

OS 2200 Client System Component
(CSC)

**4R1 Client Direct Interconnect
(CDI) Troubleshooting Guide**

July 2001

Copyright © 2001 by Storage Technology Corporation
All Rights Reserved

No part or portion of this document may be reproduced in any manner or any form without the written permission of Storage Technology Corporation.

The information in this document is confidential and proprietary to Storage Technology Corporation, and may be used only under the terms and conditions of a Storage Technology Corporation license agreement. The information in this document, including any associated software program may not be disclosed, disseminated, or distributed in any manner without the written consent of Storage Technology Corporation.

THIS DOCUMENT DOES NOT EXTEND OR CREATE WARRANTIES OF ANY NATURE, EITHER EXPRESSED OR IMPLIED. Storage Technology Corporation cannot accept any responsibility for use of the information in this document or of any associated software program. Storage Technology Corporation assumes no responsibility for any data corruption or erasure as a result of the information in this document, including any associated software program. You are responsible for backing up your data. You should be careful to ensure that use of the information complies with all applicable laws, rules, and regulations of the jurisdictions with respect to which it is used.

Unisys and EXEC are registered trademarks, as well as OS 2200 and 1100/2200, of Unisys Corporation. IBM is a registered trademark, as well as VM and MVS, of International Business Machines Corporation. Solaris is a registered trademark of Sun Microsystems, Inc. Ethernet is a registered trademark of Xerox Corporation. WolfCreek is a trademark, as well as StorageTek, of Storage Technology Corporation.

Information in this document is subject to change, and Storage Technology Corporation reserves the right to revise or modify the information in this document without prior notification. In the event of changes, this document will be revised. Contact Storage Technology Corporation to verify that you have the most current version of this document.

Comments concerning the information in this document should be directed to:

Storage Technology Corporation
2270 South 88th Street
Louisville, CO 80028-4232
(800) 678-4430

CONTENTS

PREFACE

PURPOSE.....	ix
AUDIENCE.....	ix
HOW TO USE THIS DOCUMENT	x
COMMAND SYNTAX NOTATION	xi
RELATED DOCUMENTATION	xi

1. INTRODUCTION TO CDI

ARCHITECTURE.....	1-2
OPERATIONS	1-4
STARTING CDI	1-4

2. STATUS COMMANDS

*CDI FS COMMAND	2-2
*CDI TCP CON	2-4
*CSC STATUS PATH	2-6

3. CDI TROUBLESHOOTING TOOLS AND PROCEDURES

CPATST UTILITY	3-2
Running CPATST	3-2
Setting CPA Streaming Mode	3-8
*CDI PING COMMAND.....	3-9

CHECKING YOUR HARDWARE CONFIGURATION	3-11
1. Verify that the CPA is Powered On and Online.....	3-11
2. Ensure that the CPA is Configured Properly.....	3-11
CHECKING YOUR OS 2200 CONFIGURATION	3-12
CHECKING YOUR CDI NETWORK ADDRESSING	3-12
CDI\$PARAM Configuration.....	3-12
Error Notification Displays	3-13
COMMON PROBLEMS.....	3-14
Group 1. Incorrect OS 2200 Configuration.	3-14
Group 2. CDI Aborts on Startup.....	3-14
Group 3. No successful TCP connections.	3-15

4. CDI TRACES

OVERVIEW	4-2
CDI TRACE FLAGS	4-3
TCPUI TRACE	4-5
TCP TRACE.....	4-7
SIGNAL TRACE	4-9
LAN I/O TRACE	4-9
LAN TRAFFIC TRACE	4-11
TRACE FILE EXAMPLE.....	4-11

APPENDIX A. ERROR CODES

COMMON CDI ERROR CODES	A-2
Main Errors.....	A-2
TCP Errors.....	A-3
Utilities, Internal Errors.....	A-5
IIACT PLS Internal Errors	A-5
Configuration Errors.....	A-5
SUMMARY OF INTERNAL ERROR MESSAGES	A-8

