IBM Advanced Interactive Executive/370
(AIX/370)
Diagnosis Guide
Version 1.2.1

Document Number SC23-2090-01

**IBM Advanced Interactive Executive/370
(AIX/370)**

Diagnosis Guide

Version 1.2.1

Document Number SC23-2090-01

Edition Notice

Second Edition (March 1991)

This edition applies to Version 1.2.1 of the IBM Advanced Interactive Executive for System/370 (AIX/370), Program Number 5713-AFL, and to all subsequent releases until otherwise indicated in new editions or technical newsletters. Make sure you are using the correct edition for the level of the product.

Order publications through your IBM representative or the IBM branch office serving your locality. Publications are not stocked at the address given below.

A form for reader's comments appears at the back of this publication. If the form has been removed, address your comments to:

IBM Corporation, Department 52QA MS 911
Neighborhood Road
Kingston, NY 12401
U.S.A.

Notices

References in this publication to IBM products, programs, or services do not imply that IBM intends to make these available in all countries in which IBM operates. Any reference to an IBM product, program, or service is not intended to state or imply that only IBM's product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any of IBM's intellectual property rights or other legally protectible rights may be used instead of the IBM product, program, or service. Evaluation and verification of operation in conjunction with other products, programs, or services, except those expressly designated by IBM, are the user's responsibility.

IBM may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to the IBM Director of Commercial Relations, IBM Corporation, Purchase, NY 10577.

Subtopics

Trademarks and Acknowledgments

Portions of the code and documentation were developed at the Electrical Engineering and Computer Sciences Department at the Berkeley Campus of the University of California under the auspices of the Regents of the University of California.

About This Book

Many events, some unexpected, may affect the operation of the Advanced Interactive Executive (AIX/370). When these events occur, it becomes necessary to pinpoint the cause of the problem. This guide provides procedures and tools to help you define and categorize symptoms of problems. It also helps you to determine the cause and source of an error, which are among the first steps in resolving a problem.

Subtopics

Who Should Read This Book

This manual is intended for the system administrator in an environment that uses AIX/370. It helps the system administrator to use IBM's support to resolve problems that may arise from operating AIX/370.

What You Should Know

This book assumes an understanding of:

- Computer hardware and softwar
- Communication system
- The AIX operating syste
- The hardware and software installed at your installation

How to Use This Book

If you believe a problem is a software error and you want to do error diagnosis, go directly to Chapter 2, "Procedures for Diagnosing Problems" in topic 1.2, which gives procedures for identifying errors and helps you find the source of an error. You will find an overview of error types under "Possible Problems - Overview" in topic 1.1.9.

For a general overview of problem diagnosis, read Chapter 1, "Introduction to Problem Diagnosis" in topic 1.1, which describes the AIX/370, AIX PS/2, and Transparent Computing Facility (TCF) environments. This chapter also outlines the steps you take to resolve a problem and describes your role and IBM's responsibilities in the problem determination process.

Subtopics

Highlighting

This book using the following highlighting conventions:

All AIX commands, options, parameters, names of keys, keywords, an actual file names are in **boldface** type.
Protocols are in UPPERCASE type
Environment variables in **UPPERCASE** boldface type.
New terms introduced in the text are shown in ***boldface italic***.
Variable information is in *italic* type.
Anything a user types is in **monospace** type.
Anything appearing on a display screen that is referred to in paragraph of text is in **monospace** type.
If the instruction is set off from the paragraph, it is printed i **monospace** type.

Related Publications

For additional information, you may want to refer to the following publications:

AIX Access for DOS Users Administrator's Guide, SC23-2042, explains how to install and administer the AIX Access for DOS Users program on the IBM PS/2, RT, and System/370 computers running the AIX Operating System with the AIX DOS Server. It covers the responsibilities for installation, daily operation, and maintenance of the AIX Access program.

AIX Access for DOS Users User's Guide, SC23-2041, describes the AIX Access for DOS Users program and shows how to use the file services of an AIX host while running DOS applications.

AIX C Language Reference, SC23-2058, describes the C programming language and contains reference information for writing programs in C language that run on the AIX Operating System.

AIX C Language User's Guide, SC23-2057, describes how to develop, link, and execute C language programs. This book also describes the operating dependencies of C language and shows how to use C language-related software utilities and other program development tools.

AIX Commands Reference, SC23-2292 (Vol. 1) and SC23-2184 (Vol. 2), lists and describes the AIX/370 and AIX PS/2 Operating System commands.

AIX Guide to Multibyte Character Set (MBCS) Support, GC23-2333, explains the basic concepts of AIX multibyte character set support and refers to other AIX books that contain more detailed information.

AIX Library Guide, Glossary, and Master Index, SC23-2324, describes the publications in the AIX Operating System library and contains a glossary of terms used throughout the library. This book also includes a master index to the contents of each of the publications in the library.

AIX Messages Reference, SC23-2294, lists messages displayed by the AIX Operating System and explains how to respond to them.

AIX Programming Tools and Interfaces, SC23-2304, describes the programming environment of the AIX Operating System and includes information about operating system tools that are used to develop, compile, and debug programs.

AIX TCP/IP User's Guide, SC23-2309, describes the features of TCP/IP and shows how to install and customize the program. It includes reference information on TCP/IP commands that are used to transfer files, manage the network, and log into remote systems.

AIX Technical Reference, SC23-2300 (Vol. 1) and SC23-2301 (Vol. 2), describes the system calls and subroutines a programmer uses to write application programs. This book also provides information about the AIX Operating System file system, special files, miscellaneous files, and the writing of device drivers.

AIX VS FORTRAN Reference, SC23-2050, describes the FORTRAN programming language as implemented on AIX RT, AIX PS/2, and AIX/370. This book describes all of the standard features of VS FORTRAN as well as the enhanced functions and capabilities incorporated into IBM AIX VS FORTRAN.

AIX VS FORTRAN User's Guide, SC23-2049, shows how to develop and execute FORTRAN programs on AIX RT, AIX PS/2, and AIX/370. This book also explains how to compile and execute programs that contain sections of code written in the VS Pascal and C programming languages.

AIX VS Pascal Reference, SC23-2054, describes the VS Pascal programming language as implemented on the IBM PS/2 or RT with the AIX Operating System installed. This book describes all of the standard features of Pascal as well as the enhanced functions and capabilities incorporated into IBM AIX VS Pascal.

AIX VS Pascal User's Guide, SC23-2053, shows how to develop and execute Pascal programs on the IBM PS/2 and RT using the AIX Operating System. This book also explains how to compile and execute programs that contain sections of code written in the VS FORTRAN and C programming languages.

AIX X-Windows Programmer's Reference, SC23-2118, describes the X-Windows licensed program and provides information on X-Windows library functions, FORTRAN subroutines, protocols, and extensions.

AIX X-Windows User's Guide, SC23-2017, describes the X-Windows licensed program and shows how to start, run, install, and customize this program.

AIX PS/2 DOS Merge User's and Administrator's Guide, SC23-2045, shows how to use DOS in the AIX environment, including running DOS and AIX programs simultaneously and running AIX commands from the DOS environment. It also shows how to install the DOS Merge software and how to perform essential system maintenance activities, such as adding user accounts, backing up the file system, and setting up terminals.

AIX PS/2 General Information, GC23-2055, describes the AIX PS/2 Operating System's functions and capabilities and the product's position in the AIX family of products.

AIX PS/2 INed, SC23-2001, shows how to use the INed editor to create, access, and store files. This book also includes reference information on INed commands and a listing of INed error messages.

AIX PS/2 INmail/INnet/INftp User's Guide, SC23-2076, describes the INmail/INnet/INftp/Connect programs and shows how to use these programs to send mail to and receive mail from local and remote computer systems. This book also shows how to transfer files to and from other computer systems installed on the network.

AIX PS/2 Interface Library Reference, SC23-2051, contains information about the library of system calls available with IBM AIX VS Pascal and IBM AIX VS FORTRAN as implemented for use with the IBM AIX PS/2 Operating System.

AIX PS/2 Keyboard Description and Character Reference, SC23-2037, describes the characters and keyboards supported by the AIX PS/2 Operating System. This book also provides information on keyboard position codes, keyboard states, control code points, code-sequence processing, and non-spacing character sequences.

AIX PS/2 Text Formatting Guide, SC23-2044, describes the text formatting utilities available on the PS/2 and shows how to format text with NROFF and TROFF. This book also shows how to use the **vi** editor to create, revise, and store files.

AIX PS/2 Usability Services Reference, SC23-2039, lists and describes Usability Services commands.

AIX PS/2 Usability Services User's Guide, SC23-2038, shows how to create and print text files, work with directories, start application programs, and do other basic tasks with Usability Services.

AIX/370 Administration Guide, SC23-2088, describes such administrative tasks as updating the file system, backing up files, and fine-tuning and monitoring the performance of the operating system.

AIX/370 Diagnosis Guide, SC23-2090, describes procedures and tools that can be used to define and categorize symptoms of problems that may occur during daily operation.

AIX/370 General Information, GC23-2062, describes the functions and capabilities of AIX/370 and its position in the AIX family of products.

AIX/370 Planning Guide, GC23-2065, describes the functions and capabilities of the AIX/370 Operating System and lists the requirements for all supported hardware and software. This book also includes information to assist with planning for installation and customization of the operating system.

Installing and Customizing the AIX PS/2 Operating System, SC23-2290, provides step-by-step instructions for installing the AIX PS/2 Operating System and related programs. This book also shows how to customize the operating system to suit the user's specific needs and work environment.

Installing and Customizing the AIX/370 Operating System, SC23-2066, provides step-by-step instructions for installing the AIX/370 Operating System and related programs. This book also shows how to customize the operating system to suit the user's specific needs and work environment.

Managing the AIX Operating System, SC23-2293, describes such system-management tasks as adding and deleting user IDs, creating and mounting file systems, backing up the system, repairing file system damage, and setting up an electronic mail system and other networking facilities.

Using the AIX Operating System, SC23-2291, shows the beginning user how to use AIX Operating System commands to do such basic tasks as log in and out of the system, display and print files, and set and change passwords. It includes information for intermediate to advanced users about how to use communication and networking facilities and write shell procedures.

Other Publications

The following books may also be of interest:

VM/SP Problem Solving and Reporting Guide, SC24-5282

VM/SP System Messages and Codes, SC19-6204

Environmental Record Editing and Printing Program User's Guide and Reference, GC28-1378

VM/SP OLTSEP and Error Recording Guide, SC19-6205

System/370 Principles of Operation, GA22-7000

System/370 Extended Architecture Principles of Operation, SA22-7085

Table of Contents

FIGURES Figures

1.0 Part 1: Problem Diagnosis

This part of the book gives you an overview of the terms used when diagnosing a problem. The various steps to define the type of problem are shown.

You will use the appropriate step-by-step procedure to identify the source of a problem. These procedures tell you at least what information is needed to document a problem and how to recover from it.

This part contains the following chapters:

Subtopics

1.1 Chapter 1. Introduction to Problem Diagnosis

Subtopics

1.1.1 Contents

1.1.2 About This Chapter

This chapter describes the AIX/370, AIX PS/2, and Transparent Computing Facility (TCF) environments, and the steps required for problem determination and problem source identification. Reading this chapter will help you to understand the steps and responsibilities in problem determination, and to identify the various error types and take the appropriate action.

1.1.3 Problem Diagnosis Steps

Whenever a problem occurs, follow the steps described below.

1. Determine which TCF sites are affected by the problems. Most problems only affect one or a few TCF sites. Some types of TCF communication problems (for example, LAN hardware) may affect all TCF sites. Refer to "Isolation of the Problem Within the Cluster" in topic 1.1.5.

2. Determine if the problem is caused by hardware or software. Refer to "Problem Determination" in topic 1.1.6.1.

3. If the error is caused by software, isolate the error. Refer to "Problem Source Identification" in topic 1.1.6.2.

4. Gather error-related information for in-depth investigation. Refer to Chapter 3, "Collecting Information about a Problem."

5. Recover from the error, using your locally established procedures.

6. Based on material you have collected, investigate the software error. Refer to Chapter 2, "Procedures for Diagnosing Problems."

7. Submit problem definition to IBM (if IBM code is at fault). Refer to Chapter 4, "How to Report Problems to IBM."

8. Install the IBM solution for the problem. For information on how to install problem corrections supplied by IBM, refer to *AIX/370 Administration Guide* and *AIX Operating System Commands Reference*.

9. Provide feedback to IBM on the results to help verify any fixes.

1.1.4 Overview of the Service Concept for AIX/370

Customer Activities	IBM Responsibilities
1. Do initial evaluation to determine whether the problem is in the IBM code.	
2. If IBM code is at fault, build a symptom string and report it to your IBM support center.	Enter the problem in RETAIN, and search RETAIN with customer-supplied symptom string as a search argument.
	Furnish search results to the customer:
	The problem is known:
3. Install the fix offered by IBM (it may be in form of a PTF or an APAR).	
	or:
	The problem is new:
Verify your original findings to make sure that IBM code has caused the problem.	
4. If further analysis confirms your original assumption, submit information for an APAR.	Process the APAR; provide a fix.
5. Install the IBM-supplied fix and provide feedback on the results.	

1.1.5 Isolation of the Problem Within the Cluster

Subtopics

1.1.5.1 Problems on a Single Cluster Site

Most problems are limited to a single Transparent Computing Facility (TCF) site. When such a problem occurs, the rest of the TCF cluster continues normal operation, except that resources on the site experiencing the problem may not be available. A problem may be limited to a single site if the specific indications of a problem are limited to one particular TCF site, and the rest of the TCF cluster is able to continue normal operation.

When a problem is determined to be limited to a single TCF site, it may be useful to determine if the problem can be repeated on another TCF site of the same machine type. Many system and user problems can be repeatable on other TCF sites.

Note: Do not attempt to replicate a problem that may cause severe outages or data loss.

Hardware problems and some types of software problems involving differences in the environment are not likely to be repeated on other TCF sites. Problems that involve subtle timing differences may not be repeatable even on the same TCF site. For this reason, any collection of information (such as system dumps) should be done before the system is restored to determine if the problem is repeatable.

1.1.5.2 Problems on Multiple Cluster Sites

Some types of problems affect more than one TCF site. Such problems may be indicated by more than one TCF site becoming inactive at once, the appearance of similar symptoms on more than one TCF site, or by specific messages indicating problems with specific TCF sites. One general class of problems that often involves more than one TCF site is TCF communication problems.

1.1.6 Problem Determination and Problem Source Identification

Subtopics

1.1.6.1 Problem Determination

Problem determination is the first step to resolve a problem. When you experience a problem, you have to determine whether the cause of the problem is hardware or software.

Subtopics

1.1.6.1.1 What is Hardware?

Hardware is all the physical devices and connections between those devices. The devices may include:

- The central processing unit (CPU), or mainframe
- DASD (direct access storage device)
- Tape drives
- Local Area Network (LAN) interface devices
- The physical LAN
- Terminals attached to the CPU.

Although most hardware errors are obvious, some hardware problems appear to be problems with software. Some hardware problems are intermittent; their effects may not always have the same characteristics.

1.1.6.1.2 What is Software?

Programs that run within the hardware and in most cases control the hardware are called software.

Software may be originated by:

End users
Application programmers
Vendors
IBM.

Because of the many possible interactions between programs of like or differing origins, it may be difficult to isolate the failing software code.

1.1.6.1.3 What Next?

If you established that hardware is the cause of a problem, follow up through the appropriate support procedures as described in the documentation for the hardware.

If you established that software is the cause of a problem, continue with "Problem Source Identification," which follows.

1.1.6.2 Problem Source Identification

Problem source identification is the first step in resolving a *software problem*.

In this step, you try to isolate the problem to specific functionally related code (component); that is, you narrow down the failure to one, or a combination, of the following causes:

- The AIX syste
- The VM (Virtual Machine) syste
- Remote cluster sit
- The LAN or LAN interfac
- User erro
- Non-IBM progra
- User-written program

When you have completed the problem source identification, you can contact whomever is responsible for the problem code. That source may be inside or outside your own organization, or IBM may be responsible for the problem code.

1.1.7 The UNIX Environment

The AIX/370 system with TCF is IBM's implementation of a transparent distributed operating system based on UNIX System V. This system supports several TCF sites that communicate via a shared LAN, or through channel-to-channel attachment, to provide transparent access to distributed operating system resources. The currently supported TCF site types are System/370, System/370 XA, and PS/2s. AIX/370 runs only as a guest operating system under VM. The required levels of VM or VM/XA are described in *AIX/370 General Information*.

AIX/370 does not support directly attached RS232 terminals. RS232 terminals are attached to AIX PS/2 terminals and use TCF services for access to resources. Since TCF provides a very high level of transparency, users need not be aware of this distinction. Alternatively, users have access to the TCF cluster through TCP/IP, PCI (terminal emulation or file services), or X-Windows.

This manual focuses on diagnosis of AIX TCF sites.

AIX/370 is designed to run as a virtual machine under VM. Virtual machines under VM behave as if they were in total control of the CPU. VM performs the following functions for AIX/370:

Translates real device addresses to virtual addresse
Translates virtual device addresses to real addresse
Records hardware and software error
Controls the real page storage utilizatio
Provides some supervisor support functions

AIX uses the Control Program (CP) component of VM. If CP has a problem, it displays a message with the following format:

```
DMKAAANNNL
```

where:

```
AAA = Alpha name of the issuing module
NNN = Numeric code of the exact message
L   = Alpha indicator of the level of failure
```

Note: EMSG must be set 'on' to see this message identifier.

At times, VM stops, takes a dump of storage, and reloads. If this happens, notify the VM support personnel (further information may be obtained in *VM/SP System Messages and Codes*).

Subtopics

1.1.7.1 Consoles and Terminals

It is important to understand the terminology and functions associated with consoles and terminals in the VM environment. Figure 1-1 shows an overview of the consoles and terminals used.

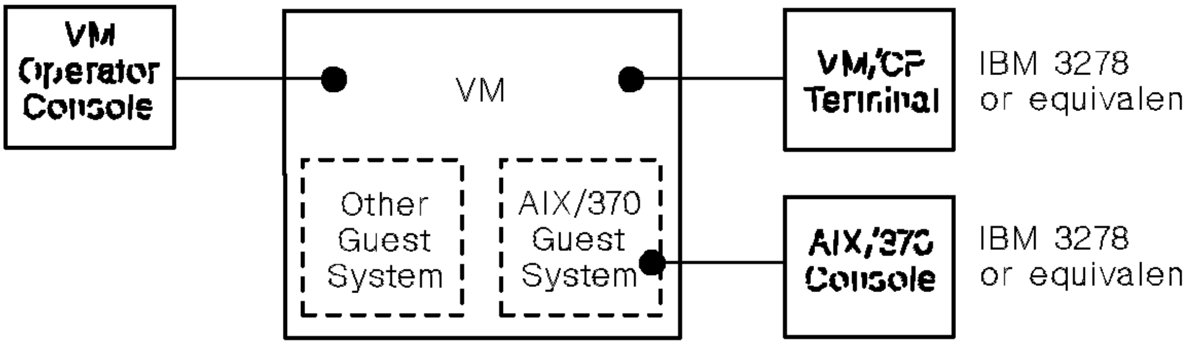

Figure 1-1. Consoles and Terminals for AIX/370

AIX/370 Console:

An individual AIX/370 TCF site system is loaded from the AIX/370 VM console. All AIX/370 operation commands can be entered from this console. Control Program (CP) commands may be entered from the AIX/370 console using the command **cpcmd**.

For information about control sequences and key mapping, refer to Appendix information in *Installing and Customizing the AIX/370 Operating System*.

Note: In all examples in this book, *AIX/370* is the user ID of the AIX/370 virtual machine. The user ID may be different at your location.

VM Operator Console:

One terminal, usually the real system console, where VM was originally loaded.

VM/CP Terminal:

This terminal is an IBM 3278 or equivalent and is defined to VM (that is, you can logon to VM or dial to another virtual machine. When the Conversational Monitor System (CMS) is loaded, CP and CMS commands are supported.

Once an AIX/370 TCF site is loaded, this VM/CP terminal has the following characteristics:

It serves as a hardware console during CP simulation of the hardwar function for AIX/370.

If the AIX/370 console is disconnected, the VM/CP console can becom the AIX/370 console to control AIX/370 by means of the following actions:

- Logon to the virtual machine in which AIX/370 is running.
- Enter your password.
- Press **ENTER**.

- Press **<PA2>**.

Personal Computer (PC) User Terminal:

Some end users use an IBM/PC to emulate an ASCII terminal attached to an AIX/370 TCF site. These terminals may be attached directly to an AIX PS/2 TCF site or indirectly through a network device using program-controlled interruption (PCI) emulation. In most cases, these terminals are logically identical to a conventional AIX PS/2 terminal.

AIX PS/2 Consoles:

In the PS/2 environment the console is more clearly identified. It is the keyboard and video display unit of the PS/2. See Figure 1-2.

AIX PS/2 Terminal:

ASCII terminals may be connected to the PS/2. These terminals are connected using a serial port or one of the ports on an RTIC card. Refer to Figure 1-2.

