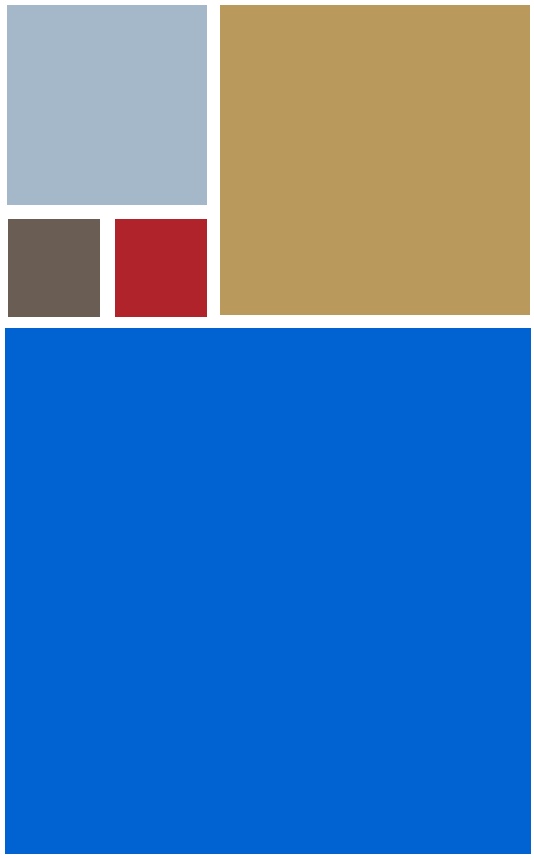


Home

OS-9® for 8xxFADS Board Guide

Version 4.7

RadiSys
THE POWER OF WE

www.radisys.com

Revision A • July 2006

Copyright and publication information

This manual reflects version 4.7 of Microware OS-9.
Reproduction of this document, in part or whole, by any means, electrical, mechanical, magnetic, optical, chemical, manual, or otherwise is prohibited, without written permission from RadiSys Microware Communications Software Division, Inc.

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.

Table of Contents

Chapter 1: Installing and Configuring OS-9®

This chapter describes installing and configuring OS-9® on the Motorola 8xxFADS target board and the following daughterboards: MPC823FADS, MPC850FADS, MPC860FADS, MPC860SARFADS, MPC860TFADS. The following sections are included:

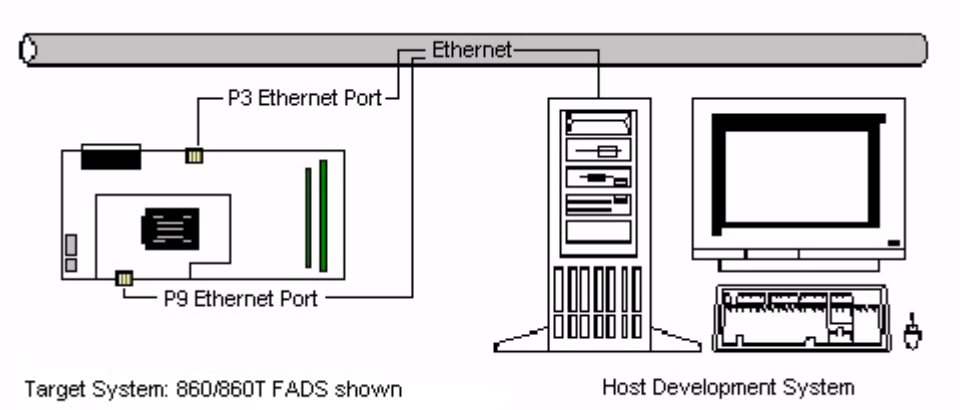
RadiSys.
MICROWARE SOFTWARE

Development Environment Overview

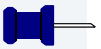
Figure 1-1 shows the development environment for the 860 and 860T FADS boards. The components shown include the minimum required to enable OS-9 to run on the 8xxFADS board. Depending on which 8xxFADS board you are using, however, you will have different capabilities:

- The **860T** board contains a chip that has capability for both 100M and 10M. The default is 100M, which is on the P9 Ethernet port.
- The **860** board contains a chip that has only 10M capability. The default is 10M, which is on the P3 Ethernet port.
- The **850** and **823** boards do not have the P9 Ethernet port; thus, the default is 10M on the P3 port.

Figure 1-1 8xxFADS Development Environment



Requirements and Compatibility



Note

Before you begin, install the ***Microware OS-9 for PowerPC*** CD-ROM on your host PC.

Host Hardware Requirements (PC Compatible)

Your host PC must meet the following minimum requirements:

- Windows 95, 98, ME, 2000, or NT
- 300-400 MB of free disk space
  - An additional 235MB of free disk space is required to run PersonalJava for OS-9.
  - The 8xxFADS Board Level Support Package requires about 100 MB of free disk space.
- 32MB of RAM (64MB recommended)
- ADI - PC board
- 37-wire flat cable included with ADI

Host Software Requirements (PC Compatible)

Your host PC must have the following applications:

- a terminal emulation program (such as Hyperterminal, which comes with Microsoft Windows 95, Windows 98, and Windows NT 4.0)
- MPC8bug or MPC8bug95

Target Hardware Requirements

Your target system requires the following hardware:

- enclosure or chassis with power supply
- display terminal
- LCD screen (optional)
- Ethernet 10BaseT and connecting cables
- RS-232 serial connectors and cables
- minimum of 4MB DRAM/2MB Flash

Special Hardware Considerations

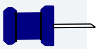
Because not all platforms supported by this package have the same set of peripherals, the example boot image contained within this package is tuned to support as many platforms as possible. After booting OS-9 initially from the sample boot explained in this chapter, you should reconfigure the system to more directly fit your requirements.

For More Information

The ***OS-9 Device Descriptor and Configuration Module Reference*** manual included with your software distribution contains information to help you understand the purpose of each of the modules contained in this distribution and the variety of ways that the software can be configured to meet your needs.

Connecting the Target to the Host

This section describes connecting the target board to the host PC via serial and Ethernet connections.



Note

Your development system must have the following basic elements to complete this procedure:

- a serial connection between the host PC and the target
- an Ethernet connection from your host and target to a network
- a terminal emulation program (for example Hyperterminal)
- an appropriate power supply to the target

Complete the following steps to connect the target to the host:

Step 1. Connect the target's RS-232 COM port to an unused RS-232 COM port on your Host PC using a serial cable.

Step 2. Connect the target board to an Ethernet network. Your Host PC must also be connected to a network.

Step 3. Start Hyperterminal on the Host PC by selecting `Start -> Programs -> Accessories -> Hyperterminal`.

Step 4. Enter a name for your Hyperterminal session.

Step 5. Select an icon for the new Hyperterminal session. A new icon will be created with the name of your session associated with it.

