EPPCBug™ Diagnostic Firmware User's Guide

EPPCDIAA/UM1

Notice

While reasonable efforts have been made to assure the accuracy of this document, Motorola, Inc. assumes no liability resulting from any omissions in this document, or from the use of the information obtained therein. Motorola reserves the right to revise this document and to make changes from time to time in the content hereof without obligation of Motorola to notify any person of such revision or changes.

It is possible that this publication may contain reference to, or information about Motorola products (machines and programs), programming, or services that are not announced in your country. Such references or information must not be construed to mean that Motorola intends to announce such Motorola products, programming, or services in your country.

Restricted Rights Legend

If the documentation contained herein is supplied, directly or indirectly, to the U.S. Government, the following notice shall apply unless otherwise agreed to in writing by Motorola, Inc.

Use, duplication, or disclosure by the Government is subject to restrictions as set forth in subparagraph (c)(1)(ii) of the Rights in Technical Data and Computer Software clause at DFARS 252.227-7013.

Motorola, Inc.
Computer Group
2900 South Diablo Way
Tempe, Arizona 85282-9602

Preface

This manual provides general information and a description of the diagnostic firmware for the EPPCBUG Debugging Package.

Use of the EPPC debugger, the debugger command set, use of the one-line assembler/disassembler, and system calls for the Debugging Package are all described in the *EPPCBUG Debugging Package User's Manual*.

This manual is intended for anyone who wants to design OEM systems, supply additional capability to an existing compatible system, or work in a lab environment for experimental purposes.

A basic knowledge of computers and digital logic is assumed.

To use this manual, you should be familiar with the publications listed in the *Related Documentation* section found in the following pages.

Manual Terminology

Throughout this manual, a convention has been maintained whereby data and address parameters are preceded by a character which specifies the numeric format, as follows:

$	dollar	specifies a hexadecimal number
%	percent	specifies a binary number
&	ampersand	specifies a decimal number

For example, "12" is the decimal number twelve, and "$12" is the decimal number eighteen. Unless otherwise specified, all address references are in hexadecimal throughout this manual.

An asterisk (*) following the signal name for signals which are level significant denotes that the signal is true or valid when the signal is low.

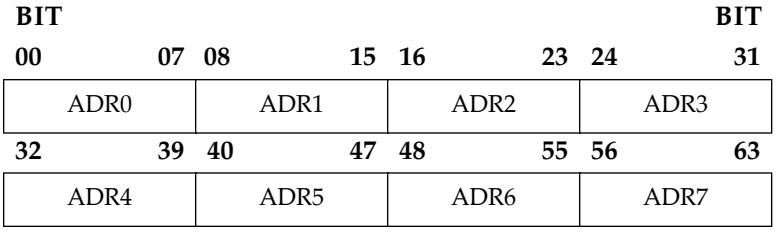
An asterisk (*) following the signal name for signals which are edge significant denotes that the actions initiated by that signal occur on high to low transition.

In this manual, *assertion* and *negation* are used to specify forcing a signal to a particular state. In particular, *assertion* and *assert* refer to a signal that is active or true; *negation* and *negate* indicate a signal that is inactive or false. These terms are used independently of the voltage level (high or low) that they represent.

Data and address sizes are defined as follows:

- A *byte* is eight bits, numbered 0 through 7, with bit 0 being the most significant.
- A two-byte is 16 bits, numbered 0 through 15, with bit 0 being the most significant. For the MBX and other PowerPC modules, this is called a *half-word*.
- A four-byte is 32 bits, numbered 0 through 31, with bit 0 being the most significant. For the MBX and other PowerPC modules, this is called a *word*.
- An eight-byte is 64 bits, numbered 0 through 63, with bit 0 being the most significant. For the MBX and other PowerPC modules, this is called a *double-word*.

Throughout this document, it is assumed that the MPU on the MBX module series is always programmed with *big-endian byte ordering*, as shown below. Any attempt to use *little-endian byte ordering* will render the EPPCBug debugger unusable.



The terms *control bit* and *status bit* are used extensively in this document. The term *control bit* is used to describe a bit in a register that can be set and cleared under software control. The term *true* is used to indicate that a bit is in the state that enables the function it controls. The term *false* is used to indicate that the bit is in the state that disables the function it controls. In all tables, the terms 0 and 1 are used to describe the actual value that should be written to the bit, or the value that it yields when read. The term *status bit* is used to describe a bit in a register that reflects a specific condition. The status bit can be read by software to determine operational or exception conditions.

The following conventions are used in this document:

bold

is used for user input that you type just as it appears. Bold is also used for commands, options and arguments to commands, and names of programs, directories, and files.

italic

is used for names of variables to which you assign values. Italic is also used for comments in screen displays and examples.

`courier`

is used for system output (e.g., screen displays, reports), examples, and system prompts.

<RETURN>

represents the carriage return key.

CTRL

represents the Control key. Execute control characters by holding down the control key while pressing the letter key, e.g., **CTRL-d**.

Related Documentation

The following publications are applicable to the EPPC module series and may provide additional helpful information. If not shipped with this product, they may be purchased by contacting your Motorola sales office.

Document Title	Motorola Publication Number
EPPCBug Firmware Package User's Manual	EPPCBUGA1/UM1

Printed in the United States of America

January 1998

Contents

List of Figures

List of Tables

General Information 1

Introduction

This manual describes the set of hardware diagnostics included in the EPPCBug Debugging Package, intended for testing and troubleshooting of Motorola's MPC8XX-based boards. This member of the EPPCBug firmware family, known as EPPCBug diagnostics, is implemented on these Motorola MPC8XX based products:

* MBX821-XXX MPC821 based embedded controllers

* MBX860-XXX MPC860 based embedded controllers

They are collectively referred to in this manual as the MBX board or board. When necessary to refer to them individually, they are called the MBX821 and MBX860 respectively.

This introductory chapter includes information about the operation and use of the diagnostics. Chapter 2 contains descriptions of the diagnostic utilities. Chapter 3 contains descriptions of the diagnostic test routines.

Before using the EPPCBug diagnostics, you should ensure that your MBX board and other hardware have been properly configured and connected according to the installation guide for your MBX board. You also need the User's manual for EPPCBug. It contains a complete description of EPPCBug, the start-up procedure, descriptions of all general software debugging commands, and other information you need to know about the debugger.

Overview of EPPCBug Firmware

The EPPCBug diagnostic firmware package resides in flash memory which has been programmed on the MBX. These flash memory devices (which also contain EPPCBug) contain a complete diagnostic monitor along with a battery of utilities and tests for exercise, test, and debug of hardware in the MBX environment. The diagnostics are menu driven for ease of use. The Help (**HE**) command displays a menu of the diagnostic functions; i.e., the tests and utilities. Several tests have a sub-test menu which may be called using the **HE** command. In addition, some utilities have subfunctions, and as such have subfunction menus.

Debugger and Diagnostic Directories

When using EPPCBug, you operate out of either the debugger directory or the diagnostic directory:

- If you are in the debugger directory, the debugger prompt EPPC-Bug> is displayed and you have all of the debugger commands at your disposal.
- If you are in the diagnostic directory, the diagnostic prompt EPPC-Diag> is displayed and you have all of the diagnostic commands at your disposal as well as all of the debugger commands.

To use the diagnostics, you must be in the diagnostic directory. If the prompt EPPC-Bug> is displayed, you are in the debugger directory and must switch to the diagnostic directory by entering SD, the debugger's Switch Directories command. The diagnostic prompt EPPC-Diag> is then be displayed.

You may examine the commands in the particular directory that you are currently in by using the Help (HE) command.The Help (HE) command displays a menu of all available diagnostic functions; i.e., the tests and utilities. Several tests have a subtest

menu which may be called using the HE command. In addition, some utilities have subfunctions, and as such have subfunction menus.

Command Entry

To invoke a diagnostic command, enter the name of the diagnostic command when the prompt EPPC-Diag> appears, and then press the RETURN or ENTER key.

The command may be the name of a diagnostic utility routine and may include one or more arguments; or it may be the name of one or more test groups listed in a main (root) directory and may include one or more subcommands (individual test names) listed in the subdirectory for a particular test group.

The utility routines are described in Chapter 2. The test groups are described in Chapter 3. Examples of command entry for both are given below.

Root-Level Command (Utility):

The utility or root-level commands affect the operation of the tests that are subsequently run. A test group name may be entered on the same command line. For example:

```
EPPC-Diag>CF RAM
```

causes an interactive dialog to begin, in which you may enter parameters for the RAM tests.

Command entry may also include a subcommand (individual test name). For example:

```
EPPC-Diag>HE RAM CODE
```

causes a help screen to appear that gives information about the CODE test in the RAM test group.

Root-Level Command (Test Group):

Entering just the name of a test group causes all individual tests that are part of that group to execute in sequence (with some exceptions). For example:

```
EPPC-Diag>RAM
```

causes all Random Access Memory (RAM) tests to execute, except for two that only execute if specified.

Subdirectory-Level Command (Individual Test):

Entering the name of a test group followed by the name of an individual test from that group causes just that test to execute.

For example, to call up a particular Random Access Memory (RAM) test, enter:

```
EPPC-Diag>RAM ADR
```

This causes the monitor to find the RAM test group subdirectory, and then to execute the Memory Addressing test command ADR from that subdirectory.

To call up a particular SCC1ETH test, enter:

```
EPPC-Diag>SCC1ETH ADDR
```

This causes the monitor to find the SCC1ETH test group subdirectory, and then to execute the ADDR subtest from that subdirectory.

Multiple Subdirectory-Level Commands (Individual Tests):

If the first part of a command is a test group name, any number and/or sequence of tests from that test group may be entered after the test group name so long as the debugger's input buffer size limit is not exceeded. For example:

```
EPPC-Diag>RAM PATS ADR
```

This causes both the Data Patterns (PATS) and the Memory Addressing (ADR) tests from the RAM test group to execute.

