_tmzone	SYS_TMZONE
m_level	OS_LEVEL
m_major	OS_VERSION
m_minor	OS_REVISION
m_edition	OS_EDITION
m_compat	COMPAT

Table 2-6 Module Header Fields (continued)

Field	Description File Macro
m_maxsigs	MAXSIGS
preio_name	PREIOS_NAME

EditMod Labels

```
2-init module contents
1-installation site code
```

Description

This field contains the installation site code. This user-definable field can be used to identify the site of the system.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `INIT/config.des` (**Figure 2-3**).

Available Values

`0` to `4294967295`

m_cputyp

MPUCHIP

EditMod Labels

```
2-init module contents
2-cpu class
```

Description

This field contains the CPU family type. For example 403, 603, 80386, etc.

Port Generic Default Value

`80386`

Port Specific Override Value

Refer to `INIT/config.des` (**Figure 2-3**).

Available Values

`0` to `4294967295`

EditMod Labels

```
2-init module contents
3-installation string
```

Description

Installation name string.

Port Generic Default Value

`NULL`

Port Specific Override Value

Refer to `INIT/config.des` (**Figure 2-3**).

Available Values

Any ASCII character string. The string may contain C-style character escapes (such as `\n` and `\012`).

os9rev_name

OS9K_REVSTR

EditMod Labels

```
2-init module contents
4-OS-9000 level/revision string
```

Description

Contains the OS-9 level revision string.

Port Generic Default Value

```
“OS-9000”
```

Port Specific Override Value

Refer to `INIT/config.des` (**Figure 2-3**).

Available Values

Any ASCII character string. The string may contain C-style character escapes (such as `\n` and `\012`).

EditMod Labels

```
2-init module contents
5-initial module name
```

Description

Contains the name string of the first executable module.

Port Generic Default Value

`NULL`

Port Specific Override Value

Refer to `INIT/config.des` (**Figure 2-3**).

Available Values

Any ASCII character string. The string may contain C-style character escapes (such as `\n` and `\012`).

sparam_string

SYS_PARAMS

EditMod Labels

```
2-init module contents
6-parameter list
```

Description

Contains the parameter string (if any) to be passed to the first executable module.

Port Generic Default Value

`NULL`

Port Specific Override Value

Refer to `INIT/config.des` (**Figure 2-3**).

Available Values

Any ASCII character string. The string may contain C-style character escapes (such as `\n` and `\012`).

EditMod Labels

```
2-init module contents
7-system RBF device
```

Description

Contains the initial default directory name string, usually `/d0` or `/h0`. The system initially does a `chd` and `chx` to this device prior to forking the initial device. If the system does not use disk, this offset must be zero.

Port Generic Default Value

```
NULL
```

Port Specific Override Value

Refer to `INIT/config.des` (**Figure 2-3**).

Available Values

Any ASCII character string. The string may contain C-style character escapes (such as `\n` and `\012`).

console_name

CONS_NAME

EditMod Labels

```
2-init module contents
8-system SCF device
```

Description

Contains the initial I/O pathlist string, usually `/term`. This pathlist is opened as the standard I/O path for the initial process. It is generally used to set up the initial I/O paths to and from a terminal. The value should be set to `NULL` if no console device is in use.

Port Generic Default Value

```
NULL
```

Port Specific Override Value

Refer to `INIT/config.des` (**Figure 2-3**).

Available Values

Any ASCII character string. The string may contain C-style character escapes (such as `\n` and `\012`).

EditMod Labels

```
2-init module contents
9-customization module list
```

Description

Contains the name string of a list of customization modules, if any. A customization module complements or changes existing standard system calls used by OS-9. These modules are searched for at start-up and are usually found in the bootfile. If found, they are executed in system state.

Module names in the name string are separated by spaces. The default name string to be searched for is `OS9P2`. If there are no customization modules, this value should be set to `NULL`.

Port Generic Default Value

```
“OS9P2“
```

Port Specific Override Value

Refer to `INIT/config.des` (**Figure 2-3**).

Available Values

Any ASCII character string. The string may contain C-style character escapes (such as `\n` and `\012`).

ticker_name

TICK_NAME

EditMod Labels

```
2-init module contents
10-ticker module name
```

Description

Contains the name string of the module used to generate the system clock tick. The kernel attempts to call this module when the first `_os_setime` system call is made.

Port Generic Default Value

`NULL`

Port Specific Override Value

Refer to `INIT/config.des` (**Figure 2-3**).

Available Values

Any ASCII character string. The string may contain C-style character escapes (such as `\n` and `\012`).

rtc_name

RTC_NAME

EditMod Labels

```
2-init module contents
11-real-time clock module name
```

Description

Contains the real-time clock module name string. The kernel attempts to call this module when the time is set, in other words when `_os_setime` is called.

Port Generic Default Value

`NULL`

Port Specific Override Value

Refer to `INIT/config.des` (**Figure 2-3**).

Available Values

Any ASCII character string. The string may contain C-style character escapes (such as `\n` and `\012`).

ioman_name

IOMAN_NAME

EditMod Labels

```
2-init module contents
12-IO manager module name
```

Description

Contains the name string of the module handling I/O system calls. This string is normally set to `ioman`.

Port Generic Default Value

```
NULL
```

Port Specific Override Value

Refer to `INIT/config.des` (**Figure 2-3**).

Available Values

Any ASCII character string. The string may contain C-style character escapes (such as `\n` and `\012`).

acct_name

USRACCT_NAME

EditMod Labels

```
2-init module contents
13-user accounting module name
```

Description

Contains the name string of the user accounting module.

Port Generic Default Value

`NULL`

Port Specific Override Value

Refer to `INIT/config.des` (**Figure 2-3**).

Available Values

Any ASCII character string. The string may contain C-style character escapes (such as `\n` and `\012`).

m_procs

PROCS

EditMod Labels

```
2-init module contents
15-number of process table entries
```

Description

This is the number of entries in the process descriptor table. If this table becomes full, it is expanded automatically.

Port Generic Default Value

```
64 (0x40)
```

Port Specific Override Value

Refer to `INIT/config.des` (**Figure 2-3**).

Available Values

`0` to `65535`

EditMod Labels

```
2-init module contents
16-number of path table entries
```

Description

This is the initial number of open paths in the system. If this table becomes full, it is expanded automatically.

Port Generic Default Value

`64 (0x40)`

Port Specific Override Value

Refer to `INIT/config.des` (**Figure 2-3**).

Available Values

`0` to `65535`

m_events

EVENTS

EditMod Labels

```
2-init module contents
17-number of system event table entries
```

Description

This is the initial number of entries allowed in the events table. If this table becomes full, it is expanded automatically.

For More Information

Refer to the ***OS-9 Technical Manual*** for specific information on events.

Port Generic Default Value

`32 (0x20)`

Port Specific Override Value

Refer to `INIT/config.des` (**Figure 2-3**).

Available Values

`0` to `65535`

EditMod Labels

```
2-init module contents
18-number of ticks per second
```

Description

This is the number of ticks into which a second of time is divided.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `INIT/config.des` (**Figure 2-3**).

Available Values

`0` to `65535`

m_slice

SLICE

EditMod Labels

```
2-init module contents
19-number of clock ticks per time slice
```

Description

This is the number of clock ticks per time-slice.

Port Generic Default Value

`2`

Port Specific Override Value

Refer to `INIT/config.des` (**Figure 2-3**).

Available Values

`0` to `65535`

EditMod Labels

```
2-init module contents
20-initial system priority
```

Description

This is the system priority at which the first module (usually `sysgo` or `shell`) is executed. This is generally the base priority at which all processes start.

Port Generic Default Value

```
128 (0x80)
```

Port Specific Override Value

Refer to `INIT/config.des` (**Figure 2-3**).

Available Values

`0` to `65535`

m_minpty

MINPTY

EditMod Labels

```
2-init module contents
21-initial minimum executable priority
```

Description

This is the initial system minimum executable priority. `m_minpty` is discussed later in this chapter and in the ***OS-9 Technical Manual***.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `INIT/config.des` (**Figure 2-3**).

Available Values

`0` to `65535`

EditMod Labels

```
2-init module contents
22-initial maximum natural process age
```

Description

This is the initial system maximum natural age. `m_maxage` is discussed later in this chapter and in the ***OS-9 Technical Manual***.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `INIT/config.des` (**Figure 2-3**).

Available Values

`0` to `65535`

m_dsptbl

DSPTBLSZ

EditMod Labels

```
2-init module contents
23-system call dispatch table entries
```

Description

This field contains the number of entries in the system call dispatch table. There must be at least 256 entries in this table, and each entry requires eight bytes.

Port Generic Default Value

`256 (0x100)`

Port Specific Override Value

Refer to `INIT/config.des` (**Figure 2-3**).

Available Values

`0` to `65535`

EditMod Labels

```
2-init module contents
24 - reserved for system specific flags
```

Description

This field is reserved for system-specific flags.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `INIT/config.des` (**Figure 2-3**).

Available Values

`0` to `65535`

m_tmzone

SYS_TMZONE

EditMod Labels

```
2-init module contents
25-system time zone
```

Description

This is the system time zone in minutes offset from Greenwich Mean Time (GMT). Therefore, this field would be 360 for a system six time zones west of GMT and -360 for a system six time zones east of GMT.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `INIT/config.des` (**Figure 2-3**).

Available Values

`-32768` to `32767`

EditMod Labels

```
2-init module contents
26-OS-9000 level
```

Description

The OS-9 level is the first byte of a four byte field that is divided into four parts: level, version, revision, and edition number. For example, level 2, version 2, revision 1, edition 0 is 2210.

Port Generic Default Value

`1`

Port Specific Override Value

Refer to `INIT/config.des` (**Figure 2-3**).

Available Values

`0` to `255`

m_major

OS_VERSION

EditMod Labels

```
2-init module contents
27-OS-9000 major release number
```

Description

The OS-9 level is the second byte of a four byte field that is divided into four parts: level, version, revision, and edition number. For example, level 2, version 2, revision 1, edition 0 is 2210.

Port Generic Default Value

`2`

Port Specific Override Value

Refer to `INIT/config.des` (**Figure 2-3**).

Available Values

`0` to `255`

EditMod Labels

```
2-init module contents
28-OS-9000 minor release level
```

Description

The OS-9 level is the third byte of a four byte field that is divided into four parts: level, version, revision, and edition number. For example, level 2, version 2, revision 1, edition 0 is 2210.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `INIT/config.des` (**Figure 2-3**).

Available Values

`0` to `255`

m_edition

OS_EDITION

EditMod Labels

```
2-init module contents
28-OS-9000 edition number
```

Description

The OS-9 level is the fourth byte of a four byte field that is divided into four parts: level, version, revision, and edition number. For example, level 2, version 2, revision 1, edition 0 is 2210.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `INIT/config.des` (**Figure 2-3**).

Available Values

`0` to `255`

EditMod Labels

```
2-init module contents
30-compatibility flags
```

Description

This byte is used for revision compatibility.

Port Generic Default Value

Macro

```
B_WIPEMEM
```

EditMod

```
0x2
```

Port Specific Override Value

Refer to `INIT/config.des` (**Figure 2-3**).

Available Values

Revision compatibility values are located in the header file, `module.h.`, and are listed in **Table 2-7**.

Table 2-7 `m_compat` Available Attribute and Revision Values

Description	Macro	EditMod
Set to ignore sticky bit in the module headers	`B_GHOST`	`0x01`
Set to patternize memory when allocated and returned	`B_WIPEMEM`	`0x02`

Table 2-7 m_compat Available Attribute and Revision Values (continued)

Description	Macro	EditMod
Set to inform the kernel not to automatically set the clock during coldstart	B_NOCLOCK	0x04
Set to not automatically expand system tables	B_EXPTBL	0x08
Set to not use SSM	B_SSM	0x16

EditMod Labels

```
2-init module contents
31-process signal queue size
```

Description

This field specifies the default maximum number of signals queued up for a process.

Port Generic Default Value

`32 (0x20)`

Port Specific Override Value

Refer to `INIT/config.des` (**Figure 2-3**).

Available Values

`0` to `65535`

preio_name

PREIOS_NAME

EditMod Labels

```
2-init module contents
32-pre-IO customization module list
```

Description

Contains the name string of a list of pre-I/O customization modules, if any. These extension modules are initialized and called prior to the initialization of the I/O system during bootstrap. For more information on customization modules, refer to the description of `m_extens`.

Port Generic Default Value

```
NULL
```

Port Specific Override Value

Refer to `INIT/config.des` (**Figure 2-3**).

Available Values

Any ASCII character string. The string may contain C-style character escapes (such as `\n` and `\012`).

Memlist Fields

The memlist fields are in the order they appear during an interactive `EditMod` session. Defined fields can appear in a different order in the description files. The fields can be changed using the `EditMod` utility or by modifying the description files. See **Init Module Field Configuration Options** for detailed instructions on changing these fields.

Table 2-8 Memlist Fields

Field	Description
`type`	Memory type code
`prior`	Memory allocation priority
`access`	Access permissions
`blksiz`	Search block size
`lolim`	Beginning block address
`hilim`	Ending block address
`desc`	Memory list description
`dma_addr`	External bus address

type

Memory Type Code

EditMod Labels

```
2-init module contents
14-memory list
1-Add additional item to list
n-memory list [n-1]
1- memory type code (color)
```

Description

This is the memory type code.

Port Generic Default Value

0 (zero)

Port Specific Override Value

Refer to INIT/config.des (**Figure 2-3**).

Available Values

Memory type values are defined in the header file, memory.h, and are listed in **Table 2-9**.

Table 2-9 type Available Values for Memory Lists

Description	Macro	EditMod
System RAM memory	MEM_SYS	0x01
Shared memory (0x8000 - 0xffff)	MEM_SHARED	0x8000

EditMod Labels

```
2-init module contents
14-memory list
1-Add additional item to list
n-memory list [n-1]
2-memory allocation priority
```

Description

This is the memory allocation priority. High priority RAM is allocated first (255 - 0).If the block priority is 0, the block can only be allocated by a request for the specific color (type) of the block.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `INIT/config.des` (**Figure 2-3**).

Available Values

`0` to `65535`

access

Access Permissions

EditMod Labels

```
2-init module contents
14-memory list
1-Add additional item to list
n-memory list [n-1]
3-access permissions
```

Description

This is the access permissions.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `INIT/config.des` (**Figure 2-3**).

Available Values

Memory type access bit are defined in the header file, `alloc.h`, and in **Table 2-10**.

Note

Only `B_USERRAM` memory can be initialized.

Table 2-10 Access Bit Definitions for Memory Type

Description		config.des Macro	EditMod Hex
bit 0	Indicates memory allocatable by user processes.	`B_USERRAM`	`0x01`
bit 1	Indicates parity memory; the kernel initializes it during start-up.	`B_PARITY`	`0x02`
bit 2	Indicates ROM; the kernel searches this for modules during start-up.	`B_ROM`	`0x04`
bit 3	Non-volatile RAM; the kernel searches this for modules during start-up.	`B_NVRAM`	`0x08`
bit 4	Shared memory.	`B_SHARED`	`0x10`

blksiz

Search Block Size

EditMod Labels

```
2-init module contents
14-memory list
1-Add additional item to list
n-memory list [n-1]
4-search block size
```

Description

This is the search block size. The kernel checks every *search block size* to see if RAM/ROM exists.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `INIT/config.des` (**Figure 2-3**).

Available Values

`0` to `65535`

EditMod Labels

```
2-init module contents
14-memory list
1-Add additional item to list
n-memory list [n-1]
5-beginning address for this type
```

Description

This is the beginning address of the block as referenced by the CPU.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `INIT/config.des` (**Figure 2-3**).

Available Values

`0` to `4294967295`

hilim

Ending Block Address

EditMod Labels

```
2-init module contents
14-memory list
1-Add additional item to list
n-memory list [n-1]
5-ending address + 1 for this type
```

Description

This is the ending address of the block as referenced by the CPU.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `INIT/config.des` (**Figure 2-3**).

Available Values

`0` to `4294967295`

EditMod Labels

```
2-init module contents
14-memory list
1-Add additional item to list
n-memory list [n-1]
6-memory list description
```

Description

This contains the memory list description name string.

Port Generic Default Value

`NULL`

Port Specific Override Value

Refer to `INIT/config.des` (**Figure 2-3**).

Available Values

Any ASCII character string. The string may contain C-style character escapes (such as `\n` and `\012`).

dma_addr

External Bus Address

EditMod Labels

```
2-init module contents
14-memory list
1-Add additional item to list
n-memory list [n-1]
7-translation address for dma’s
```

Description

External bus address of the beginning of the block. If `0`, this field does not apply.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `INIT/config.des` (**Figure 2-3**).

Available Values

`0` to `4294967295`

Cachelist Fields

The cachelist fields are in the order they appear during an interactive EditMod session. Defined fields may appear in a different order in the description files. The fields can be changed using the EditMod utility or by modifying the description files. See **Init Module Field Configuration Options** for detailed instructions on changing these fields.

Table 2-11 Cachelist Fields

Field	Description
blk_beg	Beginning address of memory region
blk_end	Ending address of memory region

blk_beg

Beginning Address of Memory Region

EditMod Labels

```
2-init module contents
33-cache list
n-cache list[n-1]
1-beginning address of memory region
```

Description

This is the beginning address of the memory region.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `INIT/config.des` (**Figure 2-3**).

Available Values

`0` to `4294967295`

blk_end

Ending Address of Memory Region

EditMod Labels

```
2-init module contents
33-cache list
n-cache list[n-1]
1-ending address + 1 of memory region
```

Description

This is the ending address of the memory region plus 1.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `INIT/config.des` (**Figure 2-3**).

Available Values

`0` to `4294967295`

Chapter 3: SCF Device Descriptors

SCF device descriptors contain configuration data specific to one serial device on an OS-9 system. Values that can be configured in the descriptor include:

- Device interrupt vector and priority
- Device I/O address
- Serial communication settings
- Special character mapping

The next section in this chapter provides a detailed example of the configuration options you can use to change configuration values in SCF descriptors.

The rest of this chapter provides a detailed list of all of the SCF device descriptor fields, including field descriptions and available values.

This chapter includes the following topics:

RadiSys.

MICROWARE SOFTWARE

SCF Field Configuration Options

To change an SCF device descriptor module configuration field, you can use either of the following methods:

1. Use the `EditMod` utility to directly modify existing SCF device descriptor modules either as a stand-alone module or as part of a merged module group (such as a boot image).
2. Modify the description file for the SCF device descriptor module and rebuild it using the makefile provided.

Direct Modification Advantages

The direct modification method has the following advantages:

Fast	No source configuration file rebuilds are necessary.
Temporary	The original module or merged-module group configuration can be easily restored through the appropriate rebuild.
Contained	Changes are limited to the individual boot image modified (merged-module option).

Description File/Rebuild Advantages

The advantage of the description file/rebuild method is that the changes are permanent and reproducible. Modifications apply to all subsequent module rebuilds and to all merged-module groups built containing the updated module.

Both methods are documented in this section. These procedures are used with the field descriptions starting with the **Module Header Fields**. For direct modification, use the `EditMod` LABELS data to navigate the

EditMod menus. The DESCRIPTION FILE MACRO data identifies the macro you need to define/modify in the configuration sources to rebuild the SCF device descriptor module.

Direct Modification

Use the Editmod utility and the following procedures to directly modify fields in the existing SCF device descriptor module. The module can stand-alone or it can be part of a merged-module group. A boot image, for example, contains multiple modules. Both situations are covered in this section. The field references later in this chapter contain a description of each configurable field, its supported values, and the sequence of menu options required by EditMod to modify that field.

For More Information

Refer to the ***Utilities Reference*** for a full description of EditMod's capabilities.

Figure 3-1 Directory Location for Modifying an SCF Device Descriptor

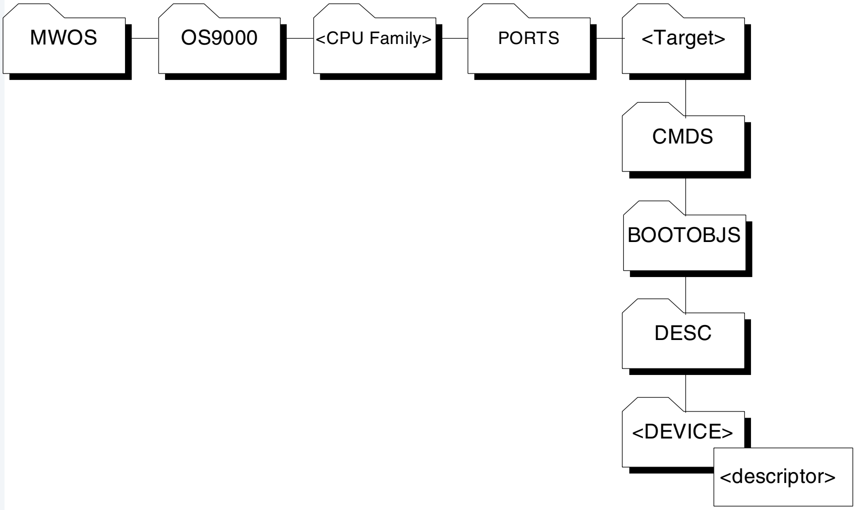
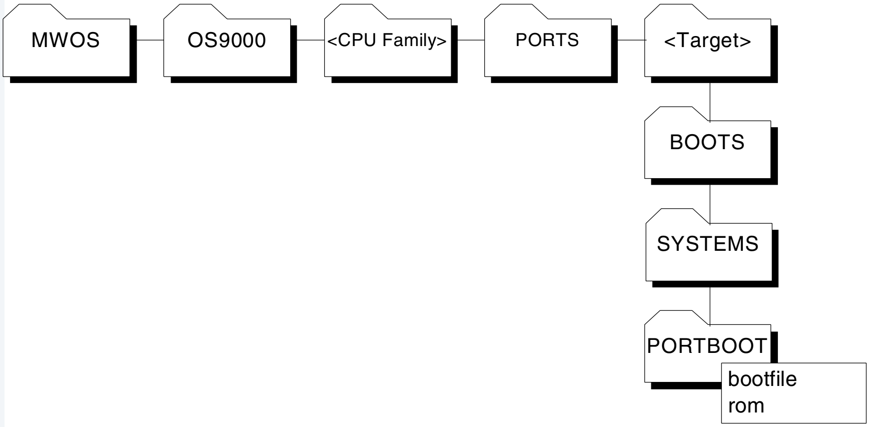


Figure 3-2 Directory Location for Modifying Low-Level Boot Images



For More Information

Refer to your board guide for information about how to modify the module lists and remake the boot images, and for specified boot image names.

Direct Modification Procedures

To modify the stand-alone module, complete the following steps:

Step 1. Go to the `CMDS/BOOTOBJS/DESC/<DEVICE>` directory (see **Figure 3-1**).

Step 2. Use EditMod to edit the module:

```
$EditMod -e <descriptor>
```

To modify the module as part of a merged module group, complete the following steps:

Step 1. Go to the `BOOTS/SYSTEMS/PORTBOOT` directory (see **Figure 3-2**).

Step 2. Use EditMod to edit the module:

```
$EditMod -e <descriptor> -f=<boot image name>
```

Step 3. Use the menu selections provided in the `EditMod` LABELS section of the field reference later in this chapter to locate the fields you want to edit.

Step 4. Select a new value for the field from the AVAILABLE VALUES section of the field reference. Enter that value at the EditMod prompt to modify the field.

Step 5. If you want to make additional modifications, use the `p` command (previous) to step backward through the EditMod menus. Repeat Steps 3 and 4 until you have made all desired modifications to the descriptor.

Step 6. Select the `w` command (write) to save the changes.

Step 7. Select the `q` command (quit) to exit `EditMod`.

Note

Unless you modified the SCF device descriptors in your boot image, you should rebuild your boot image to include the new descriptor.

Example EditMod Session

This example modifies an SCF device descriptor as part of the boot image `rom`:

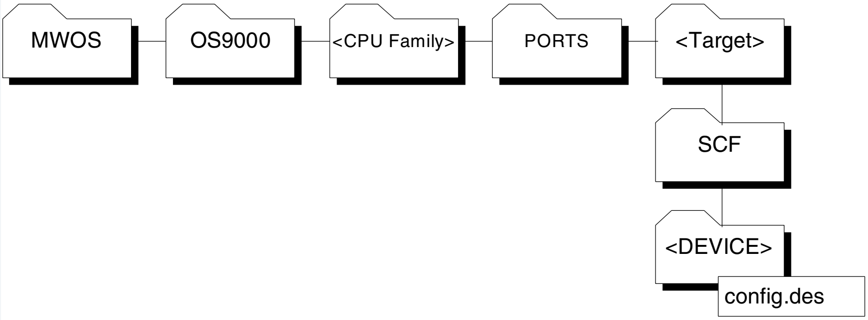
```
$ EditMod -e term -f=rom

 1. module header
 2. device descriptor data definitions
 3. SCF description block
 4. SCF logical unit static storage
 5. SCF path options

$Which? [?/1-2/p/t/a/w/q] 4

Which? [?/1-5/p/t/a/w/q] 4

 1. irq vector number                       : 0x4c
 2. irq interrupt level                     : 0x0
 3. irq polling priority                    : 5
 4. polled input flag                       : 1
 5. polled output flag                      : 1
 6. driver accessible copy of logical unit number: 0x1
 7. interrupt mask word                     : 0x80
 8. send XOFF when buffer is this full      : 246
 9. size of input buffer                    : 256
10. input buffer
11. size of output buffer                   : 256
12. output buffer
13. lines left until end of page            : 24
14. keyboard interrupt character            : ‘\x03’
15. keyboard quit character                 : ‘\x05’
16. keyboard pause character                : ‘\x17’
17. x-on character                          : ‘\x11’
18. x-off character                         : ‘\x13’
19. baud rate                               : 0xf
20. parity                                  : 0
21. stop bits                               : 0
```

```
22. word size                                : 8
23. RTS state                                : 0

$Which? [?/1-6/p/t/a/w/q] 3

irq polling priority                    : 5
New value: 8

irq polling priority                    : 8

Which? [?/1-19/p/t/a/w/q] w

Which? [?/1-19/p/t/a/w/q] q
```

Description File Modification

You can use these procedures to modify the appropriate description file and rebuild the SCF device descriptors for your port directory. The DESCRIPTION FILE MACROS section of the field reference specifies the name of the macro you modify/define in the description files to configure the field. The value used in the define is chosen from the AVAILABLE VALUES specified for the field.

Figure 3-3 Directory Location for Modifying the SCF Description File



Description File Modification Procedures

Step 1. Change to the `SCF/<DEVICE>` directory (see **Figure 3-3**).

Step 2. Edit the file `config.des` and read the included comments for more information on using the specific description files provided in your software distribution. The `config.des` file contains a list of macro names that can be defined to override the global default values for the configuration fields.

Step 3. Refer to the DESCRIPTION FILE MACRO section in the field reference later in this chapter to determine the macro name you define to configure the target field.

Step 4. Read the comments in `config.des` to determine where to place the define for this macro.

Step 5. Select the value you want to use to configure the field. See the AVAILABLE VALUES section of the field reference data for values or macros that can be used for the definition. Define the macro by entering a definition in the appropriate description files as follows:

```
#define <macro> <value>
```

Step 6. Save the changes and rebuild the SCF device descriptors, entering the following command in the `SCF/<DEVICE>/DESC` directory:

```
os9make
```

Step 7. Rebuild your boot image to include the new descriptor.

SCF Device Descriptor Field Reference

This section contains a list of the most commonly configured fields in the SCF device descriptors. Each field entry contains the following information:

- <Field name> - The call name for each field that can be reconfigured in the module.
- EditMod LABELS - `EditMod` menu selections for navigating to the proper field in an `EditMod` session.
- DESCRIPTION FILE MACRO - The macro name you modify/define in the description file.
- DESCRIPTION - A brief description of the field's purpose and use.
- EXAMPLE - An optional example of the description file entry showing how to change the value of this field.
- PORT GENERIC DEFAULT VALUE - The value set in the port generic description file for this field. This is the value the field is assigned when the module is built, unless the appropriate macro has been defined in the port specific description file to override this default value.
- PORT SPECIFIC OVERRIDE VALUE - The value set in the port specific description file for this field. If defined, this is the value the field is assigned when the module is built, overriding the port generic default value.
- AVAILABLE VALUES - Values to which the field can be set through `EditMod` or the description files. In many cases, this data is presented in a table that maps a description of the value to a numeric value appropriate for entry in EditMod, and to a pre-defined macro available for use in the description file.

Module Header Fields

The following section contains the module header fields in the order they appear during an interactive EditMod session. Defined fields can appear in a different order in config.des.

Table 3-1 Module Header Fields

Field	Description File Macro
_m_group	MH_GROUP
_m_user	MH_USER
mod_name	MH_NAME
m_access	MH_ACCESS
m_tylan	MH_TYLAN
m_attrev	MH_ATTREV
m_edit	MH_EDITION

_m_group

MH_GROUP

EditMod Labels

```
1-module header
1-module owner's group number
```

Description

Group ID of the module's owner. The group number allows people working in the same department or on the same project to share a common identification number.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

`0` to `65535`

EditMod Labels

```
1-module header
2-module owner’s user number
```

Description

User ID of the module’s owner. The user number identifies a specific user.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

`0` to `65535`

mod_name

MH_NAME

EditMod Labels

```
1-module header
3-module name
```

Description

Contains the module name string.