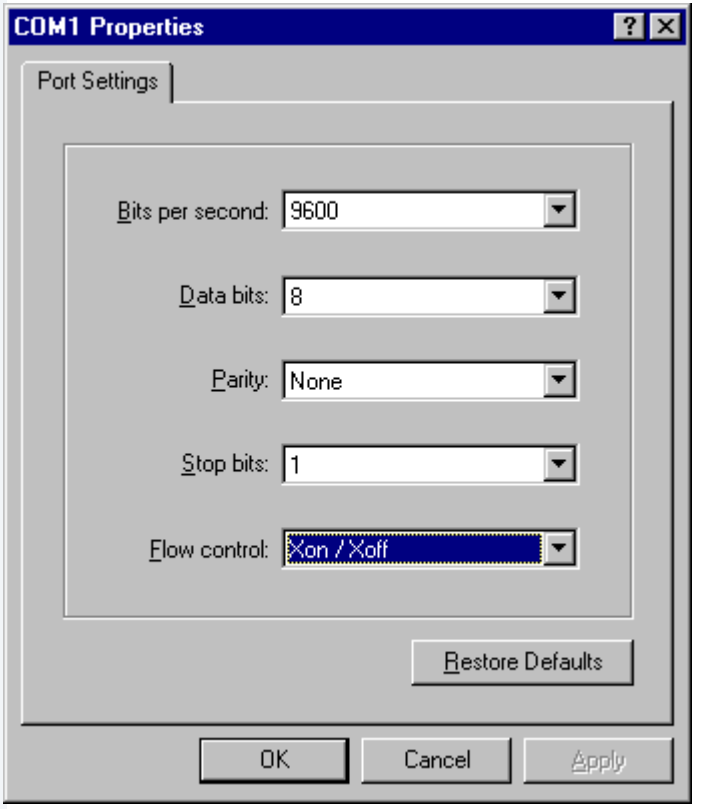
Step 6. Click `OK`.

Step 7. In the **Connect To** dialog box, go to the **Connect using** pull-down menu and enter the communications port to be used to connect to the target system.

Step 8. Click OK.

Step 9. Configure the **Port Settings** tab, as shown in **Figure 1-2**.

Figure 1-2 COM Port Settings



Step 10. Click OK.

Step 11. In the Hyperterminal window, select `File/Properties`. Click on the **Settings** tab and select the following:

```
Terminal Keys
Emulation = Auto Detect
Backscroll Buffer Lines = 500
```

Step 12. Click `OK`.

Step 13. Go to the Hyperterminal menu and select `Call/Connect` from the pull-down menu to establish your terminal session with the target. If you are connected, the bottom left corner of your Hyperterminal screen will display the word *Connected*.

Step 14. Leave the Hyperterminal window open on your desktop (or minimized); you will use the window again later in this procedure.

Building the OS-9 ROM Image

The OS-9 ROM Image is a set of files and modules that collectively make up the OS-9 operating system. The specific ROM Image contents can vary from system to system depending on hardware capabilities and user requirements.

To simplify the process of loading and testing OS-9, the ROM Image is generally divided into two parts: the low-level image, called `coreboot`, and the high-level image, called `bootfile`.

Coreboot

The coreboot image is generally responsible for initializing hardware devices and locating the high-level (or bootfile) image as specified by its configuration. For example from a FLASH part, a harddisk, or Ethernet. It is also responsible for building basic structures based on the image it finds and passing control to the kernel to bring up the OS-9 system.

Bootfile

The bootfile image contains the kernel and other high-level modules (initialization module, file managers, drivers, descriptors, applications). The image is loaded into memory based on the device you select from the boot menu. The bootfile image normally brings up an OS-9 shell prompt, but can be configured to automatically start an application.

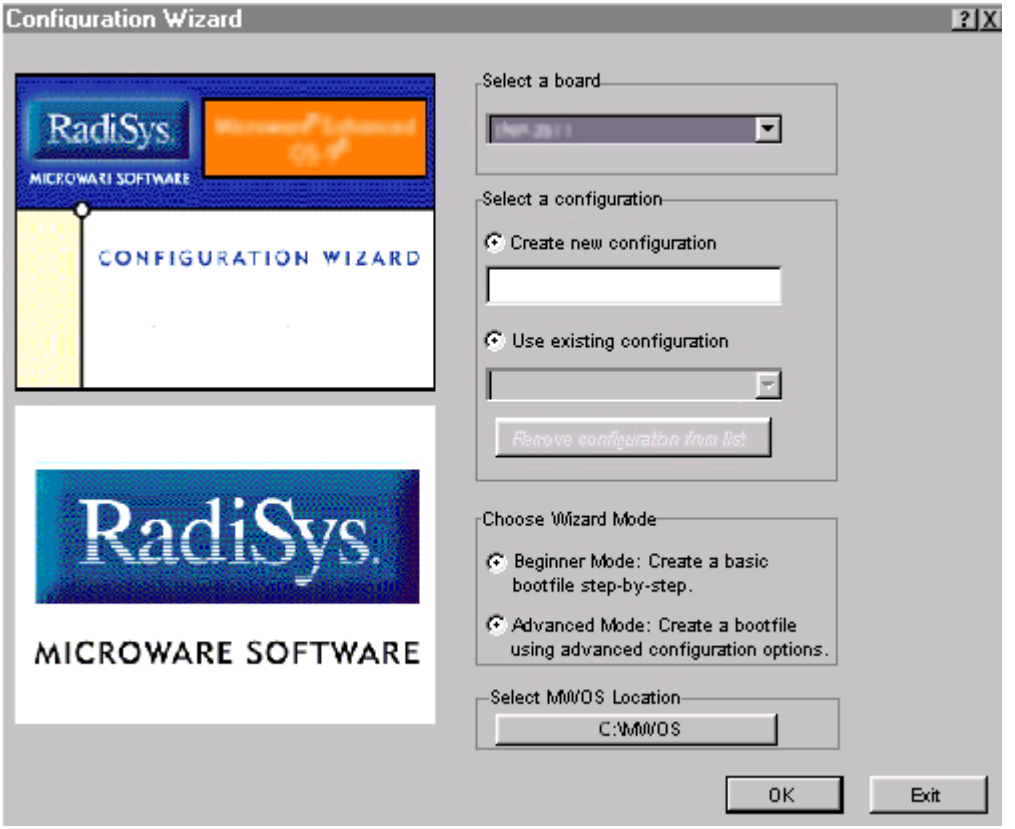
Microware provides a Configuration Wizard to create a coreboot image, a bootfile image, or an entire OS-9 ROM Image. The wizard can also be used to modify an existing image. The Configuration Wizard is automatically installed on your host PC during the OS-9 installation process.