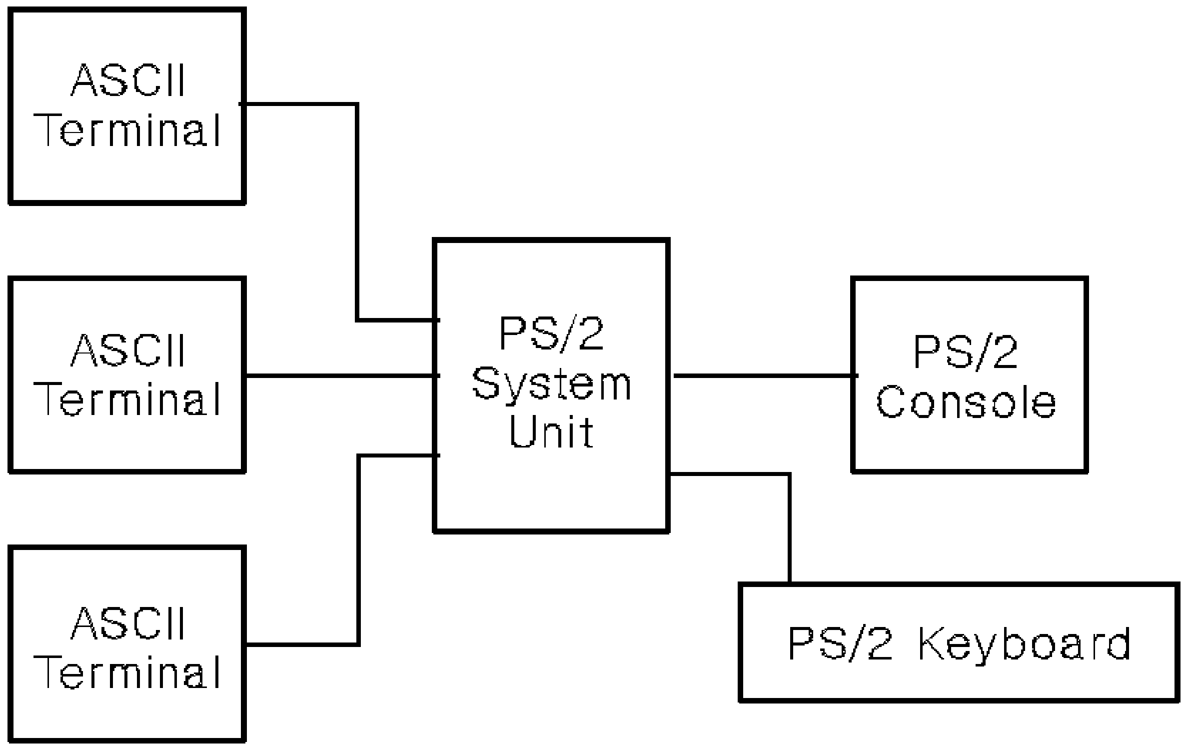

Figure 1-2. Console and Terminal PS/2 Configuration

1.1.8 The LAN Interface Device

The LAN interface device attaches an AIX TCF site to a communication LAN (Ethernet or Token-Ring). On an AIX PS/2 site the LAN interface device is a microchannel-compatible card. This device must operate correctly for the cluster site to operate within the TCF environment.

Errors on AIX/370 systems with the LAN interface device may be indicated either by the device itself (status lights, for example) in the AIX/370 environment, or by messages on the AIX console on the TCF site where the failing LAN device is attached. On AIX PS/2 sites, the errors are reported with messages on the system console and in the system error log.

Note: The **errpt** command processes a report of errors logged by the system. The default report summarizes all errors posted in the specified error file as well as information such as system starts when available. Refer to *AIX Operating System Commands Reference* for further information on **errpt**.

For details on the LAN interface device, refer to the specific manual for that device. Information about the messages that appear on the console can be found in *AIX Operating System Messages Reference*.

1.1.9 Possible Problems - Overview

One of the important steps in problem diagnosis is to analyze the type of error. This section shows how the various errors are categorized and describes each error type.

Subtopics

1.1.9.1 Diagnosis within a Cluster

In almost all cases, a failure occurs only on one particular TCF site. That is the site where the problem was observed. On rare occasions more than one TCF site may be involved. When this occurs, the unaffected part of the TCF cluster can still continue normal operation.

1.1.9.2 Error Types

A particular problem is usually associated with one of the following error types:

Messages
Codes
Panic
Wait
Loop
Cluster Communication
Initialization
Shutdown
Performance
User Errors.

The following sections define these terms.

1.1.9.3 Messages and Codes

One of the basic requirements for productive operations is communication. Messages and codes provide that communication.

Messages may:

- Ask for specific inpu
- Provide system status, hardware or softwar
- Warn of impending major error
- Provide pertinent information on system failure
- Provide insight into problems with other TCF sites

You may receive:

- Startup messages
- Standalone utility messages
- AIX/370 system messages
- Panic messages
- System call error codes
- Resident supervisor error codes
- Resident supervisor support error codes.

The format of the messages and codes depends on their origin. You can find a description of most of the various message formats in *AIX Operating System Messages Reference*. The messages and codes may be sent to the user's terminal or the system console.

Note that informational messages can be an indicator (as expressed in the message text or by the sequence in which the messages are displayed) of possible problems. Messages assist in determining the system status and in providing the information necessary to identify and resolve problems.

When a message or code appears on a system console, it usually indicates a problem on that particular TCF site. Some messages may indicate a problem with another TCF site. Messages displayed on the VM console are limited to problems on that particular TCF site. However, messages that appear on a user's terminal normally indicate a problem on the TCF site where the user's job was executing or TCF sites providing resources for the user's job. These TCF sites may not include the TCF site to which the user's terminal is attached.

1.1.9.4 Panic (AIX/370 System-Level Problem)

Subtopics

1.1.9.4.1 When does a Panic Occur?

A ***panic*** occurs when the AIX system kernel discovers an unrecoverable error condition. This may result from many sources:

- Detected hardware error
- Exhaustion of a critical system resourc
- Detection of an inconsistency within the operating system kerne
- An illegal program interrupt occurring within the operating system

The operator may also force a panic to take an AIX kernel dump.

1.1.9.4.2 System Action when a panic Occurs

When a panic condition occurs, AIX/370 does the following:

1. Halts normal operation
2. Dumps machine memory to the dump device
3. Reloads VM automatically (under normal conditions).

In some cases, it may be necessary to restart the system manually.

1.1.9.4.3 Panic messages

When a panic occurs, AIX/370 displays a message on the system console. The messages can help in solving the problem. Refer to *AIX Operating System Messages Reference* for the description of the types of panics and associated message displays.

1.1.9.5 Wait Conditions

A wait condition may be the result of:

- Nothing for the system to d
- Disabled hardware or software function
- Outstanding input/output request
- System deadlock

If you can start the system by providing the missing factor (such as an interrupt or command), it is called a ***soft wait*** condition. If the system must be loaded to recover, it is a ***hard wait*** condition. When either type of wait condition occurs on a TCF site, the site may cease to be active in the TCF cluster.

1.1.9.6 Loop Conditions

If you do not receive a response, output, or a message when you expect it, either a wait or loop condition may have occurred. Unlike a wait condition, the system is running during a loop condition. A loop condition is indicated by one or more of the following:

System or console indicator
Excessive CPU usag
Performance degradatio
New commands are not accepte
Repetition of events

A loop condition may result from software or hardware errors. When a loop condition occurs on a TCF site, it may cease to be active in the TCF cluster. Information about the loop should be collected while the system is still in the loop condition.

Note: On a PS/2, it is difficult to distinguish between a wait and a loop condition

.

1.1.9.7 Cluster Communication Problems

Cluster communication problems are those problems in which individual TCF sites seem to be functioning normally as stand-alone machines, but have difficulty communicating with other TCF sites. A TCF communication problem may involve only one TCF site, a few TCF sites, or all TCF sites.

These types of TCF problems may appear in some of the following symptoms:

- Cluster topology is difficult to stabilize
- Excessive network traffic (such as retransmissions)
- Some user commands that involve remote TCF sites indicate errors
- Performance degradation, particularly with respect to remote TC sites.

A TCF communication problem can be caused by several things. Some of these are:

- Problems with the LAN physical plan
- Problems with the LAN interface devic
- Problems with other devices attached to the LA
- Problems with system configuratio
- Problems with other TCF sites in the TCF cluste
- Problems in the AIX system software

Sometimes TCF communication problems are temporary and can be resolved automatically. Other times operator intervention is required.

1.1.9.8 Initialization Problems

An initialization problem has occurred if the system on a TCF site has been started successfully before but will not startup now. Possible causes are:

Hardware problem
Unavailable system resource
Improper shutdown

1.1.9.9 Shutdown Problems

Consider a problem as a shutdown problem whenever one of the following happens:

Shutdown does not complet
'Init state' message does not change within 10 minute
You are not able to terminate the system from your user termina logged in as a superuser.

1.1.9.10 Performance Problems

Performance problems are difficult to trace but may be due to:

- Intermittent hardware problem
- Dependent on end user activities (like editing large files
- Unfavorable system usage (like long searches through the file syste or excessive use of remote resources).

Performance problems may be indicative of other types of problems (such as loop problems) described previously. For analyzing a performance problem, you may have to investigate all available data (including changes in system generation, hardware configuration, LAN configuration, TCF configuration, use of the system, and functional requirements).

1.1.9.11 User Errors

User errors may include a variety of occurrences, such as inadvertently pressing an incorrect key or starting a program that causes an infinite loop. Sometimes user errors are caused by subtle, often unnoticed changes in the environment, such as a TCF site storing a file system critical to a user's activity being removed from the TCF cluster.

1.2 Chapter 2. Procedures for Diagnosing Problems

Subtopics

1.2.1 Contents

1.2.2 About This Chapter

This chapter contains step-by-step procedures that:

- Help you to identify an error an
- Help you find the source of the error

The chapter is divided into procedures to help you:

- Localize problem
- Perform problem determinatio
- Identify the problem's source

1.2.3 Introduction

It is necessary to localize the source of a problem within the Transparent Computing Facility (TCF) cluster. Most of the time serious problems are limited to a single TCF site. On rare occasions, problems may involve more than one TCF site.

When you encounter an error, check to see whether it is a hardware or software error. This task is called problem determination. The information under "Problem Localization" in topic 1.2.4 may help you perform this task.

In case of a software error, this chapter can help you to identify the type of the error. Once you know the type of error, further investigations are necessary to find out the cause of the error. The steps under "Problem Source Identification" in topic 1.2.6 may help you to find the source of the error.

In some cases, the problem and its determination may be machine-dependent. Machine-specific problems, and the steps taken to determine them, are labeled as '370' or 'PS/2', depending upon the kind of hardware that is involved.

When you reach the end of the applicable procedures in this chapter, you will have done one of the following:

- Called hardware support to fix the proble
- Solved the problem with a known solutio
- Identified a user erro
- Collected information that is necessary to report the problem to IBM

1.2.4 Problem Localization

Follow these steps to determine which of the TCF sites may contain information useful to the analysis of a problem.

Subtopics

1.2.4.1 Is the problem limited to only one TCF site ?

Certain types of problems may be in only one of the TCF sites.

1.2.4.2 Step 1: Did a message on a system console specify one or more other TC

Some messages may indicate inconsistencies between TCF sites or some other TCF-related problems. Did such a message appear on the consoles of one or more TCF site?

NO: ==> Step 2.

YES: Follow the procedures for problem analysis on each TCF site that produced a message or is referred to in a message. Collect the required information on all TCF sites. This information may be needed in subsequent analysis.

1.2.4.3 Step 2: Did more than one TCF site panic around the same time?

In rare cases, more than one TCF site may have a problem around the same time. In most instances, this is due to operator or system administrator error, such as providing two primary copies of the same file system, file systems with the same global file system (gfs) number mounted on two different directories, or file systems with different gfs numbers mounted on the same directory.

NO: ==> Step 3.

YES: Follow the procedures for problem analysis on each TCF site that produced a message. Collect the required information on all TCF sites that produced a panic message.

1.2.4.4 Step 3: Did more than one TCF site produce messages around the same ti

NO: ==> Step 4.

YES: Follow the procedures for problem analysis on each TCF site that produced a message. Collect the required information on each such TCF site.

1.2.4.5 Step 4: Did more than one TCF site leave the TCF cluster around the sa

Usually when TCF sites leave the TCF cluster but appear to be operating normally in other respects, there is a TCF communication problem.

NO: ==> Step 5.

YES: Investigate TCF communication problems. If that does not resolve the difficulty, then follow the procedures for problem analysis on each TCF site that left the TCF cluster. Collect the information required on all TCF sites that left the TCF cluster. This information is needed to determine the cause of the problem.

1.2.4.6 Step 5: Other multi-site problems

Were there other multi-site problems?

NO: There is no multi-site problem.

YES: Follow the procedures for problem analysis on each TCF site suspected to be involved in the problem. Collect the information required on all TCF sites involved.

1.2.5 Problem Determination

Follow these steps to find out whether the problem is a hardware or software error.

Subtopics

1.2.5.1 Step 1: Check the AIX/370 virtual machine and devices

Are the symptoms unique to the AIX/370 virtual machine and/or devices dedicated to AIX/370?

NO: Do VM Problem Determination. See *VM/SP Problem Solving and Reporting Guide*.

YES: ==> Step 2.

1.2.5.2 Step 2: Check peripheral devices

Did any of the peripheral devices behave abnormally or show hardware indicators?

Examples of indicators are: lights, runaway tape drives, improper carriage controls, or unusual noise.

NO: ==> Step 3.

YES: Call hardware support.

1.2.5.3 Step 3: Was a hardware error logged ?

Are system log (SYSLOG) records on the CPEREP Output? (370 only)

Run the CPEREP program to display the various SYSLOG records. This program is described in *Environment Recording Edit and Print User's Guide and Reference* manual.

Were any SYSLOG records written to CPEREP?

1. Were OBR, MCH, CCH or MDR records written by CPEREP ?

   YES: ==> This is a hardware problem.

   NO: ==> Go to next question.

2. Were SFT records written by CPEREP ?

   NO: ==> Step 4.

   YES: For information on the AIX trace facilities, see AIX Trace Facilities in Chapter 5. Continue with "Problem Source Identification" in topic 1.2.6.

3. Were there hardware errors in the ERROR LOG (PS/2 only)?

   Check the console and use **errpt** to determine if the error was logged. See the *AIX Commands Reference* for information on the **errpt** command.

4. Was an error logged?

   YES: This is a hardware problem.

   NO: ==> Step 4.

1.2.5.4 Step 4: Did a panic occur ?

Check the system console for a panic message.

YES: This is a software error.

NO: See "Problem Source Identification" in topic 1.2.6.

1.2.6 Problem Source Identification

Subtopics

1.2.6.1 Step 1: Was the system running ?

NO: ==> Step 2.

YES: ==> Step 4.

1.2.6.2 Step 2: Can you load the system?

NO: ==> See "Problem Loading the Program (IPL)" in topic 1.2.12.

YES: ==> Step 3.

1.2.6.3 Step 3: Was there a panic message ?

NO: ==> Step 4.

YES: ==> See "Panic" in topic 1.2.9.

1.2.6.4 Step 4: Did you receive a prompt ?

NO: ==> See "Problem Loading the Program (IPL)" in topic 1.2.12.

YES: ==> Step 5.

1.2.6.5 Step 5: Did the checks of /dev/root and /generic/dev/'sitename' find e

NO: ==> Step 6.

YES: ==> See "Problem Loading the Program (IPL)" in topic 1.2.12.

1.2.6.6 Step 6: Is the cluster's topology stable ?

Are TCF sites temporarily losing communication with the rest of the cluster?

NO: ==> See "Cluster Communication Problems" in topic 1.2.15.

YES: ==> Step 7.

1.2.6.7 Step 7: Is there a cluster communication problem ?

NO: ==> Step 8.

YES: ==> See "Cluster Communication Problems" in topic 1.2.15.

1.2.6.8 Step 8: Was there another type of message ?

NO: ==> Step 9.

YES: ==> See "Messages" in topic 1.2.8.

1.2.6.9 Step 9: Does the system respond ?

NO: ==> Step 10.

YES: ==> See "Loop Problem" in topic 1.2.10.

1.2.6.10 Step 10: Is there a Wait ?

NO: ==> Step 11.

YES: ==> See "Wait Problem" in topic 1.2.7.

1.2.6.11 Step 11: Was input/output incorrect or no terminal activity ?

NO: ==> Step 12.

YES: ==> See "Incorrect Output" in topic 1.2.11.

1.2.6.12 Step 12: Is it a performance concern ?

NO: ==> Step 13.

YES: ==> See "Performance Problems" in topic 1.2.16.

1.2.6.13 Step 13: Is it a shutdown problem ?

NO: Review the hints under "User Error" in topic 1.2.17.

YES: ==> See "Problems during Shutdown" in topic 1.2.13.

1.2.7 Wait Problem

Subtopics

1.2.7.1 Possible Causes for Waits

There are two types of wait situations: hard and soft.

A hard wait requires reloading the system. It can be caused either b hardware or programming errors.

A soft wait normally allows you to remove the wait condition withou reloading the system. Soft waits are generally caused by events that have not been completed.

Remove a soft wait condition either by entering an appropriate command after you receive a message on the system console or by forcing an I/O interrupt; for example:

- From a disk type I/O device, press the interrupt key (AIX/370 only).
- Make an I/O device unready; then make it ready again (AIX/370).
- Correct the failing hardware condition.

For each wait situation, collect information about the system status before you remove the wait condition or reload, to aid offline diagnosis.

1.2.7.2 Procedure

This procedure shows how to document and recover from a wait condition.

Subtopics

1.2.7.2.1 Step 1: Is there a program trying to read from the terminal ?

If the system is idle, a program waiting to read from a terminal or console may resemble a wait condition. For example, if you run the **cat** command with no arguments at the system console, no prompt is displayed, even after subsequent input to the keyboard. You can clear up this condition by typing the interrupt character (refer to the keyboard reference for the specific terminal type).

After the interrupt character is typed, is the prompt displayed?

NO: ==> Step 2.

YES: No wait condition exists. Continue with normal system operation.

1.2.7.2.2 Step 2: Display the current PSW (EC mode, 370 only)

The layout of the EC mode program status word (PSW) is described in *System/370 Principles of Operation* and *System/370 Extended Architecture Principles of Operation*.

To display the PSW, enter the following command from the AIX/370 console:

```
<PA1>
#CP DISPLAY PSW
B
```

1. Is PSW bit 14 on?

   NO: ==> This is not a wait problem. Continue with "Problem Source Identification" in topic 1.2.6.

   YES: ==> Next step.

   How to find the various bit settings is illustrated in the following example.

   The **cp display psw** command output looks like this:

   ```
   030E0000 00064A8A
   ```

   where the right-most 8 characters of the PSW are the instruction counter.

   Find bit 14:

   ```
   Byte 1 in this case is 03 = 00000011
                               01234567
   Byte 2 in this case is 0E = 00001110
                               89012345
   Therefore bit 14 is on.
   ```

2. Are bits 6 and 7 both on?

   NO: ==> Step 5.

   YES: ==> Step 3.

1.2.7.2.3 Step 3: Does an Enter from the system console return a prompt sign ?

NO: ==> Step 4.

YES: ==> Step 5.

1.2.7.2.4 Step 4: Were any hardware error messages displayed on the system co

YES: See the suggested action in *AIX Operating System Messages Reference*.

Did this action solve the problem?

NO: ==> Step 5.

YES: ==> Continue with normal processing.

NO: ==> Step 6.

1.2.7.2.5 Step 5: Were related errors recorded ?

For the 370: Print the SYSLOG Records with CPEREP. The EREP program is described in the *EREP User's Guide and Reference* manual.

Were hardware errors recorded on ERDS related to your problem?

NO: ==> Were software errors recorded?

YES Refer to AIX Trace Facility in Chapter 5.

NO: ==> To perform a system dump, do the following:

1. Enter: #CP STORE STATUS
2. Enter: #CP DUMP 0 -END
3. Enter: #CP CLOSE PRT
4. Print the dump.
5. Analyze the saved status information and the fixed storage location in the dump.
6. Call the IBM support center.

YES: Analyze EREP printout and call the appropriate hardware support personnel.

For the PS/2: Were hardware errors logged on the console or in the error log?

YES: Call your hardware support personnel for your machine.

NO: Call the IBM Support Center. They may direct you to perform a system dump.

1.2.8 Messages

Subtopics

1.2.8.1 Possible Causes

A message is generated whenever there is a need to communicate about:

Status of the system
Requirement for input
Pending output
Error situations (minor and severe errors)
Responses to various queries.

Because messages are issued for such a broad field of conditions and situations, they are most important for resolving problems. Messages can communicate information, for example, by their identifiers, text, timing, frequency, and codes that are returned.

1.2.8.2 Procedure

Subtopics

1.2.8.2.1 Step 1: Message on user's terminal ?

Did an error condition result in a message on the user's terminal?

NO: ==> Step 3.

YES: Follow the instructions in the message;
==> Step 2.

1.2.8.2.2 Step 2: Did the condition persist ?

NO: Continue normal processing.

YES: ==> See "User Error" in topic 1.2.17.

1.2.8.2.3 Step 3. Message on system console?

Did a message appear on the system console?

NO: ==> Step 6.

YES: ==> Step 4.

1.2.8.2.4 Step 4: Was it a Panic message (Panic screen) ?

NO: Record the information or take the recommended action. Refer to *AIX Operating System Messages*.
==> Step 5.

YES: ==> "Panic" in topic 1.2.9.

1.2.8.2.5 Step 5: Can normal processing continue?

NO: ==> See "Kernel Dump Procedure" in topic 1.2.18.

YES: Continue normal processing.

1.2.8.2.6 Step 6: Message on VM console ? (370 only)

Did the message appear on the VM console?

NO: There is no message related to the AIX/370 guest machine.

YES: ==> Step 7.

1.2.8.2.7 Step 7: Does the VM Message refer to hardware dedicated to the AIX/3

NO: Consider this as a VM problem.

YES: Investigate problems with that hardware.

1.2.9 Panic

Subtopics

1.2.9.1 Possible Causes

Internal system-level errors in AIX/370 and AIX PS/2 result in a Panic condition. When configured to do so, the system creates a kernel dump on that TCF site. When the dump is completed, the TCF site either continues normal operation or is rebooted automatically.

When there is a Panic on a TCF site, a Panic message is displayed on the system console. This message includes identification of the process that detected the error and the cause for the Panic.

In rare cases the cause of a Panic may involve a problem with another TCF site, or a discrepancy between system information on this TCF site and another TCF site. This may be the case if there is a Panic on more than one TCF site at about the same time. Also, one TCF site may experience very serious errors and force a Panic on another site and then one on itself. In these cases it may be necessary to analyze all dumps together to identify the cause of the problem.