APPENDIX B. CHANNEL STATUS WORD FORMATS

CHANNEL STATUS WORD FORMATS	B-2
-----------------------------------	-----

INDEX

EFFECTIVE PAGES

TABLES

Table 2-1. *CDI FS Display Data 2-3

Table 2-2. *CDI TCP CON Display Columns 2-5

Table 2-3. *CSC STATUS Display Data 2-7

Table 3-1. Microcodes Checksums and Uses 3-7

Table 4-1. CDI Trace Flags 4-3

Table 4-2. Description of TCPUI trace output fields 4-5

Table 4-3. IP Datagrams and TCP Segments Trace Data Descriptions 4-7

Table 4-4. Description of SIGNAL trace output fields 4-8

Table A-1. CDI Internal Error Messages..... A-8

Table B-1. CSW/TSW Formats..... B-3

FIGURES

Figure 1-1. CDI Internal Interfaces	1-2
Figure 1-2. CSC/CDI Logical Layers.....	1-3
Figure 2-1. Sample *CDI FS Display.....	2-2
Figure 2-2. Sample *CDI TCP CON Display	2-4
Figure 2-3. Sample *CDI STATUS PATH Display.....	2-6
Figure 3-1. CPATST Main Test Menu.....	3-3
Figure 3-2. Display EEPROMs Parameters	3-3
Figure 3-3. Iteration Count Prompt	3-5
Figure 3-4. Wrap-Mode Prompt.....	3-5
Figure 3-5. Sample General Test Run Results	3-6
Figure 3-6. Sample Ethernet Traffic Run Result.....	3-7
Figure 3-7. Reminder Display for Tape/Disk EEPROM Programming.....	3-8
Figure 3-8. Set Streaming Mode Display	3-8
Figure 3-9. Streaming Mode Data Transfer Message.....	3-8
Figure 4-1. CDI Trace Points	4-2
Figure 4-2. TCPUI Trace Format	4-5
Figure 4-3. TCP Trace Format	4-7
Figure 4-4. SIGNAL Trace Format	4-9
Figure 4-5. LAN I/O Trace Format	4-10
Figure 4-6. LAN Traffic Trace Format	4-11
Figure B-1. Block Multiplexer CSW/TSW Formats	B-2

PREFACE

PURPOSE

This is the *OS 2200 Client System Component (CSC) Client Direct Interconnect (CDI) Troubleshooting Guide*. This guide describes commands and tools used to monitor CDI operations, and diagnose and solve CDI problems should they occur. CDI is the software component that provides TCP/IP services and connections to the Control Path Adaptors (CPAs). CSC is the software used by the Unisys 1100/2200 Client System (the “client”) to communicate with the Solaris[®]-based Library Control System (the “server”) and the Automated Cartridge System (ACS).

AUDIENCE

This guide is written for field, software support staff, and Unisys Customer Service Engineers (CSEs) responsible for configuring and supporting the CSC/CDI environment. It assumes that you are familiar with the following hardware and software components:

- ACS
- Unisys Series 1100 and 2200 computers
- Solaris-based Library Control System

You must also have a basic knowledge of networking using TCP/IP.

HOW TO USE THIS DOCUMENT

This guide contains four chapters, two appendices, and an index.

Chapter 1. Introduction to CDI

This chapter describes the Client Direct Interconnect (CDI) architecture and its operations. It also includes instructions for starting CDI.

Chapter 2. Status Commands

This chapter explains how to use various CDI commands for determining the state of the system and for displaying status information.

Chapter 3. CDI Troubleshooting Tools and Procedures

This chapter introduces the diagnostic tools and procedures available for troubleshooting CDI operations.

Chapter 4. CDI Traces

This chapter describes CSC and CDI trace points and how to use them in troubleshooting problems.

Back Matter

This manual includes an appendix listing CDI error codes, an appendix explaining channel status word formats, and an index.

COMMAND SYNTAX NOTATION

This manual uses the following conventions for representing command syntax notation and message displays:

UPPERCASE	indicates a command or keyword.
<i>lowercase italic</i>	indicates a user- or system-supplied variable value. For example, in <i>XX=userid</i> , you enter the actual userid for <i>userid</i> .
abbreviation	indicates a command that can be abbreviated to its minimum acceptable form. For example, ENAbles can be abbreviated to ENA.
vertical bar	separates operand alternatives. For example, A B indicates that you must select either A or B.
brackets []	indicate an option that can be omitted. For example, [A B C] indicates that you can select A, B, C, or nothing.
braces {}	indicate an option that you <i>must</i> choose. For example, {A B} indicates that you must choose either A or B.
<u>underlining</u>	indicates the system default. If you do not enter a parameter or value, the system will supply the underscored value. For example, A B <u>C</u> indicates that if you do not choose an option, the system will default to C.
ellipses ...	indicate that entries can be repeated as often as necessary.
SMALLCAPS	indicate a key, such as XMIT or F1.