Starting the Configuration Wizard

The Configuration Wizard is the application used to build the coreboot, bootfile, or ROM image. To start the Configuration Wizard, perform the following steps:

Step 1. From the Windows desktop, select `Start` -> `RadiSys` -> `Microware OS-9 for <product>` -> `Configuration Wizard`. You should see the following opening screen:

Figure 1-3 Configuration Wizard Opening Screen

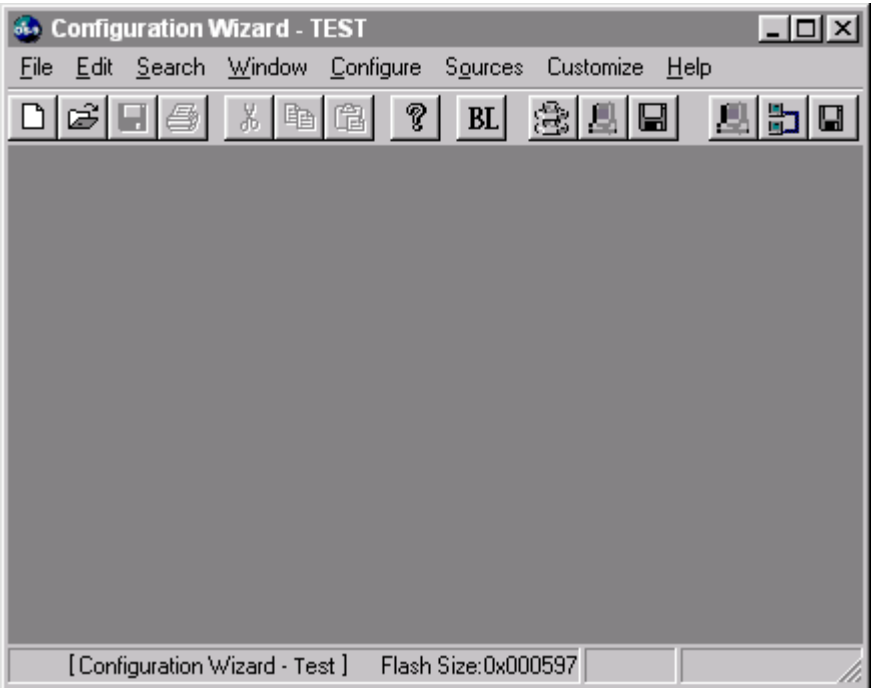


Step 2. Select your target board from the **Select a board** pull-down menu.

Step 3. Select the `Create new configuration` radio button from the **Select a configuration** menu and type in the name you want to give your ROM image in the supplied text box. This names your new configuration, which can later be accessed by selecting the **Use existing configuration** pull down menu.

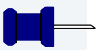
Step 4. Select the `Advanced Mode` radio button from the **Choose Wizard Mode** field and click `OK`**.** The Wizard's main window is displayed. This is the dialog from which you will proceed to build your image. An example is shown in **Figure 1-4**.

Figure 1-4 Configuration Wizard Main Window



Creating and Configuring the ROM Image

This section describes how to use the Configuration Wizard to create and configure your OS-9 ROM image.



Note

This section provides an example of an OS-9 ROM image successfully built on a Host PC and transferred to an 823FADS target board. You may have to modify your selections depending on your application.

Select System Type

Configure system type options by selecting `Configure -> Sys -> Select System Type` from the **Main Configuration** window.

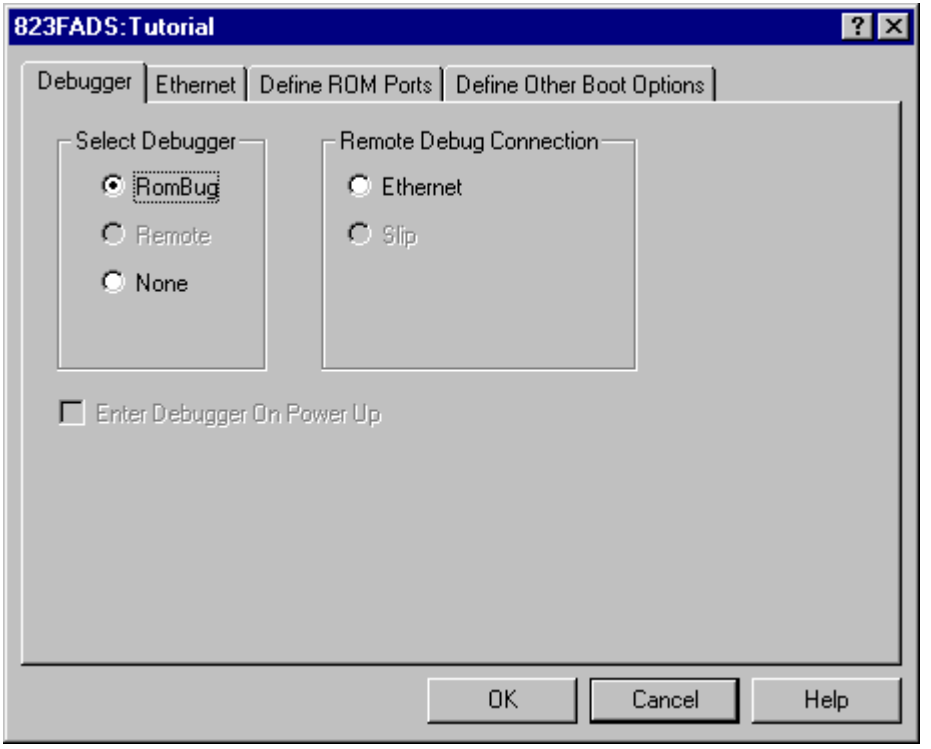
For the 823FADS target board, you can bypass this option and use the default settings.

Configure Coreboot Options

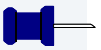
Step 1. From the **Main Configuration** window, select `Configure -> Coreboot -> Main configuration`.

Step 2. Select the **Debugger** tab. The following window is displayed.

Figure 1-5 Coreboot Configuration—Debugger Tab



Step 3. Under **Select Debugger**, select `RomBug`. This sets Ethernet as the method for user state debugging. Select `None` if you do not want to debug your program.



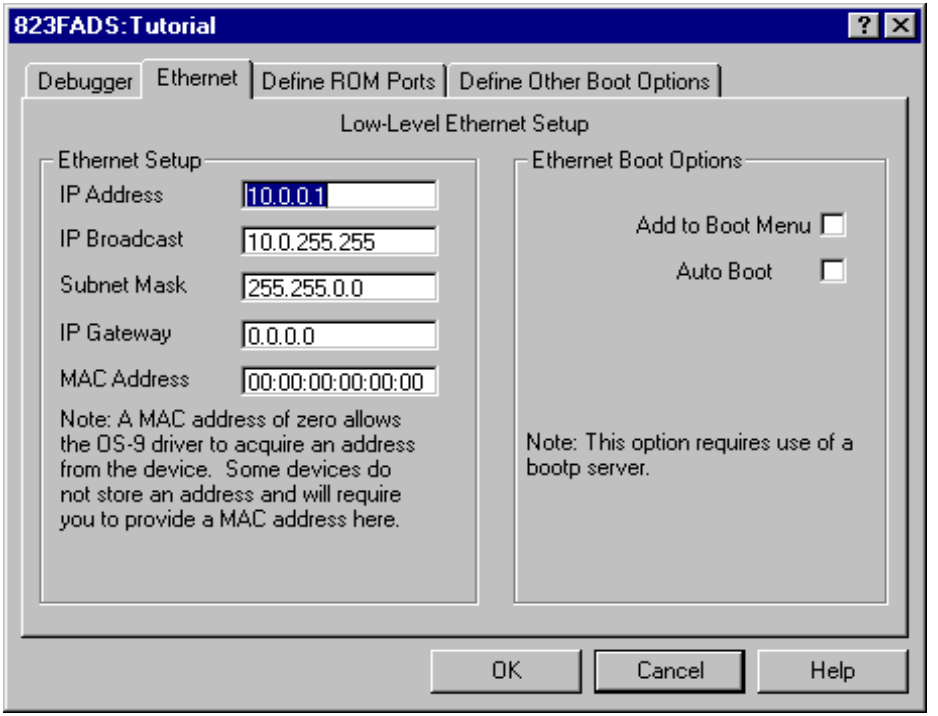
Note

To perform system state debugging, select `Ethernet` under **Remote Debug Connection**. If you set Ethernet as the method for system state debugging, you will not be able to perform user state debugging via Ethernet.