Port Generic Default Value

```
NULL
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

Any ASCII character string. The string can contain C-style character escapes (such as `\n` and `\012`).

EditMod Labels

```
1-module header
4-access permissions
```

Description

Defines the permissible module access by its owner or by other users.

Port Generic Default Value

Macro

```
MP_OWNER_READ | MP_OWNER_EXEC | MP_GROUP_READ |
MP_GROUP_EXEC | MP_WORLD_READ | MP_WORLD_EXEC
```

EditMod

```
0x555
```

Port Specific Override Value

Refer to SCF/<DEVICE>/DESC/config.des (**Figure 3-3**).

Available Values

Module access permission values are located in the header file, module.h, and are listed in **Table 3-2**.

Table 3-2 m_access **Available Values**

Description	Macro	EditMod
Read permission by owner	MP_OWNER_READ	0x0001
Write permission by owner	MP_OWNER_WRITE	0x0002

Table 3-2 `m_access` Available Values (continued)

Description	Macro	EditMod
Execute permission by owner	`MP_OWNER_EXEC`	`0x0004`
Owner permission mask	`MP_OWNER_MASK`	`0x000f`
Read permission by group	`MP_GROUP_READ`	`0x0010`
Write permission by group	`MP_GROUP_WRITE`	`0x0020`
Execute permission by group	`MP_GROUP_EXEC`	`0x0040`
Group permission mask	`MP_GROUP_MASK`	`0x00f0`
Read permission by world	`MP_WORLD_READ`	`0x0100`
Write permission by world	`MP_WORLD_WRITE`	`0x0200`
Execute permission by world	`MP_WORLD_EXEC`	`0x0400`
World permission mask	`MP_WORLD_MASK`	`0x0f00`
All permissions for owner, group, and world	`MP_WORLD_ACCESS`	`0x0777`
System permission mask	`MP_SYSTM_MASK`	`0xf000`

EditMod Labels

```
1-module header
5-type/language
```

Description

Contains the module's type (first byte) and language (second byte). The language codes indicate if the module is executable and which language the run-time system requires for execution, if any.

Port Generic Default Value

Macro

```
(MT_DATA<<8) + ML_OBJECT
```

EditMod

```
0x401
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

Module type values and language codes are located in the header file, `module.h`, and are listed in **Table 3-3** and **Table 3-4**.

Table 3-3 `m_tylan` Available Module Type Values

Description	Macro	EditMod
Not used (wildcard value in system calls)	`MT_ANY`	`0x0000`
Program module	`MT_PROGRAM`	`0x0001`
Subroutine module	`MT_SUBROUT`	`0x0002`
Multi-module (reserved for future use)	`MT_MULTI`	`0x0003`
Data module	`MT_DATA`	`0x0004`
Configuration data block data module	`MT_CDBDATA`	`0x0005`
Reserved for future use	`0xb-0xa`	`0xb-0xa`
User trap library	`MT_TRAPLIB`	`0x000b`
System module	`MT_SYSTEM`	`0x000c`
File manager module	`MT_FILEMAN`	`0x000d`
Physical device driver	`MT_DEVDRVR`	`0x000e`
Device descriptor module	`MT_DEVDESC`	`0x000f`

Table 3-3 m_tylan Available Module Type Values (continued)

Description	Macro	EditMod
User definable	0x10-0xfe	0x10-0xfe
Module type mask	MT_MASK	0xff00

Table 3-4 m_tylan Available Language Code Values

Description	Macro	EditMod
Unspecified language (wildcard in system calls)	ML_ANY	0x0
Machine language	ML_OBJECT	0x1
Basic I-code (reserved for future use)	ML_ICODE	0x2
Pascal P-code (reserved for future use)	ML_PCODE	0x3
C I-code (reserved for future use)	ML_CCODE	0x4
Cobol I-code (reserved for future use)	ML_CBLCODE	0x5
Fortran	ML_FRTNCODE	0x6
Reserved for future use	0x7-0xf	0x7-0xf
User-definable	0x10-0xfe	0x10-0xfe
Module language mask	ML_MASK	0x00ff

m_attrev

MH_ATTREV

EditMod Labels

```
1-module header
6-revision/attributes
```

Description

Contains the module’s attributes (first byte) and revision (second byte).

Port Generic Default Value

Macro

```
MA_REENT<<8
```

EditMod

```
0x8000
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

Module attribute and revision codes are located in the header file `module.h.`, and are listed in **Table 3-5**.

Note

If two modules with the same name are found in the memory search or are loaded into the current module directory, only the module with the highest revision level is kept. This enables easy substitution of modules for update or correction.

Table 3-5 m_attrev Available Attribute and Revision Values

Description	Macro	EditMod
The module is re-entrant (sharable by multiple tasks).	MA_REENT (shifted left to first byte: MA_REENT<<8)	0x80 (shifted left to first byte: 0x8000)
The module is sticky. A sticky module is not removed from memory until its link count becomes -1 or memory is required for another use.	MA_GHOST (shifted left to first byte: MA_GHOST<<8)	0x40 (shifted left to first byte: 0x4000)
The module is a system-state module.	MA_SUPER (shifted left to first byte: MA_SUPER<<8)	0x20 (shifted left to first byte: 0x2000)
User-definable revision number	0x0-0xfe	0x0 - 0xfe
Module attribute mask	MA_MASK	0xff00
Module revision mask	MR_MASK	0x00ff

m_edit

MH_EDITION

EditMod Labels

```
1-module header
7-edition
```

Description

Indicates the software release level for maintenance. OS-9 does not use this field. Whenever a program is revised (even for a small change), increase this number. We recommend internal documentation within the source program be keyed to this system.

Port Generic Default Value

`1`

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

`0` to `65535`

Device Descriptor Data Definition Fields

The following section contains the device descriptor data definition fields in the order they appear during an interactive EditMod session. Defined fields can appear in a different order in config.des.

Table 3-6 Device Descriptor Data Definition Fields

Field	Description File Macro
dd_port	PORTADDR
dd_lun	LUN
dd_pd_size	PD_SIZE
dd_type	DD_TYPE
dd_mode	DD_MODE
fmgr_name	FMGR_NAME
drvr_name	DRVR_NAME
dd_class	DD_CLASS

dd_port

PORTADDR

EditMod Labels

```
2-device descriptor data definitions
1-device port address
```

Description

Absolute physical address of the hardware controller. This is the address of the device on the bus. This is the lowest address the device has mapped. Port address is hardware dependent.

Macro Example

```
#define PORTADDR    0xfffe4000
```

Port Generic Default Value

0 (zero)

Port Specific Override Value

Refer to SCF/<DEVICE>/DESC/config.des (**Figure 3-3**).

Available Values

0 to 4294967295

EditMod Labels

```
2-device descriptor data definitions
2-logical unit number
```

Description

Distinguishes the different devices driven from a unique controller. Each unique number represents a different logical unit static storage area.

Macro Example

```
#define LUN    2
```

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

`0` to `65535`

dd_pd_size

PD_SIZE

EditMod Labels

```
2-device descriptor data definitions
3-path descriptor size
```

Description

Size of the path descriptor. IOMAN uses this value when it allocates a path descriptor.

Port Generic Default Value

`0x234`

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

`0` to `65535`

EditMod Labels

```
2-device descriptor data definitions
4-device type
```

Description

Identifies the I/O class of the device.

Port Generic Default Value

Macro

```
DT_SCF
```

EditMod

```
0x0
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

Device type values are defined in the header file `io.h`, and are listed in **Table 3-7**.

Table 3-7 `dd_type` Available Values

Description	Macro	EditMod
Sequential Character File Type	`DT_SCF`	`0x0`
Random Block File Type	`DT_RBF`	`0x1`
Pipe File Type	`DT_PIPE`	`0x2`

Table 3-7 dd_type Available Values (continued)

Description	Macro	EditMod
Sequential Block File Type	DT_SBF	0x3
Network File Type	DT_NFM	0x4
Compact Disc File Type	DT_CDFM	0x5
User Communication Manager	DT_UCM	0x6
Socket Communication Manager	DT_SOCK	0x7
Pseudo-Keyboard Manager	DT_PTTY	0x8
Graphics File Manager	DT_GFM	0x9
Inet File Manager	DT_INET	0x10
Multi-media File Manager	DT_MFM	0x11
Generic Device File Manager	DT_DVM	0x12
Null File Manager	DT_NULL	0x13
DVD File Manager	DT_DVDFM	0x14
Module Directory File System Manager	DT_MODFM	0x15
PC-DOS File Manager	DT_PCF	0xa
Non-volatile RAM File Manager	DT_NRF	0xb
ISDN File Manager	DT_ISDN	0xc
MPFM File Manager	DT_MPFM	0xd

Table 3-7 dd_type Available Values (continued)

Description	Macro	EditMod
Real-Time Network File Manager	DT_RTNFM	0xe
Serial Protocol File Manager	DT_SPF	0xf
Reserved for Microware Use Only	17-127	0xa1-0x7f

dd_mode

DD_MODE

EditMod Labels

```
2-device descriptor data definitions
5-device mode capabilities
```

Description

Used to check the validity of a caller's access mode byte in `I_CREATE` or `I_OPEN` system calls. If a bit is set, the device can perform the corresponding function. The `S_ISIZE` bit is usually set, because it is handled by the file manager or ignored. If the `S_ISHARE` bit is set, the device is non-sharable. A printer is an example of a non-sharable device.

Port Generic Default Value

Macro

```
S_ISIZE | S_IREAD | S_IWRITE
```

EditMod

```
0x2003
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

The file access modes are defined in the header file, `modes.h`, and located in **Table 3-8**. The file access permission values are defined in the header file `modes.h` and in **Table 3-9**.

Table 3-8 `dd_mode` **Available Values for File Access Modes**

Description	Macro	EditMod
Truncate on open	`S_ITRUNC`	`0x0100`
Ensure contiguous file	`S_ICONTIG`	`0x0400`
Error if file exists on create	`S_IEXCL`	`0x0400`
Create file	`S_ICREAT`	`0x0800`
Append to file	`S_IAPPEND`	`0x1000`
Non-sharable	`S_ISHARE`	`0x4000`

Table 3-9 `dd_mode` **Available Values for File Access Permissions**

Description	Macro	EditMod
Mask for permission bits	`S_IPRM`	`0xffff`
Owner read	`S_IREAD`	`0x0001`
Owner write	`S_IWRITE`	`0x0002`
Owner execute	`S_IEXEC`	`0x0004`
Search permission	`S_ISEARCH`	`0x0004`

Table 3-9 `dd_mode` Available Values for File Access Permissions

Description	Macro	EditMod
Group read	`S_IGREAD`	`0x0010`
Group write	`S_IGWRITE`	`0x0020`
Group execute	`S_IGEXEC`	`0x0040`
Group search	`S_IGSEARCH`	`0x0040`
Public read	`S_IOREAD`	`0x0100`
Public write	`S_IOWRITE`	`0x0200`
Public execute	`S_IOEXEC`	`0x0400`
Public search	`S_IOSEARCH`	`0x0400`

EditMod Labels

```
2-device descriptor data definitions
6-file manager name
```

Description

Contains the name string of the file manager module to use.

Port Generic Default Value

```
“scf”
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

Any ASCII character string. The string may contain C-style character escapes (such as `\n` and `\012`).

drvr_name

DRVR_NAME

EditMod Labels

```
2-device descriptor data definitions
7-driver name
```

Description

Contains the name string of the device driver module to use.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

Any ASCII character string. The string may contain C-style character escapes (such as `\n` and `\012`).

EditMod Labels

```
2-device descriptor data definitions
8-device class (sequential or random)
```

Description

Used to identify the class of the device, whether it is random or sequential access.

Port Generic Default Value

Macro

```
DC_SEQ
```

EditMod

```
0x1
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

Device class available values are defined in the header file, `io.h`, and in **Table 3-10**.

Table 3-10 `dd_class` **Available Values**

Description	Macro	EditMod
Sequential access device	`DC_SEQ`	`0x0001`
Random access device	`DC_RND`	`0x0002`

SCF Description Block Fields

The following section contains the SCF description block fields in the order they appear during an interactive `EditMod` session. Defined fields can appear in a different order in `config.des`.

Table 3-11 SCF Description Block Fields

Field	Description File Macro
`outdev_name`	`OUTDEVNAME`

outdev_name

OUTDEVNAME

EditMod Labels

```
3-SCF description block
1-output device name
```

Description

Macro Example

Port Generic Default Value

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (Figure 3-3).

Available Values

SCF Logical Unit Static Storage Fields

The following section contains the SCF logical unit static storage fields in the order they appear during an interactive EditMod session. Defined fields can appear in a different order in config.des.

Table 3-12 Device Descriptor Data Definition Fields

Field	Description File Macro
hardware_vector	VECTOR
v_irqlevel	IRQLEVEL
v_priority	PRIORITY
v_pollin	INPUT_TYPE
v_pollout	OUTPUT_TYPE
v_lun	LUN
v_irqmask	IRQ_MASK
v_maxbuff	MAXBUFF
v_insize	INSIZE
v_outsize	OUTSIZE
v_line	PAGE_SIZE
v_intr	KYBDINTR
v_quit	KYBDQUIT

Table 3-12 Device Descriptor Data Definition Fields (continued)

Field	Description File Macro
v_psch	KYBDPAUSE
v_xon	XON
v_xoff	XOFF
v_baud	BAUDRATE
v_parity	LUPARITY
v_stopbits	STOPBITS
v_wordsize	WORDSIZE
v_rtsstate	RTSSTATE
v_devspec	

hardware_vector

VECTOR

EditMod Labels

```
4-SCF logical unit static storage
1-irq vector number
```

Description

This is the vector passed to the processor at interrupt time. Vector is hardware/software dependent. You can program some devices to produce different vectors. See your board guide for vector mappings for specific processors.

Macro Example

```
#define VECTOR  80
```

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

`0` to `255`

EditMod Labels

```
4-SCF logical unit static storage
2-irq interrupt level
```

Description

This is the hardware priority of the console device interrupt. When a device interrupts a processor, the level of the interrupt is used to mask lower priority interrupts.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

`0` to `65535`

The number of supported interrupt levels is dependent on the processor being used (for example, 1-7 on 680x0 type CPUs).

v_priority

PRIORITY

EditMod Labels

```
4-SCF logical unit static storage
3-irq polling priority
```

Description

This is the software (polling) priority for the console device on the IRQ polling table.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

The interrupt priority value range is 0-65534 (65535 is reserved). A non-zero priority determines the position of the device within the vector. Lower values are polled first.

Some considerations to keep in mind when selecting an interrupt priority:

- A priority of 0 indicates the device desires exclusive use of the vector.
- If the priority is 1, it is polled first and no other device can have a priority of 1 on the vector. For all other priority values, more than one device can share the same priority on a vector. In this case, first-in, first-out (FIFO) scheduling determines the order of precedence in the polling table for the devices.
- OS-9 does not allow a device to claim exclusive use of a vector if another device has already been installed on the vector. Additionally, it does not allow another device to use the vector once the vector has been claimed for exclusive use.
- This value is software dependent.

See Also

`F_IRQ` system call entry in the ***OS-9 Technical Manual***.

v_pollin

INPUT_TYPE

EditMod Labels

```
4-SCF logical unit static storage
4-polled input flag
```

Description

This specifies whether input on the device is interrupt driven or polled. If the device is operated in polled mode, SCF calls the driver's read routine for every character. This value is device dependent.

Macro Example

```
#define INPUT_TYPE  IRQDRIVEN
```

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

Device input values are defined in the header file, `scf.h`, and in **Table 3-13**.

Table 3-13 `v_pollin` **Available Values**

Description	Macro	EditMod
Interrupt driven	`IRQDRIVEN`	`0x0000`
Polled	`POLLED`	`0x0001`

v_pollout

OUTPUT_TYPE

EditMod Labels

```
4-SCF logical unit static storage
5-polled output flag
```

Description

This specifies whether output on the device is interrupt driven or polled. If the device is operated in polled mode, SCF calls the driver's write routine to transmit every character. This value is device dependent.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

Device input values are defined in the header file, `scf.h`, and in **Table 3-14**.

Table 3-14 `v_pollout` Available Values

Description	Macro	EditMod
Interrupt driven	`IRQDRIVEN`	`0x0000`
Polled	`POLLED`	`0x0001`

v_lun

LUN

EditMod Labels

```
4-SCF logical unit static storage
6-driver accessible copy of logical unit number
```

Description

Since more than one device may have the same port address, the logical unit number distinguishes the devices having the same port address.

Macro Example

```
#define LUN  2
```

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

`0` to `65535`

EditMod Labels

```
4-SCF logical unit static storage
7-interrupt mask word
```

Description

This is the interrupt mask for the SCF device.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

`0` to `65535`

v_maxbuff

MAXBUFF

EditMod Labels

```
4-SCF logical unit static storage
8-send XOFF when buffer is this full
```

Description

This specifies the device to send on XOFF when the buffer is full, in bytes.

Port Generic Default Value

`246`

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

`0` to `65535`

v_insize

INSIZE

EditMod Labels

```
4-SCF logical unit static storage
9-size of input buffer
```

Description

This specifies the size of the input buffer for the logical unit.

Port Generic Default Value

256

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

0 to 65535

v_outsize

OUTSIZE

EditMod Labels

```
4-SCF logical unit static storage
11-size of output buffer
```

Description

This specifies the size of the output buffer for the logical unit.

Port Generic Default Value

```
256
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

`0` to `65535`

EditMod Labels

```
4-SCF logical unit static storage
13-lines left until end of page
```

Description

This specifies the number of lines per screen (or page).

Port Generic Default Value

`24`

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

`0` to `65535`

v_intr

KYBDINTR

EditMod Labels

```
4-SCF logical unit static storage
14-keyboard interrupt character
```

Description

This specifies the control key to use for the keyboard interrupt function.

Port Generic Default Value

Macro

```
CTRL_C
```

EditMod

```
0x03
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

The ASCII control and special characters are defined in the header file, `scf.h`, and in **Table 3-15**.

Table 3-15 ASCII Control Character Available Values

SCF/OS-9 Compatible Standard Codes	Macro	EditMod
	C_NULL	0x00
C_REPEAT	CTRL_A	0x01

Table 3-15 ASCII Control Character Available Values (continued)

SCF/OS-9 Compatible Standard Codes	Macro	EditMod
	CTRL_B	0x02
C_INTR	CTRL_C	0x03
C_REPRINT	CTRL_D	0x04
C_QUIT	CTRL_E	0x05
	CTRL_F	0x06
C_BELL	CTRL_G	0x07
C_BACKSPACE	CTRL_H	0x08
C_TAB	CTRL_I	0x09
C_LINEFEED	CTRL_J	0x0A
	CTRL_K	0x0B
C_FORMFEED	CTRL_L	0x0C
C_CR	CTRL_M	0x0D
	CTRL_N	0x0E
	CTRL_O	0x0F
	CTRL_P	0x10
C_XON	CTRL_Q	0x11
	CTRL_R	0x12

Table 3-15 ASCII Control Character Available Values (continued)

SCF/OS-9 Compatible Standard Codes	Macro	EditMod
C_XOFF	CTRL_S	0x13
	CTRL_T	0x14
	CTRL_U	0x15
	CTRL_V	0x16
C_PAUSE	CTRL_W	0x17
C_DELLINE	CTRL_X	0x18
	CTRL_Y	0x19
	CTRL_Z	0x1A
	CTRL_SPACE	0x20
	CTRL_COMMA	0x2C
	CTRL_PERIOD	0x2E
	CTRL_SLASH	0x2F
C_EOF		0x1B

EditMod Labels

```
4-SCF logical unit static storage
15-keyboard quit character
```

Description

This specifies the control key to use for the keyboard quit function.

Port Generic Default Value

Macro

```
CTRL_E
```

EditMod

```
0x05
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

See **Table 3-15**.

v_psch

KYBDPAUSE

EditMod Labels

```
4-SCF logical unit static storage
16-keyboard pause character
```

Description

This specifies the control key to use for the keyboard pause function.

Port Generic Default Value

Macro

```
CTRL_W
```

EditMod

```
0x17
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

See **Table 3-15**.

EditMod Labels

```
4-SCF logical unit static storage
17-x-on character
```

Description

This specifies the control key to use for the X-ON protocol function.

Port Generic Default Value

Macro

```
   CTRL_Q
```

EditMod

```
   0x11
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

See **Table 3-15**.

v_xoff

XOFF

EditMod Labels

```
4-SCF logical unit static storage
18-x-off character
```

Description

This specifies the control key to use for the X-OFF protocol function.

Port Generic Default Value

Macro

```
CTRL_S
```

EditMod

```
0x13
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

See **Table 3-15**.

EditMod Labels

```
4-SCF logical unit static storage
19-baud rate
```

Description

This specifies the baud rate of the device.

Port Generic Default Value

Macro

```
9600
```

EditMod

```
0xf
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

The SCF device descriptor baud rate values are located in the header file, `scf.h`, and are listed in **Table 3-16**.

Table 3-16 `cons_baudrate` **Available Values**

Description	Macro	EditMod
Hardwire baud rate	`HARDWIRE`	`0x00`
50 bits per second (bps)	`BAUD50`	`0x01`
75 bps	`BAUD75`	`0x02`

Table 3-16 `cons_baudrate` Available Values (continued)

Description	Macro	EditMod
110 bps	BAUD110	0x03
134.5 bps	BAUD134P5	0x04
150 bps	BAUD150	0x05
300 bps	BAUD300	0x06
600 pbs	BAUD600	0x07
1200 bps	BAUD1200	0x08
1800 bps	BAUD1800	0x09
2000 bps	BAUD2000	0x0A
2400 bps	BAUD2400	0x0B
3600 bps	BAUD3600	0x0C
4800 bps	BAUD4800	0x0D
7200 bps	BAUD7200	0x0E
9600 bps	BAUD9600	0x0F
19,200 bps	BAUD19200	0x10
31,250 bps	BAUD31250	0x11
38,400 bps	BAUD38400	0x12
56,000 bps	BAUD56000	0x13

Table 3-16 cons_baudrate Available Values (continued)

Description	Macro	EditMod
57,600 bps	BAUD57600	0x14
64,000 bps	BAUD64000	0x15
115,200 bps	BAUD115200	0x16

v_parity

LUPARITY

EditMod Labels

```
4-SCF logical unit static storage
20-parity
```

Description

This specifies the parity mode of the device.

Port Generic Default Value

Macro

```
NOPARITY
```

EditMod

```
0
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

Parity modes are defined in the header file, `scf.h`, and **Table 3-17**.

Table 3-17 `v_parity` Available Values

Description	Macro	EditMod
No parity	`NOPARITY`	`0x00`
Odd parity	`ODDPARITY`	`0x01`
Even parity	`EVENPARITY`	`0x02`

Table 3-17 `v_parity` Available Values (continued)

Description	Macro	EditMod
Mark parity	`MARKPARITY`	`0x03`
Space parity	`SPACEPARITY`	`0x04`

v_stopbits

STOPBITS

EditMod Labels

```
4-SCF logical unit static storage
21-stop bits
```

Description

This specifies the number of stop bits to be used for transmission.

Port Generic Default Value

Macro

```
ONESTOP
```

EditMod

```
0
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

See **Table 3-18**.

Table 3-18 `cons_stopbits` Available Values

Description	Macro	EditMod
Stop bit length of 1	`ONESTOP`	`0x0`
Stop bit length of 1.5	`ONE_5STOP`	`0x1`
Stop bit length of 2	`TWO_STOP`	`0x2`

EditMod Labels

```
4-SCF logical unit static storage
22-word size
```

Description

This specifies the number of bits per character to be used for transmission.

Port Generic Default Value

Macro

```
WORDSIZE8
```

EditMod

```
8
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

Word size values are located in the header file, `scf.h`, and are listed in **Table 3-19**.

Table 3-19 `v_wordsize` Available Values

Description	Macro	EditMod
5 bits per character	`WORDSIZE5`	`0x5`
6 bits per character	`WORDSIZE6`	`0x6`

Table 3-19 `v_wordsize` Available Values (continued)

Description	Macro	EditMod
7 bits per character	`WORDSIZE7`	`0x7`
8 bits per character	`WORDSIZE8`	`0x8`

v_rtsstate

RTSSTATE

EditMod Labels

```
4-SCF logical unit static storage
23-RTS state
```

Description

This determines the state of the Request to Send (RTS) line for hardware handshaking.

Port Generic Default Value

Macro

```
RTSDISABLED
```

EditMod

```
0
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

The Request to Send (RTS) state values are defined in the header file, `scf.h`, and in **Table 3-20**.

Table 3-20 `v_rtsstate` Available Values

Description	Macro	EditMod
rts disabled	`RTSDISABLED`	`0x0`
rts enabled	`RTSENABLED`	`0x1`

v_devspec

EditMod Labels

```
4-SCF logical unit static storage
24-<device specific storage label> (optional)
```

Description

Optional device specific information structure. Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**) to determine if structure exists, and if so, the available fields.

SCF Path Option Fields

The following section contains the SCF path option fields in the order they appear during an interactive EditMod session. Defined fields can appear in a different order in config.des.

Table 3-21 Device Descriptor Data Definition Fields

Field	Description File Macro
pd_inmap0type	TYPE0x7f
pd_inmap0func_code	FUNC0x7f
pd_inmap0size	SIZE0x7f
pd_inmap0string	STRING0x7f
pd_inmap1type	TYPE0x01
pd_inmap1func_code	FUNC0x01
pd_inmap1size	SIZE0x01
pd_inmap1string	STRING0x01
pd_inmap2type	TYPE0x02
pd_inmap2func_code	FUNC0x02
pd_inmap2size	SIZE0x02
pd_inmap2string	STRING0x02
pd_inmap3type	TYPE0x03

Table 3-21 Device Descriptor Data Definition Fields (continued)

Field	Description File Macro
pd_inmap3func_code	FUNC0x03
pd_inmap3size	SIZE0x03
pd_inmap3string	STRING0x03
pd_inmap4type	TYPE0x04
pd_inmap4func_code	FUNC0x04
pd_inmap4size	SIZE0x04
pd_inmap4string	STRING0x04
pd_inmap5type	TYPE0x05
pd_inmap5func_code	FUNC0x05
pd_inmap5size	SIZE0x05
pd_inmap5string	STRING0x05
pd_inmap6type	TYPE0x06
pd_inmap6func_code	FUNC0x06
pd_inmap6size	SIZE0x06
pd_inmap6string	STRING0x06
pd_inmap7type	TYPE0x07
pd_inmap7func_code	FUNC0x07

Table 3-21 Device Descriptor Data Definition Fields (continued)

Field	Description File Macro
pd_inmap7size	SIZE0x07
pd_inmap7string	STRING0x07
pd_inmap8type	TYPE0x08
pd_inmap8func_code	FUNC0x08
pd_inmap8size	SIZE0x08
pd_inmap8string	STRING0x08
pd_inmap9type	TYPE0x09
pd_inmap9func_code	FUNC0x09
pd_inmap9size	SIZE0x09
pd_inmap9string	STRING0x09
pd_inmap10type	TYPE0x0a
pd_inmap10func_code	FUNC0x0a
pd_inmap10size	SIZE0x0a
pd_inmap10string	STRING0x0a
pd_inmap11type	TYPE0x0b
pd_inmap11func_code	FUNC0x0b
pd_inmap11size	SIZE0x0b

Table 3-21 Device Descriptor Data Definition Fields (continued)

Field	Description File Macro
pd_inmap11string	STRING0x0b
pd_inmap12type	TYPE0x0c
pd_inmap12func_code	FUNC0x0c
pd_inmap12size	SIZE0x0c
pd_inmap12string	STRING0x0c
pd_inmap13type	TYPE0x0d
pd_inmap13func_code	FUNC0x0d
pd_inmap13size	SIZE0x0d
pd_inmap13string	STRING0x0d
pd_inmap14type	TYPE0x0e
pd_inmap14func_code	FUNC0x0e
pd_inmap14size	SIZE0x0e
pd_inmap14string	STRING0x0e
pd_inmap15type	TYPE0x0f
pd_inmap15func_code	FUNC0x0f
pd_inmap15size	SIZE0x0f
pd_inmap15string	STRING0x0f

Table 3-21 Device Descriptor Data Definition Fields (continued)

Field	Description File Macro
pd_inmap16type	TYPE0x10
pd_inmap16func_code	FUNC0x10
pd_inmap16size	SIZE0x10
pd_inmap16string	STRING0x10
pd_inmap17type	TYPE0x11
pd_inmap17func_code	FUNC0x11
pd_inmap17size	SIZE0x11
pd_inmap17string	STRING0x11
pd_inmap18type	TYPE0x12
pd_inmap18func_code	FUNC0x12
pd_inmap18size	SIZE0x12
pd_inmap18string	STRING0x12
pd_inmap19type	TYPE0x13
pd_inmap19func_code	FUNC0x13
pd_inmap19size	SIZE0x13
pd_inmap19string	STRING0x13
pd_inmap20type	TYPE0x14

Table 3-21 Device Descriptor Data Definition Fields (continued)

Field	Description File Macro
pd_inmap20func_code	FUNC0x14
pd_inmap20size	SIZE0x14
pd_inmap20string	STRING0x14
pd_inmap21type	TYPE0x15
pd_inmap21func_code	FUNC0x15
pd_inmap21size	SIZE0x15
pd_inmap21string	STRING0x15
pd_inmap22type	TYPE0x16
pd_inma22func_code	FUNC0x16
pd_inmap22size	SIZE0x16
pd_inmap22string	STRING0x16
pd_inmap23type	TYPE0x17
pd_inmap23func_code	FUNC0x17
pd_inmap23size	SIZE0x17
pd_inmap23string	STRING0x17
pd_inmap24type	TYPE0x18
pd_inmap24func_code	FUNC0x18

Table 3-21 Device Descriptor Data Definition Fields (continued)

Field	Description File Macro
pd_inmap24size	SIZE0x18
pd_inmap24string	STRING0x18
pd_inmap25type	TYPE0x19
pd_inmap25func_code	FUNC0x19
pd_inmap25size	SIZE0x19
pd_inmap25string	STRING0x19
pd_inmap26type	TYPE0x1a
pd_inmap26func_code	FUNC0x1a
pd_inmap26size	SIZE0x1a
pd_inmap26string	STRING0x1a
pd_inmap27type	TYPE0x1b
pd_inmap27func_code	FUNC0x1b
pd_inmap27size	SIZE0x1b
pd_inmap27string	STRING0x1b
pd_inmap28type	TYPE0x1c
pd_inmap28func_code	FUNC0x1c
pd_inmap28size	SIZE0x1c

Table 3-21 Device Descriptor Data Definition Fields (continued)

Field	Description File Macro
pd_inmap28string	STRING0x1c
pd_inmap29type	TYPE0x1d
pd_inmap29func_code	FUNC0x1d
pd_inmap29size	SIZE0x1d
pd_inmap29string	STRING0x1d
pd_inmap30type	TYPE0x1e
pd_inmap30func_code	FUNC0x1e
pd_inmap30size	SIZE0x1e
pd_inmap30string	STRING0x1e
pd_inmap31type	TYPE0x1f
pd_inmap31func_code	FUNC0x1f
pd_inmap31size	SIZE0x1f
pd_inmap31string	STRING0x1f
pd_eorch	EORCH
pd_eofch	EOFCH
pd_tabch	TABCH
pd_bellch	BELLCH

Table 3-21 Device Descriptor Data Definition Fields (continued)

Field	Description File Macro
pd_bspch	BSPCH
pd_case	UPC_LOCK
pd_backsp	BSB
pd_delete	LINEDEL
pd_echo	AUTOECHO
pd_alf	AUTOLF
pd_pause	PAGEPAUSE
pd_insm	INSERTMODE
pd_nulls	EOLNULLS
pd_page	PAGESIZE
pd_tabsiz	TABSIZE

pd_inmap0type

TYPE0x7f

EditMod Labels

```
5-SCF path options
1-'\x7f' character mapping type
```

Description

Input mapping type for specified character.