Multiple Root-Level Commands (Test Groups):

Multiple commands may be entered. If a command expects parameters and another command is to follow it, separate the two with a semicolon (;). For example, to invoke the command SCC1ETH ADDR after the command RAM ADR, the command line would read:

```
EPPC-Diag>RAM ADR; SCC1ETH ADDR
```

Spaces are not required before or after the semicolon but are shown here for legibility. Spaces are required between commands and their arguments. Several commands may be combined on one line.

Diagnostic Utilities 2

Introduction

This chapter contains descriptions and examples of the various diagnostic utilities available in EPPCBug.

The diagnostic tests are described in the *Test Descriptions* chapter.

Utilities

In addition to individual or sets of tests, the diagnostic package provides the utilities listed in the next table and described on the following pages. These utilities are root-level commands to the diagnostic monitor and do not require a preceeding test group name.

Table 2-1. Diagnostic Utilities

Mnemonic	Description
AEM	Append Error Messages Mode
CEM	Clear Error Messages
CF	Test Group Configuration (cf) Parameters Editor
DE	Display Error Counters
DEM	Display Error Messages
DP	Display Pass Count
HE	Help
HEX	Help Extended
LA	Loop Always Mode
LC	Loop-Continue Mode

Table 2-1. Diagnostic Utilities (Continued)

Mnemonic	Description
LE	Loop-On-Error Mode
LF	Line Feed Suppression Mode
LN	Loop Non-Verbose Mode
NV	Non-Verbose Mode
QST	Quick Self Test
SD	Switch Directories
SE	Stop-On-Error Mode
ST	Self Test
ZE	Clear (Zero) Error Counters
ZP	Zero Pass Count

Append Error Messages Mode - Command AEM

This command allows you to accumulate error messages in the internal error message buffer of the diagnostic monitor. The **AEM** command sets the internal append error messages flag of the diagnostic monitor. When the internal append error messages flag is clear, the diagnostic error message buffer is erased (cleared of all character data) before each test is executed. The duration of this command is for the life of the command line being parsed by the diagnostic monitor. The default of the internal append error messages flag is clear. The internal flag is not set until it is encountered in the command line by the diagnostic monitor.

Clear Error Messages - Command CEM

This command clears the internal error message buffer of the diagnostic monitor.

Test Group Configuration (cf) Parameters Editor - Command CF

The **cf** parameters control the operation of all tests in a test group. For example, the **RAM** test group has parameters such as starting address, ending address, parity enable, etc. At the time of initial execution of the diagnostic monitor, the default configuration parameters are copied from the firmware into the debugger work page. Here you can modify the configuration parameters using the **CF** command. When you invoke the **CF** command you are prompted with a brief parameter description and the current value of the parameter. You may enter a new value for that parameter, or a carriage return to proceed to the next configuration parameter.

You may also specify one or more test groups as argument(s) immediately following the **CF** command. If no arguments follow the **CF** command, the parameters for all test groups are listed for possible change.

Display Error Counters - Command DE

Each test or command in the diagnostic monitor has an individual error counter. As errors are encountered in a particular test, that error counter is incremented. If you were to run a self-test or just a series of tests, the results could be broken down as to which tests passed by examining the error counters.

To display all errors, enter **DE**. **DE** displays the results of a particular test if the name of that test follows **DE**. Only nonzero values are displayed.

Display Error Messages - Command DEM

This command displays (dumps) the internal error message buffer of the diagnostic monitor.

Display Pass Count - Command DP

A count of the number of passes in Loop-Continue (**LC**) mode is kept by the monitor. This count is displayed with other information at the conclusion of each pass. To display this information without using **LC**, enter **DP**.

Help - Command HE

On-line documentation has been provided in the form of a Help command (syntax: **HE** *[command name]*). This command displays a menu of the top level directory and test group names if no parameters are entered, or a menu of the subdirectory if the name of a subdirectory is entered. (The top level directory lists `(DIR)` after the name of each command that has a subdirectory).

For example, to bring up a menu of all the memory tests, enter **HE RAM**. When a menu is too long to fit on the screen, it pauses until the operator presses the carriage return, <**CR**>, again. To review a description of an individual test, enter the full name, i.e., **HE RAM CODE** displays information on the RAM Code Execution/Copy test routine.

The Help screen is shown in the following figure.

```
EPPC-Diag>he

AEM     Append Error Messages Mode

CEM     Clear Error Messages

CF      Configuration Editor

D1350   DS1350 BBRAM (DIR)

DE      Display Errors

DEM     Display Error Messages

DP      Display Pass Count

HE      Help on Tests/Commands

HEX     Help Extended

ISAB    ISA Bridge Tests (DIR)

LA      Loop Always Mode
```

```
LC      Loop Continuous Mode
LE      Loop on Error Mode
LF      Line Feed Mode
LN      Loop Non-Verbose Mode
LPT     Super I/O Parallel Port (DIR)
NV      Non-Verbose Mode
PCIB    PCI Bridge Tests (DIR)
QST     Quick Self Test (DIR)
RAM     Random Access Memory Tests (DIR)
SCC1ETH SCC1 Ethernet (DIR)
SE      Stop on Error Mode
ST      Self Test (DIR)
ZE      Zero Errors
ZP      Zero Pass Count
EPPC-Diag>
```

Figure 2-1. Help Screen

Help Extended - Command HEX

The **HEX** command goes into an interactive, continuous mode of the **HE** command. The syntax is **HEX<CR>**. The prompt displayed for **HEX** is the?. You may then type the name of a directory, or command. Type **QUIT** to exit.

Loop Always Mode - Prefix LA

To endlessly repeat a test or series of tests, enter the prefix **LA**. The **LA** command modifies the way that a failed test is endlessly repeated. The **LA** command has no effect until a test failure occurs, at which time, if the **LA** command has been previously encountered in the user command line, the failed test is endlessly repeated.

To break the loop, press the BREAK key on the diagnostic video display terminal. Certain tests disable the BREAK key interrupt, so pressing the RESET switch or power-cycling the MBX board may become necessary.

Loop-Continue Mode - Prefix LC

To endlessly repeat a test or series of tests, enter the prefix **LC**. This loop includes everything on the command line.

To break the loop, press the BREAK key on the diagnostic video display terminal. Certain tests disable the BREAK key interrupt, so pressing the RESET switch or power-cycling the MBX board may become necessary.

Loop-On-Error Mode - Prefix LE

Occasionally, when an oscilloscope or logic analyzer is in use, it becomes desirable to endlessly repeat a test (loop) while an error is detected. The **LE** command modifies the way that a failed test is endlessly repeated. The **LE** command has no effect until a test failure occurs, at which time, if the **LE** command has been previously encountered in the user command line, the failed test is re-executed as long as the previous execution returned failure status.

To break the loop, press the BREAK key on the diagnostic video display terminal. Certain tests disable the BREAK key interrupt, so pressing the RESET switch or power-cycling the MBX board may become necessary.

Line Feed Suppression Mode - Prefix LF

The **LF** command sets the internal line feed mode flag of the diagnostic monitor. The default state of the internal line feed mode flag is clear which causes the executing test title/status line(s) to be terminated with a line feed character (scrolled). The duration of the **LF** command is the life of the user command line in which it

appears. The line feed mode flag is normally used by the diagnostic monitor when executing a system mode self test. Although rarely invoked as a user command, the **LF** command is available to the diagnostic user.

Loop Non-Verbose Mode - Prefix LN

The **LN** command modifies the way that a failed test is endlessly repeated. The **LN** command has no effect until a test failure occurs, at which time, if the **LN** command has been previously encountered in the user command line, further printing of the test title and pass/fail status is suppressed. This is useful for more rapid execution of the failing test; i.e., the **LN** command contributes to a "tighter" loop.

Non-Verbose Mode - Prefix NV

Upon detecting an error, the tests display a substantial amount of data. To avoid having to watch the scrolling display, EPPCBug includes a mode that suppresses all messages except `PASSED` or `FAILED`. This mode is called non-verbose and is invoked prior to calling a command by entering **NV**. **NV ST** would cause the monitor to run the self-test, but show only the names of the subtests and the results (pass/fail).

Switch Directories - Command SD

To leave the diagnostic directory (and disable the diagnostic tests), enter **SD**. At this point, only the commands for EPPCBug function. When in the EPPCBug directory, the prompt reads "EPPC-Bug>". To return to the diagnostic directory, the command **SD** is entered again. When in the diagnostic directory, the prompt reads "EPPC-Diag>". The purpose of this feature is to allow you to access EPPCBug without the diagnostics being visible.

Stop-On-Error Mode - Prefix SE

It is sometimes desirable to stop a test or series of tests at the point where an error is detected. **SE** accomplishes that for most of the tests. To invoke **SE**, enter it before the test or series of tests that is to run in Stop-On-Error mode.

Self Test - Command ST and QST

The monitor provides an automated test mechanism called self test. This mechanism runs all the tests included in an internal self-test directory. The command **HE ST** lists the top level of the self test directory in alphabetical order.

Each test is described later in this manual.

QST operates in the same manner as ST except a lesser number of RAM tests are run. This reduces the amount of time needed to complete selftest.

Clear (Zero) Error Counters - Command ZE

The error counters originally come up with the value of zero, but it is occasionally desirable to reset them to zero at a later time. This command resets all of the error counters to zero. The error counters can be individually reset by entering the specific test name following the command. Example: **ZE RAM CODE** clears the error counter associated with **RAM CODE**.

Zero Pass Count - Command ZP

Invoking the **ZP** command resets the pass counter to zero. This is frequently desirable before typing in a command that invokes the Loop-Continue mode. Entering this command on the same line as **LC** results in the pass counter being reset every pass.

Test Descriptions 3

Detailed descriptions of EPPCBug's diagnostic tests are presented in this chapter. The test groups are described in the order shown in the following table.

Table 3-1. Diagnostic Test Groups

Test Group	Description
D1350	DS1350 BBRAM
ISAB	ISA Bridge Tests
LPT	Super I/O Parallel Port
PCIB	PCI Bridge Tests
RAM	Random Access Memory Tests
SCC1ETH	SCC1 Ethernet

Note You may enter command names in either uppercase or lowercase.