For system state debugging, you must also set the parameters in the **Ethernet** tab of the coreboot configuration.

Step 4. Select the **Ethernet** tab. The following window is displayed.

Figure 1-6 Coreboot Configuration—Ethernet Tab



Step 5. Enter the appropriate Ethernet setup information.

Note

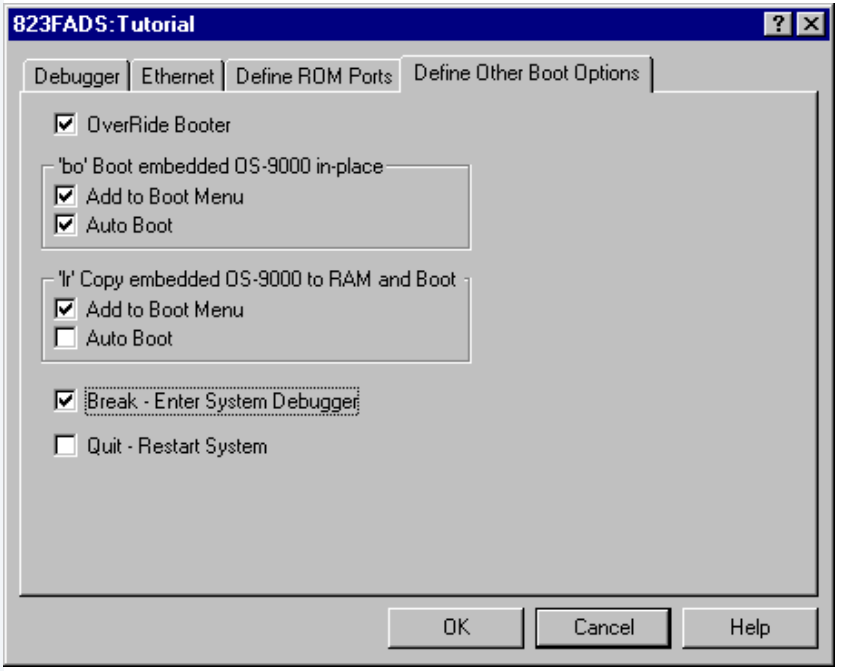
Complete the Ethernet setup information only if you intend to boot your system over a network or if you plan to use system state debugging.

Note

The addresses shown in **Figure 1-6** are for demonstration only. Contact your network administrator to obtain your Ethernet setup information.

Step 6. Select the **Define Other Boot Options** tab. The following window is displayed.

Figure 1-7 Coreboot Configuration—Define Other Boot Options Tab



Step 7. Select `Break-Enter System Debugger`.

Step 8. Click `OK` and return to the **Main Configuration** window.

Configure System Options

Configure system options by selecting `Configure -> Bootfile -> Configure System Options` from the **Main Configuration** window. You can bypass this option and use the default settings.

To use the target board across a network, you must enable the Ethernet network settings.

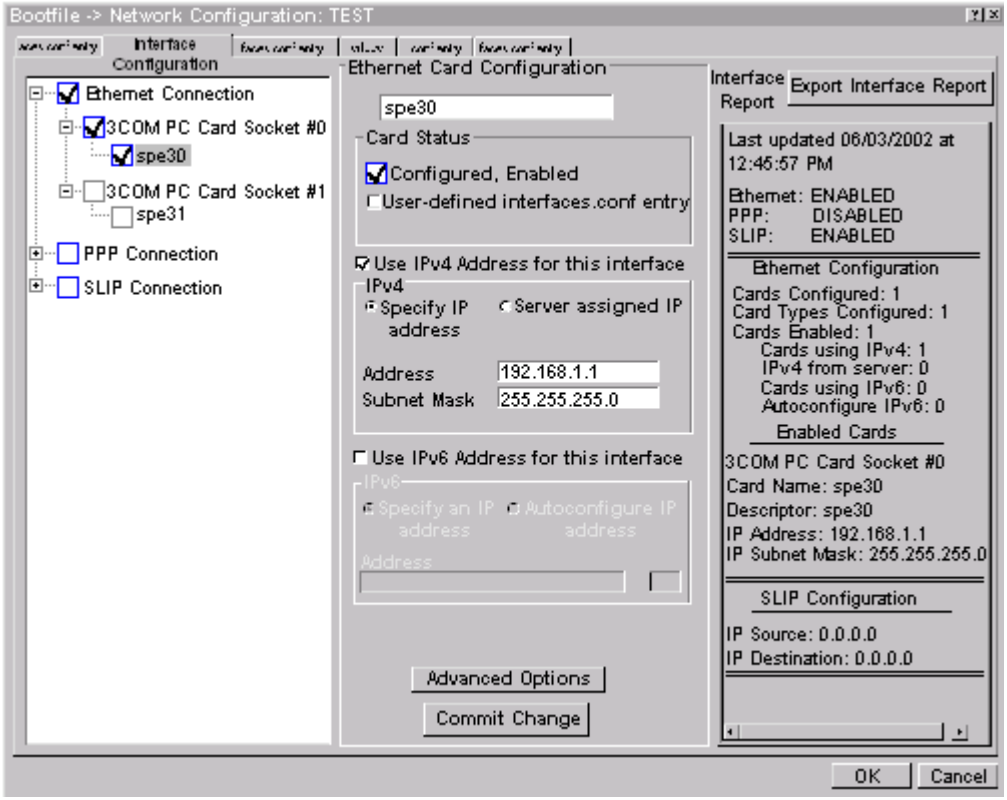
Network Configuration

To use the target board across a network, complete the following steps:

Step 9. If you want to use the target board across a network, you will need to configure the Ethernet settings within the Configuration Wizard. To do this, select `Configure -> Bootfile -> Network Configuration` from the Wizard's main menu.

Step 10. From the **Network Configuration** dialog, select the `Interface Configuration` tab. From here you can select and enable the interface. For example, you can select the appropriate Ethernet card from the list of options on the left and specify whether you would like to enable IPv4 or IPv6 addressing. **Figure 1-8** shows an example of the **Interface Configuration** tab.

Figure 1-8 Bootfile -> Network Configuration -> Interface Configuration



For More Information

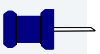
To learn more about IPv4 and IPv6 functionalities, refer to the ***Using LAN Communications*** manual, included with this product CD.