1.2.9.2 Procedure

Subtopics

1.2.9.2.1 Step 1: Did you receive a message that the kernel dump completed suc

NO: ==> See " Panic Hangs" in topic 1.2.18.1. If dumping is enabled and you have enough space on the dump device this is probably a Panic hang situation.

YES: ==> Step 2.

1.2.9.2.2 Step 2: Bring up the system after a kernel dump

On a 370: The system should automatically restart after a successful kernel core dump. If it does not, type (on the AIX/370 system console):

```
<PA1>
log off
```

Log on to the virtual machine and load the system in the normal way.

On a PS/2: The system should automatically restart after a successful kernel core dump. If it does not, power-cycle the machine and boot the system normally.

On either type: Is the version line of the kernel displayed?

NO: ==> Step 3.

YES: ==> Step 4.

1.2.9.2.3 Step 3: Did the system reboot properly ?

NO:

1. Record the Panic screen information in the way described under Step 03.

2. Note the number of the dump (if one was saved) for later analysis.

3. Go to "Problem Loading the Program (IPL)" in topic 1.2.12.

YES: ==> Step 4.

1.2.9.2.4 Step 4: Print console log and report

After you receive the prompt sign, proceed as follows:

1. Print the console log (**/usr/adm/messages**).

2. Analyze the dump (if one was saved) using **crash**.

3. Continue normal processing

1.2.10 Loop Problem

Subtopics

1.2.10.1 Possible Causes of Loops

Loops can be caused by a coding or logic error in the program, an error in setting up the program, or a malfunction of an I/O device.

Note: It is not possible to distinguish a wait from a loop on a PS/2. For the PS/2, refer to "Wait Problem" in topic 1.2.7.

1.2.10.2 Procedure

Subtopics

1.2.10.2.1 Step 1: Display current PSW (370 only)

Display the current PSW several times at the VM/CP terminal logged into the AIX/370 virtual machine with the following command:

```
#CP DISPLAY PSW
```

Did the address part of the PSW change?

NO: ==> Step 2.

YES: ==> Step 3.

1.2.10.2.2 Step 2: Is the Wait bit (PSW bit 14) on ? (370 only)

NO: ==> Step 3.

YES: ==> "Documentation for Wait" in topic 1.3.5.

How to find the various bit positions is described under "Step 2: Display the current PSW (EC mode, 370 only)" in topic 1.2.7.2.2. For a description of the various PSW bits, see the *System/370 Principles of Operation* or *System/370 Extended Architecture Principles of Operation* manuals.

1.2.10.2.3 Step 3: Record addresses (370 only)

1. Issue several #CP DISPLAY PSW commands to do one of the following:

   Identify a loop condition
   Collect status information about the problem area.

2. Record all the addresses (if not too numerous), or record several representative addresses.

3. When you press ENTER from the AIX/370 console does the system return the prompt sign?

   NO: ==> Step 4.

   YES: ==> Step 5.

1.2.10.2.4 Step 4: Save system error condition (370 only)

Generate a kernel core dump.

1.2.11 Incorrect Output

Subtopics

1.2.11.1 Possible Causes for Incorrect Output

Errors in program logic or in setting up the system for program execution may cause errors in the output. For example, the use of incorrect data for input, mistakes in device assignments, or erroneous commands frequently yield unexpected results.

1.2.11.2 Procedure

Subtopics

1.2.11.2.1 Step 1: Is there any terminal activity ?

NO: ==> Step 18.

YES: ==> Step 2.

1.2.11.2.2 Step 2: Get information about the program execution

Get the terminal output of the program in question. Get the input data for the program including any in-line data from other AIX files if possible.

Was there an incorrect operator response to a message during the execution of the program?

NO: ==> Step 3.

YES: Re-run the program with the correct response.

1.2.11.2.3 Step 3: Is input data available ?

Can the original input data be reproduced (for example, is the original tape or disk file still available)?

NO: ==> Step 4.

YES:

1. Take steps to preserve the original input file for later analysis. For example, copy the file.
2. ==> Step 4.

1.2.11.2.4 Step 4: Is it a hardware problem ?

Were there any hardware problems at the time of the program execution?

NO: ==> Step 6.

YES: ==> Step 5.

1.2.11.2.5 Step 5: Is the output incorrect?

After analysis of the hardware error, is it possible that the error caused the incorrect output?

NO: ==> Step 6.

YES: Check and repair the error and re-run the program.

1.2.11.2.6 Step 6: Was there a change in the TCF topology ?

Was there any change in the TCF topology that may have caused the incorrect output (such as some file system or necessary resource's becoming unavailable due to the unavailability of a particular TCF site)?

NO: ==> Step 7.

YES: Rerun the program when the necessary TCF topology has been restored.

1.2.11.2.7 Step 7: Were there environmental changes ?

Have there been changes such as new hardware or software, mounted or unmounted file systems?

NO: ==> Step 8.

YES: Correct problems found in environmental changes and rerun the program.

1.2.11.2.8 Step 8: Is there unreadable data on the screen or coming from the p

NO: ==> Step 9.

YES:

1. Check to see that there is no mismatch of locale between the process which generated the data and the process which sends it to the screen or printer. If the two processes use different file codes, the data will be unreadable.

2. If you cannot easily determine the locale of the process which generated the data, try switching the locale of the receiving process. Try each locale available on the system. Rerun the receiving process each time.

3. If all available locales have been tried without yielding readable data, continue to
==> Step 9.

1.2.11.2.9 Step 9: Were duplicate lines found in the output ?

NO: ==> Step 10.

YES:

1. Check for a loop in the program's I/O routine.
2. ==> Step 12.

1.2.11.2.10 Step 10: Is there data missing ?

NO: ==> Step 11.

YES:

1. Determine if any specific routines in the program failed to receive control or lost control prematurely.

2. ==> Step 12.

1.2.11.2.11 Step 11: Is there too much or unexpected data ?

Did you receive too much data or unexpected data?

NO: ==> Step 13.

YES: I/O areas may have been overwritten or may not have been cleared properly.
==> Step 12.

1.2.11.2.12 Step 12: Rerun the program

Consider rerunning the program and dumping the process to disk for further analysis; refer to "How to Perform a Kernel Dump" in topic 2.6.9.

1.2.11.2.13 Step 13: Are there VM system problems or interrupts ? (370 only)

Did a VM system problem occur during the program execution?

NO: ==> Step 14.

YES: Refer to the appropriate VM manuals to analyze the VM problem.

1.2.11.2.14 Step 14: Check the program

Consider the following:

Check the program logic flow for improper calls or process generation

Were there any call or process generation problems

NO: ==> Step 15.

YES: Correct the errors and re-run the program.

1.2.11.2.15 Step 15: Was the input faulty?

Determine if the correct input was used. Were the correct elements and programs included in your program?

NO: Correct the error and re-run the program.

YES: ==> Step 16.

1.2.11.2.16 Step 16: Compare the failing output of the program with the input

Is the output related to the input?
For example, a missing output record may be caused by a missing input record.

NO: ==> Step 17.

YES: ==> Step 12.

1.2.11.2.17 Step 17: Was there other input?

Was the input to this program the output of another program?

NO: Correct the input and re-run the program.

YES: Return to Step 02 for the other program.

1.2.11.2.18 Step 18: Is there no terminal activity ?

Does the AIX console indicate any AIX error messages, or Panic on either the local site or the site where the process was running?

NO: ==> Step 19.

YES:

1. Analyze the message and identify the mode of operation.

2. ==> See "Messages" in topic 1.2.8.

1.2.11.2.19 Step 19: Console error messages

Do any recent errors appear? (On a 370 you can use the PF7 key to scroll the console output back.) The console messages include a timestamp in GMT, to aid in identifying recent messages. Also look at **/dev/osm**.

NO: ==> Step 20.

YES: Analyze the message, go to "Messages" in topic 1.2.8.

1.2.11.2.20 Step 20: Does the Panic screen indicate the following:

ACTION => processing continues normally

NO: ==> See "Panic" in topic 1.2.9.

YES: No intervention is required. Rerun the program and check the output. Analyze the dump of the process if one was made.

1.2.12 Problem Loading the Program (IPL)

Subtopics

1.2.12.1 Possible Causes

Initialization problems are typically the result of a change outside the normal running environment, an incorrectly configured system, or a damaged file system that is critical, such as the replicated root or TCF site <**LOCAL**> file system. Some causes can be:

- Hardware problem
- Unavailable system resource
- Configuration changes (Including possible hardware or VM changes
- Improper shutdow
- Improper system generation or initializatio
- Damaged file syste
- A diskette is in the micro-diskette drive (PS/2)

1.2.12.2 Procedure

You can assume that the program was successfully loaded if the Copyright Notice and the **checking root and <LOCAL> file systems** messages appear and eventually a prompt appears on the console. Depending on the system configuration, you may then receive either the single-user prompt, or the system may become generally available.

Subtopics

1.2.12.2.1 Step 1: Could the system be loaded previously?

NO: ==> See "Problem Localization" on page 2-3.

YES: ==> Step 2.

1.2.12.2.2 Step 2: Have changes been made to the hardware ?

NO: ==> Step 4.

YES: ==> Step 3.

1.2.12.2.3 Step 3: Have there been hardware or VM changes ?

For the 370: Were the hardware and/or VM changes reflected in the AIX/370 entry in the VM directory or start-up scripts?

NO: Refer to VM manuals and:

1. Update the VM nucleus
2. Load VM
3. Try to load AIX/370.

YES: Run CPEREP and have the hardware checked.

For the PS/2: Were the hardware changes properly configured?

NO: Reconfigure with the PS/2 Reference disk.

YES: Have the hardware checked out.

1.2.12.2.4 Step 4: Have there been software changes?

You may be able to use the console log of previous executions to check whether there were changes or not.

Were software changes made to the system?

NO: ==> Step 6.

YES: ==> Step 5.

1.2.12.2.5 Step 5: Will the system be loaded successfully?

NO: ==> Step 6.

YES:
1. Review the system changes.
2. Correct any errors.
3. Try again.

1.2.12.2.6 Step 6: Is the Wait bit (PSW bit 14) on ? (370 only)

How to find the various bit positions is described under "Step 2: Display the current PSW (EC mode, 370 only)" in topic 1.2.7.2.2.

NO:

On the first attempt:

1. Reload and look at messages, if any.

   Note: If you Dial to AIX/370, a failing Autolog is indicated by the following:

   - Message: DMKDIA045E IX370 NOT LOGGED ON
   - Wait condition with message: DIALED TO IX370 200

2. ==> Step 7.

On the second attempt: ==> See "Loop Problem" in topic 1.2.10.

YES: ==> See "Wait Problem" in topic 1.2.7.

1.2.12.2.7 Step 7: Did the system reach single-user mode ?

NO: ==> Step 10.

YES: ==> Continue with system startup.

1.2.12.2.8 Step 8: Was there a problem opening the system console ?

If process 1 ('init') or its children are unable to open any of the console devices (**/dev/console, /dev/syscon, /dev/systty**), AIX/370 and AIX PS/2 cannot come up. If there is a problem, the messages:

```
INIT: Can't open ANY console devices!!
Fatal: no console available
```

and

```
Fatal: no console available
```

are displayed on the system console. AIX/370 and AIX PS/2 are then idle.

NO: => Step 11.

YES: ==> Step 9.

1.2.12.2.9 Step 9: Was a new system installed ?

Was a new system installed before this subsystem reboot? If so, it may not have been initialized or configured properly.

NO: => Step 17.

YES: ==> Step 10.

1.2.12.2.10 Step 10: Will the old system boot ?

Attempt to boot *unix.std.old* single user. Refer to *AIX/370 Administration Guide* for information on how to boot *unix.std.old*. Will the old system boot?

NO: => Step 17

YES:

1. **.cd/local**

2. **.rm unix.std**

3. **.mv unix.std.old unix.std**

4. **.mv unix.std.vold unix.std.old**

5. Reboot the regular system.

1.2.12.2.11 Step 11: Problem in checks of root file system ?

When 'init' checks the root file system, are there errors which cannot be corrected automatically by **fsck**?

NO: ==> Step 12.

YES:

1. Run

   ```
   fsck /dev/root
   ```

   and correct the errors manually. Refer to *AIX/370 Administration Guide* for information regarding correcting errors in replicated file systems.

2. Reboot AIX. Enter:

   ```
   reboot
   ```

3. Continue with normal operation.

1.2.12.2.12 Step 12: Problems in checks of <LOCAL> file system ?

When 'init' checks the file system **/generic/dev/'sitename'** are there errors which cannot be corrected automatically by **fsck**?

NO: ==> Step 13.

YES:

1. Run

   ```
   fsck /generic/dev/'sitename'
   (where 'sitename' is the name of the TCF site)
   ```

2. Reboot AIX. Enter:

   ```
   reboot
   ```

3. Continue with normal operation.

1.2.12.2.13 Step 13: Is there a warning about no pipes ?

This message normally indicates that the automatic checks of the root and **<LOCAL>** file systems failed. Do the following:

1. Run

   ```
   fsck /dev/root
   ```

   correct any errors, and reboot.

2. Run

   ```
   fsck /generic/dev/'sitename'
   (where 'sitename' is the name of the TCF site)
   ```

   and correct any errors.

3. Reboot AIX. Enter:

   ```
   reboot
   ```

4. Continue with normal operation.

1.2.12.2.14 Step 14: Is there a problem running some load module from the root

The system will produce an error message on the system console if it cannot run an important load module (for example, the console shell) from the root file system.

Did such a message appear on the console?

NO: ==> Step 15.

YES: ==> Step 21.

1.2.12.2.15 Step 15: Is there a message ?

Is there a message or code?

NO: ==> Step 16.

YES: Refer to "Messages" in topic 1.2.8.

1.2.12.2.16 Step 16: Other type of loading error

If some other type of loading error occurred:

1. Record the console output log that indicates the problem.
2. Report the problem.
3. Attempt to proceed with normal operation.

1.2.12.2.17 Step 17: Corrupted <LOCAL> file system.

If you have reached this step, you probably have a corrupted **<LOCAL>** file system. This is an extremely rare occurrence. The best way to recover from this situation is to use your maintenance system to verify the problem (that is, run **fsck**). If you have a backup, it can be restored by bringing up the TCF site as the maintenance site or from a maintenance system, if one exists. If you have a VM backup of the entire volume, you may restore those pages that are from the **<LOCAL>** file system. If it is not possible to restore only those pages, refer to Step 18 for information about restoring the root file system.

1.2.12.2.18 Step 18: Is this the primary copy of the root file system?

Does this TCF site store the primary copy of the root file system?

NO: ==> Step 19.

YES: ==> Step 21.

1.2.12.2.19 Step 19: Can the problem be repaired with the maintenance site ?

If you reach this step, there is a problem with the local copy of the replicated file system. This is an extremely rare occurrence. Since this is not the primary copy of the file system, the most likely cause of the problem is that the system was taken down during the propagation of updates. To correct the problem, do the following:

1. Bring up this TCF site as the maintenance site.
2. Mount the local copy of the root file system on '/'.
3. Run /**etc/recmstr**. The output of **recmstr** will indicate if propagations occur.
4. Enter the **sync** command.
5. Take down the generic site system.
6. Bring up the regular system.

Can the normal system now be loaded successfully?

NO: ==> Step 20.

YES: Continue with normal procedures

Note: More information on these procedures can be found in *AIX Administration Guide*.

1.2.12.2.20 Step 20: Propagate complete copy in installation mode.

If the propagation did not succeed, then it is necessary to propagate a new copy of the replicated root file system. Proceed as follows:

1. Bring up this TCF site in installation mode.
2. Initialize the disk partition that stores the local copy of the replicated root file system with /**etc/mkfs**. Be sure to make the correct type (backbone or secondary) of file system, and use the correct values for file system size and number of inodes.
3. Mount the local copy of the replicated root file system on **/**.
4. Run /**etc/recmstr**. The output of **recmstr** indicate when propagations have been completed.
5. Enter the **sync** command.
6. Take down the generic site system.
7. Bring up the regular system.
8. Continue with normal operation.

1.2.12.2.21 Step 21: Is there a backbone copy of the root file system ?

If you reach this step, the primary copy of the replicated is corrupted. This cannot occur simply as the result of improper shutdown, and is an extremely rare occurrence.

If there is a backbone copy of the root file system, recovery from this situation is a little easier. Is there a backbone copy of the root file system in the TCF cluster?

NO: ==> Step 22.

YES: Do the following:

1. Choose one of the backbone copies of the replicated file system to be turned into the primary copy. Reboot that site, stopping in maintenance (single-user) mode without enabling TCF traffic. This procedure provides a quiescent system for running the **sec2prim** utility.
2. Run the following command to verify that the conversion of the backbone copy will be successful:

   ```
   sec2prim /dev/root
   ```

   This command lists any files that would be lost as a part of the conversion if, for example, this backbone copy was an AIX/370 and the new primary site was a PS/2.
3. To convert the backbone copy of the replicated file system into a primary copy, run the **sec2prim** command with the option set to update the file system:

   ```
   sec2prim -u /dev/root
   ```

4. Repeat the **sync** command and reboot this site again, allowing the file system checks of the modified root file system to be performed, and this time allowing the site to go to multi-user mode with TCF traffic enabled. You now have a new primary copy of the file system.
5. Return to the site with the damaged primary copy of the replicated root. Bring up this site as an installation site.
6. Initialize the disk partition that stores the local copy of the replicated root file system with **/etc/mkfs**. Be sure to make the file system a backbone copy, since the primary copy is now on another TCF site.
7. Mount the local copy of the replicated root file system on **/**.
8. Run **/etc/recmstr**. The output of **recmstr** indicates when propagations have completed.
9. Enter the **sync** command.
10. Take down the installation site.
11. Bring up the regular system.
12. Continue with normal operation. (At this point, you can switch the new primary copy back to being a backbone copy, and make this TCF site the primary copy once again. It is not absolutely necessary to do so.)

1.2.12.2.22 Step 22: Recovery when no backbone copy is available.

Do the following:

1. Restore this site from a recent backup tape.
2. Force all other TCF sites out of the TCF cluster.
3. One by one, reinstall the other TCF sites into the cluster.

1.2.13 Problems during Shutdown

If shutdown does not complete or the 'init state' message does not change within 10 minutes, perform the following steps:

1. From the system console, remove the site from the TCF cluster with the following command:

   ```
   /etc/clusterstop
   ```

2. Terminate all processes with the following commands:

   ```
   /etc/killall -15
   /etc/killall -9
   ```

3. Unmount all file systems by entering

   ```
   /etc/umount -f -a
   ```

4. Enter the following command to display the file systems that are still mounted:

   ```
   /etc/mount
   ```

5. Check whether a remaining process is running. Enter:

   ```
   ps -ef
   ```

6. Stop accounting with:

   ```
   /usr/lib/acct/shutacct
   ```

7. Sync and run **shutdown**.

Note: You may want to run **halt** instead of **shutdown**. Refer to the *AIX Commands Reference* for more information on these commands.

1.2.14 Problems during Shutdown without System Console Support

If the system console is hung and the system is still up and needs to be shut down, use an available AIX terminal directly connected to the site and proceed as follows:

1. Login to AIX as a superuser.
2. Send a message to all users (**wall** command) to tell them to log off immediately.
3. Enter the command:

   ```
   sync
   ```

4. Remove this site from the TCF cluster:

   ```
   /etc/clusterstop
   ```

5. Terminate all processes with the following commands:

   ```
   /etc/killall -15
   /etc/killall -9
   ```

6. Unmount all file systems by entering the following:

   ```
   /etc/umount -f -a
   ```

7. Check whether a file system is still running or not. Enter the following command to display the file systems that are still mounted:

   ```
   /etc/mount
   ```

   If a file system is still mounted, do not forget to run the **fsck** utility after the next startup.

8. Check whether a remaining process is running. Enter:

   ```
   ps -ef
   ```

9. Stop accounting with:

   ```
   /usr/lib/acct/shutacct
   ```

10. Sync the AIX system.

    ```
    sync
    sync
    ```

11. Reboot AIX.

    ```
    /etc/reboot
    ```

1.2.15 Cluster Communication Problems

Cluster communication problems are those in which the individual TCF sites seem to be functioning correctly as individual machines, but for some reason the TCF cluster is not functioning correctly. This could range from one TCF site not being able to join in the TCF cluster to total isolation of all TCF sites. This section describes some approaches to solving TCF communication problems.

Subtopics

1.2.15.1 Step 1: Is the communication problem affecting more than one TCF site

Different procedures must be followed depending on whether the problem is affecting only one or a few TCF sites (a TCF site-specific problem), or if it is a more general problem affecting many or all TCF sites (a global TCF problem).

One indication of whether the problem is TCF site-specific or a global TCF problem is the overall stability of the TCF cluster, which can be determined by the following:

Does the AIX console history log indicate that repeated TCF topolog changes are occurring?
When the **ptn** command is run on the AIX/370 console or a user terminal at most TCF sites, does it indicate that more than one site is on the net?

Is the TCF communication problem affecting more than one TCF site?

NO: ==> Step 9.

YES: ==> Step 2.

1.2.15.2 Step 2: Are other, non-TCF activities sharing the LAN experiencing di

If other LAN activity that is not related to TCF communication is also experiencing difficulty, then the problem is probably not with AIX.

NO: ==> Step 3.

YES: ==> Step 8.

1.2.15.3 Step 3: Does the AIX osm console log indicate excessive retransmissio

Excessive retransmissions usually indicate a problem in the LAN such as reflections or shorts in the cable, or some device attached to the LAN that is not behaving correctly.

NO: ==> Step 4.

YES: ==> Step 8.

1.2.15.4 Step 4: Does the AIX osm console log indicate LAN physical problems ?

The types of physical problems may vary depending on the type of LAN used. Such problems usually indicate problems with LAN hardware or some device attached to the LAN. Examples of physical problems that might be reported for an Ethernet LAN are the detection of collisions and jams.

NO: ==> Step 5.