Port Generic Default Value

Macro

```
EDFUNCTION
```

EditMod

```
0x2
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

The input mapping type codes are defined in the header file `scf.h`, and in **Table 3-22**.

Table 3-22 ASCII Control Character Available Values

Control Character is...	Macro	EditMod
removed from the data stream.	`IGNORE`	`0x0`

Table 3-22 ASCII Control Character Available Values (continued)

Control Character is...	Macro	EditMod
passed on without editing.	`PASSTHRU`	`0x1`
removed from the data stream.	`EDFUNCTION`	`0x2`

pd_inmap0func_code

FUNC0x7f

EditMod Labels

```
5-SCF path options
2-'\x7f' editing function code
```

Description

SCF editing function mapping code for specified character.

Port Generic Default Value

Macro

```
DELCHRU
```

EditMod

```
0x07
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

The SCF editing function mapping type codes are defined in the header file `scf.h`, and in **Table 3-23**.

Table 3-23 ASCII Control Character Available Values

Description	Macro	EditMod
Move cursor to the left	`MOVLEFT`	`0x00`
Move cursor to the right	`MOVRIGHT`	`0x01`

Table 3-23 ASCII Control Character Available Values (continued)

Description	Macro	EditMod
Move cursor to the beginning of the line	`MOVBEG`	`0x02`
Move cursor to the end of the line	`MOVEND`	`0x03`
Reprint the current line to cursor position	`REPRINT`	`0x04`
Truncate the line at the cursor position	`TRUNCATE`	`0x05`
Delete character to the left	`DELCHRL`	`0x06`
Delete character under the cursor	`DELCHRU`	`0x07`
Delete word to the left	`DELWRDL`	`0x08`
Delete word to the right	`DELWRDR`	`0x09`
Delete the entire line	`DELINE`	`0x0A`
Undefined (reserved)	`UNDEF1`	`0x0B`
Input mode toggle (type over vs. insert)	`MODETOGL`	`0x0C`
Undefined (reserved)	`UNDEF2`	`0x0D`
End of record (read only)	`ENDOREC`	`0x0E`
End of file	`ENDOFILE`	`0x0F`

pd_inmap0size

SIZE0x7f

EditMod Labels

```
5-SCF path options
3-'\x7f' size of associated string
```

Description

This field specifies the size of the editing function string to echo to the terminal. If this field is specified as `0` (zero), an editing function built into SCF is executed to perform the editing function. If this field is non-zero, the string pointed to by string `0x00` is echoed to the terminal.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

`0` to `4294967295`

pd_inmap0string

STRING0x7f

EditMod Labels

```
5-SCF path options
4-'\x7f' string for key
```

Description

Character string to be echoed to the terminal.

Port Generic Default Value

`NULL`

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

Any ASCII character string. The string may contain C-style character escapes (such as `\n` and `\012`).

pd_inmap1type

TYPE0x01

EditMod Labels

```
5-SCF path options
5-'\x01' character mapping type
```

Description

Input mapping type for specified character.

Port Generic Default Value

Macro

```
EDFUNCTION
```

EditMod

```
0x2
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

The input mapping type codes are defined in the header file `scf.h`, and in **Table 3-22**.

EditMod Labels

```
5-SCF path options
6-'\x01' editing function code
```

Description

SCF editing function mapping code for specified character.

Port Generic Default Value

Macro

```
    MOVEND
```

EditMod

```
    0x07
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

The SCF editing function mapping type codes are defined in the header file `scf.h`, and in **Table 3-23**.

pd_inmap1size

SIZE0x01

EditMod Labels

```
5-SCF path options
7-'\x01' size of associated string
```

Description

This field specifies the size of the editing function string to echo to the terminal. If this field is specified as `0` (zero), an editing function built into SCF is executed to perform the editing function. If this field is non-zero, the string pointed to by string `0x00` is echoed to the terminal.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

`0` to `4294967295`

pd_inmap1string

STRING0x01

EditMod Labels

```
5-SCF path options
8-’\x01’ string for key
```

Description

Character string to be echoed to the terminal.

Port Generic Default Value

`NULL`

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

Any ASCII character string. The string may contain C-style character escapes (such as `\n` and `\012`).

pd_inmap2type

TYPE0x02

EditMod Labels

```
5-SCF path options
9-'\x02' character mapping type
```

Description

Input mapping type for specified character.

Port Generic Default Value

Macro

```
    EDFUNCTION
```

EditMod

```
    0x2
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

The input mapping type codes are defined in the header file `scf.h`, and in **Table 3-22**.

pd_inmap2func_code

FUNC0x02

EditMod Labels

```
5-SCF path options
10-'\x02' editing function code
```

Description

SCF editing function mapping code for specified character.

Port Generic Default Value

Macro

```
MOVLEFT
```

EditMod

```
0x07
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

The SCF editing function mapping type codes are defined in the header file `scf.h`, and in **Table 3-23**.

pd_inmap2size

SIZE0x02

EditMod Labels

```
5-SCF path options
11-'\x02' size of associated string
```

Description

This field specifies the size of the editing function string to echo to the terminal. If this field is specified as `0` (zero), an editing function built into SCF is executed to perform the editing function. If this field is non-zero, the string pointed to by string `0x00` is echoed to the terminal.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

`0` to `4294967295`

pd_inmap2string

STRING0x02

EditMod Labels

```
5-SCF path options
12-’\x02’ string for key
```

Description

Character string to be echoed to the terminal.

Port Generic Default Value

```
NULL
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

Any ASCII character string. The string can contain C-style character escapes (such as `\n` and `\012`).

pd_inmap3type

TYPE0x03

EditMod Labels

```
5-SCF path options
13-'\x03' character mapping type
```

Description

Input mapping type for specified character.

Port Generic Default Value

Macro

```
IGNORE
```

EditMod

```
0x2
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

The input mapping type codes are defined in the header file `scf.h`, and in **Table 3-22**.

pd_inmap3func_code

FUNC0x03

EditMod Labels

```
5-SCF path options
14-'\x03' editing function code
```

Description

SCF editing function mapping code for specified character.

Port Generic Default Value

Macro

```
    0
```

EditMod

```
    0x07
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

The SCF editing function mapping type codes are defined in the header file `scf.h`, and in **Table 3-23**.

pd_inmap3size

SIZE0x03

EditMod Labels

```
5-SCF path options
15-’\x03’ size of associated string
```

Description

This field specifies the size of the editing function string to echo to the terminal. If this field is specified as `0` (zero), an editing function built into SCF is executed to perform the editing function. If this field is non-zero, the string pointed to by string 0x00 is echoed to the terminal.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

`0` to `4294967295`

EditMod Labels

```
5-SCF path options
16-'\x03' string for key
```

Description

Character string to be echoed to the terminal.

Port Generic Default Value

`NULL`

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

Any ASCII character string. The string may contain C-style character escapes (such as `\n` and `\012`).

pd_inmap4type

TYPE0x04

EditMod Labels

```
5-SCF path options
17-’\x04’ character mapping type
```

Description

Input mapping type for specified character.

Port Generic Default Value

Macro

```
EDFUNCTION
```

EditMod

```
0x2
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

The input mapping type codes are defined in the header file `scf.h`, and in **Table 3-22**.

pd_inmap4func_code

FUNC0x04

EditMod Labels

```
5-SCF path options
18-’\x04’ editing function code
```

Description

SCF editing function mapping code for specified character.

Port Generic Default Value

Macro

```
DELCHRU
```

EditMod

```
0x07
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

The SCF editing function mapping type codes are defined in the header file `scf.h`, and in **Table 3-23**.

pd_inmap4size

SIZE0x04

EditMod Labels

```
5-SCF path options
19-'\x04' size of associated string
```

Description

This field specifies the size of the editing function string to echo to the terminal. If this field is specified as `0` (zero), an editing function built into SCF is executed to perform the editing function. If this field is non-zero, the string pointed to by string `0x00` is echoed to the terminal.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

`0` to `4294967295`

pd_inmap4string

STRING0x04

EditMod Labels

```
5-SCF path options
20-'\x04' string for key
```

Description

Character string to be echoed to the terminal.

Port Generic Default Value

```
NULL
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

Any ASCII character string. The string can contain C-style character escapes (such as `\n` and `\012`).

pd_inmap5type

TYPE0x05

EditMod Labels

```
5-SCF path options
21-’\x05’ character mapping type
```

Description

Input mapping type for specified character.

Port Generic Default Value

Macro

```
IGNORE
```

EditMod

```
0x2
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

The input mapping type codes are defined in the header file `scf.h`, and in **Table 3-22**.

pd_inmap5func_code

FUNC0x05

EditMod Labels

```
5-SCF path options
22-'\x05' editing function code
```

Description

SCF editing function mapping code for specified character.

Port Generic Default Value

Macro

```
0
```

EditMod

```
0x07
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

The SCF editing function mapping type codes are defined in the header file `scf.h`, and in **Table 3-23**.

pd_inmap5size

SIZE0x05

EditMod Labels

```
5-SCF path options
23-'\x05' size of associated string
```

Description

This field specifies the size of the editing function string to echo to the terminal. If this field is specified as `0` (zero), an editing function built into SCF is executed to perform the editing function. If this field is non-zero, the string pointed to by string `0x00` is echoed to the terminal.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

`0` to `4294967295`

pd_inmap5string

STRING0x05

EditMod Labels

```
5-SCF path options
24-'\x05' string for key
```

Description

Character string to be echoed to the terminal.

Port Generic Default Value

```
NULL
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

Any ASCII character string. The string can contain C-style character escapes (such as `\n` and `\012`).

pd_inmap6type

TYPE0x06

EditMod Labels

```
5-SCF path options
25-'\x06' character mapping type
```

Description

Input mapping type for specified character.

Port Generic Default Value

Macro

```
EDFUNCTION
```

EditMod

```
0x2
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

The input mapping type codes are defined in the header file `scf.h`, and in **Table 3-22**.

pd_inmap6func_code

FUNC0x06

EditMod Labels

```
5-SCF path options
26-’\x06’ editing function code
```

Description

SCF editing function mapping code for specified character.

Port Generic Default Value

Macro

```
MOVRIGHT
```

EditMod

```
0x07
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

The SCF editing function mapping type codes are defined in the header file `scf.h`, and in **Table 3-23**.

pd_inmap6size

SIZE0x06

EditMod Labels

```
5-SCF path options
27-'\x06' size of associated string
```

Description

This field specifies the size of the editing function string to echo to the terminal. If this field is specified as 0 (zero), an editing function built into SCF is executed to perform the editing function. If this field is non-zero, the string pointed to by string 0x00 is echoed to the terminal.

Port Generic Default Value

0 (zero)

Port Specific Override Value

Refer to SCF/<DEVICE>/DESC/config.des (**Figure 3-3**).

Available Values

0 to 4294967295

pd_inmap6string

STRING0x06

EditMod Labels

```
5-SCF path options
28-'\x06' string for key
```

Description

Character string to be echoed to the terminal.

Port Generic Default Value

`NULL`

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

Any ASCII character string. The string can contain C-style character escapes (such as `\n` and `\012`).

pd_inmap7type

TYPE0x07

EditMod Labels

```
5-SCF path options
29-'\x07' character mapping type
```

Description

Input mapping type for specified character.

Port Generic Default Value

Macro

```
PASSTHRU
```

EditMod

```
0x2
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

The input mapping type codes are defined in the header file `scf.h`, and in **Table 3-22**.

pd_inmap7func_code

FUNC0x07

EditMod Labels

```
5-SCF path options
30-'\x07' editing function code
```

Description

SCF editing function mapping code for specified character.

Port Generic Default Value

Macro

```
    0
```

EditMod

```
    0x07
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

The SCF editing function mapping type codes are defined in the header file `scf.h`, and in **Table 3-23**.

pd_inmap7size

SIZE0x07

EditMod Labels

```
5-SCF path options
31-'\x07' size of associated string
```

Description

This field specifies the size of the editing function string to echo to the terminal. If this field is specified as `0` (zero), an editing function built into SCF is executed to perform the editing function. If this field is non-zero, the string pointed to by string `0x00` is echoed to the terminal.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

`0` to `4294967295`

pd_inmap7string

STRING0x07

EditMod Labels

```
5-SCF path options
32-'\x07' string for key
```

Description

Character string to be echoed to the terminal.

Port Generic Default Value

`NULL`

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

Any ASCII character string. The string can contain C-style character escapes (such as `\n` and `\012`).

pd_inmap8type

TYPE0x08

EditMod Labels

```
5-SCF path options
33-’\x08’ character mapping type
```

Description

Input mapping type for specified character.

Port Generic Default Value

Macro

```
EDFUNCTION
```

EditMod

```
0x2
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

The input mapping type codes are defined in the header file `scf.h`, and in **Table 3-22**.

pd_inmap8func_code

FUNC0x08

EditMod Labels

```
5-SCF path options
34-’\x08’ editing function code
```

Description

SCF editing function mapping code for specified character.

Port Generic Default Value

Macro

```
DELCHRL
```

EditMod

```
0x07
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

The SCF editing function mapping type codes are defined in the header file `scf.h`, and in **Table 3-23**.

pd_inmap8size

SIZE0x08

EditMod Labels

```
5-SCF path options
35-'\x08' size of associated string
```

Description

This field specifies the size of the editing function string to echo to the terminal. If this field is specified as `0` (zero), an editing function built into SCF is executed to perform the editing function. If this field is non-zero, the string pointed to by string `0x00` is echoed to the terminal.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

`0` to `4294967295`

EditMod Labels

```
5-SCF path options
36-'\x08' string for key
```

Description

Character string to be echoed to the terminal.

Port Generic Default Value

`NULL`

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

Any ASCII character string. The string can contain C-style character escapes (such as `\n` and `\012`).

pd_inmap9type

TYPE0x09

EditMod Labels

```
5-SCF path options
37-’\x09’ character mapping type
```

Description

Input mapping type for specified character.

Port Generic Default Value

Macro

```
EDFUNCTION
```

EditMod

```
0x2
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

The input mapping type codes are defined in the header file `scf.h`, and in **Table 3-22**.

pd_inmap9func_code

FUNC0x09

EditMod Labels

```
5-SCF path options
38-'\x09' editing function code
```

Description

SCF editing function mapping code for specified character.

Port Generic Default Value

Macro

```
MODETOGL
```

EditMod

```
0x07
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

The SCF editing function mapping type codes are defined in the header file `scf.h`, and in **Table 3-23**.

pd_inmap9size

SIZE0x09

EditMod Labels

```
5-SCF path options
39-’\x09’ size of associated string
```

Description

This field specifies the size of the editing function string to echo to the terminal. If this field is specified as `0` (zero), an editing function built into SCF is executed to perform the editing function. If this field is non-zero, the string pointed to by string `0x00` is echoed to the terminal.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

`0` to `4294967295`

EditMod Labels

```
5-SCF path options
40-'\x09' string for key
```

Description

Character string to be echoed to the terminal.

Port Generic Default Value

`NULL`

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

Any ASCII character string. The string can contain C-style character escapes (such as `\n` and `\012`).

pd_inmap10type

TYPE0x0a

EditMod Labels

```
5-SCF path options
41-'\x0a' character mapping type
```

Description

Input mapping type for specified character.

Port Generic Default Value

Macro

```
PASSTHRU
```

EditMod

```
0x2
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

The input mapping type codes are defined in the header file `scf.h`, and in **Table 3-22**.

pd_inmap10func_code

FUNC0x0a

EditMod Labels

```
5-SCF path options
42-'\x0a' editing function code
```

Description

SCF editing function mapping code for specified character.

Port Generic Default Value

Macro

```
    0
```

EditMod

```
    0x07
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

The SCF editing function mapping type codes are defined in the header file `scf.h`, and in **Table 3-23**.

pd_inmap10size

SIZE0x0a

EditMod Labels

```
5-SCF path options
43-’\x0a’ size of associated string
```

Description

This field specifies the size of the editing function string to echo to the terminal. If this field is specified as 0 (zero), an editing function built into SCF is executed to perform the editing function. If this field is non-zero, the string pointed to by string 0x00 is echoed to the terminal.

Port Generic Default Value

0 (zero)

Port Specific Override Value

Refer to SCF/<DEVICE>/DESC/config.des (**Figure 3-3**).

Available Values

0 to 4294967295

EditMod Labels

```
5-SCF path options
44-'\x0a' string for key
```

Description

Character string to be echoed to the terminal.

Port Generic Default Value

`NULL`

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

Any ASCII character string. The string can contain C-style character escapes (such as `\n` and `\012`).

pd_inmap11type

TYPE0x0b

EditMod Labels

```
5-SCF path options
45-'\x0b' character mapping type
```

Description

Input mapping type for specified character.

Port Generic Default Value

Macro

```
EDFUNCTION
```

EditMod

```
0x2
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

The input mapping type codes are defined in the header file `scf.h`, and in **Table 3-22**.

pd_inmap11func_code

FUNC0x0b

EditMod Labels

```
5-SCF path options
46-'\x0b' editing function code
```

Description

SCF editing function mapping code for specified character.

Port Generic Default Value

Macro

```
TRUNCATE
```

EditMod

```
0x07
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

The SCF editing function mapping type codes are defined in the header file `scf.h`, and in **Table 3-23**.

pd_inmap11size

SIZE0x0b

EditMod Labels

```
5-SCF path options
47-'\x0b' size of associated string
```

Description

This field specifies the size of the editing function string to echo to the terminal. If this field is specified as 0 (zero), an editing function built into SCF is executed to perform the editing function. If this field is non-zero, the string pointed to by string 0x00 is echoed to the terminal.

Port Generic Default Value

0 (zero)

Port Specific Override Value

Refer to SCF/<DEVICE>/DESC/config.des (**Figure 3-3**).

Available Values

0 to 4294967295

EditMod Labels

```
5-SCF path options
48-’\x0b’ string for key
```

Description

Character string to be echoed to the terminal.

Port Generic Default Value

NULL

Port Specific Override Value

Refer to SCF/<DEVICE>/DESC/config.des (**Figure 3-3**).

Available Values

Any ASCII character string. The string can contain C-style character escapes (such as \n and \012).

pd_inmap12type

TYPE0x0c

EditMod Labels

```
5-SCF path options
49-'\x0c' character mapping type
```

Description

Input mapping type for specified character.

Port Generic Default Value

Macro

```
EDFUNCTION
```

EditMod

```
0x2
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

The input mapping type codes are defined in the header file `scf.h`, and in **Table 3-22**.

pd_inmap12func_code

FUNC0x0c

EditMod Labels

```
5-SCF path options
50-’\x0c’ editing function code
```

Description

SCF editing function mapping code for specified character.

Port Generic Default Value

Macro

```
DELWRDL
```

EditMod

```
0x07
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

The SCF editing function mapping type codes are defined in the header file `scf.h`, and in **Table 3-23**.

pd_inmap12size

SIZE0x0c

EditMod Labels

```
5-SCF path options
51-'\x0c' size of associated string
```

Description

This field specifies the size of the editing function string to echo to the terminal. If this field is specified as `0` (zero), an editing function built into SCF is executed to perform the editing function. If this field is non-zero, the string pointed to by string `0x00` is echoed to the terminal.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

`0` to `4294967295`

pd_inmap12string

STRING0x0c

EditMod Labels

```
5-SCF path options
52-'\x0c' string for key
```

Description

Character string to be echoed to the terminal.

Port Generic Default Value

```
NULL
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

Any ASCII character string. The string can contain C-style character escapes (such as `\n` and `\012`).

pd_inmap13type

TYPE0x0d

EditMod Labels

```
5-SCF path options
53-’\x0d’ character mapping type
```

Description

Input mapping type for specified character.

Port Generic Default Value

Macro

```
EDFUNCTION
```

EditMod

```
0x2
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

The input mapping type codes are defined in the header file `scf.h`, and in **Table 3-22**.

pd_inmap13func_code

FUNC0x0d

EditMod Labels

```
5-SCF path options
54-'\x0d' editing function code
```

Description

SCF editing function mapping code for specified character.

Port Generic Default Value

Macro

```
ENDOREC
```

EditMod

```
0x07
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

The SCF editing function mapping type codes are defined in the header file `scf.h`, and in **Table 3-23**.

pd_inmap13size

SIZE0x0d

EditMod Labels

```
5-SCF path options
55-'\x0d' size of associated string
```

Description

This field specifies the size of the editing function string to echo to the terminal. If this field is specified as `0` (zero), an editing function built into SCF is executed to perform the editing function. If this field is non-zero, the string pointed to by string `0x00` is echoed to the terminal.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

`0` to `4294967295`

pd_inmap13string

STRING0x0d

EditMod Labels

```
5-SCF path options
56-'\x0d' string for key
```

Description

Character string to be echoed to the terminal.

Port Generic Default Value

`NULL`

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

Any ASCII character string. The string can contain C-style character escapes (such as `\n` and `\012`).

pd_inmap14type

TYPE0x0e

EditMod Labels

```
5-SCF path options
57-'\x0e' character mapping type
```

Description

Input mapping type for specified character.

Port Generic Default Value

Macro

```
PASSTHRU
```

EditMod

```
0x2
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

The input mapping type codes are defined in the header file `scf.h`, and in **Table 3-22**.

pd_inmap14func_code

FUNC0x0e

EditMod Labels

```
5-SCF path options
58-'\x0e' editing function code
```

Description

SCF editing function mapping code for specified character.

Port Generic Default Value

Macro

```
    0
```

EditMod

```
    0x07
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

The SCF editing function mapping type codes are defined in the header file `scf.h`, and in **Table 3-23**.

pd_inmap14size

SIZE0x0e

EditMod Labels

```
5-SCF path options
59-’\x0e’ size of associated string
```

Description

This field specifies the size of the editing function string to echo to the terminal. If this field is specified as `0` (zero), an editing function built into SCF is executed to perform the editing function. If this field is non-zero, the string pointed to by string `0x00` is echoed to the terminal.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

`0` to `4294967295`

EditMod Labels

```
5-SCF path options
60-'\x0e' string for key
```

Description

Character string to be echoed to the terminal.

Port Generic Default Value

```
NULL
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

Any ASCII character string. The string can contain C-style character escapes (such as `\n` and `\012`).

pd_inmap15type

TYPE0x0f

EditMod Labels

```
5-SCF path options
61-'\x0f' character mapping type
```

Description

Input mapping type for specified character.

Port Generic Default Value

Macro

```
PASSTHRU
```

EditMod

```
0x2
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

The input mapping type codes are defined in the header file `scf.h`, and in **Table 3-22**.

pd_inmap15func_code

FUNC0x0f

EditMod Labels

```
5-SCF path options
62-’\x0f’ editing function code
```

Description

SCF editing function mapping code for specified character.

Port Generic Default Value

Macro

```
    0
```

EditMod

```
    0x07
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

The SCF editing function mapping type codes are defined in the header file `scf.h`, and in **Table 3-23**.

pd_inmap15size

SIZE0x0f

EditMod Labels

```
5-SCF path options
63-'\x0f' size of associated string
```

Description

This field specifies the size of the editing function string to echo to the terminal. If this field is specified as `0` (zero), an editing function built into SCF is executed to perform the editing function. If this field is non-zero, the string pointed to by string `0x00` is echoed to the terminal.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

`0` to `4294967295`

pd_inmap15string

STRING0x0f

EditMod Labels

```
5-SCF path options
64-’\x0f’ string for key
```

Description

Character string to be echoed to the terminal.

Port Generic Default Value

```
NULL
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

Any ASCII character string. The string can contain C-style character escapes (such as `\n` and `\012`).

pd_inmap16type

TYPE0x10

EditMod Labels

```
5-SCF path options
65-’\x10’ character mapping type
```

Description

Input mapping type for specified character.

Port Generic Default Value

Macro

```
EDFUNCTION
```

EditMod

```
0x2
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

The input mapping type codes are defined in the header file `scf.h`, and in **Table 3-22**.

pd_inmap16func_code

FUNC0x10

EditMod Labels

```
5-SCF path options
66-’\x10’ editing function code
```

Description

SCF editing function mapping code for specified character.

Port Generic Default Value

Macro

```
   REPRINT
```

EditMod

```
   0x07
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

The SCF editing function mapping type codes are defined in the header file `scf.h`, and in **Table 3-23**.

pd_inmap16size

SIZE0x10

EditMod Labels

```
5-SCF path options
67-'\x10' size of associated string
```

Description

This field specifies the size of the editing function string to echo to the terminal. If this field is specified as `0` (zero), an editing function built into SCF is executed to perform the editing function. If this field is non-zero, the string pointed to by string `0x00` is echoed to the terminal.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

`0` to `4294967295`

pd_inmap16string

STRING0x10

EditMod Labels

```
5-SCF path options
68-’\x10’ string for key
```

Description

Character string to be echoed to the terminal.

Port Generic Default Value

```
NULL
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

Any ASCII character string. The string can contain C-style character escapes (such as `\n` and `\012`).

pd_inmap17type

TYPE0x11

EditMod Labels

```
5-SCF path options
69-'\x11' character mapping type
```

Description

Input mapping type for specified character.

Port Generic Default Value

Macro

```
IGNORE
```

EditMod

```
0x2
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

The input mapping type codes are defined in the header file `scf.h`, and in **Table 3-22**.

pd_inmap17func_code

FUNC0x11

EditMod Labels

```
5-SCF path options
70-'\x11' editing function code
```

Description

SCF editing function mapping code for specified character.

Port Generic Default Value

Macro

```
0
```

EditMod

```
0x07
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

The SCF editing function mapping type codes are defined in the header file `scf.h`, and in **Table 3-23**.

pd_inmap17size

SIZE0x11

EditMod Labels

```
5-SCF path options
71-'\x11' size of associated string
```

Description

This field specifies the size of the editing function string to echo to the terminal. If this field is specified as `0` (zero), an editing function built into SCF is executed to perform the editing function. If this field is non-zero, the string pointed to by string `0x00` is echoed to the terminal.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

`0` to `4294967295`

pd_inmap17string

STRING0x11

EditMod Labels

```
5-SCF path options
72-'\x11' string for key
```

Description

Character string to be echoed to the terminal.

Port Generic Default Value

```
NULL
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

Any ASCII character string. The string can contain C-style character escapes (such as `\n` and `\012`).

pd_inmap18type

TYPE0x12

EditMod Labels

```
5-SCF path options
73-'\x12' character mapping type
```

Description

Input mapping type for specified character.

Port Generic Default Value

Macro

```
EDFUNCTION
```

EditMod

```
0x2
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

The input mapping type codes are defined in the header file `scf.h`, and in **Table 3-22**.

pd_inmap18func_code

FUNC0x12

EditMod Labels

```
5-SCF path options
74-'\x12' editing function code
```

Description

SCF editing function mapping code for specified character.

Port Generic Default Value

Macro

```
DELWRDR
```

EditMod

```
0x07
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

The SCF editing function mapping type codes are defined in the header file `scf.h`, and in **Table 3-23**.

pd_inmap18size

SIZE0x12

EditMod Labels

```
5-SCF path options
75-'\x12' size of associated string
```

Description

This field specifies the size of the editing function string to echo to the terminal. If this field is specified as `0` (zero), an editing function built into SCF is executed to perform the editing function. If this field is non-zero, the string pointed to by string `0x00` is echoed to the terminal.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

`0` to `4294967295`

EditMod Labels

```
5-SCF path options
76-'\x12' string for key
```

Description

Character string to be echoed to the terminal.

Port Generic Default Value

`NULL`

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

Any ASCII character string. The string can contain C-style character escapes (such as `\n` and `\012`).

pd_inmap19type

TYPE0x13

EditMod Labels

```
5-SCF path options
77-'\x13' character mapping type
```

Description

Input mapping type for specified character.

Port Generic Default Value

Macro

```
   IGNORE
```

EditMod

```
   0x2
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

The input mapping type codes are defined in the header file `scf.h`, and in **Table 3-22**.

pd_inmap19func_code

FUNC0x13

EditMod Labels

```
5-SCF path options
78-'\x13' editing function code
```

Description

SCF editing function mapping code for specified character.

Port Generic Default Value

Macro

```
0
```

EditMod

```
0x07
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

The SCF editing function mapping type codes are defined in the header file `scf.h`, and in **Table 3-23**.

pd_inmap19size

SIZE0x13

EditMod Labels

```
5-SCF path options
79-'\x13' size of associated string
```

Description

This field specifies the size of the editing function string to echo to the terminal. If this field is specified as `0` (zero), an editing function built into SCF is executed to perform the editing function. If this field is non-zero, the string pointed to by string `0x00` is echoed to the terminal.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

`0` to `4294967295`

pd_inmap19string

STRING0x13

EditMod Labels

```
5-SCF path options
80-'\x13' string for key
```

Description

Character string to be echoed to the terminal.

Port Generic Default Value

`NULL`

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

Any ASCII character string. The string can contain C-style character escapes (such as `\n` and `\012`).

pd_inmap20type

TYPE0x14

EditMod Labels

```
5-SCF path options
81-'\x14' character mapping type
```

Description

Input mapping type for specified character.

Port Generic Default Value

Macro

```
PASSTHRU
```

EditMod

```
0x2
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

The input mapping type codes are defined in the header file `scf.h`, and in **Table 3-22**.

pd_inmap20func_code

FUNC0x14

EditMod Labels

```
5-SCF path options
82-'\x14' editing function code
```

Description

SCF editing function mapping code for specified character.

Port Generic Default Value

Macro

```
0
```

EditMod

```
0x07
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

The SCF editing function mapping type codes are defined in the header file `scf.h`, and in **Table 3-23**.

pd_inmap20size

SIZE0x14

EditMod Labels

```
5-SCF path options
83-'\x14' size of associated string
```

Description

This field specifies the size of the editing function string to echo to the terminal. If this field is specified as 0 (zero), an editing function built into SCF is executed to perform the editing function. If this field is non-zero, the string pointed to by string 0x00 is echoed to the terminal.