For More Information

Contact your system administrator if you do not know the network values for your board.

Step 11. Once you have made your settings in the **Network Configuration** dialog, click `OK`.

Step 12. Select the `DNS Configuration` tab. The following window is displayed. More than one DNS server can be added in this dialog box. If your network does not use DNS, click `Disable DNS` and move to the **Gateway** tab. If you have DNS available, click `Enable DNS` and type your host name and domain.



Note

You add DNS IP addresses by clicking on the box directly under **DNS Server Search Order** and typing the IP address. Click the `Add` button when complete.

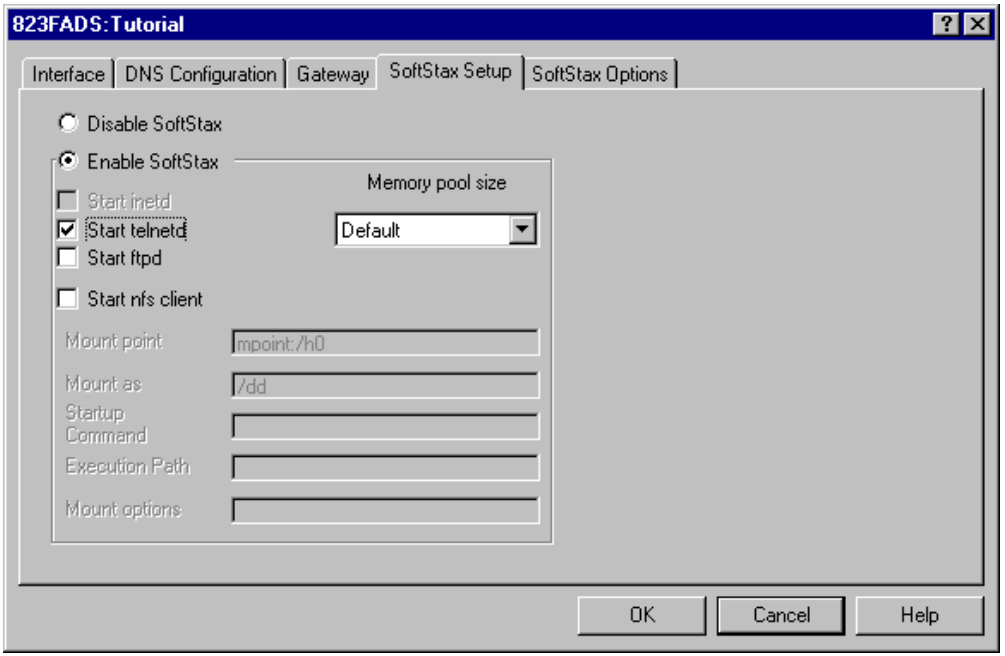
More than one DNS server can be added by repeating these steps.

Step 13. Select the **Gateway** tab. Add new gateway addresses by clicking on the box and typing in the gateway name. Click the `Add` button when complete.

Step 14. Select the `SoftStax® Setup` tab. The following window is displayed.

The options below represent daemons that can be automatically started if you want to FTP or telnet from a PC to the OS-9 target. Start NFS Client enables you to remote mount the target. For this demonstration, you will telnet to the target and establish a sender window and a receiver window.

Figure 1-9 Bootfile Configuration—SoftStax® Setup Tab



Step 15. Click `Enable SoftStax`.

Step 16. Click `Start telnetd`. (The only checked box on this tab should be the Start telnetd box.)

Step 17. Click `OK`.

Step 18. Select the **SoftStax Options** tab.

The **SoftStax Options** tab enables you to include networking utilities in the ROM image. By default, `ftp`, `hostname`, `ping`, and `netstat` are included. You can add other utilities as desired.

For More Information

The networking utilities are described in the ***Using LAN Communications*** manual.

Step 19. Click `OK` at the bottom of the **Network Configuration** menu to complete network configuration and return to the **Main Configuration** window.

Disk Configuration

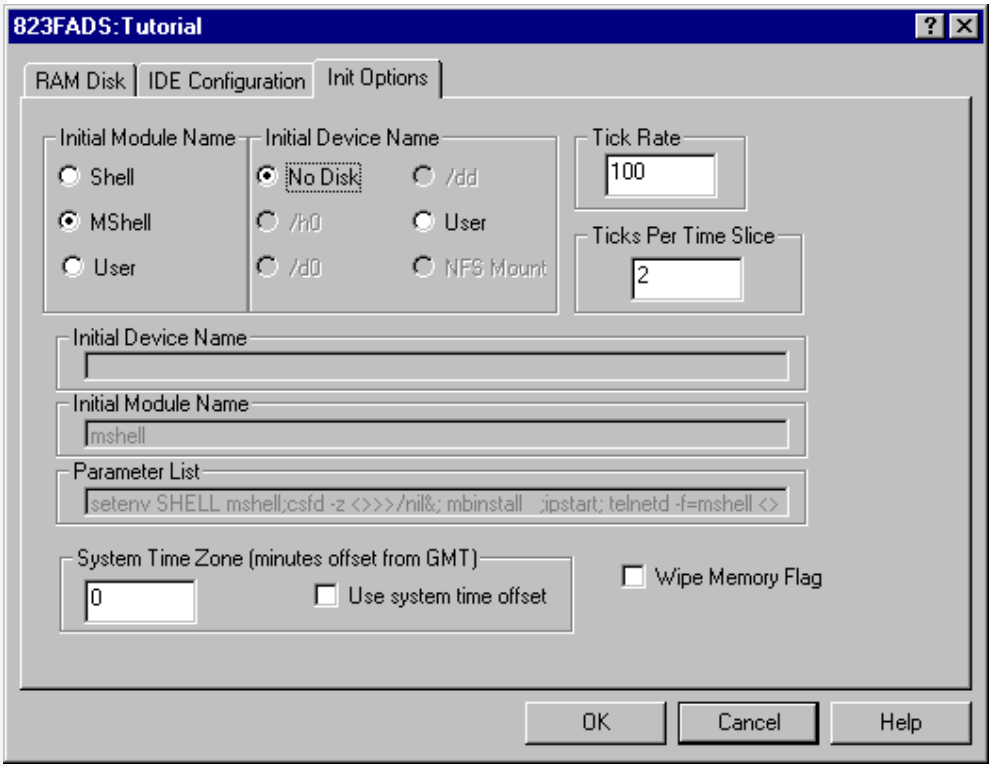
Step 1. From the main configuration window, select `Configure -> Bootfile -> Disk Configuration`.

The Disk Configuration options include the following three tabs:

- The **RAM Disk** tab enables you to create a RAM disk of any size for loading modules onto the target.
- The **IDE Configuration** tab enables you to configure IDE drives for the target.
- The **Init Options** tab sets the configuration for OS-9 to initialize itself on the target.

Step 2. Select the **Init Options** tab. The following window is displayed.