RELATED DOCUMENTATION

OS 2200 Client System Component (CSC) Technical Bulletin, Unisys Corporation (7844 8693)

OS 2200 Client System Component (CSC) System Administrator's Guide, Unisys Corporation (7844 8685)

OS 2200 Client System Component (CSC) Installation Guide, Unisys Corporation (7844 8669)

OS 2200 Client System Component (CSC) Operations Guide, Unisys Corporation (7844 8644)

OS 2200 Client System Component User Interface (CSCUI) Programmer's Reference Manual, Unisys Corporation (7844 8677)

Automated Cartridge System Library Software Product Document Set for Solaris, Storage Technology Corporation

Exec System Software Operations Reference Manual, Unisys Corporation
(7831 0281)

Executive Control Language (ECL) End Use and Programming Reference Manual, Unisys Corporation (7830 7949)

COMUS Reference Manual, Unisys Corporation (7830 7758)

SOLAR Reference Manual, Unisys Corporation (7831 0604)

Executive Requests Programming Reference Manual, Unisys Corporation

System Processor and Storage Programming Reference Manual, Unisys Corporation

1. INTRODUCTION TO CDI

This chapter describes the Client Direct Interconnect (CDI) architecture and its operations. It also includes instructions for starting CDI.

ARCHITECTURE

Architecturally, CDI is the underlying communications layer for CSC. The communications protocol used by CDI is the standard TCP/IP suite.

In the Unisys 1100/2200 environment, CDI is a run-unit in OS 2200 that accesses the Control Path Adaptor (CPA) via the EXEC Arbitrary Device Handler. Because CDI is the bottom-most layer of CSC, it must be started first to set up the communications environment for the upper-layer components.

A request starts when CSC makes an inter-program call to the CDI common bank. Figure 1-1 shows the internal layers of CDI.

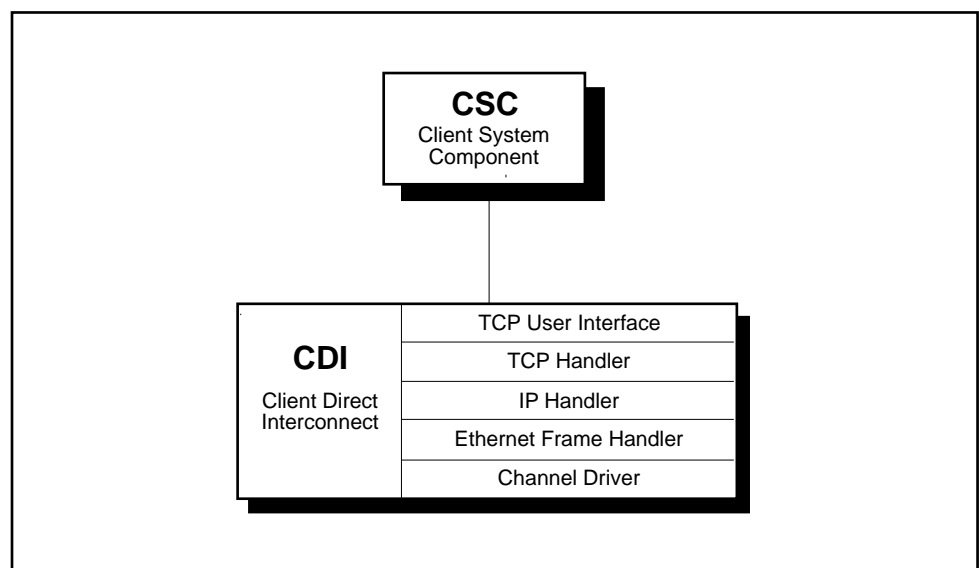


Figure 1-1. CDI Internal Interfaces

Figure 1-2 shows the logical layers of the CSC/CDI environment.