Port Generic Default Value

0 (zero)

Port Specific Override Value

Refer to SCF/<DEVICE>/DESC/config.des (**Figure 3-3**).

Available Values

0 to 4294967295

pd_inmap20string

STRING0x14

EditMod Labels

```
5-SCF path options
84-’\x14’ string for key
```

Description

Character string to be echoed to the terminal.

Port Generic Default Value

```
NULL
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

Any ASCII character string. The string can contain C-style character escapes (such as `\n` and `\012`).

pd_inmap21type

TYPE0x15

EditMod Labels

```
5-SCF path options
85-'\x15' character mapping type
```

Description

Input mapping type for specified character.

Port Generic Default Value

Macro

```
PASSTHRU
```

EditMod

```
0x2
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

The input mapping type codes are defined in the header file `scf.h`, and in **Table 3-22**.

pd_inmap21func_code

FUNC0x15

EditMod Labels

```
5-SCF path options
86-'\x15' editing function code
```

Description

SCF editing function mapping code for specified character.

Port Generic Default Value

Macro

```
0
```

EditMod

```
0x07
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

The SCF editing function mapping type codes are defined in the header file `scf.h`, and in **Table 3-23**.

pd_inmap21size

SIZE0x15

EditMod Labels

```
5-SCF path options
87-'\x15' size of associated string
```

Description

This field specifies the size of the editing function string to echo to the terminal. If this field is specified as `0` (zero), an editing function built into SCF is executed to perform the editing function. If this field is non-zero, the string pointed to by string `0x00` is echoed to the terminal.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

`0` to `4294967295`

EditMod Labels

```
5-SCF path options
88-’\x15’ string for key
```

Description

Character string to be echoed to the terminal.

Port Generic Default Value

```
NULL
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

Any ASCII character string. The string can contain C-style character escapes (such as `\n` and `\012`).

pd_inmap22type

TYPE0x16

EditMod Labels

```
5-SCF path options
89-'\x16' character mapping type
```

Description

Input mapping type for specified character.

Port Generic Default Value

Macro

```
PASSTHRU
```

EditMod

```
0x2
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

The input mapping type codes are defined in the header file `scf.h`, and in **Table 3-22**.

EditMod Labels

```
5-SCF path options
90-'\x16' editing function code
```

Description

SCF editing function mapping code for specified character.

Port Generic Default Value

Macro

```
    0
```

EditMod

```
    0x07
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

The SCF editing function mapping type codes are defined in the header file `scf.h`, and in **Table 3-23**.

pd_inmap22size

SIZE0x16

EditMod Labels

```
5-SCF path options
91-’\x16’ size of associated string
```

Description

This field specifies the size of the editing function string to echo to the terminal. If this field is specified as `0` (zero), an editing function built into SCF is executed to perform the editing function. If this field is non-zero, the string pointed to by string `0x00` is echoed to the terminal.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

`0` to `4294967295`

EditMod Labels

```
5-SCF path options
92-'\x16' string for key
```

Description

Character string to be echoed to the terminal.

Port Generic Default Value

`NULL`

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

Any ASCII character string. The string can contain C-style character escapes (such as `\n` and `\012`).

pd_inmap23type

TYPE0x17

EditMod Labels

```
5-SCF path options
93-'\x17' character mapping type
```

Description

Input mapping type for specified character.

Port Generic Default Value

Macro

```
IGNORE
```

EditMod

```
0x2
```

Port Specific Override Value

Refer to SCF/<DEVICE>/DESC/config.des (**Figure 3-3**).

Available Values

The input mapping type codes are defined in the header file scf.h, and in **Table 3-22**.

pd_inmap23func_code

FUNC0x17

EditMod Labels

```
5-SCF path options
94-'\x17' editing function code
```

Description

SCF editing function mapping code for specified character.

Port Generic Default Value

Macro

```
0
```

EditMod

```
0x07
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

The SCF editing function mapping type codes are defined in the header file `scf.h`, and in **Table 3-23**.

pd_inmap23size

SIZE0x17

EditMod Labels

```
5-SCF path options
95-'\x17' size of associated string
```

Description

This field specifies the size of the editing function string to echo to the terminal. If this field is specified as `0` (zero), an editing function built into SCF is executed to perform the editing function. If this field is non-zero, the string pointed to by string `0x00` is echoed to the terminal.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

`0` to `4294967295`

pd_inmap23string

STRING0x17

EditMod Labels

```
5-SCF path options
96-’\x17’ string for key
```

Description

Character string to be echoed to the terminal.

Port Generic Default Value

```
NULL
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

Any ASCII character string. The string can contain C-style character escapes (such as `\n` and `\012`).

pd_inmap24type

TYPE0x18

EditMod Labels

```
5-SCF path options
97-'\x18' character mapping type
```

Description

Input mapping type for specified character.

Port Generic Default Value

Macro

```
EDFUNCTION
```

EditMod

```
0x2
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

The input mapping type codes are defined in the header file `scf.h`, and in **Table 3-22**.

pd_inmap24func_code

FUNC0x18

EditMod Labels

```
5-SCF path options
98-'\x18' editing function code
```

Description

SCF editing function mapping code for specified character.

Port Generic Default Value

Macro

```
DELINE
```

EditMod

```
0x07
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

The SCF editing function mapping type codes are defined in the header file `scf.h`, and in **Table 3-23**.

pd_inmap24size

SIZE0x18

EditMod Labels

```
5-SCF path options
99-'\x18' size of associated string
```

Description

This field specifies the size of the editing function string to echo to the terminal. If this field is specified as `0` (zero), an editing function built into SCF is executed to perform the editing function. If this field is non-zero, the string pointed to by string `0x00` is echoed to the terminal.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

`0` to `4294967295`

EditMod Labels

```
5-SCF path options
100-'\x18' string for key
```

Description

Character string to be echoed to the terminal.

Port Generic Default Value

`NULL`

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

Any ASCII character string. The string can contain C-style character escapes (such as `\n` and `\012`).

pd_inmap25type

TYPE0x19

EditMod Labels

```
5-SCF path options
101-’\x19’ character mapping type
```

Description

Input mapping type for specified character.

Port Generic Default Value

Macro

```
PASSTHRU
```

EditMod

```
0x2
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

The input mapping type codes are defined in the header file `scf.h`, and in **Table 3-22**.

EditMod Labels

```
5-SCF path options
102-'\x19' editing function code
```

Description

SCF editing function mapping code for specified character.

Port Generic Default Value

Macro

```
DELCHRU
```

EditMod

```
0x07
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

The SCF editing function mapping type codes are defined in the header file `scf.h`, and in **Table 3-23**.

pd_inmap25size

SIZE0x19

EditMod Labels

```
5-SCF path options
103-'\x19' size of associated string
```

Description

This field specifies the size of the editing function string to echo to the terminal. If this field is specified as `0` (zero), an editing function built into SCF is executed to perform the editing function. If this field is non-zero, the string pointed to by string `0x00` is echoed to the terminal.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

`0` to `4294967295`

EditMod Labels

```
5-SCF path options
104-'\x19' string for key
```

Description

Character string to be echoed to the terminal.

Port Generic Default Value

```
NULL
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

Any ASCII character string. The string can contain C-style character escapes (such as `\n` and `\012`).

pd_inmap26type

TYPE0x1a

EditMod Labels

```
5-SCF path options
105-’\x1a’ character mapping type
```

Description

Input mapping type for specified character.

Port Generic Default Value

Macro

```
EDFUNCTION
```

EditMod

```
0x2
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

The input mapping type codes are defined in the header file `scf.h`, and in **Table 3-22**.

pd_inmap26func_code

FUNC0x1a

EditMod Labels

```
5-SCF path options
106-'\x1a' editing function code
```

Description

SCF editing function mapping code for specified character.

Port Generic Default Value

Macro

```
MOVBEG
```

EditMod

```
0x07
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

The SCF editing function mapping type codes are defined in the header file `scf.h`, and in **Table 3-23**.

pd_inmap26size

SIZE0x1a

EditMod Labels

```
5-SCF path options
107-'\x1a' size of associated string
```

Description

This field specifies the size of the editing function string to echo to the terminal. If this field is specified as 0 (zero), an editing function built into SCF is executed to perform the editing function. If this field is non-zero, the string pointed to by string 0x00 is echoed to the terminal.

Port Generic Default Value

0 (zero)

Port Specific Override Value

Refer to SCF/<DEVICE>/DESC/config.des (**Figure 3-3**).

Available Values

0 to 4294967295

EditMod Labels

```
5-SCF path options
108-'\x1a' string for key
```

Description

Character string to be echoed to the terminal.

Port Generic Default Value

```
NULL
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

Any ASCII character string. The string can contain C-style character escapes (such as `\n` and `\012`).

pd_inmap27type

TYPE0x1b

EditMod Labels

```
5-SCF path options
109-'\x1b' character mapping type
```

Description

Input mapping type for specified character.

Port Generic Default Value

Macro

```
EDFUNCTION
```

EditMod

```
0x2
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

The input mapping type codes are defined in the header file `scf.h`, and in **Table 3-22**.

pd_inmap27func_code

FUNC0x1b

EditMod Labels

```
5-SCF path options
110-'\x1b' editing function code
```

Description

SCF editing function mapping code for specified character.

Port Generic Default Value

Macro

```
ENDOFILE
```

EditMod

```
0x07
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

The SCF editing function mapping type codes are defined in the header file `scf.h`, and in **Table 3-23**.

pd_inmap27size

SIZE0x1b

EditMod Labels

```
5-SCF path options
111-’\x1b’ size of associated string
```

Description

This field specifies the size of the editing function string to echo to the terminal. If this field is specified as 0 (zero), an editing function built into SCF is executed to perform the editing function. If this field is non-zero, the string pointed to by string 0x00 is echoed to the terminal.

Port Generic Default Value

0 (zero)

Port Specific Override Value

Refer to SCF/<DEVICE>/DESC/config.des (**Figure 3-3**).

Available Values

0 to 4294967295

pd_inmap27string

STRING0x1b

EditMod Labels

```
5-SCF path options
112-’\x1b’ string for key
```

Description

Character string to be echoed to the terminal.

Port Generic Default Value

```
NULL
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

Any ASCII character string. The string can contain C-style character escapes (such as `\n` and `\012`).

pd_inmap28type

TYPE0x1c

EditMod Labels

```
5-SCF path options
113-’\x1c’ character mapping type
```

Description

Input mapping type for specified character.

Port Generic Default Value

Macro

```
PASSTHRU
```

EditMod

```
0x2
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

The input mapping type codes are defined in the header file `scf.h`, and in **Table 3-22**.

pd_inmap28func_code

FUNC0x1c

EditMod Labels

```
5-SCF path options
114-'\x1c' editing function code
```

Description

SCF editing function mapping code for specified character.

Port Generic Default Value

Macro

```
0
```

EditMod

```
0x07
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

The SCF editing function mapping type codes are defined in the header file `scf.h`, and in **Table 3-23**.

pd_inmap28size

SIZE0x1c

EditMod Labels

```
5-SCF path options
115-'\x1c' size of associated string
```

Description

This field specifies the size of the editing function string to echo to the terminal. If this field is specified as `0` (zero), an editing function built into SCF is executed to perform the editing function. If this field is non-zero, the string pointed to by string `0x00` is echoed to the terminal.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

`0` to `4294967295`

pd_inmap28string

STRING0x1c

EditMod Labels

```
5-SCF path options
116-'\x1c' string for key
```

Description

Character string to be echoed to the terminal.

Port Generic Default Value

`NULL`

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

Any ASCII character string. The string can contain C-style character escapes (such as `\n` and `\012`).

pd_inmap29type

TYPE0x1d

EditMod Labels

```
5-SCF path options
117-'\x1d' character mapping type
```

Description

Input mapping type for specified character.

Port Generic Default Value

Macro

```
PASSTHRU
```

EditMod

```
0x2
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

The input mapping type codes are defined in the header file `scf.h`, and in **Table 3-22**.

pd_inmap29func_code

FUNC0x1d

EditMod Labels

```
5-SCF path options
118-'\x1d' editing function code
```

Description

SCF editing function mapping code for specified character.

Port Generic Default Value

Macro

```
0
```

EditMod

```
0x07
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

The SCF editing function mapping type codes are defined in the header file `scf.h`, and in **Table 3-23**.

pd_inmap29size

SIZE0x1d

EditMod Labels

```
5-SCF path options
119-'\x1d' size of associated string
```

Description

This field specifies the size of the editing function string to echo to the terminal. If this field is specified as 0 (zero), an editing function built into SCF is executed to perform the editing function. If this field is non-zero, the string pointed to by string 0x00 is echoed to the terminal.

Port Generic Default Value

0 (zero)

Port Specific Override Value

Refer to SCF/<DEVICE>/DESC/config.des (**Figure 3-3**).

Available Values

0 to 4294967295

pd_inmap29string

STRING0x1d

EditMod Labels

```
5-SCF path options
120-’\x1d’ string for key
```

Description

Character string to be echoed to the terminal.

Port Generic Default Value

```
NULL
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

Any ASCII character string. The string can contain C-style character escapes (such as `\n` and `\012`).

pd_inmap30type

TYPE0x1e

EditMod Labels

```
5-SCF path options
121-'\x1e' character mapping type
```

Description

Input mapping type for specified character.

Port Generic Default Value

Macro

```
PASSTHRU
```

EditMod

```
0x2
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

The input mapping type codes are defined in the header file `scf.h`, and in **Table 3-22**.

EditMod Labels

```
5-SCF path options
122-’\x1e’ editing function code
```

Description

SCF editing function mapping code for specified character.

Port Generic Default Value

Macro

```
    0
```

EditMod

```
    0x07
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

The SCF editing function mapping type codes are defined in the header file `scf.h`, and in **Table 3-23**.

pd_inmap30size

SIZE0x1e

EditMod Labels

```
5-SCF path options
123-'\x1e' size of associated string
```

Description

This field specifies the size of the editing function string to echo to the terminal. If this field is specified as 0 (zero), an editing function built into SCF is executed to perform the editing function. If this field is non-zero, the string pointed to by string 0x00 is echoed to the terminal.

Port Generic Default Value

0 (zero)

Port Specific Override Value

Refer to SCF/<DEVICE>/DESC/config.des (**Figure 3-3**).

Available Values

0 to 4294967295

pd_inmap30string

STRING0x1e

EditMod Labels

```
5-SCF path options
124-'\x1e' string for key
```

Description

Character string to be echoed to the terminal.

Port Generic Default Value

`NULL`

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

Any ASCII character string. The string can contain C-style character escapes (such as `\n` and `\012`).

pd_inmap31type

TYPE0x1f

EditMod Labels

```
5-SCF path options
125-'\x1f' character mapping type
```

Description

Input mapping type for specified character.

Port Generic Default Value

Macro

```
PASSTHRU
```

EditMod

```
0x2
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

The input mapping type codes are defined in the header file `scf.h`, and in **Table 3-22**.

pd_inmap31func_code

FUNC0x1f

EditMod Labels

```
5-SCF path options
126-'\x1f' editing function code
```

Description

SCF editing function mapping code for specified character.

Port Generic Default Value

Macro

```
    0
```

EditMod

```
    0x07
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

The SCF editing function mapping type codes are defined in the header file `scf.h`, and in **Table 3-23**.

pd_inmap31size

SIZE0x1f

EditMod Labels

```
5-SCF path options
127-'\x1f' size of associated string
```

Description

This field specifies the size of the editing function string to echo to the terminal. If this field is specified as 0 (zero), an editing function built into SCF is executed to perform the editing function. If this field is non-zero, the string pointed to by string 0x00 is echoed to the terminal.

Port Generic Default Value

0 (zero)

Port Specific Override Value

Refer to SCF/<DEVICE>/DESC/config.des (**Figure 3-3**).

Available Values

0 to 4294967295

pd_inmap31string

STRING0x1f

EditMod Labels

```
5-SCF path options
128-'\x1f' string for key
```

Description

Character string to be echoed to the terminal.

Port Generic Default Value

```
NULL
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

Any ASCII character string. The string can contain C-style character escapes (such as `\n` and `\012`).

pd_eorch

EORCH

EditMod Labels

```
5-SCF path options
129-end of record character (read only)
```

Description

This specifies the end of record character.

Port Generic Default Value

Macro

EORCH (defined as C_CR in scfdesc.h)

EditMod

```
'\n'
```

Port Specific Override Value

Refer to SCF/<DEVICE>/DESC/config.des (Figure 3-3).

Available Values

The ASCII control and special characters are defined in the header file, scf.h, and in Table 3-24.

Table 3-24 ASCII Control Character Available Values

SCF/OS-9 Compatible Standard Codes	Macro	EditMod
	C_NULL	0x00
C_REPEAT	CTRL_A	0x01

Table 3-24 ASCII Control Character Available Values (continued)

SCF/OS-9 Compatible Standard Codes	Macro	EditMod
	CTRL_B	0x02
C_INTR	CTRL_C	0x03
C_REPRINT	CTRL_D	0x04
C_QUIT	CTRL_E	0x05
	CTRL_F	0x06
C_BELL	CTRL_G	0x07
C_BACKSPACE	CTRL_H	0x08
C_TAB	CTRL_I	0x09
C_LINEFEED	CTRL_J	0x0A
	CTRL_K	0x0B
C_FORMFEED	CTRL_L	0x0C
C_CR	CTRL_M	0x0D
	CTRL_N	0x0E
	CTRL_O	0x0F
	CTRL_P	0x10
C_XON	CTRL_Q	0x11
	CTRL_R	0x12

Table 3-24 ASCII Control Character Available Values (continued)

SCF/OS-9 Compatible Standard Codes	Macro	EditMod
C_XOFF	CTRL_S	0x13
	CTRL_T	0x14
	CTRL_U	0x15
	CTRL_V	0x16
C_PAUSE	CTRL_W	0x17
C_DELLINE	CTRL_X	0x18
	CTRL_Y	0x19
	CTRL_Z	0x1A
	CTRL_SPACE	0x20
	CTRL_COMMA	0x2C
	CTRL_PERIOD	0x2E
	CTRL_SLASH	0x2F
C_EOF		0x1B

EditMod Labels

```
5-SCF path options
130-end of file character
```

Description

This specifies the end of file character.

Port Generic Default Value

Macro

`EOFCH` (defined as `C_EOF` in `scfdesc.h`)

EditMod

```
0x1B
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

The ASCII control and special characters are defined in the header file, `scf.h`, and in **Table 3-24**.

pd_tabch

TABCH

EditMod Labels

```
5-SCF path options
131-tab character (0 = none)
```

Description

This defines the tab character.

Port Generic Default Value

Macro

`TABCH` (defined as `C_TAB` in `scfdesc.h`)

EditMod

`0x09`

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

The ASCII control and special characters are defined in the header file, `scf.h`, and in **Table 3-24**.

EditMod Labels

```
5-SCF path options
132-bell (line overflow)
```

Description

This defines the bell character.

Port Generic Default Value

Macro

`BELLCH` (defined as `C_BELL` in `scfdesc.h`)

EditMod

`0x07`

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

The ASCII control and special characters are defined in the header file, `scf.h`, and in **Table 3-24**.

pd_bspch

BSPCH

EditMod Labels

```
5-SCF path options
133-backspace echo character
```

Description

This defines the backspace echo character.

Port Generic Default Value

Macro

`BSPCH` (defined as `C_BACKSPACE` in `scfdesc.h`)

EditMod

```
'\b'
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

The ASCII control and special characters are defined in the header file, `scf.h`, and in **Table 3-24**.

EditMod Labels

```
5-SCF path options
134-case lock
```

Description

This specifies the state of the upper case lock character.

Port Generic Default Value

The default is upper and lower case.

Macro

`UPC_LOCK` (defined as `PLOFF` in `scfdesc.h`)

EditMod

```
0x0
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

The SCF character logic states are defined in the header file, `scf.h`, and in **Table 3-25**.

Table 3-25 `pd_case` Logic Stage Available Values

Description	Macro	EditMod
Positive logic off - Upper and lower case	`PLOFF`	`0x00`
Positive logic on	`PLON`	`0x01`
Negative logic off	`NLOFF`	`0x00`
Negative logic on	`NLON`	`0x01`

EditMod Labels

```
5-SCF path options
135-backspace
```

Description

This specifies the state of the backspace character.

Port Generic Default Value

The default is destructive backspace.

Macro

`BSB` (defined as `PLON` in `scfdesc.h`)

EditMod

```
0x01
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

The SCF character logic states are defined in the header file, `scf.h`, and in **Table 3-26**.

Table 3-26 `pd_backsp` Logic Stage Available Values

Description	Macro	EditMod
Positive logic off	`PLOFF`	`0x00`
Positive logic on - Destructive backspace	`PLON`	`0x01`
Negative logic off	`NLOFF`	`0x00`
Negative logic on	`NLON`	`0x01`

EditMod Labels

```
5-SCF path options
136-delete line
```

Description

This specifies the state of the delete line character.

Port Generic Default Value

The default is destructive line delete.

Macro

`LINDEL` (defined as `PLON` in `scfdesc.h`)

EditMod

```
0x01
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

The SCF character logic states are defined in the header file, `scf.h`, and in **Table 3-27**.

Table 3-27 `pd_delete` Logic Stage Available Values

Description	Macro	EditMod
Positive logic off - Nondestructive line delete	`PLOFF`	`0x00`
Positive logic on - Destructive line delete	`PLON`	`0x01`
Negative logic off	`NLOFF`	`0x00`
Negative logic on	`NLON`	`0x01`

EditMod Labels

```
5-SCF path options
137-echo
```

Description

This specifies whether the character echo is on or off.

Port Generic Default Value

The default is echo on.

Macro

`AUTOECHO` (defined as `PLON` in `scfdesc.h`)

EditMod

`0x01`

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

The SCF character logic states are defined in the header file, `scf.h`, and in **Table 3-28**.

Table 3-28 `pd_echo` Logic Stage Available Values

Description	Macro	EditMod
Positive logic off - Echo off	`PLOFF`	`0x00`
Positive logic on - Echo on	`PLON`	`0x01`
Negative logic off	`NLOFF`	`0x00`
Negative logic on	`NLON`	`0x01`

EditMod Labels

```
5-SCF path options
138-auto-linefeed
```

Description

This specifies whether the auto line feed is on or off.

Port Generic Default Value

The default is auto linefeed on.

Macro

`AUTOLF` (defined as `PLON` in `scfdesc.h`)

EditMod

```
0x01
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

The SCF character logic states are defined in the header file, `scf.h`, and in **Table 3-29**.

Table 3-29 `pd_alf` Logic Stage Available Values

Description	Macro	EditMod
Positive logic off - Auto linefeed off	`PLOFF`	`0x00`
Positive logic on - Auto linefeed on	`PLON`	`0x01`
Negative logic off	`NLOFF`	`0x00`
Negative logic on	`NLON`	`0x01`

pd_pause

PAGEPAUSE

EditMod Labels

```
5-SCF path options
139-end-of-page pause
```

Description

This specifies whether the page pause is on or off.

Port Generic Default Value

The default is page pause on.

Macro

`PAGEPAUSE` (defined as `PLON` in `scfdesc.h`)

EditMod

`0x01`

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

The SCF character logic states are defined in the header file, `scf.h`, and in **Table 3-30**.

Table 3-30 pd_pause Logic Stage Available Values

Description	Macro	EditMod
Positive logic off - Auto linefeed off	PLOFF	0x00
Positive logic on - Auto linefeed on	PLON	0x01
Negative logic off	NLOFF	0x00
Negative logic on	NLON	0x01

EditMod Labels

```
5-SCF path options
140-insert mode
```

Description

This specifies whether the insert mode is on or off.

Port Generic Default Value

The default is insert mode off.

Macro

`INSERTMODE` (defined as `PLOFF` in `scfdesc.h`)

EditMod

```
0x00
```

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

The SCF character logic states are defined in the header file, `scf.h`, and in **Table 3-31**.

Table 3-31 `pd_insmLogic` Stage Available Values

Description	Macro	EditMod
Positive logic off - Insert mode off	`PLOFF`	`0x00`
Positive logic on - Insert mode on	`PLON`	`0x01`
Negative logic off	`NLOFF`	`0x00`
Negative logic on	`NLON`	`0x01`

EditMod Labels

```
5-SCF path options
141-end of line null count
```

Description

This specifies the number of end of line nulls.

Port Generic Default Value

`0` (no end of line nulls)

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

`-128` to `127`

pd_page

PAGESIZE

EditMod Labels

```
5-SCF path options
142-lines per page
```

Description

This specifies the number of lines per page.

Port Generic Default Value

`24`

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

`-128` to `127`

EditMod Labels

```
5-SCF path options
143-tab field size
```

Description

This specifies the number of spaces a tab skips.

Port Generic Default Value

`4`

Port Specific Override Value

Refer to `SCF/<DEVICE>/DESC/config.des` (**Figure 3-3**).

Available Values

`-128` to `127`

Chapter 4: SBF Device Descriptors

SBF device descriptors contain configuration data specific to one OS-9 format disk device on an OS-9 system. Values which can be configured in the descriptor include:

- Device interrupt vector and priority
- Device I/O address
- Device geometry
- Logical sector size

The next section in this chapter provides a detailed example of the configuration options you can use to change configuration values for SBF (sequential block file).

The rest of this chapter provides a detailed list of all of the SBF device descriptor fields.

This chapter includes the following topics:

SBF Field Configuration Options

To change an SBF device descriptor module configuration field, you can use either of the following methods:

1. Use the `EditMod` utility to directly modify existing SBF device descriptor modules either as a stand-alone module or as part of a merged module group (such as a boot image).
2. Modify the description file for the SBF device descriptor module and rebuild it using the makefile provided.

Direct Modification Advantages

The direct modification method has the following advantages:

Fast	No source configuration file rebuilds are necessary.
Temporary	The original module or merged-module group configuration can be easily restored through the appropriate rebuild.
Contained	Changes are limited to the individual boot image modified (merged-module option).

Description File/Rebuild Advantages

The advantage of the description file/rebuild method is that the changes are permanent and reproducible. Modifications apply to all subsequent module rebuilds and to all merged-module groups built containing the updated module.

Both methods are documented in this section. These procedures are used with the field descriptions starting with the **Module Header Fields**. For direct modification, use the `EditMod` LABELS data to navigate the

`EditMod` menus. The DESCRIPTION FILE MACRO data identifies the macro you need to define/modify in the configuration sources to rebuild the SBF device descriptor module.

Direct Modification

Use the `Editmod` utility and the following procedures to directly modify fields in the existing SBF device descriptor module. The module can stand-alone or it can be part of a merged-module group. A boot image, for example, contains multiple modules. Both situations are covered in this section. The field references later in this chapter contain a description of each configurable field, its supported values, and the sequence of menu options required by `EditMod` to modify that field.

For More Information

Refer to the ***Utilities Reference*** for a full description of `EditMod`'s capabilities.

Figure 4-1 Directory Location for Modifying SBF Device Descriptors

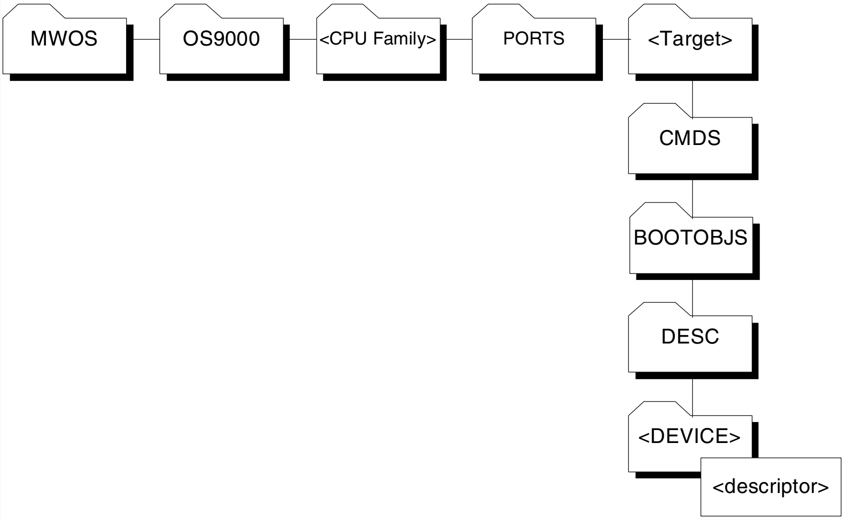
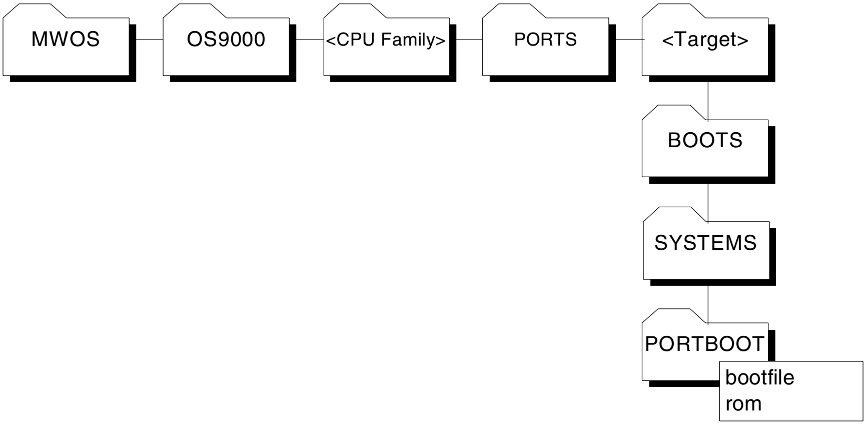


Figure 4-2 Directory Location for Modifying Low-Level Boot Images



For More Information

Refer to your board guide for information about how to modify the module lists and remake the boot images, and for specified boot image names.

Direct Modification Procedures

To modify the stand-alone module, complete the following steps:

Step 1. Change to the `CMDS/BOOTOBJS/DESC/<DEVICE>` directory (see **Figure 4-1**).

Step 2. Use EditMod to edit the module:

```
$EditMod -e <descriptor>
```

To modify the module as part of a merged module group, complete the following steps:

Step 1. Change to the `BOOTS/SYSTEMS/PORTBOOT` directory (see **Figure 4-2**).

Step 2. Use EditMod to edit the module:

```
$EditMod -e <descriptor> -f=<boot image name>
```

Step 3. Use the menu selections provided in the `EditMod` LABELS section of the field reference later in this chapter to locate the fields you want to edit.

Step 4. Select a new value for the field from the AVAILABLE VALUES section of the field reference. Enter that value at the EditMod prompt to modify the field.

Step 5. If you want to make additional modifications, use the `p` command (previous) to step backward through the EditMod menus. Repeat Steps 3 and 4 until you have made all desired modifications to the descriptor.

Step 6. Select the `w` command (write) to save the changes.

Step 7. Select the `q` command (quit) to exit `EditMod`.

Note

Unless you modified the SBF device descriptors in your boot image, you should rebuild your boot image to include the new descriptor.

Example EditMod Session

This example modifies an SBF device descriptor as part of the boot image `rom`:

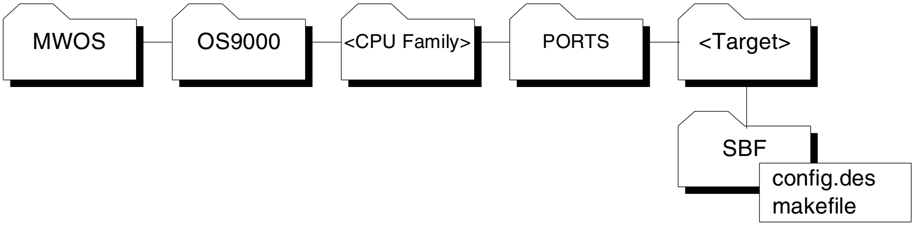
```
$ EditMod -e mt0 -f=rom

 1. module header
 2. device descriptor data definitions
 3. SBF path options structure
 4. SBF logical unit status

Which? [?/1-4/p/t/a/w/q] 4

 1. irq vector                               : 0x4b
 2. irq level                                : 0x4
 3. irq priority                             : 0xa
 4. drive flag                               : 0x0

Which? [?/1-4/p/t/a/w/q] 3

irq priority                                 : 0xa
New value: 1

 1. irq vector                               : 0x4b
 2. irq level                                : 0x4
 3. irq priority                             : 0xa
 4. drive flag                               : 0x0
```

```
Which? [?/1-19/p/t/a/w/q] w

Which? [?/1-19/p/t/a/w/q] q
```

Description File Configuration

You can use these procedures to modify the appropriate description file and rebuild the SBF device descriptors for your port directory. The DESCRIPTION FILE MACROS section of the field reference specifies the name of the macro you modify/define in the description files to configure the field. The value used in the define is chosen from the AVAILABLE VALUES specified for the field.

Figure 4-3 Directory Location for Modifying SBF Description Files



Description File Configuration Procedures

Step 1. Change to the `SBF/<DEVICE>` directory (see **Figure 4-3**).

Step 2. Edit the file `config.des` and read the included comments for more information on how to use the specific description files provided in your software distribution. The `config.des` file contains a list of macro names that can be defined to override the global default values for the configuration fields.

Step 3. Refer to the DESCRIPTION FILE MACRO section in the field reference later in this chapter to determine the macro name you define to configure the target field.

Step 4. Read the comments in `config.des` to determine where to place the define for this macro.

Step 5. Select the value you want to use to configure the field. See the AVAILABLE VALUES section of the field reference data for values or macros that can be used for the definition. Define the macro by entering a definition in the appropriate description files as follows:

```
#define <macro> <value>
```

Step 6. Save the changes and rebuild the SBF device descriptors, entering the following command in the `SBF/<DEVICE>/DESC` directory:

```
os9make
```

Step 7. Rebuild your boot image to include the new descriptor.

SBF Device Descriptor Field Reference

This section contains a list of the most commonly configured fields in the SBF device descriptors. Each field entry contains the following information:

- <Field name> - The call name for each field that can be reconfigured in the module.
- EditMod LABELS - `EditMod` menu selections for navigating to the proper field in and `EditMod` session.
- DESCRIPTION FILE MACRO - The macro name you modify/define in the description file.
- DESCRIPTION - A brief description of the field's purpose and use.
- EXAMPLE - An optional example of the description file entry showing how to change the value of this field.
- PORT GENERIC DEFAULT VALUE - The value set in the port generic description file for this field. This is the value the field is assigned when the module is built, unless the appropriate macro has been defined in the port specific description file to override this default value.
- PORT SPECIFIC OVERRIDE VALUE - The value set in the port specific description file for this field. If defined, this is the value the field is assigned when the module is built, overriding the port generic default value.
- AVAILABLE VALUES - Values to which the field can be set through `EditMod` or the description files. In many cases, this data is presented in a table that maps a description of the value to a numeric value appropriate for entry in EditMod, and to a pre-defined macro available for use in the description file.

Module Header Fields

The following section contains the module header fields in the order they appear during an interactive EditMod session. Defined fields may appear in a different order in config.des.

Table 4-1 Module Header Fields

Field	Description File Macro
_m_group	MH_GROUP
_m_user	MH_USER
mod_name	MH_NAME
m_access	MH_ACCESS
m_tylan	MH_TYLAN
m_attrev	MH_ATTREV
m_edit	MH_EDITION

_m_group

MH_GROUP

EditMod Labels

```
1-module header
1-module owner's group number
```

Description

Group ID of the module's owner. The group number allows people working in the same department or on the same project to share a common identification number.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `SBF/<DEVICE>/DESC/config.des` (**Figure 4-3**).

Available Values

`0` to `65535`

_m_user

MH_USER

EditMod Labels

`1-module header`
`2-module owner’s user number`

Description

User ID of the module’s owner. The user number identifies a specific user.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `SBF/<DEVICE>/DESC/config.des` (**Figure 4-3**).

Available Values

`0` to `65535`

mod_name

MH_NAME

EditMod Labels

```
1-module header
3-module name
```

Description

Contains the module name string.

Port Generic Default Value

String value (None)

Port Specific Override Value

Refer to `SBF/<DEVICE>/DESC/config.des` (**Figure 4-3**).

Available Values

Any ASCII character string. The string may contain C-style character escapes (such as `\n` and `\012`).

EditMod Labels

```
1-module header
4-access permissions
```

Description

Defines the permissible module access by its owner or by other users.

Port Generic Default Value

Macro

```
MP_OWNER_READ | MP_OWNER_EXEC | MP_GROUP_READ |
MP_GROUP_EXEC | MP_WORLD_READ | MP_WORLD_EXEC
```

EditMod

```
0x555
```

Port Specific Override Value

Refer to `SBF/<DEVICE>/DESC/config.des` (**Figure 4-3**).

Available Values

Module access permission values are located in the header file, `module.h`, and are listed in **Table 4-2**.

Table 4-2 `m_access` Available Values

Description	Macro	EditMod
Read permission by owner	`MP_OWNER_READ`	`0x0001`
Write permission by owner	`MP_OWNER_WRITE`	`0x0002`

Table 4-2 m_access Available Values (continued)

Description	Macro	EditMod
Execute permission by owner	MP_OWNER_EXEC	0x0004
Owner permission mask	MP_OWNER_MASK	0x000f
Read permission by group	MP_GROUP_READ	0x0010
Write permission by group	MP_GROUP_WRITE	0x0020
Execute permission by group	MP_GROUP_EXEC	0x0040
Group permission mask	MP_GROUP_MASK	0x00f0
Read permission by world	MP_WORLD_READ	0x0100
Write permission by world	MP_WORLD_WRITE	0x0200
Execute permission by world	MP_WORLD_EXEC	0x0400
World permission mask	MP_WORLD_MASK	0x0f00
All permissions for owner, group, and world	MP_WORLD_ACCESS	0x0777
System permission mask	MP_SYSTM_MASK	0xf000

EditMod Labels

```
1-module header
5-type/language
```

Description

Contains the module’s type (first byte) and language (second byte). The language codes indicate if the module is executable and which language the run-time system requires for execution, if any.

Port Generic Default Value

Macro

```
(MT_DATA<<8) + ML_OBJECT
```

EditMod

```
0x401
```

Port Specific Override Value

Refer to `SBF/<DEVICE>/DESC/config.des` (**Figure 4-3**).

Available Values

Module type values and language codes are located in the header file, `module.h`, and are listed in **Table 4-3** and **Table 4-4**.

Table 4-3 `m_tylan` Available Module Type Values

Description	Macro	EditMod
Not used (wildcard value in system calls)	`MT_ANY`	`0x0000`
Program module	`MT_PROGRAM`	`0x0001`
Subroutine module	`MT_SUBROUT`	`0x0002`
Multi-module (reserved for future use)	`MT_MULTI`	`0x0003`
Data module	`MT_DATA`	`0x0004`
Configuration data block data module	`MT_CDBDATA`	`0x0005`
Reserved for future use	`0xb-0xa`	`0xb-0xa`
User trap library	`MT_TRAPLIB`	`0x000b`
System module	`MT_SYSTEM`	`0x000c`
File manager module	`MT_FILEMAN`	`0x000d`
Physical device driver	`MT_DEVDRVR`	`0x000e`
Device descriptor module	`MT_DEVDESC`	`0x000f`

Table 4-3 `m_tylan` Available Module Type Values (continued)

Description	Macro	EditMod
User definable	`0x10-0xfe`	`0x10-0xfe`
Module type mask	`MT_MASK`	`0xff00`

Table 4-4 `m_tylan` Available Language Code Values

Description	Macro	EditMod
Unspecified language (wildcard in system calls)	`ML_ANY`	`0x0`
Machine language	`ML_OBJECT`	`0x1`
Basic I-code (reserved for future use)	`ML_ICODE`	`0x2`
Pascal P-code (reserved for future use)	`ML_PCODE`	`0x3`
C I-code (reserved for future use)	`ML_CCODE`	`0x4`
Cobol I-code (reserved for future use)	`ML_CBLCODE`	`0x5`
Fortran	`ML_FRTNCODE`	`0x6`
Reserved for future use	`0x7-0xf`	`0x7-0xf`
User-definable	`0x10-0xfe`	`0x10-0xfe`
Module language mask	`ML_MASK`	`0x00ff`

m_attrev

MH_ATTREV

EditMod Labels

```
1-module header
6-revision/attributes
```

Description

Contains the module's attributes (first byte) and revision (second byte).

Port Generic Default Value

Macro

```
MA_REENT<<8
```

EditMod

```
0x8000
```

Port Specific Override Value

Refer to `SBF/<DEVICE>/DESC/config.des` (**Figure 4-3**).

Available Values

Module attribute and revision codes are located in the header file `module.h.`, and are listed in **Table 4-5**.

Note

If two modules with the same name are found in the memory search or are loaded into the current module directory, only the module with the highest revision level is kept. This enables easy substitution of modules for update or correction.

Table 4-5 m_attrev Available Attribute and Revision Values

Description	Macro	EditMod
The module is re-entrant (sharable by multiple tasks).	MA_REENT (shifted left to first byte: MA_REENT<<8)	0x80 (shifted left to first byte: 0x8000)
The module is sticky. A sticky module is not removed from memory until its link count becomes -1 or memory is required for another use.	MA_GHOST (shifted left to first byte: MA_GHOST<<8)	0x40 (shifted left to first byte: 0x4000)
The module is a system-state module.	MA_SUPER (shifted left to first byte: MA_SUPER<<8)	0x20 (shifted left to first byte: 0x2000)
User-definable revision number	0x0-0xfe	0x0-0xfe
Module attribute mask	MA_MASK	0xff00
Module revision mask	MR_MASK	0x00ff

m_edit

MH_EDITION

EditMod Labels

```
1-module header
7-edition
```

Description

Indicates the software release level for maintenance. OS-9 does not use this field. Whenever a program is revised (even for a small change), increase this number. We recommend internal documentation within the source program be keyed to this system.

Port Generic Default Value

`1`

Port Specific Override Value

Refer to `SBF/<DEVICE>/DESC/config.des` (**Figure 4-3**).

Available Values

`0` to `65535`

Device Descriptor Data Definition Fields

The following section contains the device descriptor data definition fields in the order they appear during an interactive `EditMod` session. Defined fields can appear in a different order in `config.des`.

Table 4-6 Device Descriptor Data Definition Fields

Field	Description File Macro
`dd_port`	`PORTADDR`
`dd_lun`	`LUN`
`dd_pd_size`	`PD_SIZE`
`dd_type`	`DD_TYPE`
`dd_mode`	`DD_MODE`
`dd_port`	`MFGR_NAME`
`drvr_name`	`DRVR_NAME`
`dd_class`	`DD_CLASS`

dd_port

PORTADDR

EditMod Labels

```
2-device descriptor data definitions
1-device port address
```

Description

Absolute physical address of the hardware controller. This is the address of the device on the bus. This is the lowest address the device has mapped. Port address is hardware dependent.

Macro Example

```
#define PORTADDR    0xfffe4000
```

Port Generic Default Value

0 (zero)

Port Specific Override Value

Refer to SBF/<DEVICE>/DESC/config.des (**Figure 4-3**).

Available Values

0 to 4294967295

EditMod Labels

```
2-device descriptor data definitions
2-logical unit number
```

Description

Distinguishes the different devices driven from a unique controller. Each unique number represents a different logical unit static storage area.

Macro Example

```
#define LUN    2
```

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `SBF/<DEVICE>/DESC/config.des` (**Figure 4-3**).

Available Values

`0` to `65535`

dd_pd_size

PD_SIZE

EditMod Labels

```
2-device descriptor data definitions
3-path descriptor size
```

Description

Size of the path descriptor. IOMAN uses this value when it allocates a path descriptor.

Port Generic Default Value

`124`

Port Specific Override Value

Refer to `SBF/<DEVICE>/DESC/config.des` (**Figure 4-3**).

Available Values

`0` to `65535`

EditMod Labels

`2-device descriptor data definitions`
`4-device type`

Description

Identifies the I/O class of the device.

Port Generic Default Value

Macro

```
DT_SBF
```

EditMod

```
0x3
```

Port Specific Override Value

Refer to `SBF/<DEVICE>/DESC/config.des` (**Figure 4-3**).

Available Values

Device type values are defined in the header file `io.h`, and are listed in **Table 4-7**.

Table 4-7 `dd_type` **Available Values**

Description	Macro	EditMod
Sequential Character File Type	`DT_SCF`	`0x0`
Random Block File Type	`DT_RBF`	`0x1`
Pipe File Type	`DT_PIPE`	`0x2`

Table 4-7 dd_type Available Values (continued)

Description	Macro	EditMod
Sequential Block File Type	DT_SBF	0x3
Network File Type	DT_NFM	0x4
Compact Disc File Type	DT_CDFM	0x5
User Communication Manager	DT_UCM	0x6
Socket Communication Manager	DT_SOCK	0x7
Pseudo-Keyboard Manager	DT_PTTY	0x8
Graphics File Manager	DT_GFM	0x9
PC-DOS File Manager	DT_PCF	0xa
Non-volatile RAM File Manager	DT_NRF	0xb
ISDN File Manager	DT_ISDN	0xc
MPFM File Manager	DT_MPFM	0xd
Real-Time Network File Manager	DT_RTNFM	0xe
Serial Protocol File Manager	DT_SPF	0xf
Inet File Manager	DT_INET	0xa0
Reserved for Microware Use Only	17-127	0xa1-0x7f

EditMod Labels

```
2-device descriptor data definitions
5-device mode capabilities
```

Description

Used to check the validity of a caller's access mode byte in `I_CREATE` or `I_OPEN` system calls. If a bit is set, the device can perform the corresponding function. The `S_ISIZE` bit is usually set, because it is handled by the file manager or ignored. If the `S_ISHARE` bit is set, the device is non-sharable. A printer is an example of a non-sharable device.

Port Generic Default Value

Macro

```
S_IPRM
```

EditMod

```
0xFFFF
```

Port Specific Override Value

Refer to `SBF/<DEVICE>/DESC/config.des` (**Figure 4-3**).

Available Values

The file access modes are defined in the header file, modes.h, and located in Table 4-8. The file access permission values are defined in the header file modes.h and in Table 4-9.

Table 4-8 dd_mode Available Values for File Access Modes

Description	Macro	EditMod
Truncate on open	S_ITRUNC	0x0100
Ensure contiguous file	S_ICONTIG	0x0400
Error if file exists on create	S_IEXCL	0x0400
Create file	S_ICREAT	0x0800
Append to file	S_IAPPEND	0x1000
Non-sharable	S_ISHARE	0x4000

Table 4-9 dd_mode Available Values for File Access Permissions

Description	Macro	EditMod
Mask for permission bits	S_IPRM	0xffff
Owner read	S_IREAD	0x0001
Owner write	S_IWRITE	0x0002
Owner execute	S_IEXEC	0x0004
Search permission	S_ISEARCH	0x0004

Table 4-9 dd_mode Available Values for File Access Permissions

Description	Macro	EditMod
Group read	S_IGREAD	0x0010
Group write	S_IGWRITE	0x0020
Group execute	S_IGEXEC	0x0040
Group search	S_IGSEARCH	0x0040
Public read	S_IOREAD	0x0100
Public write	S_IOWRITE	0x0200
Public execute	S_IOEXEC	0x0400
Public search	S_IOSEARCH	0x0400

fmgr_name

FMGR_NAME

EditMod Labels

```
2-device descriptor data definitions
6-file manager name
```

Description

Contains the name string of the file manager module to use.

Port Generic Default Value

`“sbf”`

Port Specific Override Value

Refer to `SBF/<DEVICE>/DESC/config.des` (**Figure 4-3**).

Available Values

Any ASCII character string. The string can contain C-style character escapes (such as `\n` and `\012`).

drvr_name

DRVR_NAME

EditMod Labels

```
2-device descriptor data definitions
7-driver name
```

Description

Contains the name string of the device driver module to use.

Port Generic Default Value

`NULL`

Port Specific Override Value

Refer to `SBF/<DEVICE>/DESC/config.des` (**Figure 4-3**).

Available Values

Any ASCII character string. The string may contain C-style character escapes (such as `\n` and `\012`).

dd_class

DD_CLASS

EditMod Labels

```
1-module header
2-device descriptor data definitions
8-device class (sequential or random)
```

Description

Used to identify the class of the device, whether it is random or sequential access.

Port Generic Default Value

Macro

```
DC_SEQ
```

EditMod

```
0x1
```

Port Specific Override Value

Refer to `SBF/<DEVICE>/DESC/config.des` (**Figure 4-3**).

Available Values

Device class available values are defined in the header file, `io.h`, and in **Table 4-10**.

Table 4-10 `dd_class` Available Values

Description	Macro	EditMod
Sequential access device	`DC_SEQ`	`0x0001`
Random access device	`DC_RND`	`0x0002`

SBF Path Options Fields

The following section contains the SBF path options fields in the order they appear during an interactive EditMod session. Defined fields can appear in a different order in config.des.

Table 4-11 SBF Path Options Fields

Field	Description File Macro
pd_blksiz	BLKSIZE
pd_flags	FLAGS
pd_dmamode	DMAMODE
pd_sci_id	SCSIID
pd_scsilun	SCSILUN

pd_blksiz

BLKSIZE

EditMod Labels

```
3-SBF path options structure
1-size of blocks allocated
```

Description

Logical block size in bytes.

Port Generic Default Value

`512`

Port Specific Override Value

Refer to `SBF/<DEVICE>/DESC/config.des` (**Figure 4-3**).

Available Values

`0` to `4294967295`

EditMod Labels

```
3-SBF path options structure
2-SBF/driver compatability flags
```

Description

SBF driver compatibility flags.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `SBF/<DEVICE>/DESC/config.des` (**Figure 4-3**).

Available Values

Compatibility flag values are defined in the header file `sbf.h`, and in **Table 4-12**.

Table 4-12 SBF Compatibility Flag

Description	Macro	EditMod
Rewind tape on close	`DEV_REWIND_FLG`	`0x0001`
Erase to end after writing	`DEV_ERASE_FLG`	`0x0002`
Take drive off-line on close	`DEV_OFFLINE_FLG`	`0x0004`
Device can skip backwards	`DEV_SKIPBACK_FLG`	`0x0008`

pd_dmamode

DMAMODE

EditMod Labels

```
3-SBF path options structure
3-DMA type/usage
```

Description

DMA mode to be used by the driver.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `SBF/<DEVICE>/DESC/config.des` (**Figure 4-3**).

Available Values

`0` to `65535`

EditMod Labels

```
3-SBF path options structure
4-SCSI controller ID
```

Description

SCSI ID of the device’s controller.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `SBF/<DEVICE>/DESC/config.des` (**Figure 4-3**).

Available Values

`0` to `255`

pd_scsilun

SCSILUN

EditMod Labels

```
3-SBF path options structure
5-SCSI controller drive LUN
```

Description

Logical Unit Number of the tape device.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `SBF/<DEVICE>/DESC/config.des` (**Figure 4-3**).

Available Values

`0` to `255`

SBF Logical Unit Status Fields

The following section contains the SBF logical unit status fields in the order they appear during an interactive `EditMod` session. Defined fields can appear in a different order in `config.des`.

Table 4-13 SBF Logical Unit Static Storage Fields

Field	Description File Macro
`sbf_vector`	`VECTOR`
`sbf_irqlevel`	`IRQLEVEL`
`sbf_priority`	`PRIORITY`
`sbf_dflag`	`DRIVE_FLAG`

sbf_vector

VECTOR

EditMod Labels

```
4-SBF logical unit status
1-irq vector
```

Description

This is the vector number of the device interrupt.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `SBF/<DEVICE>/DESC/config.des` (**Figure 4-3**).

Available Values

`0` to `255`

sbf_irqlevel

IRQLEVEL

EditMod Labels

```
4-SBF logical unit status
2-irq level
```

Description

This is the hardware priority of the device interrupt.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `SBF/<DEVICE>/DESC/config.des` (**Figure 4-3**).

Available Values

`0` to `255`

sbf_priority

PRIORITY

EditMod Labels

```
4-SBF logical unit status
3-irq priority
```

Description

This is the software (polling) priority of the device interrupt.

Port Generic Default Value

```
5
```

Port Specific Override Value

Refer to `SBF/<DEVICE>/DESC/config.des` (**Figure 4-3**).

Available Values

`0` to `255`

sbf_dflag

DRIVE_FLAG

EditMod Labels

```
4-SBF logical unit status
4-drive flag
```

Description

Current state of SBF device.

Port Generic Default Value

0 (zero)

Port Specific Override Value

Refer to SBF/<DEVICE>/DESC/config.des (**Figure 4-3**).

Available Values

Drive flag values are defined in the header file sbf.h, and in **Table 4-14**.

Table 4-14 SBF Drive Flag

Description	Macro	EditMod
Read is in progress on device	DFLG_READFLAG	0x0001
Write is in progress on device	DFLG_WRITEFLAG	0x0002
Driver is using the device	DFLG_DRIVEBUSY	0x0004
Drive is at EOF	DFLG_EOFFLAG	0x0008

Chapter 5: RBF Device Descriptors

RBF device descriptors contain configuration data specific to one OS-9 format disk device on an OS-9 system. Values that can be configured in the descriptor include:

- Device interrupt vector and priority
- Device I/O address
- Device geometry
- Logical sector size

The next section in this chapter provides a detailed example of the configuration options you can use to change configuration values for RBF (random block file) devices.

The rest of this chapter provides a detailed list of all of the RBF device descriptor fields.

This chapter includes the following topics:

RBF Field Configuration Options

To change an RBF device descriptor module configuration field, you can use either of the following methods:

1. Use the `EditMod` utility to directly modify existing RBF device descriptor modules either as a stand-alone module or as part of a merged module group (such as a boot image).
2. Modify the description file for the RBF device descriptor module and rebuild it using the makefile provided.

Direct Modification Advantages

The direct modification method has the following advantages:

Fast	No source configuration file rebuilds are necessary.
Temporary	The original module or merged-module group configuration can be easily restored through the appropriate rebuild.
Contained	Changes are limited to the individual boot image modified (merged-module option).

Description File/Rebuild Advantages

The advantage of the description file/rebuild method is that the changes are permanent and reproducible. Modifications apply to all subsequent module rebuilds and to all merged-module groups built containing the updated module.

Both methods are documented in this section. These procedures are used with the field descriptions starting with the **Module Header Fields**. For direct modification, use the `EditMod` LABELS data to navigate through the

EditMod menus. The DESCRIPTION FILE MACRO data identifies the macro you need to define/modify in the configuration sources to rebuild the RBF device descriptor module.

Direct Modification

Use the Editmod utility and the following procedures to directly modify fields in the existing RBF device descriptor module. The module can stand-alone or it can be part of a merged-module group. A boot image, for example, contains multiple modules. Both situations are covered in this section. The field references later in this chapter contain a description of each configurable field, its supported values, and the sequence of menu options required by EditMod to modify that field.

For More Information

Refer to the ***Utilities Reference*** for a full description of `EditMod`'s capabilities.

Figure 5-1 Directory Location for Modifying RBF Device Descriptors

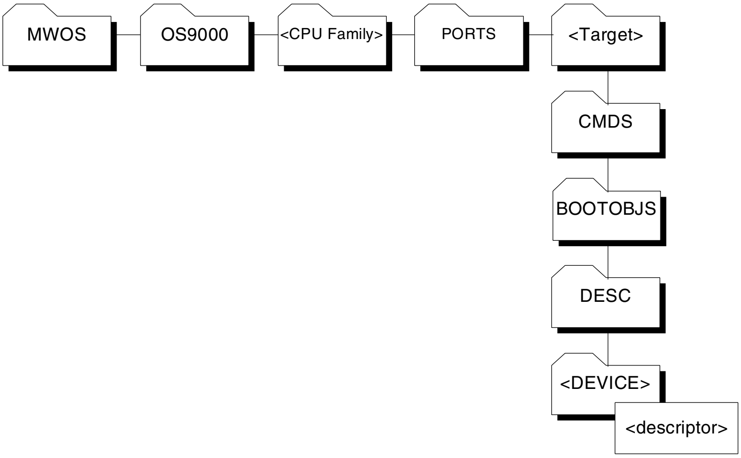
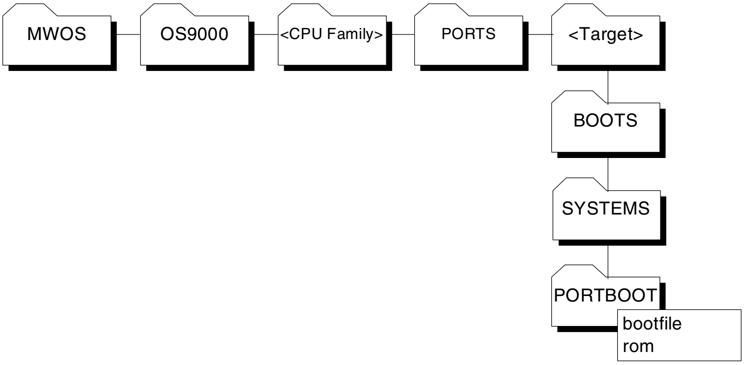


Figure 5-2 Directory Location for Modifying Low-Level Boot Images



For More Information

Refer to your board guide for information about how to modify the module lists and remake the boot images, and for specified boot image names.

Direct Modification Procedures

To modify the stand-alone module, complete the following steps:

Step 1. Change to the `CMDS/BOOTOBJS/DESC/<DEVICE>` directory (see **Figure 5-1**).

Step 2. Use EditMod to edit the module:

```
$EditMod -e <descriptor>
```

To modify the module as part of a merged module group, complete the following steps:

Step 1. Change to the `BOOTS/SYSTEMS/PORTBOOT` directory (see **Figure 5-2**).

Step 2. Use EditMod to edit the module:

```
$EditMod -e <descriptor> -f=<boot image name>
```

Step 3. Use the menu selections provided in the `EditMod` LABELS section of the field reference later in this chapter to locate the fields you want to edit.

Step 4. Select a new value for the field from the AVAILABLE VALUES section of the field reference. Enter that value at the EditMod prompt to modify the field.

Step 5. If you want to make additional modifications, use the `p` command (previous) to step backward through the EditMod menus. Repeat Steps 3 and 4 until you have made all desired modifications to the descriptor.

Step 6. Select the `w` command (write) to save the changes.

Step 7. Select the `q` command (quit) to exit `EditMod`.

Note

Unless you modified the RBF device descriptors in your boot image, you should rebuild your boot image to include the new descriptor.

Example EditMod Session

This example modifies an RBF device descriptor as part of the boot image `rom`:

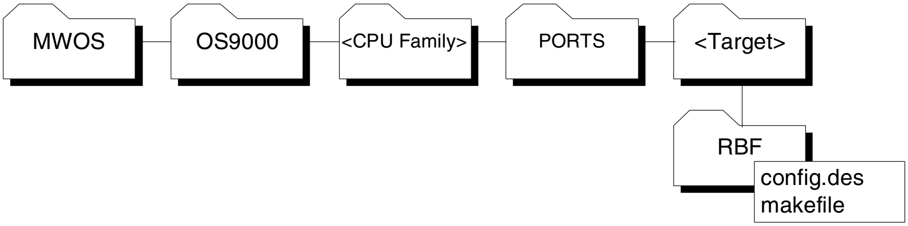
```
$ EditMod -e r0 -f=rom

 1. module header
 2. device descriptor data definitions
 3. RBF path options
 4. RBF logical unit static storage

Which? [?/1-4/p/t/a/w/q] 4

 1. interrupt vector                        : 0x0
 2. interrupt level                         : 0
 3. interrupt priority                      : 5
 4. RBF logical unit options

Which? [?/1-4/p/t/a/w/q] 3

interrupt priority                      : 5
New value: 1

 1. interrupt vector                        : 0x0
 2. interrupt level                         : 0
 3. interrupt priority                      : 1
 4. RBF logical unit options
```