Figure 1-10 Bootfile Configuration—Init Options Tab



Step 3. Select the `Mshell` option for the initial module name. This causes OS-9 to start a console shell usable from your terminal window. Initial **Device Name** should be selected as `No Disk`.

The tick rate is `100` and ticks per timeslice is set to `2`. If you look at the **Parameter List** box, you see the commands that OS-9 executes upon system start-up.

Step 4. Click `OK` to return to the **Main Configuration** window.

Build Image

Complete the following steps to build the target board image.

Step 1. From the **Main Configuration** window, select `Configure -> Build Image`. The **Master Builder** window appears.

Step 2. Select the **Coreboot + Bootfile** option.

Step 3. Select the `ROM Utility Set`, `User State Debugging Modules`, and the `SoftStax (SPF) Support` boxes under the **Include** options.

Step 4. Click `Build`. It should display progress information and show the statistics of the image just created.

Step 5. Click `Save As`. The rom and rom.s files are created in the following directory:

```
MWOS/OS9000/821/PORTS/8XXFADS/BOOTS/INSTALL/PORTBOOT
```

At this point you can either close the Configuration Wizard or leave it open for use in modifying your ROM Image. If you choose to close, you can save your configuration settings for later use.

Transferring the ROM Image to the Target

Complete the following steps to transfer your ROM image to the reference board.

For More Information

This process uses the MPC8BUG utility from Motorola. For more information about MPC8BUG, refer to the ***MPC8xxFADS User's Manual*** the appropriate user's manual for your processor.

Step 1. Connect the reference board to your host PC through the ADI card and cable.

Step 2. After making the connection, reboot the target machine by turning the power switch off and on.

Step 3. At the DOS prompt on your host PC, type the following command:

`mpc8bug a b` (if using Windows NT)

`mpc8bug95 a b` (if using Windows 95 or Windows 98)

where a is the ADI - number (between 0-3)

and b is the ADS board address (between 0-7)

Your screen should display the following message:

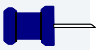
```
Microsoft(R) Windows NT(TM)
(C) Copyright 1985-1996 Microsoft Corp.

U:\mwos\OS9000\821\PORTS\8XXFADS\BOOTS\INSTALL\PORTBOOT>
mpc8bug 0 0
mpc8bug version 1.5  May 18 98
Copyright 1998 Motorola, Inc.  All Rights Reserved.
```

```
 Initializing memory controller and UPM for 50MHZ
  DRAM delay set to 60ns
  DRAM size set to 4Mbytes
Executing .mpctcl.cfg file from c:\mpc8bug directory.
Executing .mpc8xx.cfg file from c:\mpc8bug directory.
Executing .mpc823.cfg file from c:\mpc8bug directory.
Executing .mpcsdram.cfg file from c:\mpc8bug directory.
```

Note

The example above is for an MPC823FADS reference board. The displays for other screens will vary slightly.



Note

Use the following instructions to determine the values of a and b.

mpc8bug ADI ADS [-nosem] [object_file] [-c command_file] [-o log_file]

ADI- a number in the range 0-3.
On a Sparcstation: The number of the SBus slot of the ADI card.
On a PC: address of the port of the ADI card divided by 0x100.

ADS- address of ADS board (0-7). (determined by dip switches on the board.)

nosem	suppress semaphore allocation upon activation. (By default, allows only one debugger to be activated per each ADI port.) This argument is not applicable in IBM-compatible PCs.
object_file	name of a program to load.
command_file	name of a file containing debugger commands.
log_file	name of a file to which session history will be logged.
Example	mpc8bug 1 7

These instructions were taken from the ***MPC8BUG Software Users Manual***.

The prompt on your Windows host PC should read `F823bug>`. If your prompt does not display, or displays something else (such as 8xxFADS>), exit MPC8BUG by typing `exit` and return to step one.

Your prompt can also read `F850bug>`, `F860bug>`, `F860SARbug>`, or `F860Tbug>` depending on the reference board you are using.

Step 4. Reset the hardware by typing `reset :h`. The example below indicates what your screen should display:

```
f823Bug> reset :h
 Initializing memory controller and UPM for 50MHZ
  DRAM delay set to 60ns
  DRAM size set to 4Mbytes
```

Step 5. Send the `Srecord` file from the host PC to the reference board by typing `loadf rom.S 0`. The value of b is the same as in Step three. Your Host PC screen should display the following message:

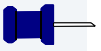
```
f823Bug> loadf rom.S 0
loadf: Loading Srecords file . . .
Loading flash mapped sections to ram memory buffer:
Programming flash :00200000 bytes  at 02800000-029fffff
Flash programming completed

Loading ram mapped sections to ram memory:
Loading block : at 02800000
Entry point (IP) set to 00000000
Heap start address set to 02a00000
f823Bug>
```

Step 6. Power the target machine off and remove the cable.

Step 7. Type `quit` to quit the debugger.

Step 8. Power the target machine on.

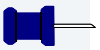


Note

An alternate way to perform steps six through eight is to type the following at the prompt:

```
f823Bug>reset :h
f823Bug>rms der 0
f823Bug>go 2800100
```

After booting, your terminal should display the prompt.



Note

The boot menu can have different selections, depending upon your selections using the Configuration Wizard.

Step 9. Type the `bo` command to select booting OS-9 in-place. The example below details what your screen should display:

```
OS-9000 Bootstrap for the PowerPC(tm)

Now trying to Override autobooters.

BOOTING PROCEDURES AVAILABLE ---------- <INPUT>

Boot embedded OS-9000 in-place -------- <bo>
Copy embedded OS-9000 to RAM and boot - <lr>
Enter ROM Debugger -------------------- <break>
Restart the System -------------------- <q>

Select a boot method from the above menu: bo

Now searching memory ($02840000 - $029fffff) for an
OS-9000 Kernel...

An OS-9000 kernel was found at $02840000
A valid OS-9000 bootfile was found.
+3
[1]$
```

Creating a Startup File

When the Configuration Wizard is set to use a hard drive, or another fixed drive such as a PC Flash Card, as the default device, it automatically sets up the init module to call the `startup` file in the `SYS` directory in the target (For example: `/h0/SYS/startup`, `/mhc1/SYS/startup`). However, this directory and file will not exist until you create it. To create the startup file, complete the following steps:

Step 1. Create a `SYS` directory on the target machine where the `startup` file will reside (for example: `makdir /h0/SYS`, `makdir /dd/SYS`).

Step 2. On the host machine, navigate to the following directory:

`MWOS/OS9000/SRC/SYS`

In this directory, you will see several files. The files related to this section are listed below:

- `motd`: Message of the day file
- `password`: User/password file
- `termcap`: Terminal description file
- `startup`: Startup file

Step 3. Transfer all files to the newly created `SYS` directory on the target machine. (You can use Kermit, or FTP in ASCII mode to transfer these files.)

Step 4. Since the files are still in DOS format, you will be required to convert them into the OS-9 format with the `cudo` utility. The following command is an example:

```
cudo -cdo password
```

This will convert the `password` file from DOS to OS-9 format.