DS13xxY (Battery Backed-Up RAM) Tests

This section describes the battery backed-up RAM tests. These tests check the RAM and battery status of the DS13xxY chips.

Entering **D1350** without parameters causes all **BBRAM** tests to execute in the order shown in the table below.

To run an individual test, add that test name to the **D1350** command.

The individual tests are described in alphabetical order on the following pages.

Table 3-2. BBRAM Test Group

Mnemonic	Description
RAM	Battery Backed-Up RAM
ADR	BBRAM address test
BTLOW	Battery Low Test

D1350 Configuration Parameters

Command Input

```
EPPC-Diag>cf d1350

D1350 Configuration Data:

ADR test: Restore BBRAM contents on test exit? [N/Y] =Y ?

BBRAM SIZE: (32/128/512) Kbytes? =32
```

Description

User configurable test parameters are available for the **D1350** test group. The **CF** command may be used to tune these parameters.

The **DS1350** test parameters are listed in the command input block above and described below.

```
ADR test: Restore BBRAM contents on test exit? [N/Y] =Y ?
```

If this parameter is Y then the contents of the Battery Backup RAM will be restored to its state prior to the running of this test, upon exit. Otherwise, the RAM will be modified upon exit.

```
BBRAM SIZE: (32/128/512) Kbytes? =32 ?
```

This parameter needs to contain the size of the BBRAM chip onboard (or smaller) in Kilobytes. If a smaller size is specified, the BBRAM will only be partially tested.

To determine the size of the specific BBRAM find the correct size to the corresponding part number in the following table.

Table 3-3. BBRAM Memory Sizes

BBRAM Part Number	BBRAM Size
DS1330	32K
DS1345	128K
DS1350	512K

BBRAM Addressing - ADR

Command Input

```
EPPC-Diag>D1350 ADR
```

Description

This test is designed to assure proper addressability of the D13xxY BBRAM. The algorithm used is to fill the BBRAM with data pattern "a", a single address line of the D13xxY is set to one, and pattern "b" is written to the resultant (test) address. All other locations in the BBRAM are checked to ensure that they were not affected by this write. The "a" pattern is then restored to the test. All address lines connected to the D13xxY are tested in this manner.

Since this test overwrites all memory locations in the BBRAM, the BBRAM contents are saved in debugger system memory prior to writing the BBRAM. The D1350 test group features a configuration parameter which overrides automatic restoration of the BBRAM contents. The default for this parameter is to restore BBRAM contents upon test completion.

Response/Messages

After the command has been issued, the following line is printed:

```
D1350 ADR: D1350Y RAM Addressing..................... Running --->
```

If all parts of the test are completed correctly, then the test passes.

```
D1350 ADR: D1350Y RAM Addressing..................... Running ---> PASSED
```

If any part of the test fails, then the display appears as follows:

```
D1350 ADR: D1350Y RAM Addressing..................... Running ---> FAILED


(error message)
```

Here, (error message) is one of the following:

If debugger system memory cannot be allocated for use as a save area for the BBRAM contents:

```
D1350/ADR Test Failure Data:
```

```
Test Initialization Error:

Not Enough Memory, Need =________ Actual =________
```

If the BBRAM cannot be initialized with pattern “a”:

```
D1350/ADR Test Failure Data:

Data Verify Error: Address =________, Expected =__, Actual =__

Memory initialization error
```

If a pattern “b” write affects any BBRAM location other than the test address:

```
D1350/ADR Test Failure Data:

Data Verify Error: Address =________, Expected =__, Actual =__

Memory addressing error
```

Battery Low Test- BTLOW

D1350 BTLOW

Command Input

```
EPPC-Diag>d1350 BTLOW
```

Description

This test checks the status of the onboard battery of the **D13xxY** chip.

Response/Messages

After the command has been issued, the following line is printed:

```
D1350 BTLOW: DS1350 Battery Low Test ............. Running --->
```

If all parts of the test are completed correctly, then the test passes.

```
D1350 BTLOW: DS1350 Battery Low Test ............. Running ---> PASSED
```

If any part of the test fails, then the display appears as follows:

```
D1350 BTLOW: DS1350 Battery Low Test ............. Running ---> FAILED


Low Battery Status
```

Battery Backed-Up RAM - RAM D1350 RAM

Command Input

```
EPPC-Diag>d1350 RAM
```

Description

This test performs a data test on each RAM location of the Dallas DS13xxY "Zeropower" RAM. RAM contents are unchanged upon completion of test, regardless of pass or fail test return status. This test is coded to test only byte data entities.

The test proceeds as follows:

1. For each of the following patterns: $55, $aa, $33, $cc, $0, and $ff:
2. For each valid byte of the "Zeropower RAM":
3. Write and verify the current data test pattern.
4. Write and verify the complement of the current data test pattern.

Response/Messages

After the command has been issued, the following line is printed:

```
D1350 RAM: DS1350Y Battery Backed Up RAM..... Running --->
```

If all parts of the test are completed correctly, then the test passes.

```
D1350 RAM: DS1350Y Battery Backed Up RAM..... Running ---> PASSED
```

If any part of the test fails, then the display appears as follows:

```
D1350 RAM: DS1350Y Battery Backed Up RAM..... Running ---> FAILED


(error message)
```

Here, (error message) is one of the following:

If debugger system memory cannot be allocated for use as a save area for the BBRAM contents:

```
D1350/RAM Test Failure Data:

Test Initialization Error:

Not Enough Memory, Need =________ Actual =________
```

If there is a problem with an address, the location and values are printed

out..

```
Address =00000, Expected =55, Actual =53
```

ISAB - PCI/ISA Bridge Tests

This section describes the individual PCI/ISA Bridge tests.

Entering **ISAB** without parameters causes all **ISAB** tests to execute in the order shown in the following table.

To run an individual test, add that test name to the **ISAB** command.

The individual tests are described in alphabetical order on the following pages.

Table 3-4. ISAB Test Group

Name	Description
REG	Register
IRQ	Interrupt

IRQ - Interrupt

Command Input

```
PPC1-Diag>ISAB IRQ
```

Description

This test verifies that the ISA Bridge can generate interrupts.

Response/Messages

After the command has been issued, the following line is printed:

```
ISAB IRQ: Interrupt..................... Running --->
```

If all parts of the test are completed correctly, then the test passes.

```
ISAB IRQ: Interrupt..................... Running ---> PASSED
```

If any failures occur, the following is displayed (more descriptive text then follows):

```
ISAB IRQ: Interrupt..................... Running ---> FAILED
```

If the test fails because an interrupt request from the ISA Bridge is pending, after masking the interrupt in the IEN register, the following is displayed:

```
ISAB/IRQ Test Failure Data:
Unexpected ISAB IRQ pending
Address =________, Expected =________, Actual =________
```

This test makes use of the ISA Bridge counters, to generate the test interrupt. If after running the counters to “terminal count”, an interrupt has not been requested by the ISAB, the following message is displayed:

```
ISAB/IRQ Test Failure Data:
ISAB IRQ not pending in IST register
Address =________, Expected =________, Actual =________
```

REG - Register

Command Input:

```
PPC1-Diag>ISAB REG
```

Description

This test verifies that the ISAB registers can be written and read. Data patterns verify that every read/write bit can be modified.

Response/Messages

After the command has been issued, the following line is printed:

```
ISAB REG: Register..................... Running --->
```

If all parts of the test are completed correctly, then the test passes.

```
ISAB REG: Register..................... Running ---> PASSED
```

If any failures occur, the following is displayed (more descriptive text then follows):

```
ISAB REG: Register..................... Running ---> FAILED
```

If the test fails because the pattern written does not match the data read back from the ISAB register, the following is printed:

```
ISAB REG Test Failure Data:
Register xxx Miscompare Error:Address =____,Expected =_,Actual =_
```

LPT - Super I/O Parallel Port Tests

This section describes the individual Super I/O Parallel Port tests.

Entering **LPT** without parameters causes all **LPT** tests to run in the order shown in the table below, except as noted.

To run an individual test, add that test name to the **LPT** command.

The individual tests are described in alphabetical order on the following pages.

Table 3-5. LPT Test Group

Name	Description
IRQ	Parallel Port Interrupt

The **IRQ** test does not require any connection to Parallel Port header J13.

IRQ - Parallel Port Interrupt

Command Input

```
EPPC-Diag>LPT IRQ
```

Description

This test verifies the capability of the Peripheral I/O Controller parallel port function to generate interrupts to the system.

The test consists of two parts.

Part 1 checks that no interrupt occurs when the parallel port interrupt output is disabled and the parallel port interrupt is unmasked at the cascaded 8259 interrupt controller and external interrupts are unmasked at the MPC 8xx interrupt controller.

Part 2 checks that an interrupt, and the correct 8259 IRQ, does occur when the parallel port interrupt output is asserted and the parallel port interrupt is unmasked at the cascaded 8259 interrupt controller and external interrupts are unmasked at the MPC 8xx interrupt controller.

The parallel port interrupt output is asserted by configuring the parallel port function into ECP Mode-ECP Test Mode and enabling the service interrupt. This asserts an interrupt since the direction is output and the Test FIFO is empty. This test does not require any offboard connections.

Response/Messages

After the command has been issued, the following line is printed:

```
LPT IRQ: Parallel Port Interrupt.............Running --->
```

If all parts of the test are completed correctly, then the test passes:

```
LPT IRQ: Parallel Port Interrupt.............Running ---> PASSED
```

If any part of the test fails, then the display appears as follows:

```
LPT IRQ: Parallel Port Interrupt.............Running ---> FAILED
```

```
LPT/IRQ Test Failure Data:
(error message)
```

Refer to the section *LPT Error Messages* for a list of the error messages and their meaning.