```
Which? [?/1-19/p/t/a/w/q] w

Which? [?/1-19/p/t/a/w/q] q
```

Description File Configuration

You can use these procedures to modify the appropriate description file and rebuild the RBF device descriptors for your port directory. The DESCRIPTION FILE MACROS section of the field reference specifies the name of the macro you modify/define in the description files to configure the field. The value used in the define is chosen from the AVAILABLE VALUES specified for the field.

Figure 5-3 Directory Location for Modifying RBF Description Files



Description File Configuration Procedures

Step 1. Change to the `RBF/<DEVICE>` directory (see **Figure 5-3**).

Step 2. Edit the file `config.des` and read the included comments for more information on how to use the specific description files provided in your software distribution. The `config.des` file contains a list of macro names that can be defined to override the global default values for the configuration fields.

Step 3. Refer to the DESCRIPTION FILE MACRO section in the field reference later in this chapter to determine the macro name you define to configure the target field.

Step 4. Read the comments in `config.des` to determine where to place the define for this macro.

Step 5. Select the value you want to use to configure the field. See the AVAILABLE VALUES section of the field reference data for values or macros that can be used for the definition. Define the macro by entering a definition in the appropriate description files as follows:

```
#define <macro> <value>
```

Step 6. Save the changes and rebuild the RBF device descriptors, entering the following command in the `RBF/<DEVICE>/DESC` directory:

```
os9make
```

Step 7. Rebuild your boot image to include the new descriptor.

RBF Device Descriptor Field Reference

This section contains a list of the most commonly configured fields in the RBF device descriptors. Each field entry contains the following information:

- <Field name> - The call name for each field that can be reconfigured in the module.
- EditMod LABELS - `EditMod` menu selections for navigating to the proper field in an `EditMod` session.
- DESCRIPTION FILE MACRO - The macro name you modify/define in the description file.
- DESCRIPTION - A brief description of the field's purpose and use.
- EXAMPLE - An optional example of the description file entry showing how to change the value of this field.
- PORT GENERIC DEFAULT VALUE - The value set in the port generic description file for this field. This is the value the field is assigned when the module is built, unless the appropriate macro has been defined in the port specific description file to override this default value.
- PORT SPECIFIC OVERRIDE VALUE - The value set in the port specific description file for this field. If defined, this is the value the field is assigned when the module is built, overriding the port generic default value.
- AVAILABLE VALUES - Values to which the field can be set through `EditMod` or the description files. In many cases, this data is presented in a table that maps a description of the value to a numeric value appropriate for entry in EditMod, and to a pre-defined macro available for use in the description file.

Module Header Fields

The following section contains the module header fields in the order they appear during an interactive EditMod session. Defined fields can appear in a different order in config.des.

Table 5-1 Module Header Fields

Field	Description File Macro
_m_group	MH_GROUP
_m_user	MH_USER
mod_name	MH_NAME
m_access	MH_ACCESS
m_tylan	MH_TYLAN
m_attrev	MH_ATTREV
m_edit	MH_EDITION

_m_group

MH_GROUP

EditMod Labels

```
1-module header
1-module owner's group number
```

Description

Group ID of the module's owner. The group number allows people working in the same department or on the same project to share a common identification number.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `RBF/<DEVICE>/DESC/config.des` (**Figure 5-3**).

Available Values

`0` to `65535`

EditMod Labels

`1-module header`
`2-module owner’s user number`

Description

User ID of the module’s owner. The user number identifies a specific user.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `RBF/<DEVICE>/DESC/config.des` (**Figure 5-3**).

Available Values

`0` to `65535`

mod_name

MH_NAME

EditMod Labels

```
1-module header
3-module name
```

Description

Contains the module name string.

Port Generic Default Value

String value (None)

Port Specific Override Value

Refer to `RBF/<DEVICE>/DESC/config.des` (**Figure 5-3**).

Available Values

Any ASCII character string. The string can contain C-style character escapes (such as `\n` and `\012`).

EditMod Labels

```
1-module header
4-access permissions
```

Description

Defines the permissible module access by its owner or by other users.

Port Generic Default Value

Macro

```
MP_OWNER_READ | MP_OWNER_EXEC | MP_GROUP_READ |
MP_GROUP_EXEC | MP_WORLD_READ | MP_WORLD_EXEC
```

EditMod

```
0x555
```

Port Specific Override Value

Refer to `RBF/<DEVICE>/DESC/config.des` (**Figure 5-3**).

Available Values

Module access permission values are located in the header file, `module.h`, and are listed in **Table 5-2**.

Table 5-2 `m_access` Available Values

Description	Macro	EditMod
Read permission by owner	`MP_OWNER_READ`	`0x0001`
Write permission by owner	`MP_OWNER_WRITE`	`0x0002`

Table 5-2 m_access Available Values (continued)

Description	Macro	EditMod
Execute permission by owner	MP_OWNER_EXEC	0x0004
Owner permission mask	MP_OWNER_MASK	0x000f
Read permission by group	MP_GROUP_READ	0x0010
Write permission by group	MP_GROUP_WRITE	0x0020
Execute permission by group	MP_GROUP_EXEC	0x0040
Group permission mask	MP_GROUP_MASK	0x00f0
Read permission by world	MP_WORLD_READ	0x0100
Write permission by world	MP_WORLD_WRITE	0x0200
Execute permission by world	MP_WORLD_EXEC	0x0400
World permission mask	MP_WORLD_MASK	0x0f00
All permissions for owner, group, and world	MP_WORLD_ACCESS	0x0777
System permission mask	MP_SYSTM_MASK	0xf000

EditMod Labels

```
1-module header
5-type/language
```

Description

Contains the module's type (first byte) and language (second byte). The language codes indicate if the module is executable and which language the run-time system requires for execution, if any.

Port Generic Default Value

Macro

```
(MT_DATA<<8) + ML_OBJECT
```

EditMod

```
0x401
```

Port Specific Override Value

Refer to `RBF/<DEVICE>/DESC/config.des` (**Figure 5-3**).

Available Values

Module type values and language codes are located in the header file, `module.h`, and are listed in **Table 5-3** and **Table 5-4**.

Table 5-3 `m_tylan` Available Module Type Values

Description	Macro	EditMod
Not used (wildcard value in system calls)	`MT_ANY`	`0x0000`
Program module	`MT_PROGRAM`	`0x0001`
Subroutine module	`MT_SUBROUT`	`0x0002`
Multi-module (reserved for future use)	`MT_MULTI`	`0x0003`
Data module	`MT_DATA`	`0x0004`
Configuration data block data module	`MT_CDBDATA`	`0x0005`
Reserved for future use	`0xb-0xa`	`0xb-0xa`
User trap library	`MT_TRAPLIB`	`0x000b`
System module	`MT_SYSTEM`	`0x000c`
File manager module	`MT_FILEMAN`	`0x000d`
Physical device driver	`MT_DEVDRVR`	`0x000e`
Device descriptor module	`MT_DEVDESC`	`0x000f`

Table 5-3 m_tylan Available Module Type Values (continued)

Description	Macro	EditMod
User definable	0x10-0xfe	0x10-0xfe
Module type mask	MT_MASK	0xff00

Table 5-4 m_tylan Available Language Code Values

Description	Macro	EditMod
Unspecified language (wildcard in system calls)	ML_ANY	0x0
Machine language	ML_OBJECT	0x1
Basic I-code (reserved for future use)	ML_ICODE	0x2
Pascal P-code (reserved for future use)	ML_PCODE	0x3
C I-code (reserved for future use)	ML_CCODE	0x4
Cobol I-code (reserved for future use)	ML_CBLCODE	0x5
Fortran	ML_FRTNCODE	0x6
Reserved for future use	0x7-0xf	0x7-0xf
User-definable	0x10-0xfe	0x10-0xfe
Module language mask	ML_MASK	0x00ff

m_attrev

MH_ATTREV

EditMod Labels

```
1-module header
6-revision/attributes
```

Description

Contains the module's attributes (first byte) and revision (second byte).

Port Generic Default Value

Macro

```
MA_REENT<<8
```

EditMod

```
0x8000
```

Port Specific Override Value

Refer to `RBF/<DEVICE>/DESC/config.des` (**Figure 5-3**).

Available Values

Module attribute and revision codes are located in the header file `module.h.`, and are listed in **Table 5-5**.

Note

If two modules with the same name are found in the memory search or are loaded into the current module directory, only the module with the highest revision level is kept. This enables easy substitution of modules for update or correction.

Table 5-5 m_attrev Available Attribute and Revision Values

Description	Macro	EditMod
The module is re-entrant (sharable by multiple tasks).	MA_REENT (shifted left to first byte: MA_REENT<<8)	0x80 (shifted left to first byte: 0x8000)
The module is sticky. A sticky module is not removed from memory until its link count becomes -1 or memory is required for another use.	MA_GHOST (shifted left to first byte: MA_GHOST<<8)	0x40 (shifted left to first byte: 0x4000)
The module is a system-state module.	MA_SUPER (shifted left to first byte: MA_SUPER<<8)	0x20 (shifted left to first byte: 0x2000)
User-definable revision number	0x0-0xfe	0x0-0xfe
Module attribute mask	MA_MASK	0xff00
Module revision mask	MR_MASK	0x00ff

m_edit

MH_EDITION

EditMod Labels

```
1-module header
7-edition
```

Description

Indicates the software release level for maintenance. OS-9 does not use this field. Whenever a program is revised (even for a small change), increase this number. We recommend internal documentation within the source program be keyed to this system.

Port Generic Default Value

`1`

Port Specific Override Value

Refer to `RBF/<DEVICE>/DESC/config.des` (**Figure 5-3**).

Available Values

`0` to `65535`

Device Descriptor Data Definition Fields

The following section contains the device descriptor data definition fields in the order they appear during an interactive EditMod session. Defined fields can appear in a different order in config.des.

Table 5-6 Device Descriptor Data Definition Fields

Field	Description File Macro
dd_port	PORTADDR
dd_lun	LUN
dd_pd_size	PD_SIZE
dd_type	DD_TYPE
dd_mode	DD_MODE
dd_port	MFGR_NAME
drvr_name	DRVR_NAME
dd_class	DD_CLASS

dd_port

PORTADDR

EditMod Labels

```
2-device descriptor data definitions
1-device port address
```

Description

Absolute physical address of the hardware controller. This is the address of the device on the bus. This is the lowest address the device has mapped. Port address is hardware dependent.

Macro Example

```
#define PORTADDR    0xfffe4000
```

Port Generic Default Value

0 (zero)

Port Specific Override Value

Refer to RBF/<DEVICE>/DESC/config.des (**Figure 5-3**).

Available Values

0 to 4294967295

dd_lun

LUN

EditMod Labels

```
2-device descriptor data definitions
2-logical unit number
```

Description

Distinguishes between the different devices driven from a unique controller. Each unique number represents a different logical unit static storage area.

Macro Example

```
#define LUN    2
```

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `RBF/<DEVICE>/DESC/config.des` (**Figure 5-3**).

Available Values

`0` to `65535`

dd_pd_size

PD_SIZE

EditMod Labels

```
2-device descriptor data definitions
3-path descriptor size
```

Description

Size of the path descriptor. IOMAN uses this value when it allocates a path descriptor.

Port Generic Default Value

`360`

Port Specific Override Value

Refer to `RBF/<DEVICE>/DESC/config.des` (**Figure 5-3**).

Available Values

`0` to `65535`

EditMod Labels

```
2-device descriptor data definitions
4-device type
```

Description

Identifies the I/O class of the device.

Port Generic Default Value

Macro

```
DT_RBF
```

EditMod

```
0x1
```

Port Specific Override Value

Refer to `RBF/<DEVICE>/DESC/config.des` (**Figure 5-3**).

Available Values

Device type values are defined in the header file `io.h`, and are listed in **Table 5-7**.

Table 5-7 `dd_type` **Available Values**

Description	Macro	EditMod
Sequential Character File Type	`DT_SCF`	`0x0`
Random Block File Type	`DT_RBF`	`0x1`
Pipe File Type	`DT_PIPE`	`0x2`

Table 5-7 `dd_type` Available Values (continued)

Description	Macro	EditMod
Sequential Block File Type	DT_SBF	0x3
Network File Type	DT_NFM	0x4
Compact Disc File Type	DT_CDFM	0x5
User Communication Manager	DT_UCM	0x6
Socket Communication Manager	DT_SOCK	0x7
Pseudo-Keyboard Manager	DT_PTTY	0x8
Graphics File Manager	DT_GFM	0x9
PC-DOS File Manager	DT_PCF	0xa
Non-volatile RAM File Manager	DT_NRF	0xb
ISDN File Manager	DT_ISDN	0xc
MPFM File Manager	DT_MPFM	0xd
Real-Time Network File Manager	DT_RTNFM	0xe
Serial Protocol File Manager	DT_SPF	0xf
Inet File Manager	DT_INET	0xa0
Reserved for Microware Use Only	17-127	0xa1-0x7f

EditMod Labels

```
2-device descriptor data definitions
5-device mode capabilities
```

Description

Used to check the validity of a caller's access mode byte in I_CREATE or I_OPEN system calls. If a bit is set, the device can perform the corresponding function. The S_ISIZE bit is usually set, because it is handled by the file manager or ignored. If the S_ISHARE bit is set, the device is non-sharable. A printer is an example of a non-sharable device.

Port Generic Default Value

Macro

```
S_IPRM
```

EditMod

```
0xFFFF
```

Port Specific Override Value

Refer to RBF/<DEVICE>/DESC/config.des (**Figure 5-3**).

Available Values

The file access modes are defined in the header file, modes.h, and located in **Table 5-8**. The file access permission values are defined in the header file modes.h and in **Table 5-9**.

Table 5-8 dd_mode Available Values for File Access Modes

Description	Macro	EditMod
Truncate on open	S_ITRUNC	0x0100
Ensure contiguous file	S_ICONTIG	0x0400
Error if file exists on create	S_IEXCL	0x0400
Create file	S_ICREAT	0x0800
Append to file	S_IAPPEND	0x1000
Non-sharable	S_ISHARE	0x4000

Table 5-9 dd_mode Available Values for File Access Permissions

Description	Macro	EditMod
Mask for permission bits	S_IPRM	0xffff
Owner read	S_IREAD	0x0001
Owner write	S_IWRITE	0x0002
Owner execute	S_IEXEC	0x0004
Search permission	S_ISEARCH	0x0004

Table 5-9 dd_mode Available Values for File Access Permissions

Description	Macro	EditMod
Group read	S_IGREAD	0x0010
Group write	S_IGWRITE	0x0020
Group execute	S_IGEXEC	0x0040
Group search	S_IGSEARCH	0x0040
Public read	S_IOREAD	0x0100
Public write	S_IOWRITE	0x0200
Public execute	S_IOEXEC	0x0400
Public search	S_IOSEARCH	0x0400

fmgr_name

FMGR_NAME

EditMod Labels

```
2-device descriptor data definitions
6-file manager name
```

Description

Contains the name string of the file manager module to use.

Port Generic Default Value

```
“rbf”
```

Port Specific Override Value

Refer to `RBF/<DEVICE>/DESC/config.des` (**Figure 5-3**).

Available Values

Any ASCII character string. The string can contain C-style character escapes (such as `\n` and `\012`).

drvr_name

DRVR_NAME

EditMod Labels

```
2-device descriptor data definitions
7-driver name
```

Description

Contains the name string of the device driver module to use.

Port Generic Default Value

`NULL`

Port Specific Override Value

Refer to `RBF/<DEVICE>/DESC/config.des` (**Figure 5-3**).

Available Values

Any ASCII character string. The string may contain C-style character escapes (such as `\n` and `\012`).

dd_class

DD_CLASS

EditMod Labels

```
1-module header
2-device descriptor data definitions
8-device class (sequential or random)
```

Description

Used to identify the class of the device, whether it is random or sequential access.

Port Generic Default Value

Macro

```
DC_RND
```

EditMod

```
0x2
```

Port Specific Override Value

Refer to `RBF/<DEVICE>/DESC/config.des` (**Figure 5-3**).

Available Values

Device class available values are defined in the header file, `io.h`, and in **Table 5-10**.

Table 5-10 `dd_class` Available Values

Description	Macro	EditMod
Sequential access device	`DC_SEQ`	`0x0001`
Random access device	`DC_RND`	`0x0002`

RBF Path Option Fields

The following section contains the RBF path option fields in the order they appear during an interactive EditMod session. Defined fields can appear in a different order in config.des.

Table 5-11 RBF Path Option Fields

Field	Description File Macro
pd_sid	SIDES
pd_vfy	VERIFY
pd_format	FORMAT
pd_cyl	CYLNDRS
pd_blk	BLKSTRK
pd_t0b	BLKSTRK0
pd_sas	SEGSIZE
pd_ilv	INTRLV
pd_toffs	TRKOFFS
pd_boffs	BLKOFFS
pd_trys	TRYS
pd_bsize	BLKSIZE
pd_cntl	CONTROL

Table 5-11 RBF Path Option Fields (continued)

Field	Description File Macro
pd_wpc	PRECOMP
pd_rwr	REDWRITE
pd_park	PARK
pd_lsnoffs	LSNOFFS
pd_xfersize	XFERSIZE

EditMod Labels

```
3-RBF path options
1-number of surfaces
```

Description

Indicates the number of surfaces (heads or sides) for a disk unit.

Port Generic Default Value

```
2
```

Port Specific Override Value

Refer to `RBF/<DEVICE>/DESC/config.des` (**Figure 5-3**).

Available Values

`-2147483648` to `2147483647`

pd_vfy

VERIFY

EditMod Labels

```
3-RBF path options
2-verify disk writes (0=verify)
```

Description

Indicates whether a write is verified by a re-read and compare. Write verify operations are generally performed on floppy disks but not hard disks because of the lower soft error rate of hard disks.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `RBF/<DEVICE>/DESC/config.des` (**Figure 5-3**).

Available Values

Device verify values are defined in the header file, `rbf.h`, and in **Table 5-12**.

Table 5-12 `pd_vfy` Available Values

Description	Macro	EditMod
Verify disk write	`0`	`0x0`
No verification	`1`	`0x01`

EditMod Labels

```
3-RBF path options
3-device format
```

Description

Indicates whether a write is verified by a re-read and compare. Write verify operations are generally performed on floppy disks but not hard disks because of the lower soft error rate of hard disks.

Port Generic Default Value

Macro

```
FMT_STDFMT + FMT_DBLBITDNS + FMT_DBLTRKDNS + FMT_DBLSIDE
```

EditMod

```
0x200e
```

Port Specific Override Value

Refer to `RBF/<DEVICE>/DESC/config.des` (**Figure 5-3**).

Available Values

Device format values are defined in the header file, `rbf.h`, and in **Table 5-13**.

Table 5-13 `pd_format` **Available Values**

Description	Macro	EditMod
Track 0 is double density.	`FMT_DBLTRK0`	`0x0001`
Device is double bit density.	`FMT_DBLBITDNS`	`0x0002`

Table 5-13 pd_format Available Values (continued)

Description	Macro	EditMod
Device is double track density.	FMT_DBLTRKDNS	0x0004
Device is double sided.	FMT_DBLSIDE	0x0008
Drive is eight inch.	FMT_EIGHTINCH	0x0010
Drive is five inch.	FMT_FIVEINCH	0x0020
Drive is three inch.	FMT_THREEINCH	0x0040
Device is high density.	FMT_HIGHDENS	0x1000
Device is standard format.	FMT_STDFMT	0x2000
Media can be removed.	FMT_REMOVABLE	0x4000
Device is a hard disk.	FMT_HARDISK	0x8000

EditMod Labels

```
3-RBF path options
4-number of cylinders
```

Description

Indicates the number of cylinders per disk.

Port Generic Default Value

`80`

Port Specific Override Value

Refer to `RBF/<DEVICE>/DESC/config.des` (**Figure 5-3**).

Available Values

`-2147483648` to `2147483647`

pd_blk

BLKSTRK

EditMod Labels

```
3-RBF path options
5-default blocks/track
```

Description

Indicates the number of blocks per track on the disk for all tracks except track 0. (See `pd_t0b` for track 0 information.)

Port Generic Default Value

`2048`

Port Specific Override Value

Refer to `RBF/<DEVICE>/DESC/config.des` (**Figure 5-3**).

Available Values

`-2147483648` to `2147483647`

EditMod Labels

```
3-RBF path options
6-default blocks/track for trk0
```

Description

Indicates the number of blocks per track 0 on the disk. Depending on the device, this may be a different number for track 0 than the other tracks on the disk.

Port Generic Default Value

```
10
```

Port Specific Override Value

Refer to `RBF/<DEVICE>/DESC/config.des` (**Figure 5-3**).

Available Values

`-2147483648` to `2147483647`

pd_sas

SEGSIZE

EditMod Labels

```
3-RBF path options
7-segment allocation size
```

Description

This value specifies the default minimum number of sectors to be allocated when a file is expanded.

Port Generic Default Value

`1`

Port Specific Override Value

Refer to `RBF/<DEVICE>/DESC/config.des` (**Figure 5-3**).

Available Values

`-2147483648` to `2147483647`

pd_ilv

INTRLV

EditMod Labels

```
3-RBF path options
8-block interleave offset
```

Description

This value determines the sector interleave factor. Sectors are arranged on a disk in a certain sequential order (1, 2, 3, ... or 1, 3, 5, ...). The interleave factor determines the arrangement. For example, if the interleave factor is 2, the sectors would be arranged by twos, (1,3,5,...) starting at the base sector. See pd_boffs for base sector information.

Port Generic Default Value

3

Port Specific Override Value

Refer to RBF/<DEVICE>/DESC/config.des (**Figure 5-3**).

Available Values

-2147483648 to 2147483647

pd_toffs

TRKOFFS

EditMod Labels

```
3-RBF path options
9-track base offset
```

Description

This is the offset to the first accessible track number. Because Track 0 is often a different density, Track 0 is sometimes not used as the base track.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `RBF/<DEVICE>/DESC/config.des` (**Figure 5-3**).

Available Values

`-2147483648` to `2147483647`

EditMod Labels

```
3-RBF path options
10-block base offset
```

Description

This is the offset to the first accessible sector number. Because Sector 0 is not always the base sector.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `RBF/<DEVICE>/DESC/config.des` (**Figure 5-3**).

Available Values

`-2147483648` to `2147483647`

pd_trys

TRYS

EditMod Labels

```
3-RBF path options
11-# tries
```

Description

This is the number of times a device tries to access a disk before returning an error.

Port Generic Default Value

```
7
```

Port Specific Override Value

Refer to `RBF/<DEVICE>/DESC/config.des` (**Figure 5-3**).

Available Values

`-2147483648` to `2147483647`

pd_bsize

BLKSIZE

EditMod Labels

```
3-RBF path options
12-size of block in bytes
```

Description

This is the logical block size in bytes.

Port Generic Default Value

`256` (256 characters)

Port Specific Override Value

Refer to `RBF/<DEVICE>/DESC/config.des` (**Figure 5-3**).

Available Values

`-2147483648` to `2147483647`

pd_cntl

CONTROL

EditMod Labels

```
3-RBF path options
13-control word
```

Description

This is the device control word.

Port Generic Default Value

Macro

```
CTRL_AUTOSIZE
```

EditMod

```
0x2
```

Port Specific Override Value

Refer to `RBF/<DEVICE>/DESC/config.des` (**Figure 5-3**).

Available Values

Control word values are defined in the header file, `rbf.h`, and in **Table 5-14**.

Table 5-14 `pd_cntl` Available Values

Description	Macro	EditMod
Disable formatting of the device	`CTRL_FMTDIS`	`0x0`
Device is capable of multi-sector transfers	`CTRL_MULTI`	`0x1`

Table 5-14 pd_cntl Available Values (continued)

Description	Macro	EditMod
Device size can be obtained from device	CTRL_AUTOSIZE	0x2
Device requires only one format command	CTRL_FMTENTIRE	0x3
Device needs a full track buffer for format	CTRL_TRKWRITE	0x4

pd_wpc

PRECOMP

EditMod Labels

```
3-RBF path options
14-first write precomp cylinder
```

Description

This number indicates at which cylinder to begin write precompensation. Only older disk drives require this information, such as MFM or RLL drives.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `RBF/<DEVICE>/DESC/config.des` (**Figure 5-3**).

Available Values

`-2147483648` to `2147483647`

EditMod Labels

```
3-RBF path options
15-first reduced write current cylinder
```

Description

This number indicates at which cylinder to begin reduced write current. Only older disk drives require this information, such as MFM or RLL drives.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `RBF/<DEVICE>/DESC/config.des` (**Figure 5-3**).

Available Values

`-2147483648` to `2147483647`

pd_park

PARK

EditMod Labels

```
3-RBF path options
16-park cylinder for hard disks
```

Description

This is the cylinder where the hard disk heads should be parked when the drive is shut down.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `RBF/<DEVICE>/DESC/config.des` (**Figure 5-3**).

Available Values

`-2147483648` to `2147483647`

EditMod Labels

```
3-RBF path options
17- lsn offset for partition
```

Description

This is the offset to be used when accessing a partitioned drive.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `RBF/<DEVICE>/DESC/config.des` (**Figure 5-3**).

Available Values

`-2147483648` to `2147483647`

pd_xfersize

XFERSIZE

EditMod Labels

```
3-RBF path options
18-max transfer size in terms of bytes
```

Description

This is the maximum size of memory the controller can transfer at one time. The size is specified in bytes.

Port Generic Default Value

`0xff00`

Port Specific Override Value

Refer to `RBF/<DEVICE>/DESC/config.des` (**Figure 5-3**).

Available Values

`0` to `4294967295`

RBF Logical Unit Static Storage Fields

The following section contains the RBF logical unit static storage fields in the order they appear during an interactive EditMod session. Defined fields can appear in a different order in config.des.

Table 5-15 RBF Logical Unit Static Storage Fields

Field	Description File Macro
v_vector	VECTOR
v_irqlevel	IRQLEVEL
v_priority	PRIORITY

v_vector

VECTOR

EditMod Labels

```
4-RBF logical unit static storage
1-interrupt vector
```

Description

This is the vector number of the device interrupt.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `RBF/<DEVICE>/DESC/config.des` (**Figure 5-3**).

Available Values

`0` to `255`

v_irqlevel

IRQLEVEL

EditMod Labels

```
4-RBF logical unit static storage
2-interrupt level
```

Description

This is the hardware priority of the device interrupt.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `RBF/<DEVICE>/DESC/config.des` (**Figure 5-3**).

Available Values

`-128` to `127`

v_priority

PRIORITY

EditMod Labels

```
4-RBF logical unit static storage
3-interrupt priority
```

Description

This is the software (polling) priority of the device interrupt.

Port Generic Default Value

`5`

Port Specific Override Value

Refer to `RBF/<DEVICE>/DESC/config.des` (**Figure 5-3**).

Available Values

`-128` to `127`

RBF Logical Unit Options

The following section contains the RBF logical unit options fields in the order they appear during an interactive EditMod session. Defined fields can appear in a different order in config.des.

Table 5-16 RBF Logical Unit Options Fields

Field	Description File Macro
lu_stp	STEP
lu_tfm	DMAMODE
lu_lun	SCSILUN
lu_ctrlrid	CTRLRID
lu_totcyls	TOTCYLS

lu_stp

STEP

EditMod Labels

```
4-RBF logical unit static storage
4-RBF logical unit options
1-step rate
```

Description

This code sets the head stepping rate used with the drive. Set the step rate to the fastest value the drive is capable of to reduce access time.

Port Generic Default Value

Macro

```
STEP_30MS
```

EditMod

```
0x00
```

Port Specific Override Value

Refer to `RBF/<DEVICE>/DESC/config.des` (**Figure 5-3**).

Available Values

Step rate values are defined in the header file, `rbf.h`, and in **Table 5-17**.

Table 5-17 `lu_stp` Available Values

Description	Macro	EditMod
30 millisecond step rate	`STEP_30MS`	`0x00`
20 millisecond step rate	`STEP_20MS`	`0x01`

Table 5-17 lu_stp Available Values (continued)

Description	Macro	EditMod
12 millisecond step rate	STEP_12MS	0x02
6 millisecond step rate	STEP_6MS	0x03

lu_tfm

DMAMODE

EditMod Labels

```
4-RBF logical unit static storage
4-RBF logical unit options
2-dma transfer mode
```

Description

This hardware specific byte can be set for use of DMA mode, if it is available. DMA requires only a single interrupt for each block of characters transferred in an I/O operation. It is much faster than methods that interrupt for each character transferred.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `RBF/<DEVICE>/DESC/config.des` (**Figure 5-3**).

Available Values

`-128` to `127`

lu_lun

SCSILUN

EditMod Labels

```
4-RBF logical unit static storage
4-RBF logical unit options
3-drive logical unit number
```

Description

This number is used in the command block to identify the drive to the controller. The driver uses this number when specifying the device.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `RBF/<DEVICE>/DESC/config.des` (**Figure 5-3**).

Available Values

`-128` to `127`

lu_ctrlrid

CTRLRID

EditMod Labels

```
4-RBF logical unit static storage
4-RBF logical unit options
4-controller ID
```

Description

This is the identification number of the controller attached to the drive. The drive uses this number when communicating with the controller.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `RBF/<DEVICE>/DESC/config.des` (**Figure 5-3**).

Available Values

`-128` to `127`

EditMod Labels