For More Information

For a complete description of all the `cudo` command options, refer to the ***Utilities Reference Manual*** located on the Microware OS-9 CD.

Step 5. Since the command lines in the startup file are system-dependent, it may be necessary to modify this file to fit your system configuration. It is recommended that you modify the file before transferring it to the target machine.

Example Startup File

Below is the example startup file as it appears in the `MWOS/OS9000/SRC/SYS` directory:

```
-tnxnp
tmode -w=1 nopause
*
*OS-9 - Version 3.0
*Copyright 2001 by Microware Systems Corporation
*The commands in this file are highly system dependent and
*should be modified by the user.
*
*setime </term              ;* start system clock
setime -s                   ;* start system clock
link mshell csl             ;* make "mshell" and "csl" stay in memory
* iniz r0 h0 d0 t1 p1 term  ;* initialize devices
* load utils                ;* make some utilities stay in memory
* tsmon /term /t1 &         ;* start other terminals
list sys/motd
setenv TERM vt100
tmode -w=1 pause
mshell<>>>/term -l&
```

For More Information

Refer to the **Making a Startup File** section in Chapter 9 of the ***Using OS-9*** manual for more information on startup files.

Optional Procedures

The following section provides optional procedures you can perform after installing and configuring OS-9 on your board.

Preliminary Testing

Once you have established an OS-9 prompt on your target system, you can perform the following procedures to test your system:

Step 1. Type `mdir` at the prompt.

`mdir` displays all the modules in memory. You may have to press the space bar to scroll the output.

Step 2. Type `procs` at the prompt.

`procs` displays the processes currently running in the system.

Step 3. Test the networking on your system.

Select a host on the Ethernet network and run the `ping` utility. The following display shows a successful `ping` to a machine called `solkanar`.

```
$ ping solkanar
PING solkanar.microware.com (172.16.2.51): 56 data bytes
64 bytes from 172.16.2.51: ttl=128 time=0 ms
```

Step 4. Test `telnet`.

Select a host machine that allows telnet access and try the OS-9 `telnet` utility. The following display shows a successful `telnet` to a machine called delta.

```
$ telnet delta
Trying 172.16.1.40...Connected to delta.microware.com.
Escape character is '^]'.
capture closed.

OS-9/68K V3.0.3 Delta VME177 - 68060 98/12/24 14:41:51
User name?: curt
Password:
Process #101 logged on 98/12/24 14:41:56
```

```
Welcome!
*************************************************************
*       WELCOME TO DELTA - THE :OS-9 68K: MACHINE *
```

Chapter 2: Board Specific Reference

This chapter contains information that is specific to the Motorola 8xxFADS reference boards. It contains the following sections:

- **Boot Options**
- **OS-9 Vector Mappings**

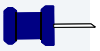
For More Information

For general information on porting OS-9, see the ***OS-9 Porting Guide***.

RadiSys.
MICROWARE SOFTWARE

Boot Options

Select your boot device menu options using the Configuration Wizard. For each boot device option, you can select whether you want it to be displayed on a boot menu, set up to autoboot, or both. The autoboot option enables the device selected to automatically boot up the high-level bootfile, bypassing the boot device menu.



Note

When using the Configuration Wizard, you should select only one device for autoboot on your system.

Following is an example of the Boot menu displayed in the terminal emulation window (using Hyperterminal):

```
OS-9000 Bootstrap for the PowerPC(tm)

Now trying to Override autobooters.

BOOTING PROCEDURES AVAILABLE ----- <INPUT>

Scan SCSI devices ---------------- <ioi>
Boot FDC floppy ------------------ <fd>
Boot from PC-Floppy -------------- <pf>
Boot from Teac SCSI floppy drive - <fs>
Boot from SCSI PC-Floppy --------- <pfs>
Boot from Viper tape drive ------- <vs>
Boot over Ethernet --------------- <eb>
Boot from SCSI(SCCS) hard drive -- <hs>
Boot embedded OS-9000 in-place --- <bo>
Enter system debugger ------------ <break>
Restart the System --------------- <q>

Select a boot method from the above menu:
```

Your boot option selections in the Configuration Wizard determine which modules are included in the coreboot image. **Table 2-1** lists some of the supported boot devices for OS-9.

Table 2-1 Supported Boot Methods

Type of Boot	Description
Boot from RBF hard disk	Boot from a standard SCSI hard disk (`hs`).
Floppy Disk	Boot from floppy disk. You must select if the floppy is controlled by a Random Block File System (RBF) (`fd` or `fs`) or PC File System (`pf` or `pfs`).
Boot embedded OS-9 in-place	Boot OS-9 from FLASH (`bo`).
Copy embedded OS-9 to RAM and Boot	Copy OS-9 from FLASH (if stored there) to RAM and boot (`lr`).

OS-9 Vector Mappings

This section contains the vector mappings and dual-port RAM mappings for the 8xxFADS processors.

The system modules `siuirq` and `cpicirq` map interrupts coming from the SIU and CPM into the OS-9 vector table according to the following mappings.

SIU (System Interface Unit) vectors are mapped starting at vector 0x40 in the order shown in the following table.

Table 2-2 System Interface Unit Vectors

Vector	Source
0x40	IRQ0
0x41	Level 0
0x42	IRQ1
0x43	Level 1
0x44	IRQ2
0x45	Level 2
0x46	IRQ3
0x47	Level 3
0x48	IRQ4
0x49	Level 4 (CPIC)
0x4a	IRQ5

Table 2-2 System Interface Unit Vectors (continued)

Vector	Source
0x4b	Level 5
0x4c	IRQ6
0x4d	Level 6
0x4e	IRQ7
0x4f	Level 7

CPM (Communications Processor Module) vectors are mapped starting at vector 0x50 in the order shown in **Table 16-43** of the ***8xxFADS User's Manual***, and in the following table.

Table 2-3 Communications Processor Module Vectors

Vector	Source
0x50	Error
0x51	Parallel I/O--PC4
0x52	Parallel I/O--PC5
0x53	SMC2/PIP
0x54	SMC1
0x55	SPI
0x56	Parallel I/O--PC6
0x57	Timer 4

Table 2-3 Communications Processor Module Vectors (continued)

Vector	Source
0x58	Reserved
0x59	Parallel I/O--PC7
0x5a	Parallel I/O--PC8
0x5b	Parallel I/O--PC9
0x5c	Timer 3
0x5d	Reserved
0x5e	Parallel I/O--PC10
0x5f	Parallel I/O--PC11
0x60	I2C
0x61	RISC Timer Table
0x62	Timer 2
0x63	Reserved
0x64	IDMA2
0x65	IDMA1
0x66	SDMA Channel Bus Error
0x67	Parallel I/O--PC12
0x68	Parallel I/O--PC13

Table 2-3 Communications Processor Module Vectors (continued)

Vector	Source
0x69	Timer 1
0x6a	Parallel I/O--PC14
0x6b	SCC4
0x6c	SCC3
0x6d	SCC2
0x6e	SCC1
0x6f	Parallel I/O--PC15