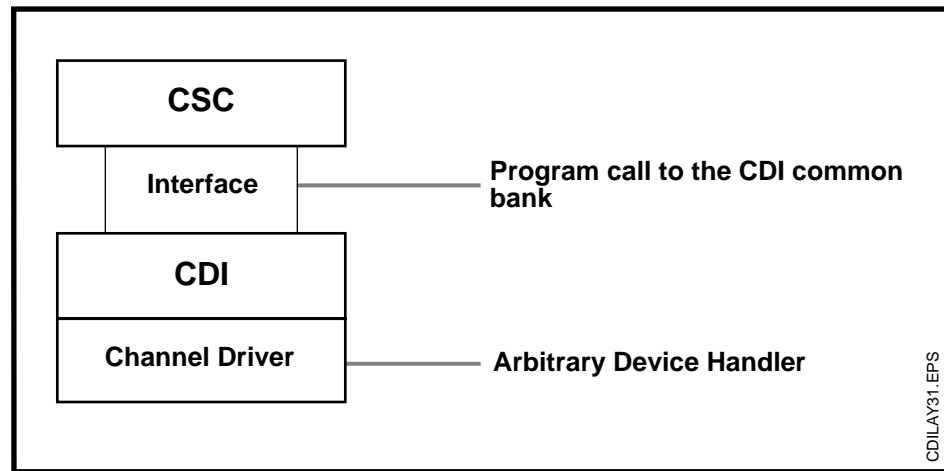


Figure 1-2. CSC/CDI Logical Layers

CSC uses a variant of Remote Procedure Call (RPC) to communicate with the server. This is referred to as “callback” RPC, because the server makes a call back to the requester to return the response. For example, when a mount request is submitted, the following events occur:

1. CSC establishes a connection to the server.
2. CSC sends the server request as an RPC.
3. CSC waits for the RPC to acknowledge that the request is received.
4. CSC closes the connection.
5. CSC issues a passive-open to wait for a server connection.
6. The server opens a connection to the client.
7. The server submits an RPC request that contains an intermediate acknowledgment from the server.
8. CSC sends an RPC acknowledgment for the server’s intermediate acknowledgment.
9. The server closes the connection.
10. CSC issues a passive-open to wait for a server connection.
11. The server opens a connection to the client.
12. The server submits an RPC request that contains a final response from the server.

13. CSC sends an RPC acknowledgment for the server's final response.

14. The server closes the connection.

The previous sequence of events presents the basic protocol used by CSC. The implementation, however, has been tuned for higher performance. There is a listening task always ready to accept the server's connect request. This is needed because Steps 11 and 12 can occur at any time after Step 8.

OPERATIONS

As you can see from the previous section, "Architecture," each request to the server uses several connections. You may be able to see the progression of these connections by issuing a *CDI TCP CON command. The *CSC QUEUES keyin shows the active and pending CSC requests. Whenever requests are active, there should be a succession of connections.

STARTING CDI

To start CDI, enter the following keyin at the system console:

```
ST CDI
```

Because CDI internally reloads its common banks on startup, you do *not* need to reload any common bank prior to starting CDI. For more information, please refer to the *CSC Operations Guide*.

2. STATUS COMMANDS

This chapter describes the following CDI and CSC console keyins used for determining the state of your system and displaying various statuses:

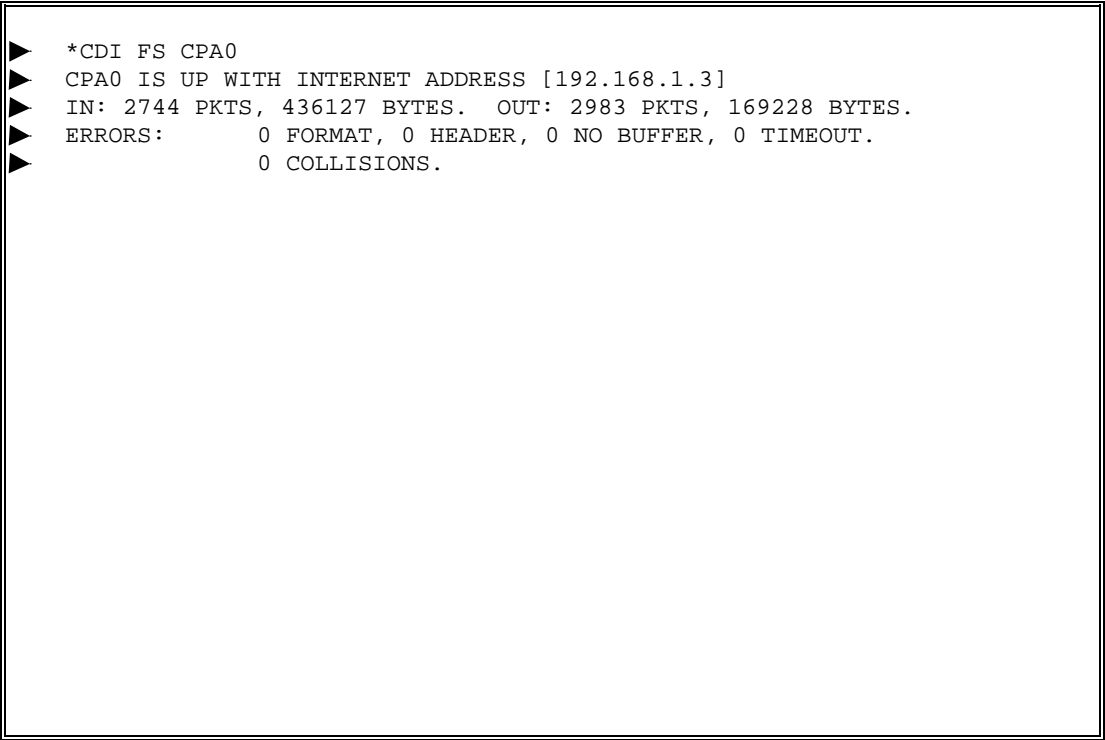
- ***CDI FS.** This command displays the current status of the specified network interface.
- ***CDI TCP CON.** This command displays all active TCP connections owned by a run.
- ***CSC STATUS PATH.** This command describes the status of each possible communication path to the server.