YES: ==> Step 8.

1.2.15.5 Step 5: Are there other LAN problems ?

Is there evidence of other problems with the LAN used for TCF communication ?

NO: ==> Step 6.

YES: ==> Step 8.

1.2.15.6 Step 6: Are the ignore masks set correctly ?

If the lowest numbered site in a TCF cluster is ignoring some TCF site that is not being ignored by other TCF sites, an unstable topology results.

Is the lowest numbered active TCF site ignoring some TCF site that is attempting to join the TCF cluster?

NO: ==> Step 7.

YES: Correct the ignore mask using the **/etc/ignore** or **/etc/unignore** command. Verify the stability of the TCF topology

1.2.15.7 Step 7: Is there a file system mismatch on different TCF sites ?

Two file system packs are considered mismatched if either they have two different global file system (**gfs**) numbers but are mounted at the same mount point, or have the same **gfs** number but are mounted at different mount points. Other mismatches include conflicting pack numbers, or differing numbers of inodes. Such mismatches prevent a stable TCF topology.

Is there such a mount mismatch?

NO: ==> Step 8.

YES: Correct mount mismatch, and verify that the TCF topology stabilizes.

1.2.15.8 Step 8: Resolving LAN problems

It is most likely that there is a problem with the LAN itself. Locate and correct LAN problems according to the procedures recommended by the manufacturer of the LAN equipment. When correct operation of the LAN has been restored, verify that TCF topology stabilizes after TCF communication is restored.

1.2.15.9 Step 9: Is the LAN interface device operating properly ?

For the System/370: the LAN interface may be failing because:

Hardware failure
The device is turned offline
The device needs to be reset
The device is not correctly attached by VM to the AIX/370 virtua machine.
The device is not varied on by the resident supervisor

For the PS/2: Is the hardware properly configured at both the hardware and software levels? Is the hardware properly installed and connected?

For the System/370 and the PS/2: Is there a problem with the LAN interface?

NO: ==> Step 10.

YES: Correct problems with the LAN interface. Verify that the TCF communication can proceed.

1.2.15.10 Step 10: Does the TCF site have TCF cluster traffic disabled ?

If a TCF site has TCF traffic disabled, it cannot join the TCF cluster. Run the **/etc/clusterstart** command, which indicates if TCF communication was disabled, and enables it if it was. Was the TCF communication disabled?

NO: ==> Step 11.

YES: Run **/etc/clusterstart** to reenable TCF communication and verify that the TCF site has rejoined the TCF cluster.

1.2.15.11 Step 11: Is the TCF site in a different TCF cluster?

AIX/370 supports several simultaneous TCF clusters in a single group of networks. If the subnet ID of this site does not match those of the cluster, communication cannot succeed. This value is indicated by the kernel value, subnetnum, and may be examined by a debugger or **crash**.

Is the TCF site in a different cluster?

NO: ==> Step 12.

YES: An installed site cannot be moved from one cluster to another because of replicated file systems, particularly the replicated root file system. Each cluster has an independent replicated root file system. TCF requires that all sites in a cluster agree on both the content of a replicated file system (such as file data and directories) and the order in which changes have been made. Therefore, if you want to move a site or a disk from one cluster to another, the old root file system on this site must be discarded. The site must be reinstalled as a new member of another cluster.

1.2.15.12 Step 12: Are the TCF versions incompatible?

Some releases of TCF may not be compatible with other TCF releases. If a partial upgrade is installed, old sites may not communicate with TCF until the upgrade is completed.

Is there a version mismatch?

NO: ==> Step 13.

YES: Complete the upgrade to the new TCF version on all sites in the TCF cluster.

1.2.15.13 Step 13: Unresolved LAN problems

The problem may be in the LAN interface or in the AIX software. Document the problem and call IBM service.

1.2.16 Performance Problems

Performance problems are among the most elusive problems to resolve.

One possible cause of performance problems is system overutilization. A system that is overloaded fails to perform properly because the system spends most of its time making resources available and scheduling processes, and less time running the process. The **loads** program can indicate a short or medium range problem of overutilization. The **ps** program can determine processes that are using excessive resources. The **sar** command can be used to get information on the system activity. System accounting can be used to keep track of system usage over a longer term.

On a 370 system, use the VM level tools to assess the load presented to VM.

Another possible cause of performance problems is the misuse of resources distributed throughout the TCF cluster. There are no absolute rules for determining where to run processes with respect to their resources, but the following may be useful:

- Terminal-intensive activity (particularly screen-oriented activity) i best run on the TCF site where the terminal is physically attached.
- Activities that perform large amounts of file activity should run o the TCF site where the data is stored.
- Computations that are CPU intensive are best run on faster or les loaded TCF sites, and those TCF sites that are not heavily used for terminal traffic.

Real computation seldom falls exactly into any one of those categories, so good judgement must be used in practice.

For more information on performance problems, refer to *AIX/370 Administration Guide* and *AIX Managing Guide*.

1.2.17 User Error

Subtopics

1.2.17.1 Possible Causes

If you did not follow the rules of a defined environment, you committed a user error. Such errors can happen at any interface between the user and AIX. Following are some common errors:

- Wrong hardware connection
- Incorrect system generation parameter
- Incorrect terminal procedure
- Wrong commands entere
- Faulty user-program codin
- Misinterpreting of message
- Execution of unsupported functions

1.2.17.2 Procedure

Here are some steps to help you find possible user errors.

Subtopics

1.2.17.2.1 Step 1: Did this program run previously ?

NO: Check for incorrect programming in user-written AIX applications.

YES: ==> Step 2.

1.2.17.2.2 Step 2: Were any changes made to the program ?

NO: ==> Step 4.

YES: ==> Step 3.

1.2.17.2.3 Step 3: Will an old version of the program still run ?

NO: ==> Step 4.

YES: Examine the changes for errors.

1.2.17.2.4 Step 4: Were service fixes applied ?

Was any IBM service (PTF or APAR fix) applied to the systems used by the process that terminated?

NO: ==> Step 5.

YES: Do the following:

1. Collect the documentation.
2. Collect information as shown in Chapter 4, "How to Report Problems to IBM" in topic 1.4.
3. Call the IBM Support Center.

1.2.17.2.5 Step 5: Was a message displayed ?

Review the message information. Have you found the reason for the problem?

NO: ==> Step 6.

YES: ==> Step 8.

1.2.17.2.6 Step 6: Did the process end abnormally ?

Did the process end abnormally with the dump file named 'core' placed in the current directory?

NO: ==> Step 07.

YES: The kernel detected a problem and wrote out a 'core image' of the process.

Analyze the core dump, using the symbolic debugger **dbx**. For information on the dbx debugger, refer to *AIX Operating System Commands Reference*. The *AIX Operating System Technical Reference* provides information on the core dump.

1.2.17.2.7 Step 07: Has your process terminated normally ?

Use the **ps -f** command to verify whether your process is on the active queue or not.

==> Step 08.

1.2.17.2.8 Step 08: List, rerun, and examine the output

1. Get a listing of the program and the input that produces the error.

2. Rerun the program.

3. Examine the output and the messages.

4. Return to Step 05.

1.2.18 Kernel Dump Procedure

Subtopics

1.2.18.1 Panic Hangs

370: Verify that VM is still functioning, the virtual machine remains functional, and the system is not configured to pause on a panic.

PS/2: Is the PS/2 awaiting user interaction, such as inserting a diskette? Is the system configured to pause on a panic?

1.3 Chapter 3. Collecting Information about a Problem

Subtopics

1.3.1 Contents

1.3.2 About This Chapter

This chapter indicates what information you should collect in case of a specific error.

The better the information you collect about a problem, the faster it can be fixed. Therefore, establish specific instructions and procedures to collect the required information before an error occurs.

Each of the following sections contains a list of recommended error documentation and a list of references. These references may point to chapters in this manual or they may mention specific commands. The commands are described in the *AIX Operating System Commands Reference* manual.

1.3.3 Documentation for any Problem

Besides your seven-digit customer access code (found on your maintenance agreement or invoice), you will need to collect documentation about a problem. The following information applies to any software problem that may occur.

System recovery may erase information that could be important for failure analysis. Therefore, write down the information available on each TCF cluster site, such as the following:

- Displayed messages and any presented recommended actio
- System statu
- System device lights, if an
- Unusual device or system function
- Program or command being ru
- Other intermittent symptoms

It is also necessary to provide information about the system's configuration. This information includes:

- TCF cluster configuration at the time the problem occurred (how man sites are on the TCF cluster, what are their numbers, what are their machine types).
- File system configuration at the time the problem occurred (fo example, what file systems are mounted on which TCF sites, which file systems are replicated, which sites store the primary copies of the file system).

If either of these configurations are different from the normal operational configuration, those differences should be identified.

Console logs, notes kept by the operator and users, and observations could also be useful, and should be provided if they are available.

If you suspect a problem involves more than one TCF site, then information from each TCF site involved in the problem should be kept, and grouped together. Sometimes this information is essential to resolve such a problem.

There is more information about some of the terms that follow in the *AIX Technical Reference*.

1.3.4 Documentation for Panic

The following information may be useful:

- Panic dum
- Panic messag
- Console listin
- Dump of active task (taskid specified in the panic message
- the AIX kernel map

In rare cases more than one TCF site may produce panic messages at approximately the same time. This information should be collected together.

Subtopics

1.3.4.1 References

Chapter 6, "AIX/370 Kernel Dump Debugging Information"
savecore
crash
osm (found in **/dev/osm**).

1.3.5 Documentation for Wait

The following information should be collected:

- AIX kernel dum
- Console listin
- Messages and note
- Copy of the AIX kerne
- CPEREP output, or error log output

Subtopics

1.3.5.1 References

Chapter 6, "AIX/370 Kernel Dump Debugging Information"
crash
savecore
osm.

1.3.6 Documentation for Loop

The following information should be collected:

- AIX kernel dump and/or a dump of the looping tas
- Corresponding AIX kerne
- Console log printou
- If available, the source listing of the user program that was runnin when the loop occurred
- Printed output of the user program (if available
- Loop instruction addresse
- PER trace output

Subtopics

1.3.6.1 References

Chapter 6, "AIX/370 Kernel Dump Debugging Information"
crash
savecore
osm.

1.3.7 Documentation for Incorrect Output

The following information should be collected:

An AIX/370-Process-ID-Dump on disk. If this is not available, re-ru the program and then execute the command

```
kill -QUIT processid
```

to terminate the program and force a dump when the incorrect output occurs again.
The source listing of the failing program
A printout of the failing output
A list of all I/O files and volumes used by the particular program
The console log printout

Subtopics

1.3.7.1 References

fsck
fsdb
crash
kill
osm.

1.3.8 Documentation for Load Problem

The following information should be collected:

- Description of failur
- CP display of the contents of low-address storage bytes and PSW fo AIX/370
- CP TRACE output for the AIX/370 useri
- Physical device address of the SYSRES.IPL volume in use when the erro occurred
- List of hardware and software change
- List of VM generation parameters, for example, directories an profiles
- Device analysis lis
- History fil
- Recent configuration change
- Recent system administrator work

Subtopics

1.3.8.1 References

VM/CP Commands

1.3.9 Documentation for User Error

The following information should be collected:

Console log printou
Listing of user programs

1.4 Chapter 4. How to Report Problems to IBM

Subtopics

1.4.1 Contents

1.4.2 About This Chapter

After you have identified that your problem is AIX-related, report it to the appropriate IBM support. Your IBM representative has the telephone number of the IBM support center. You will need your seven-digit customer access code, as well as other pertinent information about your software and hardware.

You should describe the problem to IBM in the form of a *Problem Symptom String*.

Figure 4-1 in topic 1.4.3 shows the elements of the problem symptom string. Figure 4-2 in topic 1.4.3.2 shows a problem-reporting-form that may help you to build a symptom string.

1.4.3 Problem Symptom String

Figure 4-1 below shows the elements of a problem symptom string, which you must prepare *before* reporting your problem to IBM. Make sure that you have read the preceding chapter of this guide.

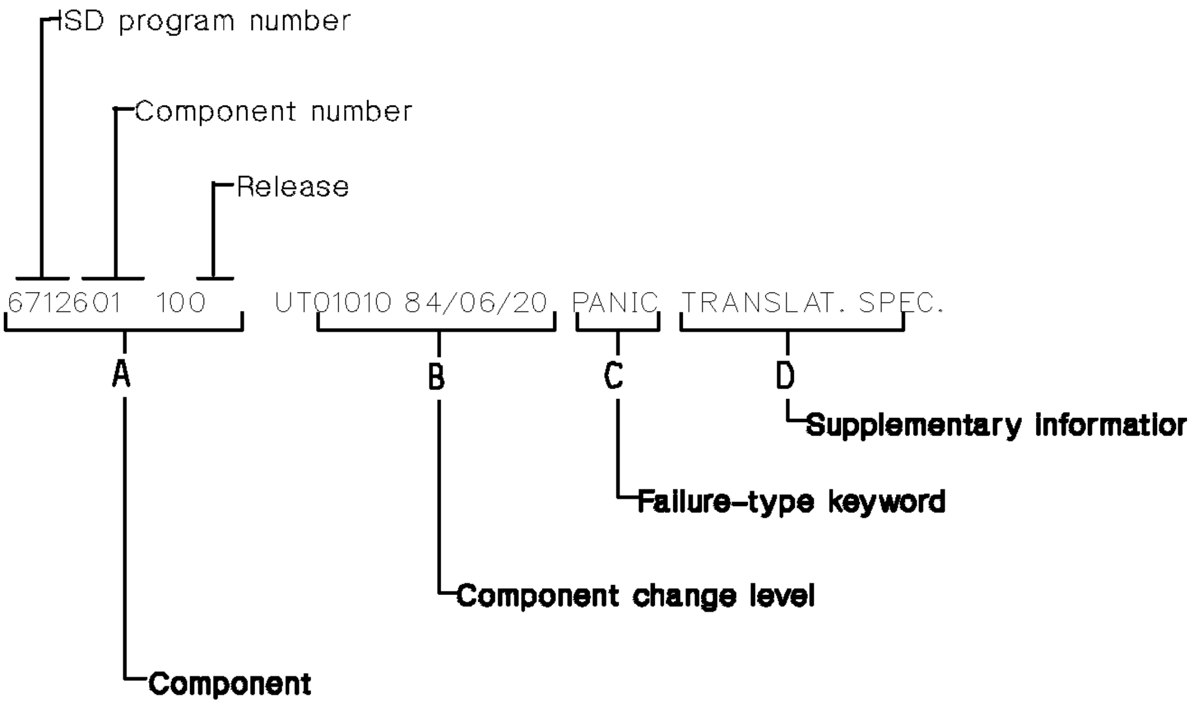

Figure 4-1. Elements of a Problem Symptom String.

These are the items shown in the previous figure.

A *Component*

The component ID (COMPID) includes the following:

IBM's ISD program number, which identifies the product

Component number

Release.

B *Component change level*

The component change level identifies the service level of the affected component by the following:

The number and application date of the program temporary fix (PTF) applied recently, or

The "correctly installed" date if *no* PTF or APAR fix has been installed up to the time the problem occurred.

C *Failure-type keyword*

Depending on the problem you want to report, you supply a specific keyword. For the keywords you can use, refer to "Failure-Type

Keywords" in topic 1.4.3.1.

D *Supplementary information*

Consists of any documentation that is available for the problem type you report. For lists of documentation for specific problem types, refer to Chapter 3, "Collecting Information about a Problem" in topic 1.3.

For the elements A and B, you can retrieve the information from your AIX/370 history file (using the **dhist** command).

AIX/370 commands and parameters use upper- and lowercase letters; entering the correct case is significant for the execution of the function. When you report a problem that relates to commands, you have to tell the IBM service support which case was used, and in which combinations.

Subtopics

1.4.3.1 Failure-Type Keywords

For a search in PROBLEM MANAGEMENT and RETAIN, a program failure should be categorized by one or a combination of the failure-type keywords listed below.

Report your problem to your IBM Support Center with the applicable failure-type keyword(s). The keywords are:

MSGxxxxx An incorrect message occurred, or a message was garbled or missing.

INCORROUT Program output is either incorrect, missing, or duplicated.

LOOP An unintended loop occurred.

WAIT A wait state occurred.

PERFM A performance problem has been noted.

DOC A publication contains incorrect information or required information is missing.

ABEND An abnormal termination of a program occurred.

PANIC A special message indicating an AIX/370 system level problem occurred.

INST Installation-specific problems have been encountered.

AIX370MOD Indicates, in an APAR, that an AIX/370 module is referenced.

CCOM A TCF cluster communication problem was encountered.

1.4.3.2 Example: Problem Reporting

```
 PRIMARY SYMPTOMS
 ----------------

 Keyword  ¦ Supplementary Information
 ---------+---------------------------------------
          ¦
 PANIC    ¦ Translation specification     -sh
          ¦

 DESCRIPTIVE SYMPTOMS
 --------------------
 Component id ¦ Release      ¦ PTF
 or           ¦ from         ¦ or
 Program id   ¦ Program id   ¦ Refresh Level
 -------------------------------------------------

 566712601    ¦ 100          ¦ UT01010
              ¦              ¦
```

Figure 4-2. Example: Problem Reporting

1.4.4 Submitting a Problem Management Record (PMR)

When the problem you found is not yet known to IBM, and if IBM software is suspected as being at fault, your IBM representative may request that you submit a Problem Management Report (PMR). This is done online through a unique database.

If necessary, your PMR will be tracked by the IBM Support Center. Your IBM representative has the telephone number. The support center representative may submit an Authorized Program Analysis Report (APAR) to track further the progress of your problem.

Refer to the following publications for more information:

You and the IBM Support Center, GA21-9824.

IBM Programming Support Center General Information, G229-2228.

Your IBM representative can supply further details about submitting a PMR.

1.4.5 Determining Problem Severity

Your IBM representative may ask you to determine the severity of your problem. Some problems may have more of a direct impact on your operations than others. You should classify your problem in one of the severity categories below.

Severity 1. You are unable to use a program. This is a critica condition that requires an immediate solution.

Severity 2. You are able to use the program but are severel restricted.

Severity 3. You are able to use the program with limited function that are not critical to the overall operation.

Severity 4. You have found a way to circumvent the problem. However an APAR should be considered and action taken as directed by the problem.

In determining the severity of your problem, neither understate nor overstate. Once you have contacted the support center, one of the staff members will assist you in confirming the severity if you need such assistance. If the severity of a problem changes at any time, you can increase or decrease it by calling the support center and asking for the status desk. If you do not specify a severity level, level 3 is assigned.

Subtopics

1.4.5.1 Problem Recording

Record the following information about your problem. Keep a record of all reported problems.

Problem Number:
Originator:
Date:
Time:
Message/External Symptom:

System Action/Reaction:

System Activity at Occurrence of Problem:

Affected:

_ VM
_ AIX/370
_ LAN or LAN Interface
_ Multiple TCF sites
_ Multiple users
_ Single user
_ Other:

Documentation Obtained/Available:
Logs:

_ Terminal
_ Console
_ CPEREP
_

File name and/or location:

Dumps:
_VM
_AIX/370
_

File name and/or location:

Listings:
_ Program
_I/O data
_

File name and/or Location:

Comments:

1.4.5.2 Symptom String

Record the following information about your problem. Keep a record of all reported problems.

For Problem Number:

Apparently Failing Component

COMPID or PID:
Release:
Maintenance or PTF Level:

Messages:
Message Identifier: None?_
Message Text Recorded YES/NO?:

Effect of Action Taken in Response to the Message:

Names (if available) of Affected Modules:

Last Executed or Listed Command:

Frequency or Trigger Factor:

1.4.5.3 Problem Reporting

Record the following information, and keep a record of all reported problems.

```
For Problem Number:

Problem Reported to IBM:

Problem Record Number:

Problem Activity
  APAR/PTF Identified:
  Obtained from:
  Queued to Level 2, Date:
  Severity:

Activity Records for Problem

  Date:                Time:
  Action:

  Date:                Time:
  Action:

  Date:                Time:
  Action:

Problem Resolution

  APAR Number:  TP.....      Applied to:            Date:
  PTF  Number:  UT.....      Applied to:            Date:

  Resolved by APAR/PTF?   YES/NO:

  Emergency Patch applied
    To:
    By:
    Date:

  Other Type of Fix applied
    To:
    By:
    Date:
```

2.0 Part 2: Diagnostic Aids and Debugging Tools

This part of the book gives an overview and examples of debugging tools. The **crash** command is used to debug the kernel on a TCF site.

You can find an overview of the various parts of the AIX system in Chapter 5, "Debugging Tools Overview" in topic 2.5.

For a detailed description of the various tools see *AIX Operating System Commands Reference*. Note that in this document the term *processid* refers to the AIX/370 kernel process identifier.

This part contains the following chapters:

Chapter 5, "Debugging Tools Overview"

Chapter 6, "AIX/370 Kernel Dump Debugging Information"

Chapter 7, "Command and Utility Examples"

Subtopics

2.5 Chapter 5. Debugging Tools Overview

Subtopics

2.5.1 Contents

2.5.2 About This Chapter

This chapter gives an overview of the various tools used for debugging in the AIX PS/2, AIX/370, and VM environments. The tools described here are used on individual TCF sites. For some types of problems, it may be necessary to use these tools and commands on more than one TCF site and combine the results in order to diagnose the problem adequately.

See Chapter 6 for more specific information on debugging the kernel. The appendixes also contain specific debugging information.

Most of the discussion in this chapter focuses on the use of these tools with an individual AIX TCF site. This site is called the ***AIX environment*** for this purpose.

2.5.3 VM Traces Overview (AIX/370 only)

Figure 5-1 shows an overview of the traces that may be useful for debugging within the AIX/370 environment.