LPT Error Messages

The **LPT** test group error messages generally take the following form:

```
LPT IRQ: Parallel Port Interrupt..... Running ---> FAILED

LPT/IRQ Test Failure Data:

IRQ 7 Failed to Occur, Interrupt Controller: IRR2 =22, IRR1 =00
```

The first line of the failure identifies what type of failure occurred. The following line provides additional information about the failure.

Table 3-6. LPT Error Messages

Error Message	Symptom or Cause
`Lookup of PC I/O product configuration option in VPD data failed`	The I2C bus access to the serial ROM device containing vital product data failed. The product configuration option tells whether PC I/O devices are installed on this board.
`Unsolicited IRQ %d occurred while device source of IRQ %d is still disabled`	This indicates a failure to part 1 of the test. The first IRQ number is the IRQ number that caused the external interrupt at the cascaded 8259 interrupt controller. The second IRQ number is the IRQ line at the cascaded 8259 interrupt controller that is connected to the parallel port.
`IRQ %d Failed to Occur, Interrupt Controller: IRR2 =XX, IRR1 =YY`	This indicates a failure to part 2 of the test. The IRQ number is the IRQ line at the cascaded 8259 interrupt controller that is connected to the parallel port . XX is the hex value of IRR register of the slave 8259 at the time the failure is reported. YY is the hex value of IRR register of the master 8259 at the time the failure is reported.
`Wrong IRQ occurred, expected =%d, actual =%d`	An interrupt from a different IRQ than expected occurred.

PCIB - PCI/ISA Bridge Tests

This section describes the individual PCI Bridge tests.

Entering **PCIB** without parameters causes all **PCIB** tests to execute in the order shown in the following table.

To run an individual test, add that test name to the **PCIB** command.

The individual tests are described in alphabetical order on the following pages.

Table 3-7. PCIB Test Group

Name	Description
REG	Register
IRQ	Interrupt

IRQ - Interrupt

Command Input

```
PPC1-Diag>PCIB IRQ
```

Description

This test verifies that the PCI Bridge can generate interrupts. It uses the software interrupt of the PCI Bridge to induce an interrupt from the PCI bridge to the CPU.

Response/Messages

After the command has been issued, the following line is printed:

```
PCIB IRQ: Interrupt..................... Running --->
```

If all parts of the test are completed correctly, then the test passes.

```
PCIB IRQ: Interrupt..................... Running ---> PASSED
```

If any failures occur, the following is displayed (more descriptive text then follows):

```
PCIB IRQ: Interrupt..................... Running ---> FAILED
```

If the test fails because an interrupt request from the PCI Bridge is pending and it is not expected the following is displayed:

```
PCIB/IRQ Test Failure Data:
Unexpected Interrupt:
SIPEND =________, SIMASK =________, SIVEC =________ INT_STAT =
```

If an interrupt has not been requested by the PCIB, the following message is displayed:

```
PCIB/IRQ Test Failure Data:
Interrupt Failed to Occur:
SIPEND =________, SIMASK =________, SIVEC =________ INT_STAT =
```

REG - Register

Command Input:

```
PPC1-Diag>PCIB REG
```

Description

This test verifies that the PCIB registers can be written and read. Data patterns verify that every read/write bit can be modified.

Response/Messages

After the command has been issued, the following line is printed:

```
PCIB REG: Register..................... Running --->
```

If all parts of the test are completed correctly, then the test passes.

```
PCIB REG: Register..................... Running ---> PASSED
```

If any failures occur, the following is displayed (more descriptive text then follows):

```
PCIB REG: Register..................... Running ---> FAILED
```

If the test fails because the pattern written does not match the data read back from the PCIB register, the following is printed:

```
PCIB REG Test Failure Data:
Data Miscompare Error: Address =____,Expected =_,Actual =_
```

RAM - Local RAM Tests

These sections describe the individual Random Access Memory (RAM) tests.

Entering **RAM** without parameters causes all **RAM** tests to execute in the order shown in the table below.

To run an individual test, add that test name to the **RAM** command.

The individual tests are described in alphabetical order on the following pages.

Table 3-8. RAM Test Group

Name	Description
QUIK	Quick Write/Read
ALTS	Alternating Ones/Zeros
PATS	Data Patterns
ADR	Memory Addressing
CODE	Code Execution/Copy
PERM	Permutations
RNDM	Random Data
BTOG	Bit Toggle
PED	Parity Error Detection
REF	Memory Refresh

ADR - Memory Addressing

Command Input

```
EPPC-Diag>RAM ADR
```

Description

This is the memory addressability test, the purpose of which is to verify addressing of memory in the range specified by the configuration parameters for the **RAM** test group. Addressing errors are sought by using a memory locations address as the data for that location. This test is coded to use only 32-bit data entities. The test proceeds as follows:

1. A Locations Address is written to its location (*n*).
2. The next location (*n*+4) is written with its address complemented.
3. The next location (*n*+8) is written with the most significant (MS) 16 bits and least significant (LS) 16 bits of its address swapped with each other.
4. Steps 1, 2, and 3 are repeated throughout the specified memory range.
5. The memory is read and verified for the correct data pattern(s) and any errors are reported.
6. The test is repeated using the same algorithm as above (steps 1 through 5) except that inverted data is used to insure that every data bit is written and verified at both “0” and “1”.

Response/Messages

After the command has been issued, the following line is printed:

```
RAM   ADR: Addressability..... Running --->
```

If all parts of the test are completed correctly, then the test passes.

```
RAM   ADR: Addressability..... Running ---> PASSED
```

If the test fails, then the display appears as follows.

```
RAM   ADR: Addressability.............. Running ---> FAILED

Data Miscompare Error:
Address =________, Expected =________, Actual =________
```

ALTS - Alternating Ones/Zeros

Command Input

```
EPPC-Diag>RAM ALTS
```

Description

This test verifies addressing of memory in the range specified by the configuration parameters for the **RAM** test group. Addressing errors are sought by using a memory locations address as the data for that location. This test is coded to use only 32-bit data entities. The test proceeds as follows:

1. Location (*n*) is written with data of all bits 0.
2. The next location (*n*+4) is written with all bits 1.
3. Steps 1 and 2 are repeated throughout the specified memory range.
4. The memory is read and verified for the correct data pattern(s) and any errors are reported.

Response/Messages

After the command has been issued, the following line is printed:

```
RAM   ALTS: Alternating Ones/Zeroes.......... Running --->
```

If all parts of the test are completed correctly, then the test passes.

```
RAM   ALTS: Alternating Ones/Zeroes.......... Running ---> PASSED
```

If the test fails, then the display appears as follows.

```
RAM   ALTS: Alternating Ones/Zeroes.......... Running ---> FAILED

Data Miscompare Error:
Address =________, Expected =________, Actual =________
```

BTOG - Bit Toggle

Command Input

```
EPPC-Diag>RAM BTOG
```

Description

The memory range is specified by the RAM test directory configuration parameters. (Refer to *CF - Test Group Configuration Parameters Editor* in Chapter 2.) The RAM test directory configuration parameters also determine the value of the global random data seed used by this test. The global random data seed is incremented after it is used by this test. This test uses the following test data pattern generation algorithm:

1. Random data seed is copied into a work register.
2. Work register data is shifted right one bit position.
3. Random data seed is added to work register using unsigned arithmetic.
4. Data in the work register may or may not be complemented.
5. Data in the work register is written to current memory location.

If the RAM test directory configuration parameter for code cache enable equals "Y", the microprocessor code cache is enabled. This test is coded to operate using the 32-bit data size only. Each memory location in the specified memory range is written with the test data pattern. Each memory location in the specified memory range is then written with the test data pattern complemented before it is written. The memory under test is read back to verify that the complement test data is properly retained. Each memory location in the specified memory range is then written with the test data pattern. The memory under test is read back to verify that the test data is properly retained.

Response/Messages

After the command has been issued, the following line is printed:

```
RAM   BTOG: Bit Toggle..................... Running --->
```

If all parts of the test are completed correctly, then the test passes.

```
RAM   BTOG: Bit Toggle..................... Running ---> PASSED
```

If the test fails, then the display appears as follows.

```
RAM   BTOG: Bit Toggle..................... Running ---> FAILED

Data Miscompare Error:
Address =________, Expected =________, Actual =________
```

CODE - Code Execution/Copy

Command Input

```
EPPC-Diag>RAM CODE
```

Description

Copy test code to memory and execute. The code in the memory under test copies itself to the next higher memory address and executes the new copy. This process is repeated until there is not enough memory, as specified by the configuration parameters, to perform another code copy and execution.

Response/Messages

After the command has been issued, the following line is printed:

```
RAM  CODE: Code Execution/Copy.............. Running --->
```

If all parts of the test are completed correctly, then the test passes.

```
RAM  CODE: Code Execution/Copy.............. Running ---> PASSED
```

The test failure mode is typified by the nonjudicial of the `PASSED` message above after more than about 1 minute, which indicates that the MPU has irrecoverably crashed. Hardware reset is required to recover from this error.

PATS - Data Patterns

Command Input

```
EPPC-Diag>RAM PATS
```

Description

If the test address range (test range) is less than 8 bytes, the test immediately returns pass status. The effective test range end address is reduced to the next lower 8-byte boundary if necessary. Memory in the test range is filled with all ones ($FFFFFFFF). For each location in the test range, the following patterns are used:

```
$00000000
$01010101
$03030303
$07070707
$0F0F0F0F
$1F1F1F1F
$3F3F3F3F
$7F7F7F7F
```

Each location in the test range is, individually, written with the current pattern and the 1's complement of the current pattern. Each write is read back and verified. This test is coded to use only 32-bit data entities.

Response/Messages

After the command has been issued, the following line is printed:

```
RAM   PATS: Patterns.................. Running --->
```

If all parts of the test are completed correctly, then the test passes.

```
RAM   PATS: Patterns.................. Running ---> PASSED
```

If the test fails, then the display appears as follows.

```
RAM   PATS: Patterns.................. Running ---> FAILED

Data Miscompare Error:
Address =________, Expected =________, Actual =_________
```

PED - Local Parity Memory Error Detection

Command Input

```
EPPC-Diag>RAM PED
```

Description

The memory range and address increment is specified by the RAM test directory configuration parameters. (Refer to *CF - Test Group Configuration Parameters Editor*.)