*CDI FS COMMAND

The *CDI FS command displays the current status of the specified network interface. You enter the command from CSC as follows:

```
*CDI FS interface
```

where *interface* is the name of the network interface whose status you want to display.

A screenshot of a terminal window showing the output of the *CDI FS CPA0 command. The output is displayed within a rectangular frame. On the left side of the frame, there are five line numbers (1 through 5) each followed by a right-pointing triangle. The text of the command and its output is aligned to the right of these markers.

```
1 ► *CDI FS CPA0
2 ► CPA0 IS UP WITH INTERNET ADDRESS [192.168.1.3]
3 ► IN: 2744 PKTS, 436127 BYTES. OUT: 2983 PKTS, 169228 BYTES.
4 ► ERRORS:          0 FORMAT, 0 HEADER, 0 NO BUFFER, 0 TIMEOUT.
5 ►                 0 COLLISIONS.
```

Figure 2-1. Sample *CDI FS Display

Table 2-1 explains the information contained in the display.

Table 2-1. *CDI FS Display Data

<i>Line</i>	Item	Description	
2	CPAn IS...	The state of the CPA interface. This field may be one of the following values:	
		Value	Description
		DN	The CPA has been DOWNed.
		UP	The CPA is UP.
		UP/DN IN PROG	The CPA UP or DN operation is in progress.
	[10.0.0.3]	Internet address assigned to the network interface.	
3	IN: <i>n</i> PKTS ...	The number of packets (PKTS) and bytes received on this interface. If this count doesn't change, there may not be any CPA activity.	
	OUT: <i>n</i> PKTS...	The number of packets (PKTS) and bytes transmitted on this interface. If this count doesn't change, there may not be any CPA activity.	
4	ERRORS: FORMAT	The number of error packets reported by the CPA. This includes the sum of frames with Frame Check Sequence (FCS) errors; frames that were too short; and other format errors.	
	ERRORS: HEADER	The number of frames not acceptable because of an invalid LAN header.	
	ERRORS: NO BUFFER	The number of packets discarded by the CPA because no buffers were available. This condition occurs when data packets aren't read from the CPA.	
	ERRORS: TIMEOUT	The number of timeouts when no data was received from the LAN within a specific time. This is not an actual error; there simply may not be any data to transfer from the server.	
5	COLLISIONS	The number of times the CPA started sending data at the same moment that an incoming data frame was starting. If the LAN is non-dedicated, collisions will result. These collisions are due to the disparate hardware environments (e.g., repeaters, various board types). If collisions run in the thousands per hour, it may signal a LAN error.	

*CDI TCP CON

The *CDI TCP CON command displays all active TCP connections. You enter the command from the console as follows:

```
*CDI TCP {CONNECTIONS | CON}
```

1	▶	*CDI TCP CON					
2	▶	CON ID	STATE	S-ADDRESS	S-PORT	D-PORT	D-ADDRESS
3	▶	1:104821	LISTEN	[0.0.0.0]	7000	⇒ 0	[0.0.0.0]
4	▶	2:104823	ESTAB	[192.168.1.3]	2125	⇒ 111	[192.168.1.1] RTXING

Figure 2-2. Sample *CDI TCP CON Display

In this display, there is a header row followed by one row of information per TCP connection. Table 2-2 describes the items reported for each connection.