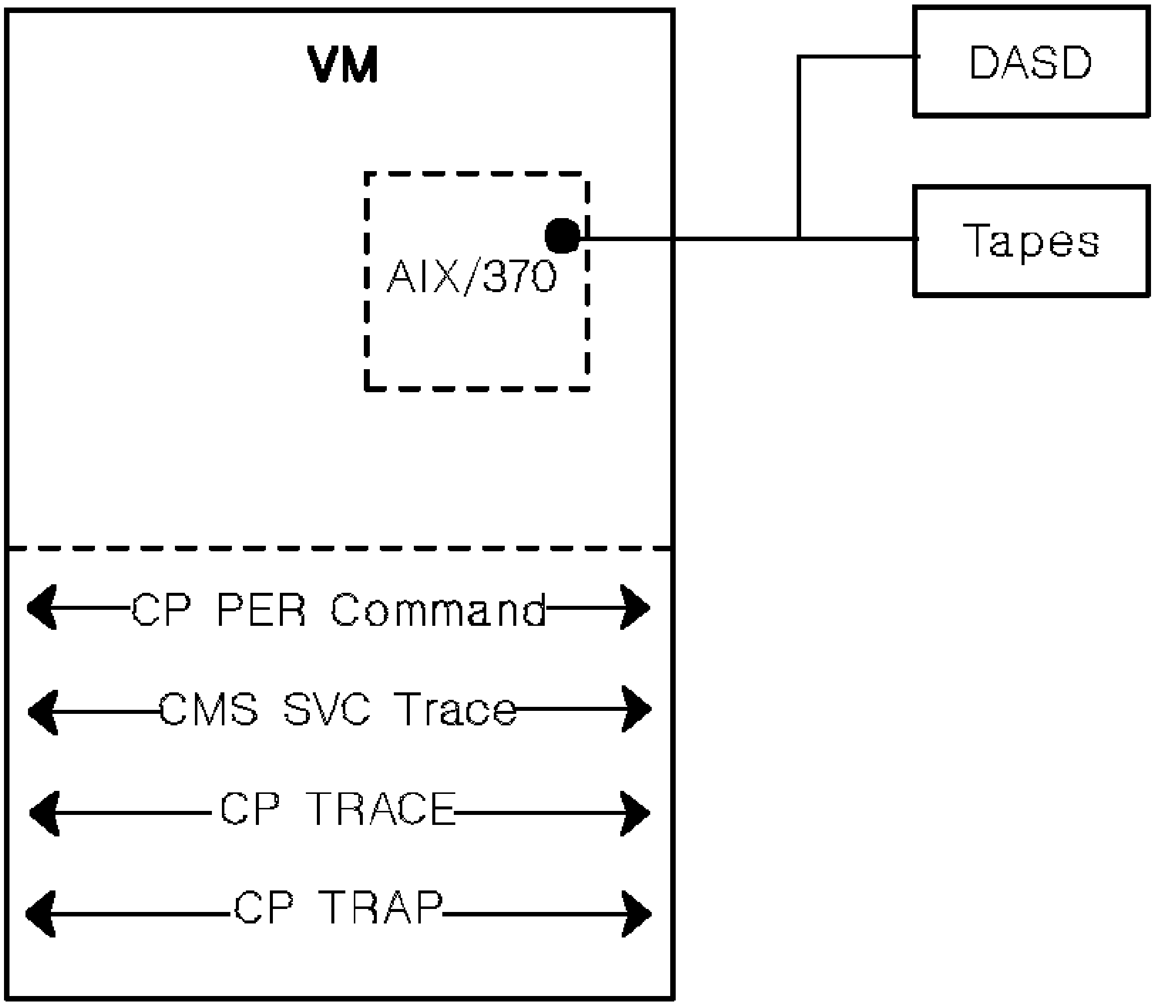

Figure 5-1. Summary of Traces Used for AIX/370

CP PER Command can be used to set up multiple address stops.

CMS SVC Trace traces all SVC interrupts.

CP TRACE traces specified virtual machine activities.

CP TRAP creates a file of selected CP interface and virtual machine interface trace table entries.

2.5.4 AIX Trace Facility

AIX includes an ***event trace facility*** to track certain events in the operation of the system. This trace facility is described in *AIX Programming Tools and Interfaces*. The **trace** and **trcstrop** commands are described in *AIX Operating System Commands Reference*.

2.5.5 Dumps Overview

Figure 5-2 shows the dumps that can be used for debugging in the AIX/370 environment.

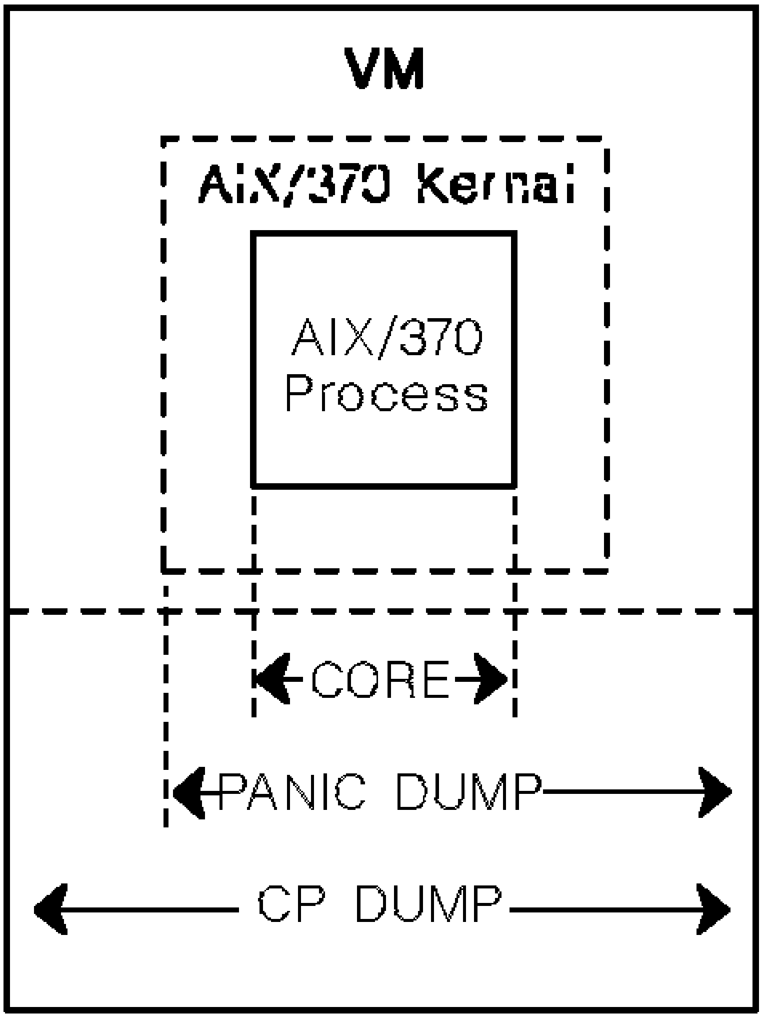

Figure 5-2. Summary of AIX/370 Cluster Site System Dumps

For a description of the various dump types use the following reference list:

CORE Dump See "CORE Dumps" in topic 2.6.3.1.

PANIC Dump See "AIX/370 Dumps after a Panic" in topic 2.6.3.2.

CP Dump See VM documentation (370 only).

2.5.6 The Diagnosis Tools

The following is a brief description of each tool used with an AIX PS/2 or an AIX/370 TCF site.

These tools are described in *AIX Operating System Commands Reference*.

These are the tools and helps that are used for debugging:

crash Interactively examines an operating system core image or the running system, and interprets and formats the various control structures in the system. It is a useful tool to analyze a dump. **crash** can be used both on a live AIX/370 or AIX PS/2 system image and on an AIX/370 or AIX PS/2 process dump.

dbx Symbolically debugs C and assembler programs. **dbx** stands for debugging executive. It may be used to examine both object files as well as core files and provides a controlled environment for their execution.

You can access all program variables symbolically and display them in their correct format. You may also set one or more breakpoints at selected statements or single step through the program line by line.

dump Dumps selected parts of an object file. It formats and prints the information in character, hexadecimal, octal, or decimal representation as appropriate.

fsck Checks and interactively repairs inconsistencies in the file system. **fsck** stands for file system checker.

fsdb Allows examination and patching of a damaged file system. **fsdb** stands for file system debugger.

lnetstat Provides statistical information on the TCF communication traffic, from the perspective of the TCF site on which it is running. It can be used to print a single report, or to generate information periodically. The information printed may be useful in determining if there is some type of TCF communication problem.

od Writes files in a selected format like octal, ASCII, signed or unsigned, decimal or hex to standard output.

osm Displays a recent history of the operating system generated messages. These messages include those that appear on the AIX console, plus additional messages that are not normally displayed on the console but which may be of use in problem diagnosis. **osm** is found in **/dev/osm**.

pstat Allows examination of certain kernel data structures, either from a dump or from the running system. Some of the data structures include the process table, the in-core inodes, and the process tracking tables. **pstat** is useful for certain types of low-level system diagnosis. Its function partially overlaps that of **crash**.

savecore Maintains the system dump file and copies AIX/370 Kernel dumps to ordinary files for archival purpose or examination by crash.

The following VM service facilities are available if the device is detached from AIX/370 and attached to a VM service machine.

DSF All **DSF** functions are available when you have attached the devices to VM with the exception of:

Initialize disk in the AIX/370 format
VTOC management (no VTOC on AIX/370 disks).

DSF stands for device support facility.

ERDS Is the VM error recording data set. This data set is also called SYSLOG.

CP EREP Edits and prints the ERDS records.

AIX/370 uses the following three record types:

Unit check/outboard record (OBR)
Miscellaneous data record (MDR)
Software record (SFT).

OLTSEP All functions supported by VM are also supported for AIX/370 when they have been attached to VM.

VM Tools For a description of the following tools see the appropriate VM documentation.

CP TRAP
CP TRACE
PER Command
CMS SVC Trace.

2.6 Chapter 6. AIX/370 Kernel Dump Debugging Information

Subtopics

2.6.1 Contents

2.6.2 About This Chapter

Both AIX/370 and AIX PS/2, by default, produce a dump of the machine memory whenever a process panics or when the operator requests a system dump.

This chapter describes the types of dumps and explains how to manage the dump file.

2.6.3 Types of Dumps

Subtopics

2.6.3.1 CORE Dumps

A core dump is written whenever the AIX user (the process) detects an error in a user process. A user can force a core dump using the QUIT character or the **kill -quit** command. The system informs a user about this event by displaying a 'core dumped' message on the user's workstation. This dump contains the core image of the failing process. It is written to the file in the working directory with the file name 'core', provided that the user has permission to create a file in the directory, and the coredump size and file size limits are not exceeded.

You may analyze these core dumps with the **dbx** command, described in *AIX Operating System Commands Reference*.

2.6.3.2 AIX/370 Dumps after a Panic

When an AIX/370 or an AIX PS/2 kernel panics, the following activities occur:

1. All activity by other processes is frozen.

2. "Panic" is displayed on the console, followed by the reason and other information.

3. A dump of the panicking kernel is automatically taken on the **/dev/dump** partition if dumping is enabled. See "How to Enable Dump Writing" in topic 2.6.6.

4. The system will then reboot.

2.6.3.3 Operator-requested Dumps

The operator may request that the system produce a kernel dump. This dump is also stored on the **/dev/dump** device.

See "How to Perform a Kernel Dump" in topic 2.6.9.

Subtopics

2.6.3.3.1 User-forced Dump

These are the error situations which may require a ***user-forced dump***:

Panic hangs
An error situation where the Panic operation cannot end and no automatic dump is written.

An AIX process hangs
An error situation where a specific process hangs.

AIX PS/2 or AIX/370 hangs
An error situation where the AIX/370 or AIX PS/2 operating system hangs.

2.6.4 The Dump Partition

The ***dump partition*** is where AIX makes a copy of memory when a panic occurs or a dump is requested. In AIX/370, the dump partition is normally on a minidisk that is formatted for dumps instead of an AIX file system. In AIX PS/2, the dump partition may be either an unformatted minidisk or the floppy disk drive.

2.6.5 How to Define the Dump Partition (AIX/370 only)

The **minidisk** command is used to assign a minidisk as a dump device.

Note: The **dumpdev** parameter in the system configuration file must be set to correspond to this minidisk.

The dump partition:

- Cannot also be used for a file system

- Is initialized in AIX/370 with the **savecore -I -d /dev/dump** command. **mkfs** should never be used to create a normal AIX/370 file system on the dump partition once that partition has been set up for dumping. Creating a normal file system on the dump partition can destroy dump data and cause failure of future dumps.

- Should be at least as large as the machine memory size so that a least one dump can be stored.

2.6.6 How to Enable Dump Writing

Make sure that the partition has enough space to store the dump. For an AIX/370 site the directory **<LOCAL>/dumplocks** must exist and the dump partition must be initialized with **savecore -I**.

2.6.7 How to Maintain the Dump Partition (AIX/370 only)

The **savecore** command is used to maintain the dump partition. It can also be used to copy dumps to ordinary files for archival or for examination by crash.

Note: The **savecore** command must be run on the TCF site where the dump partition of interest is located.

Subtopics

2.6.7.1 The savecore Command (AIX/370 only)

You may perform the following **savecore** functions one at a time:

Copy the dump specified by **dumpid** from the dump partition along with its symbol table information to the file named **filename**.

```
savecore -e dumpid -d dumpfile > filename
```

Remove the dump specified by **dumpid** from the dump partition and reclaim all space. The **dumpid** is not be reused until the dump partition is reinitialized with the -I option.

```
savecore -r dumpid -d dumpfile
```

Clear all dumps from the dump partition and reinitialize it for use The dump partition must be cleared and initialized before it can be used.

```
savecore -I -d dumpfile
```

Print a summary of all dumps in the dump partition including dump ID size, date, time, and system name. Panic strings and operator comments are listed if present. The listing is written to standard output.

```
savecore -t -d dumpfile
```

dumpid Stands for the numerical dumpid.

dumpfile Stands for the name of the dump partition. This name can be used to access the dump partition of another subsystem. The default specification is /dev/dump.

filename Stands for the name of the ordinary file where you want to copy the dump.

This is the default command which is executed if you enter **savecore** without an argument:

```
savecore -t -d /dev/dump
```

2.6.7.2 Dump Handling Sequence Example

The **savecore** command is also used to copy dumps to ordinary files for examination by crash. Figure 6-1 shows an example of a typical dump handling sequence. These are the steps shown in Figure 6-1:

1. List the dumpids and other data about dumps present in the dump partition.
2. Copy dump 12 to an ordinary file.
3. Remove the dump from the dump partition to allow its space to be reused.
4. Execute crash to examine the dump interactively.

```
savecore -t
savecore -e 12 >dump12
savecore -r 12
crash -d dump12
```

Figure 6-1. Dump Handling Sequence

For a more detailed description of the **savecore** command, see *AIX Operating System Commands Reference*.

2.6.8 How to Produce the Kernel Map

An important requisite for problem source identification is the ***AIX/370 map***. This map reflects the actual kernel module names with all their relevant addresses. Print this list after each AIX/370 kernel reconfiguration and the subsequent kernel regenerations.

Use one of the following commands on the appropriate TCF site to get the AIX Kernel map. You may print the map with this command:

```
nm -vex /unix | print
```

With these options, the map is displayed in increasing symbol order, which is useful for finding things between symbols..

You display the map on your screen with the following command:

```
nm -vex /unix
```

You can put the map into a file with this command:

```
nm -vex /unix > /tmp/unix.map
```

The **nm** command is described in *AIX Operating System Commands Reference*.

2.6.9 How to Perform a Kernel Dump

Occasionally important processes may work in an unusual manner while the system is still up. In those cases, you may do the following:

1. Obtain a dump of the troublesome process.
2. Analyze the dump to find the reason for the problem.

You have to find out the process ID of the process which you want to dump first. Figure 6-2 shows the system response to the either form of dump request.

```
02:11:47
02:11:47 panic: Kernel protection exception
02:11:47 dpdump: dump number 1 in progress
```

Figure 6-2. Panic Screen

The dump has been performed and processing continues normally.

2.6.10 How to Force a Nonfatal System Dump in AIX/370

The following console log extract illustrates the user-process dump activities. The system operator can use the **sysdump** command to force a nonfatal system dump. The **sysdump -p** option reboots the system after the dump is completed.

Subtopics

2.6.10.1 How to Force a Panic

Even when commands cannot be executed from the AIX/370 system console, the system operator can force a panic:

In AIX/370, the operator presses ¬5! **Enter** key).

In AIX PS/2, the operator simultaneously presses **Ctrl-Alt-Right Shift** followed by the **!** key.

If the kernel is in a disabled loop, you can also force a dump using **snoop**. First, enter **snoop** using the PA1 key. Then enter **SYS RESTART**. Then enter the following:

```
DP        (to display the PSW)
MP 1      (oldpsw1 + 1)
GO
```

where oldpsw1 is what you got at the second word from the DP. This causes the kernel to trap.

2.6.10.2 Display /dev/dump Status Information

In AIX/370, a display of the contents of the **/dev/dump** file also shows the user process dumps and the current used space. Enter the following command to display the dump file status information:

```
savecore
```

Figure 6-3 shows the output of the **savecore** command.

```
1      1560    ETTEBEH Jun  11 19:09
       Kernel trap: page
       Dump flags: x71, " shared private full frozen"
2      1560 ETTEBEH Jun  13 19:09
       Kernel trap: segment
       Dump flags: x71, " shared private full frozen"
3120 blocks used, 2 dumps used
```

Figure 6-3. Savecore Command Output

2.6.11 How to Copy a Dump to a File (AIX/370 Only)

You have to **copy** the dump which you want to analyze from the **/dev/dump** partition to an ordinary file before you can use the **crash** command to display or print dump information. Use the following command to copy the dump:

```
savecore -e dumpid > filename
```

This command copies the dump identified by **dumpid** to a file named **filename**.

Note: Full dumps of the AIX/370 kernel can be quite large. Hence, *filename* should be stored in a file system where sufficient space exists to hold the dump.

2.6.12 How to Start crash

You use the **crash** command to investigate AIX kernel dumps.

```
crash -d filename
```

With the options of that command you may extract system information such as:

Selected fields from the interrupt storage are
System stac
Process tabl
User bloc
Global and local file system mount table
Network message histor
Site table and dat
Topology change variable
Recent console messages

WARNINGDo not use the **crash** command in the background, as shown in this example:

```
crash -d filename print &
```

After the prompt is returned, enter the desired **crash** command options.

2.6.13 How to Direct the crash Output

The dump information extracted with crash is directed to your terminal if you do not redirect it explicitly. You may use one of the following methods to direct the **crash** output.

To direct the output only to the terminal, enter:

```
crash -d filename
```

To direct the output only to a printer, enter:

```
crash -d filename | print
```

To direct the output to a file (named *outfile*), enter:

```
crash -d filename > outfile
```

2.6.14 How to End the Output Operation

End the **crash** session with the quit option.

```
quit
```

2.6.15 How to Analyze a Dump with crash

This section describes an example of a typical crash session. It shows the crash commands and their output. The result of this session is a file that contains the output of the entered commands. This example displays the following information:

Statistical dat
Interrupt save are
System stack informatio
User bloc
Process tabl
Last runable proces
Recent network message histor
Network message buffe
Global and local mount table
Site table and dat
Topology change variable
Server process table

Depending on the type of problem, the following dump information may be useful for error analysis.

crash options to display information of common interest:

stat	Extracts statistical data
page 0	System page 0 parameters
t	Extract the system stack information
netlog	Network message history log

The following information, depending on the error, is useful:

In case of *program checks*:

user	User block
proc	Process table
dump	Dump of user area

For TCF cluster communication problem

netbuf	Network message buffers
netlist	Network message free list and headers

For all other kind of errors

proc -r	Last runable process If you do not get the correct output from that option, use the following steps:
user	Displays the user block. Find the process pointer (**procp**) in that block.
p slot	Use the process table slot number to address the **slot** of the affected process.

For a more detailed description of the **crash** command see *AIX Operating System Commands Reference*.

Subtopics

2.6.15.1 Start Display Session

Initiate the display session with the following command:

```
crash -d filename > outfile
```

The data is recorded in the file named outfile and the prompt sign is displayed at your terminal.