Dual-port RAM Mapping

The MPC8xx processors include at least 5120 bytes of dual-port RAM for buffer descriptor and microcode use. Since the high and low-level drivers both use this area and must agree on their usage of it, the following locations have been reserved for the following uses:

Table 2-4 Dual Port RAM Use Map

Offset into DPRAM	Use
0x0 - 0x0f	SCC1
0x10 - 0x1f	SCC2
0x20 - 0x2f	SCC3
0x30 - 0x3f	SCC4

Table 2-4 Dual Port RAM Use Map (continued)

Offset into DPRAM	Use
0x40 - 0x4f	SMC1
0x50 - 0x5f	SMC2
0x60 - 0xff	reserved
0x100 - 0x17f	Ethernet
0x180 - 0x200	reserved

Appendix A: Board Specific Modules

This appendix contains lists of high and low-level modules. The following sections are included:

- **Low-Level System Modules**
- **High-Level System Modules**
- **Common System Modules List**

RadiSys.
MICROWARE SOFTWARE

Low-Level System Modules

The following low-level system modules are tailored specifically for the PowerPC MPC8xxFADS platform. The functionality of many of these modules can be altered through changes to the configuration data module (`cnfgdata`). These modules are located in the following directory:

`MWOS/OS9000/821/PORTS/8XXFADS/CMDS/BOOTOBJS/ROM`

`cnfgdata`	provides low-level configuration data including configuration of a serial console.
`confgfunc`	retrieves configuration parameters from the `cnfgdata` module.
`conscnfg`	retrieves the name of the low-level console driver from the `cnfgdata` module.
`iosmc`	provides console services for the SMC UART on the 823/850/860.
`ll860t`	provides network driver services for the 860T fast Ethernet port.
`llquicc`	provides network driver services for the 823/850/860 Ethernet port
`portmenu`	retrieves a list of configured booter names from the ROM `cnfgdata` module.
`romcore`	gives bootstrap code.
`romstart`	is a vector table.
`tbtimer`	provides polling timer services using the `tblo` and `tbhi` registers in the 8xxFADS processors.
`usedebug`	is a debugger configuration module.

High-Level System Modules

The following OS-9 system modules are tailored specifically for your MPC8xxFADS platform. Unless otherwise specified, each module can be found in a file of the same name in the following directory:

`<MWOS>/OS9000/821/PORTS/8XXFADS/CMDS/BOOTOBJS`

`cpicirq`	provides the communications processor pic module.
`llqd`	disables `llquicc`.
`lltd`	disables `ll860t`.
`picsub`	provides interrupt enable and disable routines to handle platform specific interrupt controller issues for device drivers. This module is called by all drivers and should be included in your bootfile.
`rb1003`	provides support for IDE and EIDE drives up to 4GB. Many descriptors are provided for use with this driver. Among the descriptors provided are several modules named `h0` and `dd`. These descriptors are contained in files of unique names and located in the following directory: `MWOS/OS9000/821/PORTS/8XXFADS/CMDS/BOOTOBJS/DESC/RB1003`
`rtc821`	provides OS-9 access to the real time clock. In this release, `rtc821` is the name of the ticker regardless of the CPU in use on your platform.
`sc16550`	provides support for the external 16550 serial ports. This driver is used to drive the console over the com1 port in the sample boots provided in the package.
`sccpm`	provides support for the CPM SMC and SCC UARTS serial port.

`siuirq`	provides the system interface unit pic module.
`tk821pit`	provides the system ticker based on the SIU periodic interrupt timer.
`tkcpm`	provides the system ticker based on the CPM general purpose timer.
`tkdec`	provides the system ticker based on the PowerPC decrementer.
`nodisk`	indicates no default device is to be used.
`cdb`	is a MAUI® configuration description block.
`gfx`	is a MAUI graphics descriptor.
`gx_lcd821`	is a MAUI graphics driver module.
`atm`	provides ATM network services.
`inetdb`	is a data module containing Internet configurations.
`inetdb2`	is a data module containing Internet configurations.
`rpcdb`	is an NFS/RPC database module.
`sp823`	is an Ethernet driver.
`sp860sar`	is an Ethernet driver.
`sp860t`	is an Ethernet driver.
`spfe0`	is an Ethernet descriptor.
`spqe0`	is an Ethernet descriptor.

Common System Modules List

The following low-level system modules provide generic services for OS9000 modular ROM. They are located in the following directory:

`MWOS/OS9000/PPC/CMDS/BOOTOBJS/ROM`

Table 2-5 Common System Modules List

Module	Description
`bootsys`	provides booter services.
`console`	provides high-level I/O hooks into low-level console serial driver.
`dbgentry`	provides hooks to low-level debugger server.
`dbgserv`	is a debugger server module.
`excption`	is a service module.
`fdc765`	provides PC style floppy support.
`fdman`	is a target-independent booter support module providing general booting services for RBF file systems.
`flboot`	is a SCSI floptical drive disk booter.
`flshcach`	provides the cache flushing routine.
`fsboot`	is a SCSI TEAC floppy disk drive booter.
`hlproto`	allows user-state debugging.
`hsboot`	is a SCSI hard disk driver booter.

Table 2-5 Common System Modules List (continued)

Module	Description
`ide`	provides target-specific standard IDE support, including PCMCIA ATA PC cards.
`iovcons`	is a hardware independent virtual console driver that provides a telnetd-like interface to the low-level system console.
`llbootp`	is a target-independent BOOTP protocol booter module.
`llip`	is a target-independent internet protocol module.
`llkermit`	is a kermit booter (serial down loader).
`llslip`	is a target-independent serial line internet protocol module. This modules uses the auxiliary communications port driver to perform serial I/O
`lltcp`	is a target-independent transmission control protocol module.
`lludp`	is a target-independent user datagram protocol modules.
`notify`	coordinates use of low-level I/O drivers in system and user-state debugging.
`override`	enables overriding of the autobooter. If the space bar is pressed within three seconds after booting the target, a boot menu is displayed. Otherwise, booting proceeds with the first autobooter.
`parser`	parses key fields from the `cnfgdata` module and the user parameter fields.

Table 2-5 Common System Modules List (continued)

Module	Description
pcman	is a target-independent booter support module providing general booting services for PCF file systems (PC FAT file systems).
protoman	is a target-independent protocol module manager. This module provides the initial communication entry points into the protocol module stack.
restart	restarts boot process.
romboot	locates the OS-9 bootfile in ROM, FLASH, NVRAM.
rombreak	enables break option from the boot menu.
rombug	is a debugger client module.
scsiman	is a target-independent booter support module that provides general SCSI command protocol services
sndp	is a target-independent system-state network debugging protocol module. This module acts as a debugging client on the target, invoking the services of dbgserv to perform debug tasks.
srecord	receives a Motorola S-record format file from the communications port and loads it into memory.
swtimer	is a software timer.
tsboot	is a SCSI TEAC tape drive booter.
type41	is a primary partition type.

Table 2-5 Common System Modules List (continued)

Module	Description
`vcons`	is the console terminal pathlist.
`vsboot`	is a SCSI archive viper tape drive booter.