First, each memory location to be tested has the data portion verified by writing/verifying all zeros, and all ones. Each memory location to be tested is tested once with parity interrupt disabled, and once with parity interrupt enabled. Parity checking is enabled, and data is written and verified at the test location that causes the parity bit to toggle on and off (verifying that the parity bit of memory is good). Next, data with incorrect parity is written to the test location. The data is read, and if a parity error exception does occur, the fault address is compared to the test address. If the addresses are the same, the test passed and the test location is incremented until the end of the test range has been reached.

Response/Messages

After the command has been issued, the following line is printed:

```
RAM   PED: Local Parity Memory Detection.... Running --->
```

If the board under test does not support Parity error detection the test is bypassed

```
RAM   PED: Local Parity Memory Detection. Running --> BYPASS
```

If all parts of the test are completed correctly, then the test passes.

```
RAM   PED: Local Parity Memory Detection.... Running ---> PASSED
```

If any part of the test fails, then the display appears as follows.

```
RAM   PED: Local Parity Memory Detection.... Running ---> FAILED (error message)
```

Here *(error message)* is one of the following:

If a data verification error occurs:

```
RAM/PED Test Failure Data:

Data Miscompare Error:
Address =________, Expected =________, Actual =________
```

If an unexpected exception, such as a parity error being detected as the parity bit was being toggled:

```
RAM/PED Test Failure Data:

Unexpected Exception Error, Vector =________
Address Under Test =________
```

If no exception occurred when data with bad parity was read:

```
RAM/PED Test Failure Data:

Parity Error Detection Exception Did Not Occur

Exception Vector =________
Address Under Test =________
```

If the exception address was different from that of the test location:

```
RAM/PED Test Failure Data:

Fault Address Miscompare, Expected =________, Actual =________
```

PERM - Permutations

Command Input

```
EPPC-Diag>RAM PERM
```

Description

This command performs a test which verifies that the memory in the test range can accommodate 8-bit, 16-bit, and 32-bit writes and reads in any combination. The test range is the memory range specified by the **RAM** test group configuration parameters for starting and ending address. If the test address range (test range) is less than 16 bytes, the test immediately returns pass status. The effective test range end address is reduced to the next lower 16-byte boundary if necessary.

This test performs three data size test phases in the following order: 8, 16, and 32 bits. Each test phase writes a 16-byte data pattern (using its data size) to the first 16 bytes of every 256-byte block of memory in the test range. The 256-byte blocks of memory are aligned to the starting address configuration parameter for the **RAM** test group. The test phase then reads and verifies the 16-byte block using 8-bit, 16-bit, and 32-bit access modes.

Response/Messages

After the command has been issued, the following line is printed:

```
RAM   PERM: Permutations..... Running --->
```

If all parts of the test are completed correctly, then the test passes.

```
RAM   PERM: Permutations..... Running ---> PASSED
```

If the test fails, then the display appears as follows.

```
RAM   PERM: Permutations..... Running ---> FAILED

Data Miscompare Error:
Address =________, Expected =________, Actual =________
```

QUIK - Quick Write/Read

Command Input

```
EPPC-Diag>RAM QUIK
```

Description

Each pass of this test fills the test range with a data pattern by writing the current data pattern to each memory location from a local variable and reading it back into that same register. The local variable is verified to be unchanged only after the write pass through the test range. This test uses a first pass data pattern of 0, and $FFFFFFFF for the second pass. This test is coded to use only 32-bit data entities.

Response/Messages

After the command has been issued, the following line is printed:

```
RAM   QUIK: Quick Write/Read.......... Running --->
```

If all parts of the test are completed correctly, then the test passes.

```
RAM   QUIK: Quick Write/Read.......... Running ---> PASSED
```

If the test fails, then the display appears as follows.

```
RAM   QUIK: Quick Write/Read.......... Running ---> FAILED

Data Miscompare Error:
Expected =________, Actual =________
```

REF - Memory Refresh Testing

Command Input

```
EPPC-Diag>RAM REF
```

Description

The memory range and address increment is specified by the **RAM** test directory configuration parameters. (Refer to *CF - Test Group Configuration Parameters Editor*.)

First, the real time clock is checked to see if it is functioning properly. Second, each memory location to be tested has the data portion verified by writing/verifying all zeros, and all ones. Next a data pattern is written to the test location. After all the data patterns are filled for all test locations, a refresh wait cycle is executed. After the wait cycle, the data is read, and if the previously entered data pattern does not match the data pattern read in, a failure occurs. If the data patterns match, then the test is passed.

Response/Messages

After the command has been issued, the following line is printed:

```
RAM   REF: Memory Refresh Test..............  Running --->
```

If all parts of the test are completed correctly, then the test passes.

```
RAM   REF: Memory Refresh Test..............  Running ---> PASSED
```

If any part of the test fails, then the display appears as follows.

```
RAM   REF: Memory Refresh Test..............  Running ---> FAILED
(error message)
```

Here *(error message)* is one of the following:

If the real time clock is not functioning properly, one of the following is printed:

```
RAM/REF Test Failure Data:

RTC is stopped, invoke SET command.
```

or:

```
RAM/REF Test Failure Data:
RTC is in write mode, invoke SET command.
```

or:

```
RAM/REF Test Failure Data:
RTC is in read mode, invoke SET command.
```

If a data verification error occurs before the refresh wait cycle:

```
RAM/REF Test Failure Data:
Immediate Data Miscompare Error:
Address =________, Expected =________, Actual =________
```

If a data verification error occurs following the refresh wait cycle:

```
RAM/REF Test Failure Data:
Unrefreshed Data Miscompare Error:
Address =________, Expected =________, Actual =________
```

RNDM - Random Data

Command Input


```
EPPC-Diag>RAM RNDM
```

Description

The test block is the memory range specified by the **RAM** test group configuration parameters. The test proceeds as follows:

1. A random pattern is written throughout the test block.
2. The random pattern complemented is written throughout the test block.
3. The complemented pattern is verified.
4. The random pattern is rewritten throughout the test block.
5. The random pattern is verified.

This test is coded to use only 32-bit data entities. Each time this test is executed, the random seed in the **RAM** test group configuration parameters is post incremented by 1.

Response/Messages

After the command has been issued, the following line is printed:

```
RAM   RNDM: Random Data..................... Running --->
```

If all parts of the test are completed correctly, then the test passes.

```
RAM   RNDM: Random Data..................... Running ---> PASSED
```

If the test fails, then the display appears as follows.

```
RAM   RNDM: Random Data..................... Running ---> FAILED

Data Miscompare Error:
Address =________, Expected =________, Actual =________
```

SCC1ETH - SCC1 Ethernet Tests

These sections describe the individual on-board SCC1 Ethernet Controller tests.

Entering **SCC1ETH** without parameters causes all **SCC1ETH** tests to run in the order shown in the table below, except as noted.

To run an individual test, add that test name to the **SCC1ETH** command.

The individual tests are described in alphabetical order on the following pages.

Table 3-9. SCC1ETH Test Group

Name	Description
ADDR	Ethernet Address
REGA	Transceiver Control Register Access
TCLK	Transmit Clock Frequency
ILOOP	Interface Transceiver Loopback
ILOOPQ	Interface Transceiver Loopback SQE
Executed only when specified:	
TPLOOP	10BaseT Connector External Loopback
AULOOP	AUI Connector External Loopback

The **ADDR**, **REGA**, **TCLK**, **ILOOP**, **ILOOPQ**, tests do not require, are not affected by, and do not interfere with, the connection to an active ethernet through the 10BASE-T connector or the AUI connector.

The **TPLOOP** test requires an external loopback “plug” connected to the 10BASE-T connector.

The **AULOOP** test requires an external loopback “plug” connected to the AUI header.

ADDR - Ethernet Address

Command Input

```
EPPC-Diag>SCC1ETH ADDR
```

Description

This test verifies that the ethernet address stored in the Vital Product Data Serial ROM is a valid Motorola ethernet address. Valid Motorola ethernet addresses start with 0x08003E.

Response/Messages

After the command has been issued, the following line is printed:

```
SCC1ETH ADDR: Ethernet Address..............Running --->
```

If all parts of the test are completed correctly, then the test passes:

```
SCC1ETH ADDR: Ethernet Address..............Running ---> PASSED
```

If any part of the test fails, then the display appears as follows:

```
SCC1ETH ADDR: Ethernet Address..............Running ---> FAILED

SCC1ETH/ADDR Test Failure Data:
(error message)
```

Refer to the section *SCC1ETH Error Messages* for a list of the error messages and their meaning.

AULOOP - AUI Connector External Loopback

Command Input

```
EPPC-Diag>SCC1ETH AULOOP
```

Description

This test verifies that the signal path through the AUI header is operating correctly by transmitting and receiving packets and comparing the data. This test requires the presence of an external loopback "plug" in the AUI header.

Note It is recommended that the board under test not be connected to a live network while this test is running. The suggested "loopback" setup for this test is to connect pins of the 16 pin AUI header as follows: connect pin 5 to pin 9 and connect pin 4 to pin 8.

Response/Messages

After the command has been issued, the following line is printed:

```
SCC1ETH AULOOP: AUI Connector External Loopback...Running --->
```

If all parts of the test are completed correctly, then the test passes:

```
SCC1ETH AULOOP: AUI Connector External Loopback...Running ---> PASSED
```

If any part of the test fails, then the display appears as follows:

```
SCC1ETH AULOOP: AUI Connector External Loopback...Running ---> FAILED

SCC1ETH/AULOOP Test Failure Data:
(error message)
```

Refer to the section *SCC1ETH Error Messages* for a list of the error messages and their meaning.

LOOP - Interface Transceiver Loopback

Command Input

```
EPPC-Diag>SCC1ETH ILOOP
```

Description

This test verifies the signal path through the on-board ethernet transceiver device. The ethernet transceiver is operated in internal loopback mode. The test transmits and receives packets and compares the data.