2.6.15.2 Display Available crash Command Options

You may enter a **?** (question mark) to display a list of the available options at your terminal as shown in Figure 6-4. The aliases are alternate ways to invoke the **crash** command. They are provided either to maintain compatibility with previous versions of **crash** or for the convenience of the **crash** user.

```
AIX Dump Analyzer
Available commands:
active    display active entries only
addr      Print address of an array item, eg Buf 10  aliases:  a
all       all information available
buffer    buffer data                                aliases:  b
buf       buffer headers                             aliases:  bufhdr hdr
bufhash   buffer hash chains                         aliases:  bufh
buflist   buffer free lists                          aliases:  bufl
dcache    directory cache                            aliases:  dca
ds        namelist search by address
file      file table                                 aliases:  f files
gensw     gensw structure
inode     inode table                                aliases:  ino i
mount     mount table - global and local             aliases:  m mnt
netbuf    net message buffers                        aliases:  netb netmsg
netlist   net message free list and nmheader
netlog    log of recent net messages                 aliases:  netl
netswitch net switch structure                       aliases:  netsw
nm        namelist search by name
dump      dump symbol values                         aliases:  hd od rd
osm       most recent printfs                        aliases:  printf printf
pcb       process control block
page0     Page 0                                     aliases:  pa
proc      process table                              aliases:  p
prop      update propagation queue
pvseg     process vseg structures
quit      exit                                       aliases:  q
rmsleep   remote sleep table                         aliases:  rms
rmwakeup  remote wakeup table                        aliases:  rmw
site      site table and site data
slot      Print name and slot for an address         aliases:  s
sptab     server process table                       aliases:  sp
stat      dump statistics
tabgrow   used/avail ratio of various system tables  aliases:  systab
token     token control block table                  aliases:  tok
tokreq    token site request table
topology  topology change variables                  aliases:  top
callout   callout table                              aliases:  calls call t
trace     kernel or user stack trace                 aliases:  t
ts        namelist search by address
tty       tty structures
user      user area                                  aliases:  u_area uarea
var       system variables                           aliases:  tunable tunab
vseg      kernel vseg structures
!         escape to shell
help      description of commands                    aliases:  ?
usage     usage info for subcommands
          age of system: 1 min.
          panic: vsunlock
```

Figure 6-4. crash Command Option Summary

2.6.15.3 Display page0 Save Area

Enter the following crash command option to display the interrupt save area:

```
page0
```

Figure 6-5 shows the output of this option.

```
OLD Program Status Words:
Restart                  00000000 00000000
External Interrupt       00000000 00000000
SVC                      00000000 00000000
Machine Check            00000000 00000000
I/O                      00000000 00000000

NEW Program Status Words:
Restart                  00000000 00000000
External Interrupt       00000000 00000000
SVC                      00000000 00000000
Machine Check            00000000 00000000
I/O                      00000000 00000000

SVC Id:                  00000000
Program Check ID:        00000000
Translation Exception:   00000000

Kernel Addresses:
&kstack                  0x3ff00000
Ksp                      0x3ff01c90
Stack contains 2064 bytes
from 3ff01c90 to 3ff024a0.
```

Figure 6-5. Page0 Option Output

2.6.15.4 Dump the System Stack

The system stack contains the various registers at the time of error. Use the following commands to display and record the user and the system stack:

```
trace       returns the system stack
trace -u    displays the user stack



KERNEL STACK (pid  1084):
==========================

ADDRESS         CONTENTS

3ff01c90               1 (R13) stack pointer
3ff01c94        3ff01cf8 <-- back chain
3ff01c98        30081954
3ff01c9c        3ff01d58
3ff01ca0               0
3ff01ca4               0
3ff01ca8        30252000
3ff01cac               1
3ff01cb0               0
3ff01cb4        3ff01e70
3ff01cb8            2000
3ff01cbc        3055ee80
3ff01cc0               0
3ff01cc4        300d8838
3ff01cc8        30098434
3ff01ccc        3ff01c90
3ff01cd0        b00984b8 R14  (_panic+0x84)????
3ff01cd4        3001d208
3ff01cd8        300d88b8
3ff01cdc        300d08a0
3ff01ce0               0
3ff01ce4               0
3ff01ce8        300d9168
3ff01cec        30530324
3ff01cf0               0
3ff01cf4               0
3ff01cf8        2e627373 (R13) stack pointer
3ff01cfc        3ff01d58 <-- back chain
3ff01d00         1009890
3ff01d04         1009890
3ff01d08            7c28
3ff01d0c               0
3ff01d10               0
3ff01d14               0
3ff01d18               0
3ff01d1c              80
3ff01d20               0
3ff01d24               0
3ff01d28               0
3ff01d2c        3055ee80
3ff01d30        3002f424
3ff01d34        3ff01cf8
3ff01d38        b002f45c R14  (_vsunlock+0x38)
3ff01d3c        30098434
```

```
3ff01d40         300d08a0
3ff01d44         23c8fcla
3ff01d48            20000
3ff01d4c              417
3ff01d50           1c0203
3ff01d54                0
3ff01d58                0 (R13) stack pointer
3ff01d5c         3ff01ea0 <-- back chain
3ff01d60                0
3ff01d64         300b3336
3ff01d68                0
3ff01d6c                0
3ff01d70         30252000
3ff01d74                1
3ff01d78                0
3ff01d7c         3ff01e70
3ff01d80             2000
3ff01d84         3055ee80
3ff01d88                0
3ff01d8c         3ff024a0
3ff01d90         30065b24
3ff01d94         3ff01d58
3ff01d98         b00664fc R14 (_getcoff+0x9d8)
3ff01d9c         3002f424
3ff01da0         30530324
3ff01da4         3ff01e40
3ff01da8         300d1b8c
3ff01dac         300e0178
3ff01db0         300812ec
3ff01db4         3ff01d78
3ff01db8         b0081368
3ff01dbc         300a1b58
3ff01dc0             1000
3ff01dc4                7
3ff01dc8               c0
3ff01dcc         3ff01e40
3ff01dd0                0
3ff01dd4         30407080
3ff01dd8                4
3ff01ddc              380
3ff01de0         303e29c4
3ff01de4         3009b730
3ff01de8                1
3ff01dec               16
3ff01df0               12
3ff01df4         303c6c1c
3ff01df8                0
3ff01dfc         3009b7c4
3ff01e00         3ff01f06
3ff01e04                1
3ff01e08         300d1b8c
3ff01e0c         3ff01e68
3ff01e10                2
3ff01e14         3ff01dd8
3ff01e18         b0081124
3ff01e1c         300812ec
3ff01e20                0
3ff01e24                0
3ff01e28                0
3ff01e2c                0
```

```
3ff01e30        3ff01e70
3ff01e34               4
3ff01e38               0
3ff01e3c               0
3ff01e40        300b3678
3ff01e44        3ff01ea0
3ff01e48          7ff000
3ff01e4c        3ff01eb0
3ff01e50               1
3ff01e54               0
3ff01e58            93fb
3ff01e5c               1
3ff01e60              21
3ff01e64        3012cd70
3ff01e68               0
3ff01e6c        3ff024a0
3ff01e70               2
3ff01e74            9448
3ff01e78        3008d324
3ff01e7c        303e29c4
3ff01e80              e0
3ff01e84            1000
3ff01e88               0
3ff01e8c           15800
3ff01e90         1000000
3ff01e94           16000
3ff01e98            9890
3ff01e9c            7c28
3ff01ea0        30097390 (R13) stack pointer
3ff01ea4        3ff01f08 <-- back chain
3ff01ea8               0
3ff01eac               0
3ff01eb0        3ff01f06
3ff01eb4               0
3ff01eb8        30252000
3ff01ebc               1
3ff01ec0              2d
3ff01ec4        303e29c4
3ff01ec8        303c6c1c
3ff01ecc        3055ee80
3ff01ed0        3ff024a0
3ff01ed4               0
3ff01ed8        30065914
3ff01edc        3ff01ea0
3ff01ee0        b0065984 R14 ( _bodygetxfile+0x70)
3ff01ee4        30065b24
3ff01ee8               1
3ff01eec        3ff024a0
3ff01ef0               2
3ff01ef4            9448
3ff01ef8        3008d324
3ff01efc        3ff01ec0
3ff01f00        b008d4be
3ff01f04        30080001
3ff01f08            8000 (R13) stack pointer
3ff01f0c        3ff01f70 <-- back chain
3ff01f10        3008dcf0
3ff01f14             fff
3ff01f18              2d
3ff01f1c               0
```

```
3ff01f20         30252000
3ff01f24                1
3ff01f28               2d
3ff01f2c         303e29c4
3ff01f30              2bc
3ff01f34             1922
3ff01f38                0
3ff01f3c         3ff024a0
3ff01f40         30065240
3ff01f44         3ff01f08
3ff01f48         b00652ca R14 (_getxfile+0x8a)
3ff01f4c         30065914
3ff01f50                0
3ff01f54         3ff024a0
3ff01f58         300780b0
3ff01f5c         3ff01f20
3ff01f60                0
3ff01f64         3008326c
3ff01f68         b008c116
3ff01f6c         300a8f28
3ff01f70               14 (R13) stack pointer
3ff01f74         3ff02000 <-- back chain
3ff01f78         30077d64
3ff01f7c         3ff01f40
3ff01f80               2d
3ff01f84             1922
3ff01f88         30252000
3ff01f8c                1
3ff01f90               2d
3ff01f94                0
3ff01f98         3ff020c0
3ff01f9c                0
3ff01fa0                0
3ff01fa4              2bc
3ff01fa8         30064cc4
3ff01fac         3ff01f70
3ff01fb0         b0064d16 R14 (_prep_exec+0x52)
3ff01fb4         30065240
3ff01fb8         30252000
3ff01fbc                1
3ff01fc0                0
3ff01fc4         3008326c
3ff01fc8                2
3ff01fcc                0
3ff01fd0                0
3ff01fd4         3ff024a0
3ff01fd8         30061f20
3ff01fdc         3ff01fa0
3ff01fe0         b0062026
3ff01fe4         fffffffc
3ff01fe8         3009b924
3ff01fec                0
3ff01ff0           220060
3ff01ff4               2d
3ff01ff8                0
3ff01ffc                8
3ff02000         300b3678 (R13) stack pointer
3ff02004         3ff02158 <-- back chain
3ff02008         3009c67c
3ff0200c         3ff01fd0
```

```
3ff02010            1922
3ff02014               2
3ff02018        30252000
3ff0201c               1
3ff02020               0
3ff02024               0
3ff02028        3ff020c0
3ff0202c        303c6c1c
3ff02030               0
3ff02034        3ff024a0
3ff02038        300605d8
3ff0203c        3ff02000
3ff02040        b0060f40 R14 (_svrfpr+0x968)
3ff02044        30064cc4
3ff02048        30085a4c
3ff0204c        3ff02010
3ff02050        b0085b66
3ff02054        30086a64
3ff02058              2d
3ff0205c              12
3ff02060             2bb
3ff02064               0
3ff02068               0
3ff0206c        3ff020c0
3ff02070        b00936cc
3ff02074        3009c67c
3ff02078               4
3ff0207c        303e29c4
3ff02080               2
3ff02084            1922
3ff02088               2
3ff0208c               0
3ff02090               4
3ff02094        3008dcf0
3ff02098          6383d
3ff0209c        300b3312
3ff020a0               1
3ff020a4               0
3ff020a8        303c2bcc
3ff020ac        30079faa
3ff020b0               2
3ff020b4               0
3ff020b8        300b3338
3ff020bc           62e56
3ff020c0        78780028
3ff020c4        2853454d
3ff020c8        55535a2a
3ff020cc        5345afd4
3ff020d0         15f0003
3ff020d4        23c8fcla
3ff020d8           20000
3ff020dc             417
3ff020e0          1c0203
3ff020e4         1080000
3ff020e8           15758
3ff020ec            9890
3ff020f0            7c28
3ff020f4           100b4
3ff020f8           100a8
3ff020fc         1000000
```

```
3ff02100        2e746578
3ff02104        74000000
3ff02108           100a8
3ff0210c           100a8
3ff02110           15758
3ff02114              a8
3ff02118               0
3ff0211c               0
3ff02120               0
3ff02124              20
3ff02128        2e646174
3ff0212c        61000000
3ff02130         1000000
3ff02134         1000000
3ff02138            9890
3ff0213c           16000
3ff02140               0
3ff02144               0
3ff02148               0
3ff0214c              40
3ff02150        2e627373
3ff02154        303dce5c
3ff02158        3001a280 (R13) stack pointer
3ff0215c        3ff021f0 <-- back chain
3ff02160        b001a424
3ff02164        3001e9ec
3ff02168             330
3ff0216c               4
3ff02170        30134af0
3ff02174        305334fc
3ff02178        300e0e20
3ff0217c        3ff024a0
3ff02180               1
3ff02184               0
3ff02188               8
3ff0218c        3007617c
3ff02190        30075ccc
3ff02194        3ff02158
3ff02198        b0076044 R14 (_svrproc+0x378)
3ff0219c        300605d8
3ff021a0        300d65c8
3ff021a4               2
3ff021a8             80a
3ff021ac              1e
3ff021b0        303c490c
3ff021b4               1
3ff021b8              15
3ff021bc        30097af0
3ff021c0        300aae00
3ff021c4        30097af0
3ff021c8           1dea3
3ff021cc        300b3336
3ff021d0        300b06d0
3ff021d4        300b331a
3ff021d8        b00b2c66
3ff021dc        30150004
3ff021e0             330
3ff021e4           1dea3
3ff021e8               8
3ff021ec               0
```

```
3ff021f0         31373a32 (R13) stack pointer
3ff021f4         3ff02260 <-- back chain
3ff021f8                0
3ff021fc                0
3ff02200                c
3ff02204         3ff024a0
3ff02208                7
3ff0220c         300aae00
3ff02210         300d2db8
3ff02214         3ff024a0
3ff02218         ffffffff
3ff0221c                1
3ff02220         300c7e8c
3ff02224         300c7e88
3ff02228         3004d918
3ff0222c         3ff021f0
3ff02230         b004dbc0 R14 (_startsvrs+0x2a8)
3ff02234         30075ccc
3ff02238         3004d8a0
3ff0223c         3ff02200
3ff02240         b004d8dc
3ff02244         300b2a64
3ff02248         3061c088
3ff0224c         3061c888
3ff02250         300b3544
3ff02254           400000
3ff02258         3ff02780
3ff0225c         3004a534
3ff02260                1 (R13) stack pointer
3ff02264         3ff022c8 <-- back chain
3ff02268                5
3ff0226c         3ff022c8
3ff02270         300b3678
3ff02274         3ff024a0
3ff02278                1
3ff0227c         ffffffff
3ff02280         300e6148
3ff02284         300d9168
3ff02288                1
3ff0228c                2
3ff02290         300d2db8
3ff02294         3004d918
3ff02298         3004d66c
3ff0229c         3ff02260
3ff022a0         b004d706 R14 (_startnet+0x9a)
3ff022a4         3004d918
3ff022a8         b0097b4a
3ff022ac         30097cb4
3ff022b0         3002ec7c
3ff022b4         3ff02338
3ff022b8         b002ecce
3ff022bc         3002ed60
3ff022c0         300da868
3ff022c4         3003d010
3ff022c8                5 (R13) stack pointer
3ff022cc         3ff02338 <-- back chain
3ff022d0                9
3ff022d4         3ff02338
3ff022d8         300b3678
3ff022dc         3ff024a0
```

```
3ff022e0                1
3ff022e4         ffffffff
3ff022e8         300de954
3ff022ec                1
3ff022f0               40
3ff022f4         3ff024a0
3ff022f8                1
3ff022fc         3ff02538
3ff02300         3007ea9c
3ff02304         3ff022c8
3ff02308         b007eb8a R14 (_netctr1+0xee)
3ff0230c         3004d66c
3ff02310         300d7658
3ff02314         300d7640
3ff02318         300b3678
3ff0231c         3ff024a0
3ff02320                1
3ff02324                a
3ff02328         3053a6e4
3ff0232c               e0
3ff02330                0
3ff02334         300b3338
3ff02338         3ff02538 (R13) stack pointer
3ff0233c         3ff023c0 <-- back chain
3ff02340         b0097ad0
3ff02344         30097cb4
3ff02348         300b3678
3ff0234c            10740
3ff02350         3ff02538
3ff02354         303c4804
3ff02358         300de954
3ff0235c                0
3ff02360               40
3ff02364         3ff024a0
3ff02368                1
3ff0236c                6
3ff02370         300c444c
3ff02374         3ff02338
3ff02378         b00c4672 R14 (_svctrap+0x226)
3ff0237c         3007ea9c
3ff02380           1cc000
3ff02384                0
3ff02388           10f880
3ff0238c           13b7d0
3ff02390         b00afdf2
3ff02394         3ff024a0
3ff02398         3001a1cc
3ff0239c         3ff02360
3ff023a0                0
3ff023a4            10000
3ff023a8                0
3ff023ac             79c2
3ff023b0           764000
3ff023b4                6
3ff023b8               40
3ff023bc             79c2
3ff023c0                0 (R13) stack pointer
3ff023c4                0
3ff023c8                0
3ff023cc                0
```

```
3ff023d0        300b3678
3ff023d4           10740
3ff023d8               0
3ff023dc        30145900
3ff023e0               1
3ff023e4               1
3ff023e8           20008
3ff023ec        3ff02418
3ff023f0          7ffe04
3ff023f4           1073a
3ff023f8           10154
3ff023fc        3ff023c0
3ff02400        b00afe96 R14 ( svc_ent+0x76)
3ff02404        300c444c
3ff02408        3ff02418
3ff0240c               0
3ff02410              12
3ff02414          1cefd0
3ff02418              80
3ff0241c          7ffe04
3ff02420          7ffecc
3ff02424               0
3ff02428               0
3ff0242c          7ffec4
3ff02430               1
3ff02434               1
3ff02438           20008
3ff0243c           1436c
3ff02440          7ffe04
3ff02444           1073a
3ff02448           10154
3ff0244c          7ffd50 <-- R13 trapblk sp (back chain)
3ff02450        40010274
3ff02454              40
3ff02458               0
3ff0245c               0
3ff02460               0
3ff02464               0
3ff02468               0
3ff0246c               0
3ff02470               0
3ff02474               0
3ff02478         7ed0000
3ff0247c           10740 PSW trapblk
3ff02480           2126c
3ff02484          152bf0
3ff02488         c000000
3ff0248c           2000a
3ff02490               0
3ff02490        300b3679
3ff02498        6b6b636b
3ff0249c               0
```

Figure 6-6. Trace Output

2.6.15.5 Display the User Block

What you usually need to find out next is the process pointer of the last process involved before the error came up. That information is contained in the user block. Enter the following crash command option to display that user block:

```
u
```

Figure 6-7 shows a screen output example of the crash option *u* (user block). Find the *procp=xx* entry in that block which contains this process pointer.

You will find the following information in that output example:

- Process pointe
- Terminal numbe
- Last command executed

```
                          U-BLOCK

u_comm = xx, uid=6434, gid=2, ruid=6434, rgid=2, groups = {2, 6}
error=0, seg flag=1, u_ar0=x3ff02418
pid=115985, proc=50, xprocp=303c6c1c, rtrn-val="x00000000 x00000040"

FILE I/O:      base=x3ff01d20, count=0, offset=xa8, i-count=-1
OPEN FILES:    file desc:     0   1   2
               file slot:   151 164 96
               file flag:   x00 x00 x00
DIRECTORIES:   cdir=x303e70dc, rdir=x0, dirp=x30280400,
               Local fs: /swords, gfs=240, d-inode=2
SYSCALL ARGS:  x00000000 x007ffe04 x00000000 x00000000 x00000000
MEMORY USAGE:  tsize   x0005   dsize   x0026   ssize   x0001
SIGNALS: hex
00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
00000000 00000000 00000000 00000000 00000000 00000001 00000001 00000000
00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
00000000 00000000 00000000 00000000 00000000 00000000 00000000
TRAP CODE:      4001    Trap block:     x3ff02418
LABELs for goto's on interrupts: x    3a3 x      5 x30134af0 x3052f01c x300e0e
 x3ff024a0 x3052f01c x300876ac x3ff020f8 xb00876fa x300b35ce x     0 x  0
USER SETTABLE FLAGS:     NONE
Pointer to Specific Request Structure: 3ff020f2

PROFILE ARGUMENTS:
  Buffer Base:  0
  Buffer Size:  0
  PC Offset:    0
  PC Scaling:   0
END of SYSCALL ACTION (decimal):        0

START:         Mon Jan  9  10:41:33 1989

ACFLAG:        1
```

```
NUM Locked SVR's: 1
File Creation Mask:    2

LIMITS:
       0 7fffffff 7fffffff 7fffffff 7fffffff 7fffffff 7fffffff 7fffffff
USER STATISTICS (decimal):
  User Time:             0 sec 0 usec
  System Time:           0 sec 0 usec
  Maximum RSS:           0
  Integral of Text RSS: 0
  Integral of d+s RSS:  0
  Minor Page Faults:     0
  Major Page Faults:     0
                Fill from Swap:          0
                Fill from File Sys:      0
                Demand Fill with Zeroes:         0
  Number of Swaps:       0
  Block Reads:           0
  Block Writes:          0
  Messages Sent:         0
  Messages Received:     0
  Signals Received:      0
  Voluntary Context switches:  1
  Involuntary Context Switches:          0
  pfault: Copy on Write:        0
  pfault: Reclaimed pages:      0


CHILD STATISTICS (decimal):
  User Time:             0 sec 40000 usec
  System Time:           0 sec 129597 usec
  Maximum RSS:           112
  Integral of Text RSS: 84
  Integral of d+s RSS:  644
  Minor Page Faults:     0
  Major Page Faults:     0
                Fill from Swap:          0
                Fill from File Sys:      0
                Demand Fill with Zeroes:         0
  Number of Swaps:       0
  Block Reads:           3
  Block Writes:          0
  Messages Sent:         0
  Messages Received:     0
  Signals Received:      0
  Voluntary Context switches:  6
  Involuntary Context Switches:          0
  pfault: Copy on Write:        10
  pfault: Reclaimed pages:      1

A.OUT HDR:
        magic= 410     tsize=x000016   dsize=x00000a
        symsize=x100b4 trelsize=x1000000    drelsize=x0     entry=x0100a8
        bssize=x000008 entry=x100a8
```

Figure 6-7. User Block Display

2.6.15.6 Display Process Table Entries

You display the table entry of the last runable process with the following crash option:

```
p -r
```

The displayed table entry contains information about the state of the process and the process ID. If no information is displayed, the **procp=xx** of the user block displayed previously will supply the entry to the slot of the process structure. The process number in our example is 40. The example shown below shows screen output of the following crash option:

```
p 40
```

You will find the following information in that output example:

Involved AIX/370 device is 0700 (ttyd=0700) which is the AIX/37 console in our example
Command execute
State of the process

```
PROCESS TABLE
active listp: x303c766c   free listp:     x303c787c
active slot :     60      free slot :         62

p  40 addr=x303c61cc state=asleep     flags(x900001)=" load ousig"
     nice:    20 sig.1: x0    sig.h: x0  vsegs: x3055be44
     pgrp:  100127 pid:  100127           ppid:  1
     uid:    0    suid:    0 :
     parent: 1    parentp: x303c3994
     plink:  53   plinkp:  x303c6f34
     nxtact: 39   nxtactivep: x303c60c4
     command="/usr/pci/bin/pciconsvr.ip -D0 -L0 -B"
     start="Fri Jan 6  09:27:33 1989"
```

Figure 6-8. Process of procp of U-Block

Remember that entries found in the **/dev** directory name all the system device-drivers and special files that can be related to devices most commonly available within your system.