Table 2-2. *CDI TCP CON Display Columns

Column	Description	
CON ID	An identifier that indicates the age and usage of the connection. This identifier is supplied by CSC. For responses from the server, this is the time (in <i>hhmmss</i> format) that the passive-open was issued for that connection. For requests to the server, this has the format <i>##Cmss</i> , where <i>##</i> is an internal command code, <i>C</i> is the letter 'C', and <i>mss</i> is the low-order portion of the time this connection was opened to send the request.	
STATE	Status of the connection, as follows:	
	Value	Description
	ESTAB	The establishment exchange has completed and the connection is open to transfer data.
	SYN-SENT SYN-RECV	Initial startup of a TCP/IP connection requires a three-way handshake. These states indicate the connection process has begun. SYN-SENT means CDI started the sequence. SYN-RECV means the server started it.
	LISTEN	CDI is waiting for a request to establish a connection.
	CLS-WAIT FINWAIT1	These states are entered when a request to close a connection is received (CLS-WAIT) or sent (FINWAIT1)
	CLOSING FINWAIT2 LAST-ACK TIMEWAIT	These states control the exchanges used to close a connection.
S-ADDRESS	Internet address of the CPA. The listen address 0.0.0.0 listens to all CPAs.	
S-PORT	Source port number.	
D-PORT	Destination port number on the server.	
D-ADDRESS	Internet address of the Ethernet adaptor on the server. The listen address 0.0.0.0 is for any server address.	
	If RXTING appears after the D-ADDRESS, then CDI is retransmitting a frame because it did not receive a server acknowledgement. If this appears often then it may indicate a communication problem.	

*CSC STATUS PATH

The *CSC STATUS command displays information about the internal status of CSC. This keyin is described in the *OS 22000 Client System Component (CSC) 4R1 Operations Guide*. When entered as *CSC STATUS PATH, CSC's output for this command includes its view of the state of communications with the server, including the number of active requests. The status of each path is not continuously updated and it can be up to one minute out of date.

In this display, lines 1 are typical output from the *CSC STATUS keyin. The "PATH" option produces one line, like line 2, for each configured CPA. The general form of this line is

PATH *cpaname* (via *intf-name:stat*) IS *status* AS OF *time* *n a/r*

```
1  ► *csc status path
1  ►   CSC 4R1 (3-5-5)
1  ►   CSC status as of Fri Jan 7 11:20:11 2000
1  ►     Active since Fri Jan 7 11:08:58 2000
1  ►     CSC is active
1  ►     Active tasks = 5
1  ►     Total tasks = 7
1  ►     Lock id = 7810
1  ►     Total mounts completed = 0
1  ►     Control path is ALPHA via CDI
2  ►     PATH ALPHA (via CDIINT:UP) is Active      as of 11:19:34  2 a/r
2  ►     PATH BRAVO (via CDIINT:UP) is Available as of 11:20:02
```

Figure 2-3. Sample *CSC STATUS PATH Display

Table 2-3. *CSC STATUS Display Data

Item	Description	
cpaname	The PATH name of the CPA.	
intf-name	The name of the interface this path is using. For CPAs, it must be the CDI interface.	
stat	The status of the interface: UP, DOWN, or unavailable.	
status	Value	Description
	DOWN	The CPA is down in CDI. CSC sets a CPA to this state when it notices that the CPA is down in CDI. CSC will not try to use a CPA that is in this state.
	UNTESTED	No communication with the server has been attempted using this CPA. CSC puts the CPA in this state when it notices that CDI has changed the CPA state from down to up. CSC periodically initiates a test of CPAs in the UNTESTED state.
	UNUSABLE	Communication with the server was tried and failed through this CPA. There is no delay between failure detection and setting the CPA to the UNUSABLE state. CSC periodically initiates a test of CPAs in the UNUSABLE state.
	AVAILABLE	Successful communication with the server has occurred through this CPA. It is available for use. CSC periodically initiates a test of CPAs in the AVAILABLE state to verify that they are actually available.
	ACTIVE	CSC uses the CPA in this state for sending requests to the server. In the absence of any server requests, CSC periodically initiates a test of the ACTIVE CPA to verify that communication with the server is possible.
time	The last time the path through this CPA was either tested by CSC or used for a server connection.	
n	The number of requests that are active through this PATH.	

3. CDI TROUBLESHOOTING TOOLS AND PROCEDURES

This chapter describes troubleshooting tools you can use to verify that CDI is functioning properly. It also describes ways to check your hardware and software configurations if you encounter problems with CDI. Finally, this chapter includes some background information and a discussion of common problems.

CPATST UTILITY

CPATST (CPA Test) is an interactive diagnostic tool that allows you to display EEPROM values, set channel types, download microcode to the CPA, and listen to Ethernet traffic on the LAN. Using CPATST requires knowledge of communication protocols and networking and should be reserved for your Unisys CSE and Network Administrator.

During the CSC product build, a file is created that contains all the elements used by CSC: the default filename is SYS\$LIB\$*CSC. When the CSC product group is installed, elements used by CPATST are loaded into this file. These elements include four CPA microcode elements and the ECL statements used to execute CPATST.

The ECL is built during the installation process, and uses the CDI configuration statements to determine the arbitrary device name assignments for your site.