Response/Messages

After the command has been issued, the following line is printed:

```
SCC1ETH ILOOP: Interface Transceiver Loopback.....Running --->
```

If all parts of the test are completed correctly, then the test passes:

```
SCC1ETH ILOOP: Interface Transceiver Loopback.....Running ---> PASSED
```

If any part of the test fails, then the display appears as follows:

```
SCC1ETH ILOOP: Interface Transceiver Loopback.....Running ---> FAILED

SCC1ETH/ILOOP Test Failure Data:
(error message)
```

Refer to the section *SCC1ETH Error Messages* for a list of the error messages and their meaning.

ILOOPQ - Interface Transceiver Loopback SQE

Command Input

```
EPPC-Diag>SCC1ETH ILOOPQ
```

Description

This test verifies the signal path through the on-board ethernet transceiver device while Signal Quality Error testing is enabled in SCC1 and the transceiver. The ethernet transceiver is operated in internal loopback mode. The test transmits and receives packets and compares the data. For Signal Quality Error testing the transceiver is configured to generate a collision detect pulse after each transmission and Heartbeat Checking is enabled in SCC1.

Response/Messages

After the command has been issued, the following line is printed:

```
SCC1ETH ILOOPQ: Interface Transceiver Loopback SQE..Running --->
```

If all parts of the test are completed correctly, then the test passes:

```
SCC1ETH ILOOPQ: Interface Transceiver Loopback SQE..Running ---> PASSED
```

If any part of the test fails, then the display appears as follows:

```
SCC1ETH ILOOPQ: Interface Transceiver Loopback SQE..Running ---> FAILED

SCC1ETH/ILOOPQ Test Failure Data:
(error message)
```

Refer to the section *SCC1ETH Error Messages* for a list of the error messages and their meaning.

REGA - Transceiver Control Register Access

Command Input

```
EPPC-Diag>SCC1ETH REGA
```

Description

This test verifies the write and read capability of the 5 bits in the board Control and Status Register 1 that interface to the ethernet interface transceiver chip.

Response/Messages

After the command has been issued, the following line is printed:

```
SCC1ETH REGA: Transceiver Control Register Access...Running --->
```

If all parts of the test are completed correctly, then the test passes:

```
SCC1ETH REGA: Transceiver Control Register Access...Running ---> PASSED
```

If any part of the test fails, then the display appears as follows:

```
SCC1ETH REGA: Transceiver Control Register Access...Running ---> FAILED

SCC1ETH/REGA Test Failure Data:
(error message)
```

Refer to the section *SCC1ETH Error Messages* for a list of the error messages and their meaning.

TCLK- Transmit Clock Frequency

Command Input

```
EPPC-Diag>SCC1ETH TCLK
```

Description

This verifies the frequency of the ethernet transmitter clock by measuring the average time it takes to transmit a frame while in internal loopback mode.

Response/Messages

After the command has been issued, the following line is printed:

```
SCC1ETH TCLK: Transmit Clock Frequency........Running --->
```

If all parts of the test are completed correctly, then the test passes:

```
SCC1ETH TCLK: Transmit Clock Frequency........Running ---> PASSED
```

If any part of the test fails, then the display appears as follows:

```
SCC1ETH TCLK: Transmit Clock Frequency........Running ---> FAILED

SCC1ETH/TCLK Test Failure Data:
(error message)
```

Refer to the section *SCC1ETH Error Messages* for a list of the error messages and their meaning.

TPLOOP -10BaseT Connector External Loopback

Command Input

```
EPPC-Diag>SCC1ETH TPLOOP
```

Description

This test verifies that the signal path through the 10BaseT connector is operating correctly by transmitting and receiving packets and comparing the data. This test requires the presence of an external loopback "plug" in the 10BaseT connector.

Note It is recommended that the board under test not be connected to a live network while this test is running. The suggested "loopback" setup for this test is to connect pins of the RJ45 10BaseT connector as follows: connect pin 1 to pin 3 and connect pin 2 to pin 6. A cable at least 6 feet long should be used for the "loopback".

Response/Messages

After the command has been issued, the following line is printed:

```
SCC1ETH TPLOOP: 10BaseT Connector External Loopback.Running --->
```

If all parts of the test are completed correctly, then the test passes:

```
SCC1ETH TPLOOP: 10BaseT Connector External Loopback.Running ---> PASSED
```

If any part of the test fails, then the display appears as follows:

```
SCC1ETH TPLOOP: 10BaseT Connector External Loopback.Running ---> FAILED

SCC1ETH/TPLOOP Test Failure Data:
(error message)
```

Refer to the section *SCC1ETH Error Messages* for a list of the error messages and their meaning.

SCC1ETH Error Messages

The **SCC1ETH** test group error messages generally take the following form:

```
SCC1ETH ILOOP: Interface Transceiver Loopback.Running ---> FAILED

SCC1ETH/ILOOP Test Failure Data:

Time-out error while receiving packet on SCC1
```

The first line of the failure identifies what type of failure occurred. The following line provides additional information about the failure.

Table 3-10. SCC1ETH Error Messages

Error Message	Symptom or Cause
`Software error- couldn't find network driver node control memory`	No node control memory was found for the SCC1ETH driver. The NIOT command must also show a non-zero value for the Node Control Memory to prevent this failure.
`Driver error while initializing SCC1, status =XXYY`	The initialization call of the SCC1ETH driver returned an error code. XX is the controller independent status code. YY is the MPC8XX SCC1 ethernet controller dependent status code. See the Network Communication Status Codes section of the PowerPC EPPCBUG Firmware Package Users Manual.
`Driver error while sending packet on SCC1, status =XXYY`	The send packet call of the SCC1ETH driver returned an error code. XX is the controller independent status code. YY is the MPC8XX SCC1 ethernet controller dependent status code. See the Network Communication Status Codes section of the PowerPC *EPPCBUG Firmware Package Users Manual*.

Table 3-10. SCC1ETH Error Messages (Continued)

Error Message	Symptom or Cause
`Driver error while receiving packet on SCC1, status =XXYY`	The packet receive call of the SCC1ETH driver returned an error code. XX is the controller independent status code. YY is the MPC8XX SCC1 ethernet controller dependent status code. See the Network Communication Status Codes section of the PowerPC *EPPCBUG Firmware Package Users Manual*.
`Timeout error while receiving packet on SCC1`	A loopback packet was not received within the time-out period. Indicates a malfunctioning circuit or connection in the loopback path including the loopback connector if one is required. If this error occurs during the TPLOOP test it may indicate that the length of the loopback cable is too short.
`Transmit buffer size %d should be %d to %d inclusive. Proceeding anyway`	This warns the user that the transmit buffer size has been changed by using the CF SCC1ETH command to a value outside of the range allowed by the protocol.

Table 3-10. SCC1ETH Error Messages (Continued)

Error Message	Symptom or Cause
`Sizes are different. Sent packet size =%d, Received packet size =%d`	An incorrect number of bytes was received based on the size of the transmitted loopback packet. Note that normally the size of the received packet is four bytes greater than the size of the transmitted packet due to the addition of the frame checksum.
`Send/receive data miscompare: Sent[%d] =XX, Received[%d] =XX`	The loopback packet received differs from the loopback packet transmitted at the index [%d]. Only the first few occurrences of this error will be displayed.
`Bad ethernet address`	The ethernet address stored in the Vital Product Data Serial ROM does not begin with 0x08003E. It may be un-initialized or corrupted.
`Bad ethernet address length`	Internal firmware error.

Table 3-10. SCC1ETH Error Messages (Continued)

Error Message	Symptom or Cause
`Lookup of AUI connector product configuration option in VPD data failed`	A problem was encountered accessing the AUI header Product Configuration Option in the Vital Product Data Serial ROM. The nature of the problem is displayed just before this message.
`Error writing XX to CR1, SR1 Expected =XX, Actual =XX`	This is caused by an onboard problem accessing the board Control/Status register 1.
`Frame transmit time failure, Expected =%dus to %dus, Actual =%dus`	The time required to transmit a frame is out of range. The expected range and the actual time taken are displayed. The clock source to the MPC8xx transceiver may have diverged from 10MHz.
`Lookup of TP connector product configuration option in VPD data failed`	A problem was encountered accessing the TP connector Product Configuration Option in the Vital Product Data Serial ROM. The nature of the problem is displayed just before this message.

Glossary

Abbreviations, Acronyms, and Terms to Know

This glossary defines some of the abbreviations, acronyms, and key terms used in this document.

10Base-5 An Ethernet implementation in which the physical medium is a doubly shielded, 50-ohm coaxial cable capable of carrying data at 10 Mbps for a length of 500 meters (also referred to as thicknet). Also known as thick Ethernet.

10Base-2 An Ethernet implementation in which the physical medium is a single-shielded, 50-ohm RG58A/U coaxial cable capable of carrying data at 10 Mbps for a length of 185 meters (also referred to as AUI or thinnet). Also known as thin Ethernet.

10Base-T An Ethernet implementation in which the physical medium is an unshielded twisted pair (UTP) of wires capable of carrying data at 10 Mbps for a maximum distance of 185 meters. Also known as twisted-pair Ethernet.

100Base-TX An Ethernet implementation in which the physical medium is an unshielded twisted pair (UTP) of wires capable of carrying data at 100 Mbps for a maximum distance of 100 meters. Also known as fast Ethernet.

ACIA **A**synchronous **C**ommunications **I**nterface **A**dapter

AIX **A**dvanced **I**nteractive e**X**ecutive (IBM version of UNIX)

architecture The main overall design in which each individual hardware component of the computer system is interrelated. The most common uses of this term are 8-bit, 16-bit, or 32-bit architectural design systems.

ASCII **A**merican **S**tandard **C**ode for **I**nformation **I**nterchange. This is a 7-bit code used to encode alphanumeric information. In the IBM-compatible world, this is expanded to 8-bits to encode a total of 256 alphanumeric and control characters.