2.6.15.7 Display the Network Message Log

You display the Network Message Log with the following command:

```
netlog
```

Note: The **netlog** command is a **crash** subcommand. This log is a history of recent TCF communication activity. Each entry in the log indicates whether the message was incoming or outgoing, its acknowledgement number, the network message address, the type of the message, the length of the message, the other TCF site, and the time the entry was logged. It also displays the contents of the major type dependent fields. An abbreviated example of the output is provided in Figure 6-9.

```
LAST 100 NETWORK MESSAGES
       ack#     Nmaddr  Len Type     Site     Long1  Short    Long2       Ti
        dec        hex  dec           dec       dec    dec      hex        d

Out    199 S  30532764  192 SNDIND    11          1      6  2000002  6003743
In     151+S  30532764 1216 RSNDIND   11          0      6        0  6003743
Out    151 S  00000000  192 NOP       11         11    144       97  6003743
Out     18 S  30532764  192 SNDIND    12          1      6        0  6003743
Out   1755    006421f1 1216 RREAD     25          0  11868       6c  6003743
In      12+S  30532764 1216 RSNDIND   12          0      6        1  6003743
Out     12 S  00000000  192 NOP       12         12    122        c  6003743
In      675   487e21f1  192 USLOCKF   25        171  11868       26  6003743
Out    1756   487e21f1  192 RUSLOCK   25          0  11868       26  6003743
Out     197 S 30532764  192 SNDIND    14          1      6        1  6003743
In      676   00000000 1216 WRITE     25        171  11868       6c  6003743
In      677+  00000000 1208 WRITE     25        171  11868       6d  6003743
Out     677   00000000  192 NOP       25         25    100      2a5  6003743
In      678   607d21f1  192 USCOMMI   25        171  11868       81  6003743
Out    1757   607d21f1  192 RUSCOMM   25          0  11868       81  6003743
```

Figure 6-9. Network Messages

For analyzing some types of TCF cluster communication problems it is useful to compare the the network message history logs from dumps performed at the same time or from logs taken from the running system. This analysis may show whether the network is losing messages or if a TCF site is overloaded with TCF cluster traffic.

2.6.15.8 Display a Network Message Buffer

The network message address field from the network message log in the previous example can be used to locate a particular network message in the pool. The command:

```
slot c25b98
```

produces:

```
netmsg:[58]+x0
```

indicating that network message buffer 58 is the slot at address 0xc25b98. This buffer can be displayed using the command:

```
netbuf 58
```

The output of this command is:

```
NETWORK MESSAGE BUFFERS (nnetmsg = 200.)
msg#     addr     flag      frw   svrseq      ack# chan rst fst ret      typ r
          hex      hex      hex      dec       dec  dec dec dec dec    ascii h
  58 30531b9c        0 305368ac     1354     52987   73  31  25   0    WRITE
```

Figure 6-10. Network Message Buffers

2.6.15.9 Display the Mount Table

The global mount table can be displayed using this command:

```
mount
```

The display includes an entry for every file system mounted somewhere within the TCF cluster. The entries for file systems that are mounted on the local TCF site include additional information, such as the mount point, the total number and the number of available inodes and blocks. If the file system is replicated, information on the other packs is provided. An example of output from this command is shown in Figure 6-11.

```
Global Mount Table

Address     GFS     Css    Gflag     Lflag   Oino#    Ogfs     Next

                  Local Mount
Address     GFS     Css    Gflag     Lflag   Oino#    Ogfs     Next
3050e768     32       1     x0000    x0008    5664       1     30516ffc
      Dev     Maj    Min    Buf        BUfp
     x001b    x0000  x001b  11900461           x303be414
     FNAME    PACK   BLOCKS INODES     BFREE   IFREE
     /usr/s   VOLXXX 17637  12824        350    5332

30516ffc     160      14    x0000     x0000   5098       1     3050e920

                  Local Mount
Address     GFS     Css    Gflag     Lflag   Oino#    Ogfs     Next
305e920      256      1    x0000     x0008   1684        1     0
      Dev     Maj    Min    Buf        Bufp
     x40041  x0004  x0041  11900632          x303c1180
     FNAME    PACK   BLOCKS INODES     BFREE   IFREE
     /u6      VOLXXX 18332  13328      1155    9064

                  Local Mount
Address     GFS     Css    Gflag     Lflag   Oino#    Ogfs     Next
3052e060     1        1    x0030     x0009     2         1     30517164
      Dev     Maj    Min    Buf        Bufp
      x0001   x0000  x0001  11900777           x303c3804
     FNAME    PACK   BLOCKS INODES     BFREE   IFREE
     /               28750  9984       2495    3950
     PLADDR          PACKN  SITE       FLAGS   FSTORE
     x300e30a0       30     30         20      e00
     x300e3020       29     29         20      e00
     x300e3060       27     27         20      e00
     x300e31e0       26     26         20      e00
     x300e3050       25     25         20      1e00
     x300e3030       20     24         20      1e00
     x300e31c0       23     23         20      e00
     x300e31d0       22     22         20      e00
     x300e3130       31     20         20      e00
     x300e30f0       19     19         20      1e00
     x300e31a0       17     17         20      e00
     x300e30e0       16     16         20      e00
     x300e3200       15     15         a0      ffffffff
     x300e3070       14     14         a0      ffffffff
     x300e31b0       12     12         20      1e00
     x300e3220       11     11         a0      ffffffff
```

```
      x300e3120        8      8        20       e00
      x300e3170        7      7        20       e00
      x300e3210        4      4        20       e00
      x300e3180        3      3        20       e00
      x300e3100        1      1        30       ffffffff

 30517164    257      16    x0000     x0000   8842      1        30517194
 30517194    289       7    x0000     x0000    130      7               0
                  Local Mount
 Address     GFS     Css    Gflag     Lflag   Oino#    Ogfs     Next
 3050e25c     2        1    x0000     x0008   5095      1        3050e08c
       Dev      Maj    Min    Buf         Bufp
       x0003    x0000  x0003  11900752            x303c3160
       FNAME    PACK   BLOCKS INODES      BFREE   IFREE
       /fafni          5758   1920        2012    864

 3050e08c    258        8    x0020     x0000   792       1        3050e978

                  Local Mount
 Address     GFS     Css    Gflag     Lflag   Oino#    Ogfs     Next
 3050e978    290       1    x0000     x0008   395       1        0
       DEV      MAJ    MIN    BUF         BUFP
       x40049   x004   x0049  11900711            x303c267c
       FNAME    PACK   BLOCKS INODES      BFREE   IFREE
       /mnt/d   VOLXXX 124460 65496       67432   46032
```

Figure 6-11. Mount Table

2.6.15.10 Display the Site Table

The system site table can be displayed using this command:

```
site
```

This table contains information about each TCF site and statistics regarding the TCF cluster communication with that site. Figure 6-12 shows the output of the site command for one site. Similar output is produced for all TCF sites.

```
SITE LIST
Abbreviations:  xhd/xtl, ahd/atl = transmit/ack queue head/tail
win = ack window size, ret = retransmits, tim = timeout count
Uppercase names refer to special messages, lowercase to normal
All network, site & netmsg pointers are converted into indices

SITE NUMBER  3: status: UP channel: OPEN  flags: CHECKSUM,  conn: 0d 4h 21m
    PARMS   bsize   net  rout  win  WIN  ret/max  tim/max  TIM/MAX  ackw/max
             1024     0    3    12    2   0/3       0/4      0/4      2/0

    QUEUES   site    xhd  xtl   ahd  atl  xhd xtl  ahd atl   rsv
              -1     -1    -1   -1   -1    -1  -1   -1  -1   305398b4
    ACK/CHN  my next  he   ackd  hislast   MY NEXT HE ACKD  HISLAST  chan#
                 474        473      374      35        34       28   158
    STATS   pkts-in   pktsout    dups-in   retrans  ackblks  ACKBLKS  opens
             210607    226008      2609      2300     5253      371    6
    ERRORS  no msg    no buf     bad crc   bad site bad seq  bad msg  xmtfail
                 0         0           0          0    1053        0   0

   SUMMARY OF MESSAGE TRAFFIC

     message class       receive      transmt      avg time
     misc                  14168        23503         143
     nops                  28823        16215           0
     read                  16982        11725          80
     writes                14532         4499         117
     open, stat            58000        11992          98
     rcd ops                  26          999         126
     synch                     0            0         ---
     proc crt/dst            158          173         129
     proc sig/sts            123          392         204
     cls/cmt/trunc         31518        10759         756
     read rsp              38612        95307         153
     other rsp             14002        50207         173
     css-ss                  574           62          74
     topchg/update            97         1151        1487
     netwrk maint           2177         2185          92
     user-1                    0            0         ---
     user-2                    0            0         ---
     user-3                    0            0         ---
     user-4                    0            0         ---
     user-5                    0            0         ---
     UNTIMABLE MSGS                        16
     COMBINED TOTALS      219792       229185
```

Figure 6-12. Site List

2.6.15.11 Display the Topology Change Variables

Information about the current topology can be displayed using this command:

```
topology
```

This information can be useful when TCF cluster communication and/or topology problems are observed.

```
TOPOLOGY STATUS:
top_stat                     :       8         (stable)
actsite          (dec)       :       0
pollsite         (dec)       :       0
inpart           (dec)       :       0
partset          (hex)       :       37edeccd
newpartset       (hex)       :       37edeccd
newtop           (dec)       :       0
```

Figure 6-13. Network Topology Status

2.6.15.12 Display the Server Process Table

Server processes are special, system-created processes that are used to support TCF cluster communication. The server process table contains information about each server process currently allocated. Included in the listing is the process ID, the run state of the process, and the type of TCF cluster communication services being provided. The server process table is displayed using this command:

```
sptab
```

Sample output is shown in Figure 6-14.

```
SPTAB                                      SERVER STATUS
PID   SEQFUNC      SEQNUM   Stat          Type Prev Next
dec       hex         dec                       pid  pid

Active Server Processes

  7         0         816  BUSY          NAMEI   0    0

Free Server Processes

  10        0       12874  IDLE    USCLOSENC    7    9
   9        0         797  IDLE    USCLOSENC   10    4
   4        0       12119  IDLE        PROBE    9    8
   8        0        2327  IDLE    SSCLOSENC   10    6
   6        0        2342  IDLE    SSCLOSENC    8    5
   5        0        8659  IDLE    USCLOSENC    8   13
  13        0         931  IDLE    USCLOSENC    7   11
  11        0       11406  IDLE    USCLOSENC   13   12
  12        0         850  IDLE   GETCOMLIST   10   14
  14        0         850  IDLE         OPEN   12    3
   3        0        1574  IDLE    SSCLOSENC    8   -1
```

Figure 6-14. Server Process Table

2.7 Chapter 7. Command and Utility Examples

Subtopics

2.7.1 Contents

2.7.2 About This Chapter

This chapter contains example sessions of the following utilities:

fsck File system checker

fsdb File system debugger

For a description of these commands, see *AIX Operating System Commands Reference* manual.

2.7.3 The fsck System Checker

Subtopics

2.7.3.1 System Checker Example

When a TCF site is first booted, the root and **<LOCAL>** file systems are checked automatically. When the operator brings the TCF site to the multi-user level with the **startup** command, those file systems that are normally used on that TCF site are checked and mounted automatically. When these checks fail in a way that cannot be corrected without operator intervention, or when a file system that is not normally used is being mounted, it is necessary to run **fsck** explicitly.

The following command checks the file system **/dev/root**, permitting the operator to correct (or not correct) errors:

```
fsck /dev/root
```

More than one file system may be specified on the command line. Figure 7-1 shows a possible response to that **fsck** command. Note that **fsck** found three files in error.

```
  FSCK: /dev/root
  FSCK: ** Checking /dev/root (MOUNTED FILE SYSTEM)
  FSCK: File system info:
gfsS1Kgfspack 1
  FSCK: ** Phase 1 - Check Blocks and Sizes
  FSCK: missing commit indoe=138 commit count=28237 (enter?)
  OPER  y
  FSCK: missing commit indoe=726 commit count=28237 (enter?)
  OPER  y
  FSCK: ** Phase 2 - Check Pathnames
  FSCK: ** Phase 3 - Check Connectivity
  FSCK: ** Phase 4 - Check Reference Counts
  FSCK: ** Phase 5 - Check Free List
  FSCK: 1728 files 9845 blocks 11348 free
  FSCK  FILE SYSTEM NOT MARKED AS CLEAN ..FIX?
  OPER  y
  FSCK
  FSCK  ***** FILE SYSTEM WAS MODIFIED *****
```

Figure 7-1. Example: fsck Command Output

Note in the above example, the file system was the replicated root file system and hence was considered mounted. In general it is better to check file systems when they are not mounted.

In the above example, the operator chose to correct all errors. See "fsck Error Conditions" in this chapter for a description of the possible errors that can be found and corrected with **fsck**.

When checking replicated file systems, the operator must exercise caution. If a non-primary copy of the file system is repaired, it may be necessary (such as, in the case where a file was removed) to cause the propagation of certain files from the primary site. This is most easily accomplished by determining the files involved ("Find Files in Error" in topic 2.7.3.2) and forcing them to be updated using the **touch** command. If a primary copy is modified, propagations should occur automatically.

2.7.3.2 Find Files in Error

Use the following command to find the files using the inode numbers.

```
ff -i 138,726 /dev/root
```

Figure 7-2 shows the output of that **ff** command.

```
ff: /dev/root: 2 files selected
./devl/ph/trashd2 138
./devl/ph/panic1 726
```

Figure 7-2. Example: ff Command Output

This information may then be used to help determine how to repair the file system.

2.7.3.3 Verify File System is Correct

Enter the following command to check the file system again after it has been corrected.

```
fsck /dev/root
```

Figure 7-3 shows the screen output of the **fsck** command.

```
  FSCK: /dev/root
  FSCK: ** Checking /dev/diskuxrt (MOUNTED FILE SYSTEM)
  FSCK: File system info:
gfsS1Kgfspack 1
  FSCK: ** Phase 1 - Check Blocks and Sizes
  FSCK: ** Phase 2 - Check Pathnames
  FSCK: ** Phase 3 - Check Connectivity
  FSCK: ** Phase 4 - Check Reference Counts
  FSCK: ** Phase 5 - Check Free List
  FSCK: 1728 files 9845 blocks 11348 free
```

Figure 7-3. Example: fsck Command Output

2.7.3.4 fsck Error Conditions

The **fsck** command is a multi-pass file system checker. Each pass over the file system invokes a different phase of the **fsck** program. After the initial setup, **fsck** performs successive passes over the file system, checking blocks and sizes, pathnames, connectivity, reference counts, the free block list (possibly rebuilding it), and performing some cleanup. **fsck** runs all phases of the checks on the primary packs of replicated file systems and on all non-replicated file systems. Some of these phases are skipped for the non-primary packs of replicated file systems, because these packs may not store copies of all the files, and could appear in error in those phases of **fsck**.

When an inconsistency is detected, **fsck** reports the error condition to the operator. If a response is required, **fsck** prints a prompt message and waits for a response. With one exception (where a file name is required), the permissible responses are **y** for yes or **n** for no. Both uppercase and lowercase responses are valid.

There are two general types of decisions expected from the operator. In some cases, **fsck** requests permission to take a corrective action; the operator is asked to concur with the recommended corrective action. In other cases, the operator must decide whether to continue or abandon the checking of a file system. Often, most of the errors are related to a single corrupted file. Allowing **fsck** to proceed with its checks and destroying that file may solve the problem. Other times, the errors may be symptomatic of much deeper problems, and repair will be deemed hopeless. When this becomes apparent, abandoning the check may be the best course of action.

The **fsck** command messages are documented in *AIX Operating System Messages*.

2.7.4 File System Debugger

fsdb can be used to patch up a damaged file system after a crash, or examine an existing file system.

fsdb contains several error checking routines to verify inode and block addresses.

Numbers are considered decimal by default. Octal numbers must be prefixed with a zero.

The output facilities generate a formatted output in various styles.

Note: **fsdb** does not return a prompt!

2.7.5 fsdb Example Session

Subtopics

2.7.5.1 Find the Inode Number of a File

Enter the following command at the AIX/370 console to find the inode number of the file that you want to debug. For this example, consider that a previous file check showed file PANIC1 in the file system **/dev/root** in error.

```
ls -i PANIC1
```

Figure 7-4 shows the screen output of the **ls** command.

```
1552 PANIC1
```

Figure 7-4. Example: ls Command Output

The returned inode number is 1552.

2.7.5.2 Start fsdb Execution

Enter the following command at the system console to start **fsdb**:

```
fsdb /dev/root
```

Figure 7-5 shows the screen output which indicates statistics showing that **fsdb** is activated and waiting for input. Included in the display is an identification of the TCF site's machine type, the total number of blocks in the file system, the total number of inodes, the global file system number, and the global file system pack number.

```
IBM370 UNIX T/S assembly
FSIZE = 23275, ISIZE = 11984
GFS = 1, GFSPACK = 1
```

Figure 7-5. Example: fsdb Command Output

2.7.5.3 Find the Block Address

Enter the inode followed by an **i** as shown below to find the block for file PANIC1.

```
  1552i
```

Figure 7-6 shows the screen output which gives the address of block 0 (entry a0). 13153 is the address of block 0.

```
i#:  1552  md: f---rw-r--r--  ln:  1  uid:  0  gid:  0  sz:  2596864
qid:   19  dflag:0x0012
  commitcnt:191758  fstore:  0000
a0: 13153 a1:13154  a2:13155  a3:13156  a4:13157  a5:13158  a6:13159
a7: 13160  a8: 13161  a9: 13162  a10: 13163  a11:     0  a12:     0
at: Thu Jan 23 14:29:04 1986
mt: Thu Jan 23 14:15:04 1986
ct: Thu Jan 23 14:15:04 1986
```

Figure 7-6. Example: Information Block

2.7.5.4 Display the Block in Octal

Enter the following command to display the block in octal notation as shown in Figure 7-7:

```
13153b.p0o
```

```
315410000: 000000  000000  000000  000000  000000  000000  000000  000000

315410120: 072162  060560  027143  020055  020144  060564  060563  062564
315410140: 020160  060547  064556  063440  064556  072000  000000  000000

315410240: 000000  055434  017066  006314  044530  031467  030040  020040
315410260: 000151  074063  033460  000000  000000  032456  030056  030400
315410300: 000000  000062  032123  062560  034065  000000  051457  031467
315410320: 030000  000000  000000  000000  000000  000006  000000  000000
315410340: 000000  000001  000206  175274  000000  000000  000000  000000

315417760: 000000  000000  000000  000000  000000  000000  000000  000000
```

Figure 7-7. Example: Display Block in Octal Notation

2.7.5.5 Display the Block in Decimal

Enter the following command to display the block 13153 in decimal notation as shown in Figure 7-8:

```
13153b.p0e
```

```
315410000:00000   00000   00000   00000   00000   00000   00000   00000

315410120:29810   24944   11875    8237    8292   24948   24947   25972
315410140: 8304   24935   26990   26400   26990   29696       0       0

315410240:    0   23324    7734    3276   18776   13111   12320    8224
315410260:  105   30771   14128       0       0   13614   12334   12544
315410300:    0      50   13395   25968   14389       0   21295   13111
315410320:12288       0       0       0       0       6       0       0
315410340:    0       1     134   -1348       0       0       0       0

315417760:    0       0       0       0       0       0       0       0
```

Figure 7-8. Example: Display Block in Decimal Notation

2.7.5.6 Display the Block in Character

Enter the following command to display the block 13153 in character notation as shown in Figure 7-9:

```
13153b.p0c
```

```
315410000:  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
315410020:  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
315410040:  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
315410060:  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
315410100:  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
315410120:  t  r  a  p  .  c     -     d  a  t  a  s  e  t
315410140:     p  a  g  i  n  g     i  n  t  0  0  0  0  0
315410160:  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
315410200:  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
315410220:  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
315410240:  0  0     ?  ?  6  ?  ?  I  X  3  7  0
315410260:  0  i  x  3  7  0  0  0  0  0  5  .  0  .  1  0
315410300:  0  0  0  2  4  S  e  p  8  5  0  0  S  /  3  7
315410320:  0  0  0  0  0  0  0  0  0  0  0  ?  0  0  0  0
315410340:  0  0  0  ?  0  ?  ?  ?  0  0  0  0  0  0  0  0
315417760:  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
```

Figure 7-9. Example: Display Block in Character

2.7.5.7 Display Specific Bytes

This is the command syntax to display a word in a block:

```
block-number b + displacement . p number-of-words o
```

Suppose the block displayed in octal format looks like this:

```
               ----- displacement ---->
               0       1       2       3
315410000:     000000  000000  000000  000000  000000  .......
                                      ¦-word-¦¦-word-¦
```

Figure 7-10. Position in the Block

Enter the following command to display two words at displacement 3 in block 13153.

```
13153b+3.p2o
```

The output of that command is shown in Figure 7-11.

```
315410006:  000000  000000
```

Figure 7-11. Example: Display Specific Words in Block

2.7.5.8 Modify Block

This is the syntax of the command to change a word in a block.

```
  block-number b + displacement . W = new-bytes
```

Enter the following command to patch the word at displacement 3 in block 13153 to 000044.

```
  13153b+3.W=000044
```

A redisplay using the command 13153b+3.p2o would display the changed word as shown below.

```
  315410006:  000044  000000
```

Figure 7-12. Example: Display Changed Word

Now the block would look like this:

```
              ----- displacement ---->
              0       1       2       3
  315410000:  000000  000000  000000  000044  000000  .......
                                     ¦-word-¦¦-word-¦
```

Figure 7-13. Position in the Block

2.7.5.9 End fsdb Processing

Leave **fsdb** processing with the **quit** command.

```
q
```

2.8 Chapter 8. SFT: Software Record

The SFT record provides information for the debugging of detected AIX/370 system problems.

The software record contains information like the error code, the task ID, the error registers, PSW at time of error, and the interrupt code. The SYSERR message, which appears on the screen, is also a part of the SFT record.