Running CPATST

To run CPATST, first ensure that either CDI is down or the CPA is down in CDI. This is needed because CPATST obtains exclusive use of the CPA using arbitrary device assignment statements. If SYS\$LIB\$*CSC is the CSC filename you configured during the CSC build, then enter the following statement from a demand terminal:

```
@ADD SYS$LIB$*CSC.CPATST/ECL
```

If your site has multiple CPAs, then CPATST must know which CPA to use. It presents a menu containing all of the CPAs in your configuration. You then enter the number of the CPA to be used. In the following example, ALPHA and BRAVO will be replaced by your CPA names.

The following CPAs are defined in CDI\$PARAM

```
1 - ALPHA
```

```
2 - BRAVO
```

Enter the number for the desired CPA:

CPATST then displays a menu similar to the one in Figure 3-1.

```

CPATST Status : Initializing Channel Units
#####
CPATST Version 2R1A (2-1-4) Copyright 2000 StorageTek
      MAIN TEST MENU
CONTROL UNIT ADDRESS : 00  MODEL NUMBER : 03  CHECKSUM : 9FEB
1) Display EEPROMS Parameters.
2) General Tests.
3) Listen to ETHERNET traffic. (Not available in CETI mode)
4) Download ELC MODEL 1 (6.3 CHECKSUM - 0EFE) microcode.
4) Download ELC MODEL 1 (6.8 CHECKSUM - 9FEB) microcode.
4) Download ELC MODEL 2 (6.9 CHECKSUM - 9805) microcode.
4) Download ELC MODEL 2 (7.1 CHECKSUM - B407) microcode.
5) Set streaming mode only data transfer (default) .
6) Set non-streaming mode data transfer (400 type) .
E) Exit Program

ENTER CHOICE FROM ABOVE ?

```

Figure 3-1. CPATST Main Test Menu

Enter “E” to exit the CPATST main menu. Your other menu choices are described below:

❶ Display EEPROMs Parameters.

This choice displays parameters stored in EEPROM (Electrically Erasable Programmable Read Only Memory) in the CPA. These parameters are not affected by downloading new microcode. Figure 3-2 shows the screen used to display the parameters.

```

***** Display EEPROMs Parameters *****
Set Up Configuration Parameters for CPA:

Ethernet ID, CU Address, ATN+UEX Mode, Broadcast flag, Buffers

Ethernet ID is : 2:E6:D3:7:80:0
CU Address is (00-switch) : 00
ATN mode is (8x-inhibit,0x-normal) : 08
UEX mode is (x8-UEX,x0-normal) : 08
Reject Broadcast (x8-Reject, x0-Accept) : 00
EEPROM Type      : 0000
Streaming mode data transfer only is set (non-400 type)
Press Xmit to continue ...

```

Figure 3-2. Display EEPROMs Parameters

EEPROM parameters are described below:

Ethernet ID. A unique 48-bit Ethernet address assigned to the controller.

CU Address. Control Unit address entry is provided as an alternative to DIP-switch settings. If this is an all-zeros value the DIP switches dictate the control unit address. A non-zero value overrides the DIP switch setting.

ATN Mode. This defines how the CPA indicates that received data is available. When set, the CPA raises an attention interrupt to indicate packets that are received. This should be set for normal operation.

UEX Mode. This defines how the CPA responds to a read request when no received data is available. When set, UEX mode causes the CPA to present Unit Exception, rather than Unit Check. This should be set for normal operation.

Reject Broadcast. When set, the CPA rejects all inbound packets with a broadcast destination address.

EEPROM Type. EEPROM is set as type 0000.

Streaming Mode Data Transfer Only is set (non-400 type). This is the default setting.

② General Tests

This choice runs tests that verify the operability of the CPA hardware. These include general write/read/compare tests. Each test iteration performs the following steps:

1. Start prepare-read CCW (Channel Command Word) chain on subchannel 0.
2. Start prepare-read CCW chain on subchannel 2.
3. Start writes CCW chain on subchannel 1.
4. Wait for data to be transferred.
5. Compare all data for validity.
6. Exit.

You should consider running the General Tests if any of the following circumstances apply:

- When you started CDI, the Power On Confidence (POC) test was successful, but data transfer between the server and the client was not successful. The CDI print file shows I/O errors.

- You suspect that the channel addresses configured in OS 2200 are not the same as the CPA settings.
- You're running a 2200/600 platform with a 2200/400-type channel. Refer to the section, "Setting CPA Memory," below.
- You're running a 2200/400 platform. Refer to the section, "Setting CPA Memory," below.