ASIC **A**pplication-**S**pecific **I**ntegrated **C**ircuit

AUI **A**ttachment **U**nit **I**nterface

BBRAM Battery Backed-up Random Access Memory

bi-endian Having big-endian and little-endian byte ordering capability.

big-endian A byte-ordering method in memory where the address *n* of a word corresponds to the most significant byte. In an addressed memory word, the bytes are ordered (left to right) 0, 1, 2, 3, with 0 being the most significant byte.

BIOS **B**asic **I**nput/**O**utput **S**ystem. This is the built-in program that controls the basic functions of communications between the processor and the I/O (peripherals) devices. Also referred to as ROM BIOS.

BitBLT **Bit** Boundary **BL**ock **T**ransfer. A type of graphics drawing routine that moves a rectangle of data from one area of display memory to another. The data specifically need not have any particular alignment.

BLT **BL**ock **T**ransfer

board The term more commonly used to refer to a PCB (printed circuit board). Basically, a flat board made of nonconducting material, such as plastic or fiberglass, on which chips and other electronic components are mounted. Also referred to as a circuit board or card.

bpi **b**its **p**er **i**nch

bps **b**its **p**er **s**econd

bus The pathway used to communicate between the CPU, memory, and various input/output devices, including floppy and hard disk drives. Available in various widths (8-, 16-, and 32-bit), with accompanying increases in speed.

cache A high-speed memory that resides logically between a central processing unit (CPU) and the main memory. This temporary memory holds the data and/or

instructions that the CPU is most likely to use over and over again and avoids accessing the slower hard or floppy disk drive.

CAS **C**olumn **A**ddress **S**trobe. The clock signal used in dynamic RAMs to control the input of column addresses.

CD **C**ompact **D**isc. A hard, round, flat portable storage unit that stores information digitally.

CD-ROM **C**ompact **D**isk **R**ead-**O**nly **M**emory

CFM **C**ubic **F**eet per **M**inute

CHRP See Common Hardware Reference Platform (CHRP).

CHRP-compliant See Common Hardware Reference Platform (CHRP).

CHRP Spec See Common Hardware Reference Platform (CHRP).

CISC **C**omplex-**I**nstruction-**S**et **C**omputer. A computer whose processor is designed to sequentially run variable-length instructions, many of which require several clock cycles, that perform complex tasks and thereby simplify programming.

CODEC **CO**der/**DEC**oder

Color Difference (CD) The signals of (R-Y) and (B-Y) without the luminance (-Y) signal. The Green signals (G-Y) can be extracted by these two signals.

Common Hardware Reference Platform (CHRP)
A specification published by Apple, IBM, and Motorola which defines the devices, interfaces, and data formats that make up a CHRP-compliant system using a PowerPC processor.

Composite Video Signal (CVS/CVBS)
Signal that carries video picture information for color, brightness and synchronizing signals for both horizontal and vertical scans. Sometimes referred to as “Baseband Video”.

cpi **c**haracters **p**er **i**nch

cpl **c**haracters **p**er **l**ine

CPU **C**entral **P**rocessing **U**nit. The master computer unit in a system.

DCE **D**ata **C**ircuit-terminating **E**quipment.

DLL **D**ynamic **L**ink **L**ibrary. A set of functions that are linked to the referencing program at the time it is loaded into memory.

DMA **D**irect **M**emory **A**ccess. A method by which a device may read or write to memory directly without processor intervention. DMA is typically used by block I/O devices.

DOS **D**isk **O**perating **S**ystem

dpi **d**ots **p**er **i**nch

DRAM **D**ynamic **R**andom **A**ccess **M**emory. A memory technology that is characterized by extreme high density, low power, and low cost. It must be more or less continuously refreshed to avoid loss of data.

DTE **D**ata **T**erminal **E**quipment.

ECC **E**rror **C**orrection **C**ode

ECP **E**xtended **C**apability **P**ort

EEPROM **E**lectrically **E**rasable **P**rogrammable **R**ead-**O**nly **M**emory. A memory storage device that can be written repeatedly with no special erasure fixture. EEPROMs do not lose their contents when they are powered down.

EIDE **E**nhanced **I**ntegrated **D**rive **E**lectronics. An improved version of **IDE**, with faster data rates, 32-bit transactions, and DMA. Also known as **Fast ATA-2**.

EISA (bus) **E**xtended **I**ndustry **S**tandard **A**rchitecture (bus) (IBM). An architectural system using a 32-bit bus that allows data to be transferred between peripherals in 32-bit chunks instead of 16-bit or 8-bit that most systems use. With the transfer of larger bits of information, the machine is able to perform much faster than the standard ISA bus system.

EPP **E**nhanced **P**arallel **P**ort

EPROM — **E**rasable **P**rogrammable **R**ead-**O**nly **M**emory. A memory storage device that can be written once (per erasure cycle) and read many times.

ESCC — **E**nhanced **S**erial **C**ommunication **C**ontroller

ESD — **E**lectro-**S**tatic **D**ischarge/Damage

Ethernet — A local area network standard that uses radio frequency signals carried by coaxial cables.

Falcon — The DRAM controller chip developed by Motorola for the MVME2600 and MVME3600 series of boards. It is intended to be used in sets of two to provide the necessary interface between the Power PC60*x* bus and the 144-bit ECC DRAM (system memory array) and/or ROM/Flash.

fast Ethernet — See 100Base-TX.

FDC — **F**loppy **D**isk **C**ontroller

FDDI — **F**iber **D**istributed **D**ata **I**nterface. A network based on the use of optical-fiber cable to transmit data in non-return-to-zero, invert-on-1s (NRZI) format at speeds up to 100 Mbps.

FIFO — **F**irst-**I**n, **F**irst-**O**ut. A memory that can temporarily hold data so that the sending device can send data faster than the receiving device can accept it. The sending and receiving devices typically operate asynchronously.

firmware — The program or specific software instructions that have been more or less permanently burned into an electronic component, such as a ROM (read-only memory) or an EPROM (erasable programmable read-only memory).

frame — One complete television picture frame consists of 525 horizontal lines with the NTSC system. One frame consists of two Fields.

graphics controller — On EGA and VGA, a section of circuitry that can provide hardware assist for graphics drawing algorithms by performing logical functions on data written to display memory.

HAL — **H**ardware **A**bstraction **L**ayer. The lower level hardware interface module of the Windows NT operating system. It contains platform specific functionality.

hardware A computing system is normally spoken of as having two major components: hardware and software. Hardware is the term used to describe any of the physical embodiments of a computer system, with emphasis on the electronic circuits (the computer) and electromechanical devices (peripherals) that make up the system.

HCT **H**ardware **C**onformance **T**est. A test used to ensure that both hardware and software conform to the Windows NT interface.

I/O **I**nput/**O**utput

IBC PCI/**I**SA **B**ridge **C**ontroller

IDC **I**nsulation **D**isplacement **C**onnector

IDE **I**ntegrated **D**rive **E**lectronics. A disk drive interface standard. Also known as **ATA** (**A**dvanced **T**echnology **A**ttachment).

IEEE **I**nstitute of **E**lectrical and **E**lectronics **E**ngineers

interlaced A graphics system in which the even scanlines are refreshed in one vertical cycle (field), and the odd scanlines are refreshed in another vertical cycle. The advantage is that the video bandwidth is roughly half that required for a non-interlaced system of the same resolution. This results in less costly hardware. It also may make it possible to display a resolution that would otherwise be impossible on given hardware. The disadvantage of an interlaced system is flicker, especially when displaying objects that are only a few scanlines high.

IQ Signals Similar to the color difference signals (R-Y), (B-Y) but using different vector axis for encoding or decoding. Used by some USA TV and IC manufacturers for color decoding.

ISA (bus) **I**ndustry **S**tandard **A**rchitecture (bus). The de facto standard system bus for IBM-compatible computers until the introduction of VESA and PCI. Used in the reference platform specification. (IBM)

ISASIO **ISA** **S**uper **I**nput/**O**utput device

ISDN	**I**ntegrated **S**ervices **D**igital **N**etwork. A standard for digitally transmitting video, audio, and electronic data over public phone networks.
LAN	**L**ocal **A**rea **N**etwork
LED	**L**ight-**E**mitting **D**iode
LFM	**L**inear **F**eet per **M**inute
little-endian	A byte-ordering method in memory where the address *n* of a word corresponds to the least significant byte. In an addressed memory word, the bytes are ordered (left to right) 3, 2, 1, 0, with 3 being the most significant byte.
MBLT	**M**ultiplexed **BL**ock **T**ransfer
MCA (bus)	**M**icro **C**hannel **A**rchitecture
MCG	**M**otorola **C**omputer **G**roup
MFM	**M**odified **F**requency **M**odulation
MIDI	**M**usical **I**nstrument **D**igital **I**nterface. The standard format for recording, storing, and playing digital music.
MPC	**M**ultimedia **P**ersonal **C**omputer
MPC105	The PowerPC-to-PCI bus bridge chip developed by Motorola for the Ultra 603/Ultra 604 system board. It provides the necessary interface between the MPC603/MPC604 processor and the Boot ROM (secondary cache), the DRAM (system memory array), and the PCI bus.
MPC601	Motorola's component designation for the PowerPC 601 microprocessor.
MPC603	Motorola's component designation for the PowerPC 603 microprocessor.
MPC604	Motorola's component designation for the PowerPC 604 microprocessor.
MPIC	**M**ulti-**P**rocessor **I**nterrupt **C**ontroller
MPU	**M**icro**P**rocessing **U**nit
MTBF	**M**ean **T**ime **B**etween **F**ailures. A statistical term relating to reliability as expressed in power on hours (poh). It was originally developed for the military and can be calculated

	several different ways, yielding substantially different results. The specification is based on a large number of samplings in one place, running continuously, and the rate at which failure occurs. MTBF is not representative of how long a device, or any individual device is likely to last, nor is it a warranty, but rather, a gauge of the relative reliability of a family of products.
multisession	The ability to record additional information, such as digitized photographs, on a CD-ROM after a prior recording session has ended.
non-interlaced	A video system in which every pixel is refreshed during every vertical scan. A non-interlaced system is normally more expensive than an interlaced system of the same resolution, and is usually said to have a more pleasing appearance.
nonvolatile memory	A memory in which the data content is maintained whether the power supply is connected or not.
NTSC	**N**ational **T**elevision **S**tandards **C**ommittee (USA)
NVRAM	**N**on-**V**olatile **R**andom **A**ccess **M**emory
OEM	**O**riginal **E**quipment **M**anufacturer
OMPAC	**O**ver - **M**olded **P**ad **A**rray **C**arrier
OS	**O**perating **S**ystem. The software that manages the computer resources, accesses files, and dispatches programs.
OTP	**O**ne-**T**ime **P**rogrammable
palette	The range of colors available on the screen, not necessarily simultaneously. For VGA, this is either 16 or 256 simultaneous colors out of 262,144.
parallel port	A connector that can exchange data with an I/O device eight bits at a time. This port is more commonly used for the connection of a printer to a system.
PCI (local bus)	**P**eripheral **C**omponent **I**nterconnect (local bus) (Intel). A high-performance, 32-bit internal interconnect bus used for data transfer to peripheral controller components, such as those for audio, video, and graphics.