Subtopics

2.8.1 Software Error Record

AIX/370 will write SFT EREP records to the VM ERDS. These records will be passed to VM via SVC 76. AIX/370 will prepare the error SFT record in MVS like format. The CP EREP program is used to format and print this record. The following entries in the EREP printout contain AIX/370 information:

ABENDING PROGRAM NAME
Reflects the TSI name.

NAME OF MODULE INVOLVED
Contains the module name.

REGS AT TIME OF ERROR
Contains the General Purpose Registers.

EC PSW AT TIME OF ABEND
Contains the Program Status Word at the time of the error

INTERRUPT CODE
Contains the interrupt code.

A.0 Appendix A. Performing Kernel Dumps on the AIX PS/2

This appendix describes two procedures for dumping the AIX kernel on the PS/2. The first procedure should be used if you are dumping the kernel to diskettes. The second should be used if you have a dump partition set up on your hard disk.

Subtopics

A.1 Dumping to Diskette

If the system panics, the following message is displayed on the console:

```
PANIC xxxxxx
Kernel coredump: you will need (n) formatted diskettes.
Please insert diskette #1 and press any key (or ESC to abort)...
```

The *xxxxxx* represents a message describing the nature of the particular panic, and (*n*) is the number of diskettes that you will need. The number depends on the size of your machine's memory. It is suggested that you have one preformatted diskette for each megabyte of memory.

After you have the indicated number of formatted diskettes for taking the kernel dump, use the following procedure:

1. Insert the first diskette into the microdiskette drive. Press **Enter**. (When the diskette is full a prompt appears.)

2. Remove the diskette from the drive and insert the second diskette. Press **Enter**.

3. Repeat this procedure for all subsequent diskettes that are requested by the system.

4. When all diskettes have been written, the system displays a

   ```
   Dump complete
   ```

   message. The system pauses 30 seconds, then reboots.

For more information about dumping to diskette, refer to the discussion of maintaining the AIX operating system in Chapter 3 of *Managing the AIX Operating System*.

This concludes the procedure for taking kernel dumps on the PS/2. When the diskettes containing the kernel dump information have been appropriately labeled, follow the appropriate diagnostic procedure to determine the source of the problem.

A.2 Dumping to a Partition on a Fixed Disk

If the system panics, a message is displayed on the console. This message may appear like the following:

```
PANIC xxxxxx
ehddump: dumping system to hard disk...
```

where *xxxxxx* represents a message describing the nature of the particular panic.

When the dump is finished, the system displays the following message:

```
System dump complete
```

After a 30-second pause, the system reboots.

B.0 Appendix B. Kernel Debugging Tools

This appendix describes AIX kernel debugging tools, for AIX/370 and AIX PS/2.

Subtopics

B.1 Snoop System/370 Debugger

The Snoop program is loaded as a co-resident object by the bootstrap and runs as a cooperative but independent routine of the kernel. It operates like a disabled interrupt routine, doing synchronous I/O to the console and hardcopy printer. Because the console is a 3270 display in full-screen mode, the console and printer are isolated, and the system is enabled for interrupts from these devices only. In System/370 XA mode, the console and printer use a different I/O subclass from all other devices. In System/370 mode, they are placed alone on a channel determined by the system and based on the number of channels configured.

Any 3270 display with up to 43 lines and an 80-character width is supported. Since the SNOOP and AIX consoles are not 3215s, spooling the VM/CP console captures only the data displayed on the console while in VM/CP mode. The mode name (Snoop, VM/CP, or AIX) appears in the lower-right corner of the display.

VM/CP is entered with a blank screen. Upon entry, Snoop and AIX reformat the screen and redisplay the screen as it appeared when the screen mode changed. The single-step Snoop command, **SS**, is the only exception to this rule. The single-step command switches into AIX mode for the execution of a single instruction and returns to Snoop. The console remains in Snoop mode for the duration of the single-step command.

The Snoop console screen has the following format elements:

Right half of last line: status area (protected).

Next-to-last line plus left half of last line: input area. The first byte of next-to-last line contains an attribute byte. Use the tab keys to move the cursor to the first byte of the input area.

Top n-2 lines: output area (protected).

Status area: contains the following indicators:

+HC Indicates that each line displayed on the SNOOP console is also written to the printer. The Snoop print device is the same as the AIX print device. Snoop and AIX hardcopy output are kept in chronological order.

READ or MORE Indicates whether Snoop is waiting for another command (READ) or is working on the previous command but cannot display the output because the screen is full (MORE). Snoop writes the screen only when the screen is full or before waiting for the next command. When MORE is displayed, press **PF8** or **PF11** to provide space on the screen for more output. Press **PF12** to flush the remaining output and wait for the next command.

nnnn Indicates how many lines are between the last output line displayed and the last line currently displayed. In general, this number indicates how far the screen has been scrolled backward.

Snoop Indicates that the console is in Snoop mode.

When Snoop is active, the keys function as follows:

CLEAR	Refreshes the screen. CLEAR is simulated whenever the console switches from VM/CP mode to Snoop or AIX mode.
PA1	CP (the **Break** key to VM/CP).
PA2	Scrolls forward to end. Places the last line in the middle of the screen.
PA3	Not used.
PF1	Help key. Displays a description of how the keys function.
PF2	Toggles the hardcopy switch on and off. When the hardcopy switch is on, **+HC** is displayed in the console status area.
PF3	**Print Screen** key. Prints the contents of the Snoop screen on the hardcopy log.
PF4	Displays the AIX console screen as it appeared when it switched from AIX mode to Snoop mode.
PF5	Retrieves the previously entered command and places it in the input area, simulating the operator keystroking; similar to the VM/CP retrieve command. The input can be modified before pressing the Enter key to input another command. Pressing **PF5** repeatedly retrieves up to 20 successively earlier commands. Pressing **PF5** the 21st time retrieves the most recent command.
PF6	Retrieves next command; similar to **PF5** but retrieves commands in the reverse order.
PF7	Scrolls backward one full screen. The number in the status area increments to indicate that older output is displayed.
PF8	Scrolls forward one full screen. Also used to clear the screen to allow more output to be displayed when **MORE** appears in the status area.
PF9	Scrolls backward one line. Used to align the output on the screen for **PF3** (**Print Screen** key).
PF10	Scrolls backward half a screen; similar to **PF7** except it scrolls only half the screen.
PF11	Scrolls forward a half screen; similar to **PF8** except it scrolls only half the screen.
PF12	Scrolls forward to end and suppresses output; similar to **PA2** except that it suppresses additional output from the currently executing command.
PF13 through PF24	Folded to have the same functions as **PF1** thru **PF12**.

Subtopics

B.1.1 Using Snoop

You can access the Snoop debugger after loading the kernel, or you can force entry. After loading the kernel, control is returned to Snoop in initial entry mode. You can execute the kernel by using the **GO** command or enter any Snoop command except **SS**.

You can force entry into Snoop using the **RESTART** function (CP command **SYSTEM RESTART**). **RESTART** forces entry even when the AIX kernel is in a disabled state. You can also press PF4 on the AIX console to enter Snoop. PF4 should not be pressed when a trace back is required since the current trace back facility cannot pass a trap frame on the stack.

Note: Pressing the **Break** key (**PA1**) allows you to issue any CP command when Snoop or the kernel has control. The screen clears and **CP READ** displays in the lower-right corner.

When program execution encounters a break, SNOOP receives control.

To exit to the kernel, type **go**. If you are finished debugging and want to resume normal operation, you should clear all breakpoints before issuing the **go** command.

To exit to CP, press **PA1**.

B.1.2 Snoop Commands

Snoop accepts command parameters defined in an expression containing identifiers, operators, and parentheses. The operators (**+**, **-**, *****, and **=**) take integer operands. (Note: The **=** operator can be used to save the computed value in a temporary storage location.) Identifiers are decimal strings, hex strings (followed by 0x), symbols in the symbol table, or symbols representing a temporary storage location. The search order for identifiers is: temporary storage table, symbol table, hex constant. To indicate an identifier is a symbol in the symbol table, place **#** immediately before the identifier. The period (**.**) is evaluated as the indirect operator and is used to run pointer chains. Expressions are delimited by one or more spaces.

SNOOP commands are listed below. The minimum command abbreviations are indicated by the capital letters. The lowercase letters represent optional characters.

?	Evaluate expression
.	Chase pointer chain
Begin	Synonym for **GO**
BR	Set break
CB	Clear break
DEf	Set default
DM	Display real memory
DP	Display PWS
DR	Display registers
DT	Display interrupt trace table entries
Dv	Display virtual memory
FS	Find symbol with real address
Go	Exit SNOOP and begin kernel execution
Help	Obtain help information
MM	Modify real memory
MP	Modify PWS
MR	Modify register
MV	Modify virtual memory
SS	Execute single instruction
STat	Display SNOOP status
Tb	Trace back to original kernel entry

TF Trace kernel stack forward

Un Unassemble instructions

VS Find symbol with virtual address

VTR Display real address from virtual (virtual to real)

WAit Put system in disabled wait state

When you are using Snoop, you can get help on any command by typing **HELP** and the command name.

Subtopics

B.1.2.1 SNOOP Command Summaries

The SNOOP commands are summarized below.

? [**-<option>**] **<expression>**

Evaluates <**expression**> and displays the result according to <**option**>. The option is **h** for hex or **d** for decimal. The default is decimal.

. [**-<option>**] **<expression>**

or

. [**-<option>**] **r<number>**

Chases down a pointer chain. The memory location indicated by the operand is used as a pointer, and the word the pointer points to is displayed. In the first format, the address of the pointer is given by <**expression**>. In the second format, the pointer is in the register (0 to 15) given by <**number**>. The option is **h** for hex or **d** for decimal. The default is hex.

Begin is a synonym of **Go**. See the description of **GO** on page B.1.2.1.

BR [**<expression>**] **<expression>**

Sets a break at the instruction whose address is given by the second expression. You can set up to 24 breaks. The first expression indicates which break is to be set. In typical usage, it is an integer between 0 and 23. Any previous settings of that break are lost. If the first expression is omitted, the first unused break is set.

CB [**-<option>**] **<expression>**

Clears the break indicated by <**expression**>, which must have a value of 0 to 23. In typical usage, the expression is an integer. If <**option**> is **a**, the expression is omitted, and all breaks are cleared.

DEf -<option>

Sets the default display mode according to <**option**>. The option is **h** for hex, **d** for decimal, **c** for character, or **n** for none.

DM <expression> [**<expression>**]

Displays memory starting at the address in bytes given by the first expression. The number of bytes displayed is the lowest multiple of 16 greater than the number indicated by the second expression. Sixteen bytes of memory are displayed per line in hex, followed by their character representation. There must be a blank in the command line before the first expression.

DP

Displays the program status word.

DT [**<expression-1>** [**<expression-2>**]]

Displays interrupt trace table entries starting with the entry indicated by the value of <**expression-1**> for the number of entries specified by <**expression-2**>. A value of 1 is assigned to the most recent entry.

Larger numbers represent older entries. The maximum number depends on the size of the trace table. If no <**expression-1**> is specified or it is specified as *, the current display continues from where the previous one ended. A **Go** command always resets the implied starting expression to 1. The default for <**expression2**>is 20.

Dv <expression> [**<expression>**]

Same as **DM**, except that the first expression is a virtual address.

DR [**<expression>**]

Displays the register indicated by the value of <**expression**>, an integer from 0 to 15. If no expression is given, all 16 registers are displayed.

FS <expression>

Locates the nearest symbol less than the address given by <**expression**>. If it refers to a symbol in the symbol table, the expression is treated as a virtual address; otherwise, it is treated as real.

Go [**<expression>**]

Causes the program to resume execution where it left off. The optional expression causes execution to halt at the point indicated by <**expression**>. This is similar to a break except it is cleared after use.

Help [**<option>**]

Displays help information on the Snoop console. The option is a Snoop command. The format and a short description of the specified command is displayed.

MM [**-<option>**] **<expression>**

Modifies real memory starting at the address given by <**expression**>. Sequential addresses are modified according to user input. No input (that is, a carriage return) exits. The option is **b** for modifying one byte or **w** for modifying a word. If the option is **b** and the input expression is larger than a byte, the least significant byte is used. The default is **w**. There must be a blank in the command line before the first expression.

Strings can also be used as the replacement expression. When strings are used, the option is ignored and the ASCII value of each character is put in consecutive memory. Strings should be put in double quotes (") to distinguish them from expressions to be evaluated.

A space followed by a carriage return causes the memory location to increment to the next location without changing memory. A **-** followed by a carriage return decrements to the previous memory location.

MP <expression> <expression>

Modifies the first or second word of the program status word. The first expression indicates which word (0 or 1) to modify. The designated word is modified to the value of the second expression.

MR <expression> <expression>

Modifies the register indicated by the first expression with the value of the second expression.

MV [**-<option>**] **<expression>**

Same as **MM**, except <**expression**> is a virtual address.

SS [**<expression>**]

Causes a single instruction of the program to be executed. The expression indicates the number of single steps to be executed. If no expression is used, it defaults to one step.

STat

Displays the status of the debugger by listing all breaks set.

Tb

Decodes the kernel stack back to the original entry into the kernel. Entry and return addresses are decoded into a format of external symbols plus displacement. Four parameters are displayed. If a routine has less than 4 parameters passed to it, the excess parameters are invalid.

Un <expression> [**<expression>**]

Unassembles one or more instructions located at the address specified by the first expression. The number of instructions to be unassembled is given by the second expression. The second expression defaults to 10. The first expression is treated as a virtual address if it refers to a symbol in the symbol table; otherwise, it defaults to real.

VS <expression>

This is the same as **FS** except it takes the value of <**expression**> as a virtual address.

VTR <expression>

Takes the value of <**expression>** as a virtual address, and displays the corresponding real address.

WAit

Puts the system into a disabled wait state. This command is used with the VM command **savesys** to save an image of the system for debugging with SNOOP. When the saved system is IPLed, the system is returned to the disabled wait state. Use the VM command **system restart** to return Snoop to a command prompt.

B.2 Kerndbg PS/2 Debugger

In order to debug new kernel code (such as device drivers) and diagnose problems within the kernel (or applications) a basic kernel debugger is provided. Before using, this debugger must be configured into a kernel. It is used to examine and modify the state of the software.

Note: For diagnosis tools see Chapter 5 of this guide, "Debugging Tools Overview".

Subtopics

B.2.1 Configuring the Kernel Debugger into a System

The inclusion of the kernel debugger in a system is controlled by the *kerndbg* parameter in the **sysparms** stanza of the system configuration file **/etc/system**. To include the kernel debugger in a system, add the line:

```
kerndbg = 1
```

to the **sysparms** stanza of your system file. To exclude the kernel debugger from a system, change that line to read:

```
kerndbg = 0
```

Once you have made the necessary changes to the system file, you can build a new kernel with the **newkernel** command. For more information, see the **newkernel** command in *AIX Operating System Commands Reference*.

B.2.2 Using the Kernel Debugger

After the basic kernel debugger is configured, there are three ways in which it can be invoked:

It can be manually invoked by pressing the key-combinatio **CTRL-ALT-NUM4** on the console keyboard.

It is automatically invoked when a previously set breakpoint i encountered, or after single-stepping. In this case, the debugger prints out the address of the breakpoint that was encountered.

It can be called explicitly within the kernel for debugging in th following way:

```
debugger((intA)0);
```

When the kernel debugger is invoked, it prints out a simple herald of **DEBUG** and prompt you for input with a minus sign. Once you have invoked the kernel debugger, all other system activity ceases. No system activity, other than processing and responding to your commands, takes place until you exit the kernel debugger.

To exit the kernel debugger and resume normal system activities, you can issue the **go** command to the debugger. Note that while you have the system stopped, you may lose data and connections. Incoming serial data may overrun and active network connections may time out.

When you are in the kernel debugger, all command lines should be terminated with a carriage return (the **Enter** key on the console). Input case is ignored. It is not necessary to type the entire name of a command. For instance, the contents of the registers can be displayed by typing either **registers** or just **reg**.

You can obtain a list of available commands by typing **help**. The acceptable abbreviation for each command is shown in upper case. You can obtain usage information on any of the available commands by typing the name of that command, followed by a question mark (?).

All of the commands that accept numeric arguments expect hexadecimal.

Any command that expects an address as an argument accepts any of the following:

A hexadecimal constant

An asterisk followed by a hex constant

A register name

A single quote

That particular address

Indirect that address

Indirect that register

The last address that was actually typed in.

Many of the commands can be modified by flag arguments. Such arguments are always preceded with a slash **/**, and can be specified in any order and in any field of the command line.

B.2.3 Command Descriptions

The commands can be divided into four groups:

Basic commands that examine and modify the state of the machin

Commands that aid in the debugging of kernel cod

Commands that display the state of the operating syste

Commands that perform special functions

B.2.4 Examining and Modifying Machine State

The following commands display and modify the contents of memory, the general registers, or the I/O ports:

Command Function

Dump Dumps the contents of kernel virtual memory.

This command displays (in hexadecimal) the contents of a specified area in the kernel's virtual address space. If no address is specified, the dump continues from the address at which the previous dump left off. By default, the dump command displays 128 bytes of memory.

The specified area is displayed as bytes, shorts or longs, depending on whether the **/b, /s** or **/l** flag is specified. In any case, the contents is also displayed as ASCII characters. An optional second argument specifies the number of lines to dump (each representing 16 bytes).

Enter Writes data into registers or memory.

This command writes data to a registers or into the kernel's virtual address space. If two values are specified, the first is taken to be an address and the second is the value to be stored. If no address is specified, the data is stored at the address following the previous store.

If the **/r** flag is specified, the destination address is treated as a register name, and 32 bits of data are stored into the specified register. Otherwise, the amount of data stored is 1, 2 or 4 bytes, according to whether the **/b, /s** or **l** flag is specified.

Registers Dumps out the contents of the registers.

This command prints out the contents of the general registers at the time of the interrupt or trap that caused the debugger to be entered. It also prints out the contents of the major debug and control registers if the **/l** flag is specified.

Inport Reads data from an input port.

This command reads a byte or a short from the specified input port. If no address is specified, the same port address used in the previous input command is used. The **/b** and **/s** flags are used to specify whether a byte or a short should be performed.

Outport Writes data to an output port.

This command writes a byte or a short to the specified output port. If two values are specified, it is assumed that the first is to be written. If one value is specified it is taken to be the value to be written, and the port number from the previous out command is used. The **/b** and **/s** flags control whether a byte or short should be performed.

HEXarith Hexadecimal addition and subtraction.

This command takes two hexadecimal arguments and prints out their sum and difference in both hexadecimal and decimal notation. It handles a surprisingly large portion of your hexadecimal arithmetic needs including hexadecimal to decimal conversion.

B.2.5 Debugging Kernel Code Commands

The following commands are used for stack tracing, single stepping, break points, or hexadecimal conversion:

Command Function

BAcktrace Prints out a kernel stack backtrace.

This command prints nested calls, parameters, local variables and saved registers from the time the kernel was entered until the point where the debugger was entered. Normally, this trace starts with the trap or interrupt that caused the debugger to be entered; however, if the **/f** flag (for a full backtrace) is specified, the backtrace starts from the current top of stack (in the **backtrace** routine).

Each frame is printed out individually, with the saved previous frame pointer and return pc first, followed by all of the parameters, locals and saved registers that were pushed into the previous frame before the call.

Go Resumes the interrupted execution.

This command causes you to exit from the kernel debugger. The system then resumes its normal activities at the point of interruption.

Trace Single steps.

This command can only be used if the debugger was entered as the result of a breakpoint or a previous single-step operation. It causes one more instruction to be executed and then re-enters the debugger.

BReakpoint Sets, clears and displays breakpoints.

If no arguments are specified, this command prints out a list of the currently set breakpoints.

If the **/d** flag is specified, the argument is interpreted as the number of the breakpoint to be disabled. Otherwise, the argument is taken as the address where a new breakpoint should be set. The type of the breakpoint is determined by the following flags:

/r Sets a hardware read breakpoint

/w Sets a hardware write breakpoint

/c Sets a hardware execution breakpoint

/s Sets a software execution breakpoint

/1 For the **/r, /w,** and **/e** flags, set the breakpoint length to one byte

/2 For the **/r, /w,** and **/e** flags, set the breakpoint length to two bytes

/4 For the **/r, /w,** and **/e** flags, set the breakpoint length to four bytes.

Hardware breakpoints are implemented using the breakpoint hardware on the PS/2. Software execution breakpoints are implemented by replacing the byte at the specified location with a breakpoint instruction.

B.2.6 Displaying Operating System Information

The following commands display important information regarding the state of the operating system:

Command	Function
Gdt	Prints the contents of the global descriptor table.
Ldt	Prints the contents of the local descriptor table. The **Gdt** and **Ldt** commands print the entire contents of the Global or Local descriptor tables--descriptor numbers, type, length, address and flags.
MEMfree	Displays information about the free memory pool.
Process	Prints information from the process table. If the **/a** flag is specified, only information on active processes is printed. This command can also take one or two numeric arguments. The first argument specifies which process table slot is to be printed. The second argument specifies the number of process table slots to be printed. If the second argument is not specified, all processes are printed.
PVseg	Prints out **procvseg** structures. With no arguments, **pvseg** prints the **procvseg** structure of the process in context. If an argument is given, it is used as an address of a specific **procvseg** structure to be displayed.
SWAPfree	Displays information about the available swap space.
Version	Prints system version information. This command prints a string indicating when and where the base system was built.
VSeg	Prints **vseg** structures. With no arguments, the **vseg** command prints out all of the **vseg** structures on the **vseg** busylist. If an argument is given, it is used as an address of a specific **vseg** to be displayed.

B.2.7 Special Functions

The following commands perform special functions:

Command Function

Help Displays a help menu for debugger commands.

REBoot Reboots the system.

This command prompts you about rebooting the system. If you respond with a **y**, the system halts immediately without flushing system buffers to disk (which would synchronize disk and buffer versions).

Note: To use a symbolic debugger for your AIX software, see **dpx** in the *AIX Operating System Commands Reference*.