```
4-RBF logical unit static storage
4-RBF logical unit options
5-total number of cylinders
```

Description

This is the actual number of cylinders on a partitioned drive. The driver uses this value to correctly initialize the drive.

Port Generic Default Value

`5`

Port Specific Override Value

Refer to `RBF/<DEVICE>/DESC/config.des` (**Figure 5-3**).

Available Values

`-2147483648` to `2147483647`

Chapter 6: PCF Device Descriptors

PCF device descriptors contain configuration data specific to one OS-9 format disk device on an OS-9 system. Values which can be configured in the descriptor include:

- Device interrupt vector and priority
- Device I/O address
- Device geometry
- Logical sector size

The next section in this chapter provides a detailed example of the configuration options you can use to change configuration values for PCF (PC-DOS file) devices.

The rest of this chapter provides a detailed list of all of the PCF device descriptor fields.

This chapter includes the following topics:

- **PCF Field Configuration Options**
- **PCF Device Descriptor Field Reference**
    - **Module Header Fields**
    - **Device Descriptor Data Definition Fields**
    - **PCF Path Option Fields**
    - **PCF Logical Unit Static Storage Fields**
        - **PCF Logical Unit Options**

PCF Field Configuration Options

To change a PCF device descriptor module configuration field, you can use either of the following methods:

1. Use the `EditMod` utility to directly modify existing PCF device descriptor modules either as a stand-alone module or as part of a merged module group (such as a boot image).
2. Modify the description file for the PCF device descriptor module and rebuild it using the makefile provided.

Direct Modification Advantages

The direct modification method has the following advantages:

Fast	No source configuration file rebuilds are necessary.
Temporary	The original module or merged-module group configuration can be easily restored through the appropriate rebuild.
Contained	Changes are limited to the individual boot image modified (merged-module option).

Description File/Rebuild Advantages

The advantage of the description file/rebuild method is that the changes are permanent and reproducible. Modifications apply to all subsequent module rebuilds and to all merged-module groups built containing the updated module.

Both methods are documented in this section. These procedures are used with the field descriptions starting with the **Module Header Fields**. For direct modification, use the `EditMod` LABELS data to navigate the

EditMod menus. The DESCRIPTION FILE MACRO data identifies the macro you need to define/modify in the configuration sources to rebuild the PCF device descriptor module.

Direct Modification

Use the Editmod utility and the following procedures to directly modify fields in the existing PCF device descriptor module. The module can stand-alone or it may be part of a merged-module group. A boot image, for example, contains multiple modules. Both situations are covered in this section. The field references later in this chapter contain a description of each configurable field, its supported values, and the sequence of menu options required by EditMod to modify that field.

For More Information

Refer to the ***Utilities Reference*** for a full description of `EditMod`'s capabilities.

Figure 6-1 Directory Location for Modifying PCF Device Descriptors

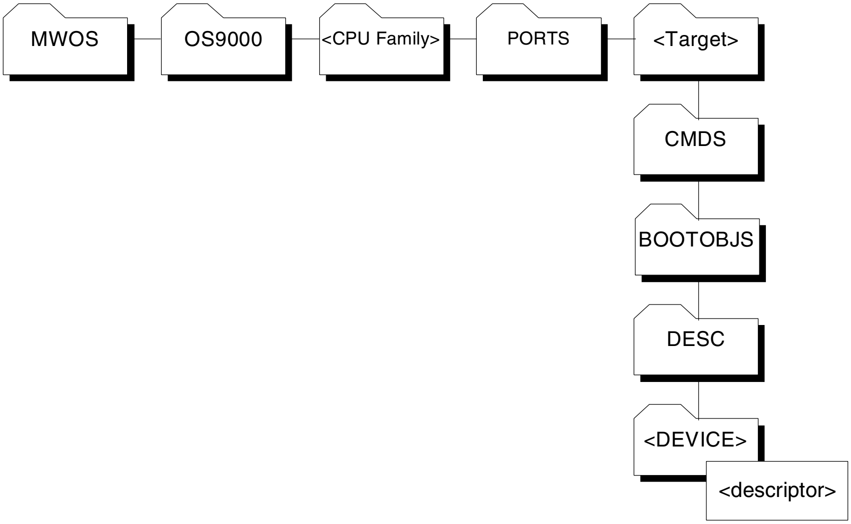
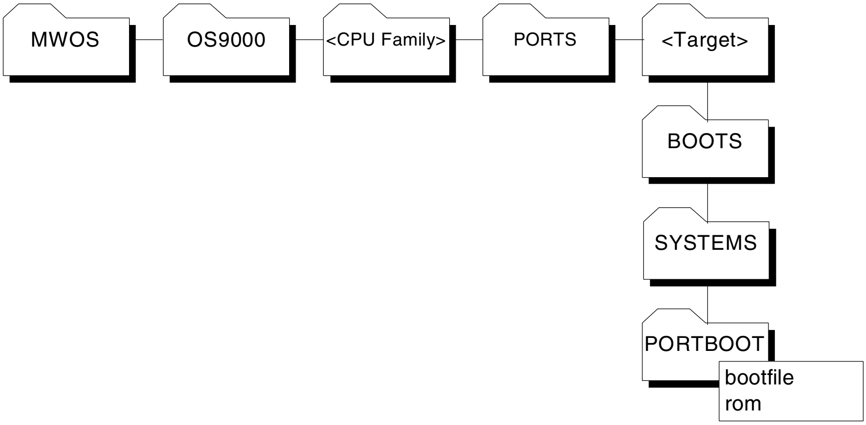


Figure 6-2 Directory Location for Modifying Low-Level Boot Images



For More Information

Refer to your board guide for information about how to modify the module lists and remake the boot images, and for specified boot image names.

Direct Modification Procedures

To modify the stand-alone module, complete the following steps:

Step 1. Change to the `CMDS/BOOTOBJS/DESC/<DEVICE>` directory (see **Figure 6-1**).

Step 2. Use EditMod to edit the module:

```
$EditMod -e <descriptor>
```

To modify the module as part of a merged module group, complete the following steps:

Step 1. Change to the `BOOTS/SYSTEMS/PORTBOOT` directory (see **Figure 6-2**).

Step 2. Use EditMod to edit the module:

```
$EditMod -e <descriptor> -f=<boot image name>
```

Step 3. Use the menu selections provided in the `EditMod` LABELS section of the field reference later in this chapter to locate the fields you want to edit.

Step 4. Select a new value for the field from the AVAILABLE VALUES section of the field reference. Enter that value at the EditMod prompt to modify the field.

Step 5. If you want to make additional modifications, use the `p` command (previous) to step backward through the EditMod menus. Repeat Steps 3 and 4 until you have made all desired modifications to the descriptor.

Step 6. Select the `w` command (write) to save the changes.

Step 7. Select the `q` command (quit) to exit `EditMod`.

Note

Unless you modified the PCF device descriptors in your boot image, you should rebuild your boot image to include the new descriptor.

Example EditMod Session

This example modifies a PCF device descriptor as part of the boot image `rom`:

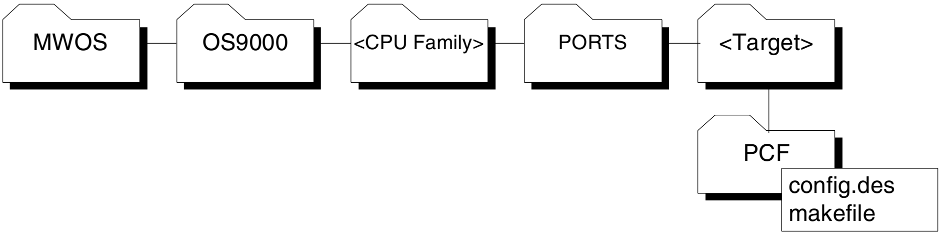
```
$  EditMod -e mhs0 -f=rom

 1. module header
 2. device descriptor data definitions
 3. PCF path options
 4. PCF logical unit static storage
```

```
Which? [?/1-4/p/t/a/w/q] 4

 1. interrupt vector                        : 0x0
 2. interrupt level                         : 0
 3. interrupt priority                      : 5
 4. PCF logical unit options

Which? [?/1-4/p/t/a/w/q] 3

interrupt priority                      : 5
New value: 1

 1. interrupt vector                        : 0x0
 2. interrupt level                         : 0
 3. interrupt priority                      : 1
 4. PCF logical unit options

Which? [?/1-19/p/t/a/w/q] w

Which? [?/1-19/p/t/a/w/q] q
```

Description File Configuration

You can use these procedures to modify the appropriate description file and rebuild the PCF device descriptors for your port directory. The DESCRIPTION FILE MACROS section of the field reference specifies the name of the macro you modify/define in the description files to configure the field. The value used in the define is chosen from the AVAILABLE VALUES specified for the field.

Figure 6-3 Directory Location for Modifying PCF Description Files



Description File Configuration Procedures

Step 1. Change to the `PCF/<DEVICE>` directory (see **Figure 6-3**).

Step 2. Edit the file `config.des` and read the included comments for more information on using the specific description files provided in your software distribution. The `config.des` file contains a list of macro names that can be defined to override the global default values for the configuration fields.

Step 3. Refer to the DESCRIPTION FILE MACRO section in the field reference later in this chapter to determine the macro name you define to configure the target field.

Step 4. Read the comments in `config.des` to determine where to place the define for this macro.

Step 5. Select the value you want to use to configure the field. See the AVAILABLE VALUES section of the field reference data for values or macros that can be used for the definition. Define the macro by entering a definition in the appropriate description files as follows:

```
#define <macro> <value>
```

Step 6. Save the changes and rebuild the PCF device descriptors, entering the following command in the `PCF/<DEVICE>/DESC` directory:

```
os9make
```

Step 7. Rebuild your boot image to include the new descriptor.

PCF Device Descriptor Field Reference

This section contains a list of the most commonly configured fields in the PCF device descriptors. Each field entry contains the following information:

- <Field name> - The call name for each field that can be reconfigured in the module.
- EditMod LABELS - `EditMod` menu selections for navigating to the proper field in an `EditMod` session.
- DESCRIPTION FILE MACRO - The macro name you modify/define in the description file.
- DESCRIPTION - A brief description of the field's purpose and use.
- EXAMPLE - An optional example of the description file entry showing how to change the value of this field.
- PORT GENERIC DEFAULT VALUE - The value set in the port generic description file for this field. This is the value the field is assigned when the module is built, unless the appropriate macro has been defined in the port specific description file to override this default value.
- PORT SPECIFIC OVERRIDE VALUE - The value set in the port specific description file for this field. If defined, this is the value the field is assigned when the module is built, overriding the port generic default value.
- AVAILABLE VALUES - Values to which the field can be set through `EditMod` or the description files. In many cases, this data is presented in a table that maps a description of the value to a numeric value appropriate for entry in EditMod, and to a pre-defined macro available for use in the description file.

Module Header Fields

The following section contains the module header fields in the order they appear during an interactive EditMod session. Defined fields can appear in a different order in config.des.

Table 6-1 Module Header Fields

Field	Description File Macro
_m_group	MH_GROUP
_m_user	MH_USER
mod_name	MH_NAME
m_access	MH_ACCESS
m_tylan	MH_TYLAN
m_attrev	MH_ATTREV
m_edit	MH_EDITION

_m_group

MH_GROUP

EditMod Labels

```
1-module header
1-module owner's group number
```

Description

Group ID of the module's owner. The group number allows people working in the same department or on the same project to share a common identification number.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `PCF/<DEVICE>/DESC/config.des` (**Figure 6-3**).

Available Values

`0` to `65535`

_m_user

MH_USER

EditMod Labels

```
1-module header
2-module owner’s user number
```

Description

User ID of the module’s owner. The user number identifies a specific user.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `PCF/<DEVICE>/DESC/config.des` (**Figure 6-3**).

Available Values

`0` to `65535`

mod_name

MH_NAME

EditMod Labels

```
1-module header
3-module name
```

Description

Contains the module name string.

Port Generic Default Value

String value (None)

Port Specific Override Value

Refer to `PCF/<DEVICE>/DESC/config.des` (**Figure 6-3**).

Available Values

Any ASCII character string. The string can contain C-style character escapes (such as `\n` and `\012`).

EditMod Labels

```
1-module header
4-access permissions
```

Description

Defines the permissible module access by its owner or by other users.

Port Generic Default Value

Macro

```
MP_OWNER_READ | MP_OWNER_EXEC | MP_GROUP_READ |
MP_GROUP_EXEC | MP_WORLD_READ | MP_WORLD_EXEC
```

EditMod

```
0x555
```

Port Specific Override Value

Refer to `PCF/<DEVICE>/DESC/config.des` (**Figure 6-3**).

Available Values

Module access permission values are located in the header file, `module.h`, and are listed in **Table 6-2**.

Table 6-2 `m_access` Available Values

Description	Macro	EditMod
Read permission by owner	`MP_OWNER_READ`	`0x0001`
Write permission by owner	`MP_OWNER_WRITE`	`0x0002`

Table 6-2 `m_access` Available Values (continued)

Description	Macro	EditMod
Execute permission by owner	`MP_OWNER_EXEC`	`0x0004`
Owner permission mask	`MP_OWNER_MASK`	`0x000f`
Read permission by group	`MP_GROUP_READ`	`0x0010`
Write permission by group	`MP_GROUP_WRITE`	`0x0020`
Execute permission by group	`MP_GROUP_EXEC`	`0x0040`
Group permission mask	`MP_GROUP_MASK`	`0x00f0`
Read permission by world	`MP_WORLD_READ`	`0x0100`
Write permission by world	`MP_WORLD_WRITE`	`0x0200`
Execute permission by world	`MP_WORLD_EXEC`	`0x0400`
World permission mask	`MP_WORLD_MASK`	`0x0f00`
All permissions for owner, group, and world	`MP_WORLD_ACCESS`	`0x0777`
System permission mask	`MP_SYSTM_MASK`	`0xf000`

EditMod Labels

```
1-module header
5-type/language
```

Description

Contains the module’s type (first byte) and language (second byte). The language codes indicate if the module is executable and which language the run-time system requires for execution, if any.

Port Generic Default Value

Macro

```
(MT_DATA<<8) + ML_OBJECT
```

EditMod

```
0x401
```

Port Specific Override Value

Refer to `PCF/<DEVICE>/DESC/config.des` (**Figure 6-3**).

Available Values

Module type values and language codes are located in the header file, `module.h`, and are listed in **Table 6-3** and **Table 6-4**.

Table 6-3 `m_tylan` Available Module Type Values

Description	Macro	EditMod
Not used (wildcard value in system calls)	`MT_ANY`	`0x0000`
Program module	`MT_PROGRAM`	`0x0001`
Subroutine module	`MT_SUBROUT`	`0x0002`
Multi-module (reserved for future use)	`MT_MULTI`	`0x0003`
Data module	`MT_DATA`	`0x0004`
Configuration data block data module	`MT_CDBDATA`	`0x0005`
Reserved for future use	`0xb-0xa`	`0xb-0xa`
User trap library	`MT_TRAPLIB`	`0x000b`
System module	`MT_SYSTEM`	`0x000c`
File manager module	`MT_FILEMAN`	`0x000d`
Physical device driver	`MT_DEVDRVR`	`0x000e`
Device descriptor module	`MT_DEVDESC`	`0x000f`

Table 6-3 m_tylan Available Module Type Values (continued)

Description	Macro	EditMod
User definable	0x10-0xfe	0x10-0xfe
Module type mask	MT_MASK	0xff00

Table 6-4 m_tylan Available Language Code Values

Description	Macro	EditMod
Unspecified language (wildcard in system calls)	ML_ANY	0x0
Machine language	ML_OBJECT	0x1
Basic I-code (reserved for future use)	ML_ICODE	0x2
Pascal P-code (reserved for future use)	ML_PCODE	0x3
C I-code (reserved for future use)	ML_CCODE	0x4
Cobol I-code (reserved for future use)	ML_CBLCODE	0x5
Fortran	ML_FRTNCODE	0x6
Reserved for future use	0x7-0xf	0x7-0xf
User-definable	0x10-0xfe	0x10-0xfe
Module language mask	ML_MASK	0x00ff

m_attrev

MH_ATTREV

EditMod Labels

```
1-module header
6-revision/attributes
```

Description

Contains the module's attributes (first byte) and revision (second byte).

Port Generic Default Value

Macro

```
MA_REENT<<8
```

EditMod

```
0x8000
```

Port Specific Override Value

Refer to `PCF/<DEVICE>/DESC/config.des` (**Figure 6-3**).

Available Values

Module attribute and revision codes are located in the header file `module.h.`, and are listed in **Table 6-5**.

Note

If two modules with the same name are found in the memory search or are loaded into the current module directory, only the module with the highest revision level is kept. This enables easy substitution of modules for update or correction.

Table 6-5 `m_attrev` Available Attribute and Revision Values

Description	Macro	EditMod
The module is re-entrant (sharable by multiple tasks).	`MA_REENT` (shifted left to first byte: `MA_REENT<<8)`	`0x80` (shifted left to first byte: `0x8000`)
The module is sticky. A sticky module is not removed from memory until its link count becomes -1 or memory is required for another use.	`MA_GHOST` (shifted left to first byte: `MA_GHOST<<8)`	`0x40` (shifted left to first byte: `0x4000`)
The module is a system-state module.	`MA_SUPER` (shifted left to first byte: `MA_SUPER<<8)`	`0x20` (shifted left to first byte: `0x2000`)
User-definable revision number	`0x0-0xfe`	`0x0-0xfe`
Module attribute mask	`MA_MASK`	`0xff00`
Module revision mask	`MR_MASK`	`0x00ff`

m_edit

MH_EDITION

EditMod Labels

```
1-module header
7-edition
```

Description

Indicates the software release level for maintenance. OS-9 does not use this field. Whenever a program is revised (even for a small change), increase this number. We recommend internal documentation within the source program be keyed to this system.

Port Generic Default Value

`1`

Port Specific Override Value

Refer to `PCF/<DEVICE>/DESC/config.des` (**Figure 6-3**).

Available Values

`0` to `65535`

Device Descriptor Data Definition Fields

The following section contains the device descriptor data definition fields in the order they appear during an interactive EditMod session. Defined fields can appear in a different order in config.des.

Table 6-6 Device Descriptor Data Definition Fields

Field	Description File Macro
dd_port	PORTADDR
dd_lun	LUN
dd_pd_size	PD_SIZE
dd_type	DD_TYPE
dd_mode	DD_MODE
dd_port	MFGR_NAME
drvr_name	DRVR_NAME
dd_class	DD_CLASS

dd_port

PORTADDR

EditMod Labels

```
2-device descriptor data definitions
1-device port address
```

Description

Absolute physical address of the hardware controller. This is the address of the device on the bus. This is the lowest address the device has mapped. Port address is hardware dependent.

Macro Example

```
#define PORTADDR    0xfffe4000
```

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `PCF/<DEVICE>/DESC/config.des` (**Figure 6-3**).

Available Values

`0` to `4294967295`

EditMod Labels

```
2-device descriptor data definitions
2-logical unit number
```

Description

Distinguishes between the different devices driven from a unique controller. Each unique number represents a different logical unit static storage area.

Macro Example

```
#define LUN    2
```

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `PCF/<DEVICE>/DESC/config.des` (**Figure 6-3**).

Available Values

`0` to `65535`

dd_pd_size

PD_SIZE

EditMod Labels

```
2-device descriptor data definitions
3-path descriptor size
```

Description

Size of the path descriptor. IOMAN uses this value when it allocates a path descriptor.

Port Generic Default Value

`360`

Port Specific Override Value

Refer to `PCF/<DEVICE>/DESC/config.des` (**Figure 6-3**).

Available Values

`0` to `65535`

EditMod Labels

```
2-device descriptor data definitions
4-device type
```

Description

Identifies the I/O class of the device.

Port Generic Default Value

Macro

```
DT_PCF
```

EditMod

```
0xa
```

Port Specific Override Value

Refer to `PCF/<DEVICE>/DESC/config.des` (**Figure 6-3**).

Available Values

Device type values are defined in the header file `io.h`, and are listed in **Table 6-7**.

Table 6-7 `dd_type` Available Values

Description	Macro	EditMod
Sequential Character File Type	`DT_SCF`	`0x0`
Random Block File Type	`DT_RBF`	`0x1`
Pipe File Type	`DT_PIPE`	`0x2`

Table 6-7 dd_type Available Values (continued)

Description	Macro	EditMod
Sequential Block File Type	DT_SBF	0x3
Network File Type	DT_NFM	0x4
Compact Disc File Type	DT_CDFM	0x5
User Communication Manager	DT_UCM	0x6
Socket Communication Manager	DT_SOCK	0x7
Pseudo-Keyboard Manager	DT_PTTY	0x8
Graphics File Manager	DT_GFM	0x9
PC-DOS File Manager	DT_PCF	0xa
Non-volatile RAM File Manager	DT_NRF	0xb
ISDN File Manager	DT_ISDN	0xc
MPFM File Manager	DT_MPFM	0xd
Real-Time Network File Manager	DT_RTNFM	0xe
Serial Protocol File Manager	DT_SPF	0xf
Inet File Manager	DT_INET	0xa0
Reserved for Microware Use Only	17-127	0xa1-0x7f

EditMod Labels

```
2-device descriptor data definitions
5-device mode capabilities
```

Description

Used to check the validity of a caller's access mode byte in `I_CREATE` or `I_OPEN` system calls. If a bit is set, the device can perform the corresponding function. The `S_ISIZE` bit is usually set, because it is handled by the file manager or ignored. If the `S_ISHARE` bit is set, the device is non-sharable. A printer is an example of a non-sharable device.

Port Generic Default Value

Macro

```
S_IPRM
```

EditMod

```
0xFFFF
```

Port Specific Override Value

Refer to `PCF/<DEVICE>/DESC/config.des` (**Figure 6-3**).

Available Values

The file access modes are defined in the header file, `modes.h`, and located in **Table 6-8**. The file access permission values are defined in the header file `modes.h` and in **Table 6-9**.

Table 6-8 `dd_mode` Available Values for File Access Modes

Description	Macro	EditMod
Truncate on open	`S_ITRUNC`	`0x0100`
Ensure contiguous file	`S_ICONTIG`	`0x0400`
Error if file exists on create	`S_IEXCL`	`0x0400`
Create file	`S_ICREAT`	`0x0800`
Append to file	`S_IAPPEND`	`0x1000`
Non-sharable	`S_ISHARE`	`0x4000`

Table 6-9 `dd_mode` Available Values for File Access Permissions

Description	Macro	EditMod
Mask for permission bits	`S_IPRM`	`0xffff`
Owner read	`S_IREAD`	`0x0001`
Owner write	`S_IWRITE`	`0x0002`
Owner execute	`S_IEXEC`	`0x0004`
Search permission	`S_ISEARCH`	`0x0004`

Table 6-9 dd_mode Available Values for File Access Permissions

Description	Macro	EditMod
Group read	S_IGREAD	0x0010
Group write	S_IGWRITE	0x0020
Group execute	S_IGEXEC	0x0040
Group search	S_IGSEARCH	0x0040
Public read	S_IOREAD	0x0100
Public write	S_IOWRITE	0x0200
Public execute	S_IOEXEC	0x0400
Public search	S_IOSEARCH	0x0400

fmgr_name

FMGR_NAME

EditMod Labels

```
2-device descriptor data definitions
6-file manager name
```

Description

Contains the name string of the file manager module to use.

Port Generic Default Value

```
"pcf"
```

Port Specific Override Value

Refer to `PCF/<DEVICE>/DESC/config.des` (**Figure 6-3**).

Available Values

Any ASCII character string. The string may contain C-style character escapes (such as `\n` and `\012`).

drvr_name

DRVR_NAME

EditMod Labels

```
2-device descriptor data definitions
7-driver name
```

Description

Contains the name string of the device driver module to use.

Port Generic Default Value

`NULL`

Port Specific Override Value

Refer to `PCF/<DEVICE>/DESC/config.des` (**Figure 6-3**).

Available Values

Any ASCII character string. The string can contain C-style character escapes (such as `\n` and `\012`).

dd_class

DD_CLASS

EditMod Labels

```
1-module header
2-device descriptor data definitions
8-device class (sequential or random)
```

Description

Used to identify the class of the device, whether it is random or sequential access.

Port Generic Default Value

Macro

```
DC_RND
```

EditMod

```
0x2
```

Port Specific Override Value

Refer to `PCF/<DEVICE>/DESC/config.des` (**Figure 6-3**).

Available Values

Device class available values are defined in the header file, `io.h`, and in **Table 6-10**.

Table 6-10 `dd_class` Available Values

Description	Macro	EditMod
Sequential access device	`DC_SEQ`	`0x0001`
Random access device	`DC_RND`	`0x0002`

PCF Path Option Fields

The following section contains the PCF path option fields in the order they appear during an interactive EditMod session. Defined fields can appear in a different order in config.des.

Table 6-11 PCF Path Option Fields

Field	Description File Macro
pd_sid	SIDES
pd_vfy	VERIFY
pd_format	FORMAT
pd_cyl	CYLNDRS
pd_blk	BLKSTRK
pd_t0b	BLKSTRK0
pd_sas	SEGSIZE
pd_ilv	INTRLV
pd_toffs	TRKOFFS
pd_boffs	BLKOFFS
pd_trys	TRYS
pd_bsize	BLKSIZE
pd_cntl	CONTROL

Table 6-11 PCF Path Option Fields (continued)

Field	Description File Macro
pd_wpc	PRECOMP
pd_rwr	REDWRITE
pd_park	PARK
pd_lsnoffs	LSNOFFS
pd_xfersize	XFERSIZE

EditMod Labels

```
3-PCF path options
1-number of surfaces
```

Description

Indicates the number of surfaces (heads or sides) for a disk unit.

Port Generic Default Value

`2`

Port Specific Override Value

Refer to `PCF/<DEVICE>/DESC/config.des` (**Figure 6-3**).

Available Values

`-2147483648` to `2147483647`

pd_vfy

VERIFY

EditMod Labels

```
3-PCF path options
2-verify disk writes (0=verify)
```

Description

Indicates whether a write is verified by a re-read and compare. Write verify operations are generally performed on floppy disks but not hard disks because of the lower soft error rate of hard disks.

Port Generic Default Value

0 (zero)

Port Specific Override Value

Refer to PCF/<DEVICE>/DESC/config.des (**Figure 6-3**).

Available Values

Device verify values are defined in the header file, PCF.h, and in **Table 6-12**.

Table 6-12 pd_vfy Available Values

Description	Macro	EditMod
Verify disk write	0	0x0
No verification	1	0x01

EditMod Labels

```
3-PCF path options
3-device format
```

Description

Indicates whether a write is verified by a re-read and compare. Write verify operations are generally performed on floppy disks but not hard disks because of the lower soft error rate of hard disks.

Port Generic Default Value

Macro

```
FMT_STDFMT + FMT_DBLBITDNS + FMT_DBLTRKDNS + FMT_DBLSIDE
```

EditMod

```
0x200e
```

Port Specific Override Value

Refer to `PCF/<DEVICE>/DESC/config.des` (**Figure 6-3**).

Available Values

Device format values are defined in the header file, `PCF.h`, and in **Table 6-13**.

Table 6-13 `pd_format` **Available Values**

Description	Macro	EditMod
Track 0 is double density.	`FMT_DBLTRK0`	`0x0001`
Device is double bit density.	`FMT_DBLBITDNS`	`0x0002`

Table 6-13 pd_format Available Values (continued)

Description	Macro	EditMod
Device is double track density.	FMT_DBLTRKDNS	0x0004
Device is double sided.	FMT_DBLSIDE	0x0008
Drive is eight inch.	FMT_EIGHTINCH	0x0010
Drive is five inch.	FMT_FIVEINCH	0x0020
Drive is three inch.	FMT_THREEINCH	0x0040
Device is high density.	FMT_HIGHDENS	0x1000
Device is standard format.	FMT_STDFMT	0x2000
Media can be removed.	FMT_REMOVABLE	0x4000
Device is a hard disk.	FMT_HARDISK	0x8000

EditMod Labels

```
3-PCF path options
4-number of cylinders
```

Description

Indicates the number of cylinders per disk.

Port Generic Default Value

`80`

Port Specific Override Value

Refer to `PCF/<DEVICE>/DESC/config.des` (**Figure 6-3**).

Available Values

`-2147483648` to `2147483647`

pd_blk

BLKSTRK

EditMod Labels

```
3-PCF path options
5-default blocks/track
```

Description

Indicates the number of blocks per track on the disk for all tracks except track 0. (See `pd_t0b` for track 0 information.)

Port Generic Default Value

```
16
```

Port Specific Override Value

Refer to `PCF/<DEVICE>/DESC/config.des` (**Figure 6-3**).

Available Values

`-2147483648` to `2147483647`

EditMod Labels

```
3-PCF path options
6-default blocks/track for trk0
```

Description

Indicates the number of blocks per track 0 on the disk. Depending on the device, this can be a different number for track 0 than the other tracks on the disk.

Port Generic Default Value

```
10
```

Port Specific Override Value

Refer to `PCF/<DEVICE>/DESC/config.des` (**Figure 6-3**).

Available Values

`-2147483648` to `2147483647`

pd_sas

SEGSIZE

EditMod Labels

```
3-PCF path options
7-segment allocation size
```

Description

This value specifies the default minimum number of sectors to be allocated when a file is expanded.

Port Generic Default Value

`1`

Port Specific Override Value

Refer to `PCF/<DEVICE>/DESC/config.des` (**Figure 6-3**).

Available Values

`-2147483648` to `2147483647`

EditMod Labels

```
3-PCF path options
8-block interleave offset
```

Description

This value determines the sector interleave factor. Sectors are arranged on a disk in a certain sequential order (1, 2, 3, ... or 1, 3, 5, ...). The interleave factor determines the arrangement. For example, if the interleave factor is 2, the sectors would be arranged by twos, (1,3,5,...) starting at the base sector. (See `pd_boffs` for base sector information.)

Port Generic Default Value

`3`

Port Specific Override Value

Refer to `PCF/<DEVICE>/DESC/config.des` (**Figure 6-3**).

Available Values

`-2147483648` to `2147483647`

pd_toffs

TRKOFFS

EditMod Labels

```
3-PCF path options
9-track base offset
```

Description

This is the offset to the first accessible track number. Because Track 0 is often a different density, Track 0 is sometimes not used as the base track.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `PCF/<DEVICE>/DESC/config.des` (**Figure 6-3**).

Available Values

`-2147483648` to `2147483647`

EditMod Labels

```
3-PCF path options
10-block base offset
```

Description

This is the offset to the first accessible sector number. Because Sector 0 is not always the base sector.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `PCF/<DEVICE>/DESC/config.des` (**Figure 6-3**).

Available Values

`-2147483648` to `2147483647`

pd_trys

TRYS

EditMod Labels

```
3-PCF path options
11-# tries
```

Description

This is the number of times a device tries to access a disk before returning an error.

Port Generic Default Value

`7`

Port Specific Override Value

Refer to `PCF/<DEVICE>/DESC/config.des` (**Figure 6-3**).

Available Values

`-2147483648` to `2147483647`

EditMod Labels

```
3-PCF path options
12-size of block in bytes
```

Description

This is the logical block size in bytes.

Port Generic Default Value

`256` (256 characters)

Port Specific Override Value

Refer to `PCF/<DEVICE>/DESC/config.des` (**Figure 6-3**).

Available Values

`-2147483648` to `2147483647`

pd_cntl

CONTROL

EditMod Labels

```
3-PCF path options
13-control word
```

Description

This is the device control word.

Port Generic Default Value

Macro