PCMCIA (bus) — **P**ersonal **C**omputer **M**emory **C**ard **I**nternational **A**ssociation (bus). A standard external interconnect bus which allows peripherals adhering to the standard to be plugged in and used without further system modification.

PCR — **P**CI **C**onfiguration **R**egister

PDS — **P**rocessor **D**irect **S**lot

PHB — **P**CI **H**ost **B**ridge

physical address — A binary address that refers to the actual location of information stored in secondary storage.

PIB — **P**CI-to-**I**SA **B**ridge

pixel — An acronym for picture element, and is also called a pel. A pixel is the smallest addressable graphic on a display screen. In RGB systems, the color of a pixel is defined by some Red intensity, some Green intensity, and some Blue intensity.

PLL — **P**hase-**L**ocked **L**oop

PMC — **P**CI **M**ezzanine **C**ard

POWER — **P**erformance **O**ptimized **W**ith **E**nhanced **R**ISC architecture (IBM)

PowerPC™ — The trademark used to describe the **P**erformance **O**ptimized **W**ith **E**nhanced **R**ISC microprocessor architecture for **P**ersonal **C**omputers developed by the IBM Corporation. PowerPC is superscalar, which means it can handle more than one instruction per clock cycle. Instructions can be sent simultaneously to three types of independent execution units (branch units, fixed-point units, and floating-point units), where they can execute concurrently, but finish out of order. PowerPC is used by Motorola, Inc. under license from IBM.

PowerPC 601™ — The first implementation of the PowerPC family of microprocessors. This CPU incorporates a memory management unit with a 256-entry buffer and a 32KB unified (instruction and data) cache. It provides a 64-bit data bus and a separate 32-bit address bus. PowerPC 601 is used by Motorola, Inc. under license from IBM.

PowerPC 603™ The second implementation of the PowerPC family of microprocessors. This CPU incorporates a memory management unit with a 64-entry buffer and an 8KB (instruction and data) cache. It provides a selectable 32-bit or 64-bit data bus and a separate 32-bit address bus. PowerPC 603 is used by Motorola, Inc. under license from IBM.

PowerPC 604™ The third implementation of the PowerPC family of microprocessors currently under development. PowerPC 604 is used by Motorola, Inc. under license from IBM.

PowerPC Reference Platform (PRP) A specification published by the IBM Power Personal Systems Division which defines the devices, interfaces, and data formats that make up a PRP-compliant system using a PowerPC processor.

PowerStack™ RISC PC (System Board) A PowerPC-based computer board platform developed by the Motorola Computer Group. It supports Microsoft's Windows NT and IBM's AIX operating systems.

PRP See PowerPC Reference Platform (PRP).

PRP-compliant See PowerPC Reference Platform (PRP).

PRP Spec See PowerPC Reference Platform (PRP).

PROM **P**rogrammable **R**ead-**O**nly **M**emory

PS/2 **P**ersonal **S**ystem/**2** (IBM)

QFP **Q**uad **F**lat **P**ackage

RAM **R**andom-**A**ccess **M**emory. The temporary memory that a computer uses to hold the instructions and data currently being worked with. All data in RAM is lost when the computer is turned off.

RAS **R**ow **A**ddress **S**trobe. A clock signal used in dynamic RAMs to control the input of the row addresses.

Raven The PowerPC-to-PCI local bus bridge chip developed by Motorola for the MVME2600 and MVME3600 series of boards. It provides the necessary interface between the PowerPC 60x bus and the PCI bus, and acts as interrupt controller.

Reduced-Instruction-Set Computer (RISC) A computer in which the processor's instruction set is limited to constant-length instructions that can usually be executed in a single clock cycle.

RFI **R**adio **F**requency **I**nterference

RGB The three separate color signals: **R**ed, **G**reen, and **B**lue. Used with color displays, an interface that uses these three color signals as opposed to an interface used with a monochrome display that requires only a single signal. Both digital and analog RGB interfaces exist.

RISC See Reduced Instruction Set Computer (RISC).

ROM **R**ead-**O**nly **M**emory

RTC **R**eal-**T**ime **C**lock

SBC **S**ingle **B**oard **C**omputer

SCSI **S**mall **C**omputer **S**ystems **I**nterface. An industry-standard high-speed interface primarily used for secondary storage. SCSI-1 provides up to 5 Mbps data transfer.

SCSI-2 (Fast/Wide) An improvement over plain SCSI; and includes command queuing. Fast SCSI provides 10 Mbps data transfer on an 8-bit bus. Wide SCSI provides up to 40 Mbps data transfer on a 16- or 32-bit bus.

serial port A connector that can exchange data with an I/O device one bit at a time. It may operate synchronously or asynchronously, and may include start bits, stop bits, and/or parity.

SIM **S**erial **I**nterface **M**odule

SIMM **S**ingle **I**nline **M**emory **M**odule. A small circuit board with RAM chips (normally surface mounted) on it designed to fit into a standard slot.

SIO	**S**uper **I/O** controller
SMP	**S**ymmetric **M**ulti**P**rocessing. A computer architecture in which tasks are distributed among two or more local processors.
SMT	**S**urface **M**ount **T**echnology. A method of mounting devices (such as integrated circuits, resistors, capacitors, and others) on a printed circuit board, characterized by not requiring mounting holes. Rather, the devices are soldered to pads on the printed circuit board. Surface-mount devices are typically smaller than the equivalent through-hole devices.
software	A computing system is normally spoken of as having two major components: hardware and software. Software is the term used to describe any single program or group of programs, languages, operating procedures, and documentation of a computer system. Software is the real interface between the user and the computer.
SRAM	**S**tatic **R**andom **A**ccess **M**emory
SSBLT	**S**ource **S**ynchronous **BL**ock **T**ransfer
standard(s)	A set of detailed technical guidelines used as a means of establishing uniformity in an area of hardware or software development.
SVGA	**S**uper **V**ideo **G**raphics **A**rray (IBM). An improved VGA monitor standard that provides at least 256 simultaneous colors and a screen resolution of 800 x 600 pixels.
Teletext	One way broadcast of digital information. The digital information is injected in the broadcast TV signal, VBI, or full field, The transmission medium could be satellite, microwave, cable, etc. The display medium is a regular TV receiver.
thick Ethernet	See 10base-5.
thin Ethernet	See 10base-2.
twisted-pair Ethernet	See 10Base-T.
UART	**U**niversal **A**synchronous **R**eceiver/**T**ransmitter

Universe ASIC developed by Tundra in consultation with Motorola, that provides the complete interface between the PCI bus and the 64-bit VMEbus.

UV **U**ltra**V**iolet

UVGA **U**ltra **V**ideo **G**raphics **A**rray. An improved VGA monitor standard that provides at least 256 simultaneous colors and a screen resolution of 1024 x 768 pixels.

Vertical Blanking Interval (VBI) The time it takes the beam to fly back to the top of the screen in order to retrace the opposite field (odd or even). VBI is in the order of 20 TV lines. Teletext information is transmitted over 4 of these lines (lines 14-17).

VESA (bus) **V**ideo **E**lectronics **S**tandards **A**ssociation (or VL bus). An internal interconnect standard for transferring video information to a computer display system.

VGA **V**ideo **G**raphics **A**rray (IBM). The third and most common monitor standard used today. It provides up to 256 simultaneous colors and a screen resolution of 640 x 480 pixels.

virtual address A binary address issued by a CPU that indirectly refers to the location of information in primary memory, such as main memory. When data is copied from disk to main memory, the physical address is changed to the virtual address.

VL bus See **V**ESA **L**ocal bus (VL bus).

VMEchip2 MCG second generation VMEbus interface ASIC (Motorola)

VME2PCI MCG ASIC that interfaces between the PCI bus and the VMEchip2 device.

volatile memory A memory in which the data content is lost when the power supply is disconnected.

VRAM **V**ideo (Dynamic) **R**andom **A**ccess **M**emory. Memory chips with two ports, one used for random accesses and the other capable of serial accesses. Once the serial port has been initialized (with a transfer cycle), it can operate independently of the random port. This frees the random

port for CPU accesses. The result of adding the serial port is a significantly reduced amount of interference from screen refresh. VRAMs cost more per bit than DRAMs.

Windows NT™ The trademark representing **Windows N**ew **T**echnology, a computer operating system developed by the Microsoft Corporation.

XGA E**X**tended **G**raphics **A**rray. An improved IBM VGA monitor standard that provides at least 256 simultaneous colors and a screen resolution of 1024 x 768 pixels.

Y Signal Luminance. This determines the brightness of each spot (pixel) on a CRT screen either color or B/W systems, but not the color.

Index

L

M

N

P

Q

R

S

T

U

W

Z

INDEX