```
CTRL_MULTI
```

EditMod

```
0x1
```

Port Specific Override Value

Refer to `PCF/<DEVICE>/DESC/config.des` (**Figure 6-3**).

Available Values

Control word values are defined in the header file, `PCF.h`, and in **Table 6-14**.

Table 6-14 `pd_cntl` Available Values

Description	Macro	EditMod
Disable formatting of the device	`CTRL_FMTDIS`	`0x0`
Device is capable of multi-sector transfers	`CTRL_MULTI`	`0x1`

Table 6-14 `pd_cntl` Available Values (continued)

Description	Macro	EditMod
Device size can be obtained from device	`CTRL_AUTOSIZE`	`0x2`
Device requires only one format command	`CTRL_FMTENTIRE`	`0x3`
Device needs a full track buffer for format	`CTRL_TRKWRITE`	`0x4`

pd_wpc

PRECOMP

EditMod Labels

```
3-PCF path options
14-first write precomp cylinder
```

Description

This number indicates at which cylinder to begin write precompensation. Only older disk drives require this information, such as MFM or RLL drives.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `PCF/<DEVICE>/DESC/config.des` (**Figure 6-3**).

Available Values

`-2147483648` to `2147483647`

EditMod Labels

```
3-PCF path options
15-first reduced write current cylinder
```

Description

This number indicates at which cylinder to begin reduced write current. Only older disk drives require this information, such as MFM or RLL drives.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `PCF/<DEVICE>/DESC/config.des` (**Figure 6-3**).

Available Values

`-2147483648` to `2147483647`

pd_park

PARK

EditMod Labels

```
3-PCF path options
16-park cylinder for hard disks
```

Description

This is the cylinder where the hard disk heads should be parked when the drive is shut down.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `PCF/<DEVICE>/DESC/config.des` (**Figure 6-3**).

Available Values

`-2147483648` to `2147483647`

EditMod Labels

```
3-PCF path options
17- lsn offset for partition
```

Description

This is the offset to be used when accessing a partitioned drive.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `PCF/<DEVICE>/DESC/config.des` (**Figure 6-3**).

Available Values

`-2147483648` to `2147483647`

pd_xfersize

XFERSIZE

EditMod Labels

```
3-PCF path options
max transfer size in terms of bytes
```

Description

This is the maximum size of memory the controller can transfer at one time. The size is specified in bytes.

Port Generic Default Value

`0xff00`

Port Specific Override Value

Refer to `PCF/<DEVICE>/DESC/config.des` (**Figure 6-3**).

Available Values

`0` to `4294967295`

PCF Logical Unit Static Storage Fields

The following section contains the PCF logical unit static storage fields in the order they appear during an interactive EditMod session. Defined fields may appear in a different order in config.des.

Table 6-15 PCF Logical Unit Static Storage Fields

Field	Description File Macro
v_vector	VECTOR
v_irqlevel	IRQLEVEL
v_priority	PRIORITY

v_vector

VECTOR

EditMod Labels

```
4-PCF logical unit static storage
1-interrupt vector
```

Description

This is the vector number of the device interrupt.

Port Generic Default Value

`80`

Port Specific Override Value

Refer to `PCF/<DEVICE>/DESC/config.des` (**Figure 6-3**).

Available Values

`0` to `255`

EditMod Labels

```
4-PCF logical unit static storage
2-interrupt level
```

Description

This is the hardware priority of the device interrupt.

Port Generic Default Value

`3`

Port Specific Override Value

Refer to `PCF/<DEVICE>/DESC/config.des` (**Figure 6-3**).

Available Values

`-128` to `127`

v_priority

PRIORITY

EditMod Labels

```
4-PCF logical unit static storage
3-interrupt priority
```

Description

This is the software (polling) priority of the device interrupt.

Port Generic Default Value

`10`

Port Specific Override Value

Refer to `PCF/<DEVICE>/DESC/config.des` (**Figure 6-3**).

Available Values

`-128` to `127`

PCF Logical Unit Options

The following section contains the PCF logical unit options fields in the order they appear during an interactive EditMod session. Defined fields can appear in a different order in config.des.

Table 6-16 PCF Logical Unit Options Fields

Field	Description File Macro
lu_stp	STEP
lu_tfm	DMAMODE
lu_lun	SCSILUN
lu_ctrlrid	CTRLRID
lu_totcyls	TOTCYLS

lu_stp

STEP

EditMod Labels

```
4-PCF logical unit static storage
4-PCF logical unit options
1-step rate
```

Description

This code sets the head stepping rate used with the drive. Set the step rate to the fastest value the drive is capable of to reduce access time.

Port Generic Default Value

Macro

```
STEP_30MS
```

EditMod

```
0x00
```

Port Specific Override Value

Refer to `PCF/<DEVICE>/DESC/config.des` (**Figure 6-3**).

Available Values

Step rate values are defined in the header file, `PCF.h`, and in **Table 6-17**.

Table 6-17 `lu_stp` Available Values

Description	Macro	EditMod
30 millisecond step rate	`STEP_30MS`	`0x00`
20 millisecond step rate	`STEP_20MS`	`0x01`

Table 6-17 lu_stp Available Values (continued)

Description	Macro	EditMod
12 millisecond step rate	STEP_12MS	0x02
6 millisecond step rate	STEP_6MS	0x03

lu_tfm

DMAMODE

EditMod Labels

```
4-PCF logical unit static storage
4-PCF logical unit options
2-dma transfer mode
```

Description

This hardware specific byte can be set for use of DMA mode, if it is available. DMA requires only a single interrupt for each block of characters transferred in an I/O operation. It is much faster than methods that interrupt for each character transferred.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `PCF/<DEVICE>/DESC/config.des` (**Figure 6-3**).

Available Values

`-128` to `127`

lu_lun

SCSILUN

EditMod Labels

```
4-PCF logical unit static storage
4-PCF logical unit options
3-drive logical unit number
```

Description

This number is used in the command block to identify the drive to the controller. The driver uses this number when specifying the device.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `PCF/<DEVICE>/DESC/config.des` (**Figure 6-3**).

Available Values

`-128` to `127`

lu_ctrlrid

CTRLRID

EditMod Labels

```
4-PCF logical unit static storage
4-PCF logical unit options
4-controller ID
```

Description

This is the identification number of the controller attached to the drive. The drive uses this number when communicating with the controller.

Port Generic Default Value

0 (zero)

Port Specific Override Value

Refer to PCF/<DEVICE>/DESC/config.des (**Figure 6-3**).

Available Values

-128 to 127

EditMod Labels

```
4-PCF logical unit static storage
4-PCF logical unit options
5-total number of cylinders
```

Description

This is the actual number of cylinders on a partitioned drive. The driver uses this value to correctly initialize the drive.

Port Generic Default Value

`80`

Port Specific Override Value

Refer to `PCF/<DEVICE>/DESC/config.des` (**Figure 6-3**).

Available Values

`-2147483648` to `2147483647`

Chapter 7: Pipe Device Descriptors

Pipe device descriptors contain configuration data for the pipe pseudo-device used on OS-9. The most common value configured in the pipe device descriptor is the default pipe size.

The next section in this chapter provides a detailed example of the two configuration options you can use to change configuration values in pipe device descriptors.

The rest of this chapter provides a detailed list of all of the pipe device descriptor fields, including field descriptions and available values.

This chapter includes the following topics:

Pipe Device Descriptor Field Configuration Options

To change a pipe device descriptor module configuration field, you can use either of the following methods:

1. Use the EditMod utility to directly modify existing pipe device descriptor modules either as a stand-alone module or as part of a merged module group (such as a boot image).
2. Modify the description file for the pipe device descriptor module and rebuild it using the makefile provided.

Direct Modification Advantages

The direct modification method has the following advantages:

Fast	No source configuration file rebuilds are necessary.
Temporary	The original module or merged-module group configuration can be easily restored through the appropriate rebuild.
Contained	Changes are limited to the individual boot image modified (merged-module option).

Description File/Rebuild Advantages

The advantage of the description file/rebuild method is that the changes are permanent and reproducible. Modifications apply to all subsequent module rebuilds and to all merged-module groups built containing the updated module.

Both methods are documented in this section. These procedures are used with the field descriptions starting with the **Module Header Fields**. For direct modification, use the EditMod LABELS data to navigate the

`EditMod` menus. The DESCRIPTION FILE MACRO data identifies the macro you need to define/modify in the configuration sources to rebuild the pipe device descriptor module.

Direct Modification

Use the `Editmod` utility and the following procedures to directly modify fields in the existing pipe device descriptor module. The module can stand-alone or it can be part of a merged-module group. A boot image, for example, contains multiple modules. Both situations are covered in this section. The field references later in this chapter contain a description of each configurable field, its supported values, and the sequence of menu options required by `EditMod` to modify that field.

For More Information

Refer to the ***Utilities Reference*** for a full description of `EditMod`'s capabilities.

Figure 7-1 Directory Location for Modifying Pipe Device Descriptors

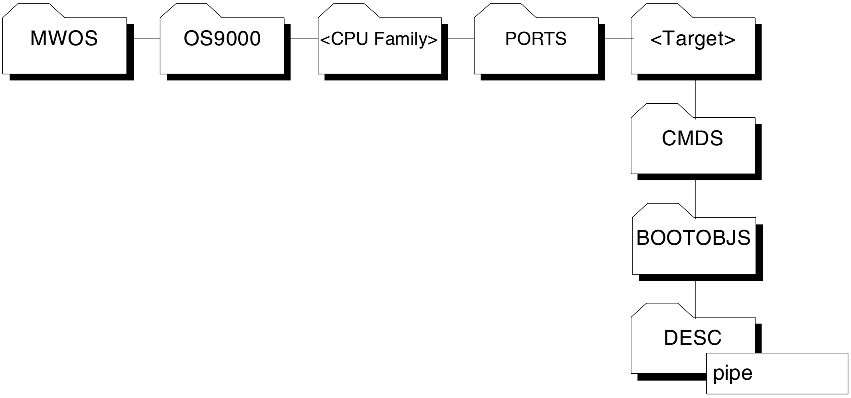
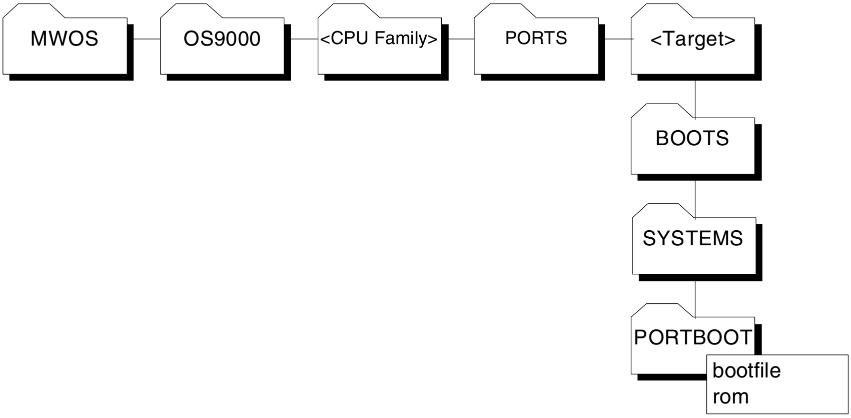


Figure 7-2 Directory Location for Modifying Low-Level Boot Images



For More Information

Refer to your board guide for information about how to modify the module lists and remake the boot images, and for specified boot image names.

Direct Modification Procedures

To modify the stand-alone module, complete the following steps:

Step 1. Change to the `CMDS/BOOTOBJS/DESC/<DEVICE>` directory (see **Figure 7-1**).

Step 2. Use EditMod to edit the module:

```
$EditMod -e <descriptor>
```

To modify the module as part of a merged module group, complete the following steps:

Step 1. Change to the `BOOTS/SYSTEMS/PORTBOOT` directory (see **Figure 7-2**).

Step 2. Use EditMod to edit the module:

```
$EditMod -e <descriptor> -f=<boot image name>
```

Step 3. Use the menu selections provided in the `EditMod` LABELS section of the field reference later in this chapter to locate the fields you want to edit.

Step 4. Select a new value for the field from the AVAILABLE VALUES section of the field reference. Enter that value at the EditMod prompt to modify the field.

Step 5. If you want to make additional modifications, use the `p` command (previous) to step backward through the EditMod menus. Repeat Steps 3 and 4 until you have made all desired modifications to the descriptor.

Step 6. Select the `w` command (write) to save the changes.

Step 7. Select the `q` command (quit) to exit `EditMod`.

Note

Unless you modified the pipe device descriptors in your boot image, you should rebuild your boot image to include the new descriptor.

Example EditMod Session

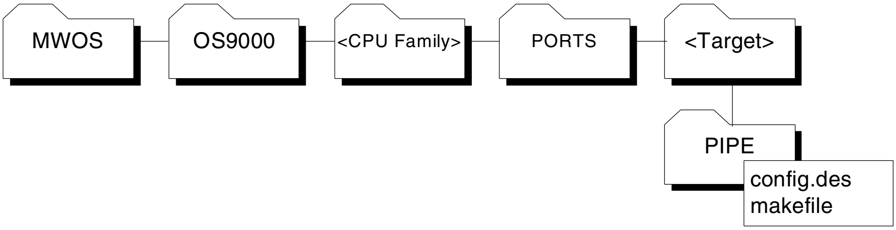
This example modifies an pipe device descriptor as part of the boot image `rom`:

```
$ EditMod -e pipe

 1. module header
 2. device descriptor data definitions
 3. pipeman logical unit static storage

Which? [?/1-3/p/t/a/w/q] 3

 1. pipe FIFO buffer size                   : 0x100

$Which? [?/1-6/p/t/a/w/q] 1

pipe FIFO buffer size                   : 0x100
New value: 0x200

 1. pipe FIFO buffer size                   : 0x200

Which? [?/1-19/p/t/a/w/q] w

Which? [?/1-19/p/t/a/w/q] q
```

Description File Modification

You can use these procedures to modify the appropriate description file and rebuild the pipe device descriptors for your port directory. The DESCRIPTION FILE MACROS section of the field reference specifies the name of the macro you modify/define in the description files to configure the field. The value used in the define is chosen from the AVAILABLE VALUES specified for the field.

Figure 7-3 Directory Location for Modifying PIPE Description Files



Description File Modification Procedures

Step 1. Change to the `PIPE/<DEVICE>` directory (see **Figure 7-3**).

Step 2. Edit the file `config.des` and read the included comments for more specific information on using the specific description files provided in your software distribution. The `config.des` file contains a list of macro names that can be defined to override the global default values for the configuration fields.

Step 3. Refer to the DESCRIPTION FILE MACRO section in the field reference later in this chapter to determine the macro name you define to configure the target field.

Step 4. Read the comments in `config.des` to determine where to place the define for this macro.

Step 5. Select the value you want to use to configure the field. See the AVAILABLE VALUES section of the field reference data for values or macros that can be used for the definition. Define the macro by entering a definition in the appropriate description files as follows:

```
#define <macro> <value>
```

Step 6. Save the changes and rebuild the pipe device descriptors, entering the following command in the `PIPE/<DEVICE>/DESC` directory:

```
os9make
```

Step 7. Rebuild your boot image to include the new descriptor.

Pipe Device Descriptor Field Reference

This section contains a list of the most commonly configured fields in the pipe device descriptors. Each field entry contains the following information:

- <Field name> - The call name for each field that can be reconfigured in the module.
- EditMod LABELS - `EditMod` menu selections for navigating to the proper field in an `EditMod` session.
- DESCRIPTION FILE MACRO - The macro name you modify/define in the description file.
- DESCRIPTION - A brief description of the field's purpose and use.
- EXAMPLE - An optional example of the description file entry showing how to change the value of this field.
- PORT GENERIC DEFAULT VALUE - The value set in the port generic description file for this field. This is the value the field is assigned when the module is built, unless the appropriate macro has been defined in the port specific description file to override this default value.
- PORT SPECIFIC OVERRIDE VALUE - The value set in the port specific description file for this field. If defined, this is the value the field is assigned when the module is built, overriding the port generic default value.
- AVAILABLE VALUES - Values to which the field can be set through `EditMod` or the description files. In many cases, this data is presented in a table that maps a description of the value to a numeric value appropriate for entry in EditMod, and to a pre-defined macro available for use in the description file.

Module Header Fields

The following section contains the module header fields in the order they appear in the EditMod utility. Defined fields can appear in a different order in the description files.

Table 7-1 Module Header Fields

Field	Description File Macro
_m_group	MH_GROUP
_m_user	MH_USER
mod_name	MH_NAME
m_access	MH_ACCESS
m_tylan	MH_TYLAN
m_attrev	MH_ATTREV
m_edit	MH_EDITION

_m_group

MH_GROUP

EditMod Labels

```
1-module header
1-module owner's group number
```

Description

Group ID of the module's owner. The group number allows people working in the same department or on the same project to share a common identification number.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `PIPE/config.des` (**Figure 7-3**).

Available Values

`0` to `65535`

EditMod Labels

```
1-module header
2-module owner’s user number
```

Description

User ID of the module’s owner. The user number identifies a specific user.

Port Generic Default Value

0 (zero)

Port Specific Override Value

Refer to PIPE/config.des (**Figure 7-3**).

Available Values

0 to 65535

mod_name

MH_NAME

EditMod Labels

```
1-module header
3-module name
```

Description

Contains the module name string.

Port Generic Default Value

`NULL`

Port Specific Override Value

Refer to `PIPE/config.des` (**Figure 7-3**).

Available Values

Any ASCII character string. The string can contain C-style character escapes (such as `\n` and `\012`).

m_access

MH_ACCESS

EditMod Labels

```
1-module header
4-access permissions
```

Description

Defines the permissible module access by its owner or by other users.

Port Generic Default Value

Macro

```
MP_OWNER_READ | MP_OWNER_EXEC | MP_GROUP_READ |
MP_GROUP_EXEC | MP_WORLD_READ | MP_WORLD_EXEC
```

EditMod

```
0x555
```

Port Specific Override Value

Refer to `PIPE/config.des` (**Figure 7-3**).

Available Values

Module access permission values are located in the header file, `module.h`, and are listed in **Table 7-2**.

Table 7-2 `m_access` Available Values

Description	Macro	EditMod
Read permission by owner	`MP_OWNER_READ`	`0x0001`
Write permission by owner	`MP_OWNER_WRITE`	`0x0002`
Execute permission by owner	`MP_OWNER_EXEC`	`0x0004`
Owner permission mask	`MP_OWNER_MASK`	`0x000f`
Read permission by group	`MP_GROUP_READ`	`0x0010`
Write permission by group	`MP_GROUP_WRITE`	`0x0020`
Execute permission by group	`MP_GROUP_EXEC`	`0x0040`
Group permission mask	`MP_GROUP_MASK`	`0x00f0`
Read permission by world	`MP_WORLD_READ`	`0x0100`
Write permission by world	`MP_WORLD_WRITE`	`0x0200`
Execute permission by world	`MP_WORLD_EXEC`	`0x0400`
World permission mask	`MP_WORLD_MASK`	`0x0f00`
All permissions for owner, group, and world	`MP_WORLD_ACCESS`	`0x0777`
System permission mask	`MP_SYSTM_MASK`	`0xf000`

EditMod Labels

```
1-module header
5-type/language
```

Description

Contains the module's type (first byte) and language (second byte). The language codes indicate if the module is executable and which language the run-time system requires for execution, if any.

Port Generic Default Value

Macro

```
    (MT_DATA<<8) + ML_OBJECT
```

EditMod

```
    0x401
```

Port Specific Override Value

Refer to `PIPE/config.des` (**Figure 7-3**).

Available Values

Module type values and language codes are located in the header file, `module.h`, and are listed in **Table 7-3** and **Table 7-4**.

Table 7-3 `m_tylan` Available Module Type Values

Description	Macro	EditMod
Not used (wildcard value in system calls)	`MT_ANY`	`0x0000`
Program module	`MT_PROGRAM`	`0x0001`
Subroutine module	`MT_SUBROUT`	`0x0002`
Multi-module (reserved for future use)	`MT_MULTI`	`0x0003`
Data module	`MT_DATA`	`0x0004`
Configuration data block data module	`MT_CDBDATA`	`0x0005`
Reserved for future use	`0xb-0xa`	`0xb-0xa`
User trap library	`MT_TRAPLIB`	`0x000b`
System module	`MT_SYSTEM`	`0x000c`
File manager module	`MT_FILEMAN`	`0x000d`
Physical device driver	`MT_DEVDRVR`	`0x000e`
Device descriptor module	`MT_DEVDESC`	`0x000f`

Table 7-3 m_tylan Available Module Type Values (continued)

Description	Macro	EditMod
User definable	0x10-0xfe	0x10-0xfe
Module type mask	MT_MASK	0xff00

Table 7-4 m_tylan Available Language Code Values

Description	Macro	EditMod
Unspecified language (wildcard in system calls)	ML_ANY	0x0
Machine language	ML_OBJECT	0x1
Basic I-code (reserved for future use)	ML_ICODE	0x2
Pascal P-code (reserved for future use)	ML_PCODE	0x3
C I-code (reserved for future use)	ML_CCODE	0x4
Cobol I-code (reserved for future use)	ML_CBLCODE	0x5
Fortran	ML_FRTNCODE	0x6
Reserved for future use	0x7-0xf	0x7-0xf
User-definable	0x10-0xfe	0x10-0xfe
Module language mask	ML_MASK	0x00ff

m_attrev

MH_ATTREV

EditMod Labels

```
1-module header
6-revision/attributes
```

Description

Contains the module's attributes (first byte) and revision (second byte).

Port Generic Default Value

Macro

```
   MA_REENT<<8
```

EditMod

```
   0x8000
```

Port Specific Override Value

Refer to `PIPE/config.des` (**Figure 7-3**).

Available Values

Module attribute and revision codes are located in the header file `module.h.`, and are listed in **Table 7-5**.

Note

If two modules with the same name are found in the memory search or are loaded into the current module directory, only the module with the highest revision level is kept. This enables easy substitution of modules for update or correction.

Table 7-5 m_attrev Available Attribute and Revision Values

Description	Macro	EditMod
The module is re-entrant (sharable by multiple tasks).	MA_REENT (shifted left to first byte: MA_REENT<<8)	0x80 (shifted left to first byte: 0x8000)
The module is sticky. A sticky module is not removed from memory until its link count becomes -1 or memory is required for another use.	MA_GHOST (shifted left to first byte: MA_GHOST<<8)	0x40 (shifted left to first byte: 0x4000)
The module is a system-state module.	MA_SUPER (shifted left to first byte: MA_SUPER<<8)	0x20 (shifted left to first byte: 0x2000)
User-definable revision number	0x0-0xfe	0x0-0xfe
Module attribute mask	MA_MASK	0xff00
Module revision mask	MR_MASK	0x00ff

m_edit

MH_EDITION

EditMod Labels

```
1-module header
7-edition
```

Description

Indicates the software release level for maintenance. OS-9 does not use this field. Whenever a program is revised (even for a small change), increase this number. We recommend internal documentation within the source program be keyed to this system.

Port Generic Default Value

`1`

Port Specific Override Value

Refer to `PIPE/config.des` (**Figure 7-3**).

Available Values

`0` to `65535`

Device Descriptor Data Definition Fields

The following section contains the device descriptor data definition fields in the order they appear during an interactive EditMod session. Defined fields can appear in a different order in config.des.

Table 7-6 Device Descriptor Data Definition Fields

Field	Description File Macro
dd_port	PORTADDR
dd_lun	LUN
dd_pd_size	PD_SIZE
dd_type	DD_TYPE
dd_mode	DD_MODE
fmgr_name	FMGR_NAME
drvr_name	DRVR_NAME
dd_class	DD_CLASS

dd_port

PORTADDR

EditMod Labels

```
2-device descriptor data definitions
1-device port address
```

Description

Absolute physical address of the hardware controller. This is the address of the device on the bus. This is the lowest address the device has mapped. Port address is hardware dependent.

Macro Example

```
#define PORTADDR    0xfffe4000
```

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `PIPE/config.des` (**Figure 7-3**).

Available Values

`0` to `4294967295`

EditMod Labels

```
2-device descriptor data definitions
2-logical unit number
```

Description

Distinguishes between the different devices driven from a unique controller. Each unique number represents a different logical unit static storage area.

Macro Example

```
#define LUN    2
```

Port Generic Default Value

0 (zero)

Port Specific Override Value

Refer to PIPE/config.des (**Figure 7-3**).

Available Values

0 to 65535

dd_pd_size

PD_SIZE

EditMod Labels

```
2-device descriptor data definitions
3-path descriptor size
```

Description

Size of the path descriptor. IOMAN uses this value when it allocates a path descriptor.

Port Generic Default Value

`108`

Port Specific Override Value

Refer to `PIPE/config.des` (**Figure 7-3**).

Available Values

`0` to `65535`

EditMod Labels

```
2-device descriptor data definitions
4-device type
```

Description

Identifies the I/O class of the device.

Port Generic Default Value

Macro

```
DT_PIPE
```

EditMod

```
0x2
```

Port Specific Override Value

Refer to PIPE/config.des (**Figure 7-3**).

Available Values

Device type values are defined in the header file io.h, and are listed in **Table 7-7**.

Table 7-7 dd_type Available Values

Description	Macro	EditMod
Sequential Character File Type	DT_SCF	0x0
Random Block File Type	DT_RBF	0x1
Pipe File Type	DT_PIPE	0x2

Table 7-7 `dd_type` Available Values (continued)

Description	Macro	EditMod
Sequential Block File Type	`DT_SBF`	`0x3`
Network File Type	`DT_NFM`	`0x4`
Compact Disc File Type	`DT_CDFM`	`0x5`
User Communication Manager	`DT_UCM`	`0x6`
Socket Communication Manager	`DT_SOCK`	`0x7`
Pseudo-Keyboard Manager	`DT_PTTY`	`0x8`
Graphics File Manager	`DT_GFM`	`0x9`
PC-DOS File Manager	`DT_PCF`	`0xa`
Non-volatile RAM File Manager	`DT_NRF`	`0xb`
ISDN File Manager	`DT_ISDN`	`0xc`
MPFM File Manager	`DT_MPFM`	`0xd`
Real-Time Network File Manager	`DT_RTNFM`	`0xe`
Serial Protocol File Manager	`DT_SPF`	`0xf`
Inet File Manager	`DT_INET`	`0xa0`
Reserved for Microware Use Only	`17-127`	`0xa1-0x7f`

EditMod Labels

```
2-device descriptor data definitions
5-device mode capabilities
```

Description

Used to check the validity of a caller's access mode byte in `I_CREATE` or `I_OPEN` system calls. If a bit is set, the device can perform the corresponding function. The `S_ISIZE` bit is usually set, because it is handled by the file manager or ignored. If the `S_ISHARE` bit is set, the device is non-sharable. A printer is an example of a non-sharable device.

Port Generic Default Value

Macro

```
S_IREAD | S_IWRITE
```

EditMod

```
0x3
```

Port Specific Override Value

Refer to `PIPE/config.des` (**Figure 7-3**).

Available Values

The file access modes are defined in the header file, modes.h, and located in Table 7-8. The file access permission values are defined in the header file modes.h and in Table 7-9.

Table 7-8 dd_mode Available Values for File Access Modes

Description	Macro	EditMod
Truncate on open	S_ITRUNC	0x0100
Ensure contiguous file	S_ICONTIG	0x0400
Error if file exists on create	S_IEXCL	0x0400
Create file	S_ICREAT	0x0800
Append to file	S_IAPPEND	0x1000
Non-sharable	S_ISHARE	0x4000

Table 7-9 dd_mode Available Values for File Access Permissions

Description	Macro	EditMod
Mask for permission bits	S_IPRM	0xffff
Owner read	S_IREAD	0x0001
Owner write	S_IWRITE	0x0002
Owner execute	S_IEXEC	0x0004

Table 7-9 dd_mode Available Values for File Access Permissions (continued)

Description	Macro	EditMod
Search permission	S_ISEARCH	0x0004
Group read	S_IGREAD	0x0010
Group write	S_IGWRITE	0x0020
Group execute	S_IGEXEC	0x0040
Group search	S_IGSEARCH	0x0040
Public read	S_IOREAD	0x0100
Public write	S_IOWRITE	0x0200
Public execute	S_IOEXEC	0x0400
Public search	S_IOSEARCH	0x0400

fmgr_name

FMGR_NAME

EditMod Labels

```
1-module header
2-device descriptor data definitions
6-file manager name
```

Description

Contains the name string of the file manager module to use.

Port Generic Default Value

`“pipe”`

Port Specific Override Value

Refer to `PIPE/config.des` (**Figure 7-3**).

Available Values

Any ASCII character string. The string can contain C-style character escapes (such as `\n` and `\012`).

EditMod Labels

```
1-module header
2-device descriptor data definitions
7-driver name
```

Description

Contains the name string of the device driver module to use.

Port Generic Default Value

`0` (zero)

Port Specific Override Value

Refer to `PIPE/config.des` (**Figure 7-3**).

Available Values

Any ASCII character string. The string can contain C-style character escapes (such as `\n` and `\012`).

dd_class

DD_CLASS

EditMod Labels

```
2-device descriptor data definitions
8-device class (sequential or random)
```

Description

Used to identify the class of the device, whether it is random or sequential access.

Port Generic Default Value

Macro

```
DC_SEQ
```

EditMod

```
0x1
```

Port Specific Override Value

Refer to `PIPE/config.des` (**Figure 7-3**).

Available Values

Device class available values are defined in the header file, `io.h`, and in **Table 7-10**.

Table 7-10 `dd_class` **Available Values**

Description	Macro	EditMod
Sequential access device	`DC_SEQ`	`0x0001`
Random access device	`DC_RND`	`0x0002`

Pipeman Logical Unit Static Storage

The following section contains the Pipeman logical unit static storage fields in the order they appear during an interactive `EditMod` session. Defined fields can appear in a different order in `config.des`.

Table 7-11 Pipeman Logical Unit Static Storage Fields

Field	Description File Macro
`bufsz`	`BUFSZ`

bufsz

BUFSZ

EditMod Labels

```
3-pipeman logical unit static storage
1-pipe FIFO buffer size
```

Description

Used to define the buffer size of the pipe.

Port Generic Default Value

`256`

Port Specific Override Value

Refer to `PIPE/config.des` (**Figure 7-3**).

Available Values

`0` to `4294967295`

Index

Product Discrepancy Report

To: Microware Customer Support

FAX: 515-224-1352

From:____

Company:____

Phone:____

Fax:______________________Email:___________________

Product Name: OS-9

Description of Problem:

____

____

____

____

____

____

____

____

____

____

Host Platform_______________________________________

Target Platform_____________________________________

RadiSys.

MICROWARE SOFTWARE