After selecting Option 2 (General Tests) from the main menu, you are prompted to enter an iteration count and a wrap mode selection.

Iteration Count Prompt

After selecting Option 2 (General Tests) from the main menu, you are prompted to enter an iteration count. The iteration count prompt is shown in Figure 3-3.

```
Enter Iteration Count (0-indefinite)
```

Figure 3-3. Iteration Count Prompt

Enter the number of iterations you want to run. Any integer value between 0 and 999999 is valid. An iteration count of at least 50 provides a statistically acceptable sample size. An iteration count of 0 (zero) causes the test to run indefinitely.

Wrap-Mode Prompt

After entering an iteration count, you are prompted to enter a wrap-mode selection. This determines how far the data passes through the CPA before it wraps around to be compared. The wrap mode prompt is shown in Figure 3-4.

```
Iteration Count:  n
You may run wrap-mode in two ways  :
    1) Using Source E-net Addr = Dest E-net Addr
    2) Wrap around prior to entering E-net (use SLM Wrap mode)

Enter Your Choice:
```

Figure 3-4. Wrap-Mode Prompt

Choose option 1 to send test data to the LAN before returning it. Option 2 uses hardware within the CPA to return the test data. Each test iteration displays the data sent and the data received. The output from your tests will scroll on your screen.

NOTE

To stop your test output from scrolling, press the interrupt on your keyboard and enter @@X TIO Then restart CPATST from the beginning.

If you did not manually interrupt your test output, and the number of iterations you specified is reached, control automatically returns to the main menu. Sample general test results are shown in Figure 3-5.

```
Start General Test
  K200 Read Activity Started
Data Sent :
S-A2A3A4A5A6A7A8A9A0B1B2B3B4B5B6B7B8B9B0C1C2C3C4C5C6C7C8C9C0
Data Received :
R-A2A3A4A5A6A7A8A9A0B1B2B3B4B5B6B7B8B9B0C1C2C3C4C5C6C7C8C9C0
Data Sent :
S-A2A3A4A5A6A7A8A9A0B1B2B3B4B5B6B7B8B9B0C1C2C3C4C5C6C7C8C9C0
Data Received :
R-A2A3A4A5A6A7A8A9A0B1B2B3B4B5B6B7B8B9B0C1C2C3C4C5C6C7C8C9C0
Data Sent :
S-A2A3A4A5A6A7A8A9A0B1B2B3B4B5B6B7B8B9B0C1C2C3C4C5C6C7C8C9C0
Data Received :
R-A2A3A4A5A6A7A8A9A0B1B2B3B4B5B6B7B8B9B0C1C2C3C4C5C6C7C8C9C0
```

Figure 3-5. Sample General Test Run Results

③ Listen to ETHERNET traffic.

This choice listens to LAN traffic. For each packet on Ethernet, it displays the length, type, source, destination, and contents. Figure 3-6 shows a sample run result.

NOTE

To interrupt this display and return control to the main menu, press the ENTER key at any time. On occasion, the Start of Entry (SOE) character may not be present when the main menu is displayed. If this occurs, enter the SOE manually.

```

Enter S to stop listening
Len: 85  Type:00  Source:08 00 0B 00 02 00  Dest:00 00 C0 B9 1D 1B

HEX: 45 00 00 47 DF 5F 00 00 20 06 F5 E0 5F 00 02 01 5F 02 05 6E
HEX: 00 66 03 E9 01 9A 85 F5 00 07 BA 53 50 1B 3F 67 35 71 00 00
HEX: 03 00 00 1F 02 F0 80 8E 81 9D C0 0F 45 4E 54 45 52 20 54 4F
HEX: 43 20 4D 4F 44 45 2E 92 93 95 90 92 00 00 00 00 00 00 00 00
HEX: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

```

Figure 3-6. Sample Ethernet Traffic Run Result

④ Download ELC MODEL 1 (CHECKSUM - 0EFE) microcode from disk to EEPROMs.

Download ELC MODEL 1 (CHECKSUM - 9FEB) microcode from disk to EEPROMs.

Download ELC MODEL 2 (CHECKSUM - 9805) microcode from disk to EEPROMs

Download ELC MODEL 2 (CHECKSUM - B407) microcode from disk to EEPROMs

These choices download operational microcode to the CPS. Microcode is identified by a hexadecimal checksum. The following table relates microcode checksums to their uses.

Table 3-1. Microcodes Checksums and Uses

Checksum	Microcode Used for:
0EFE	CPA prior level
9FEB	CPA level 6.8
9805	CPA level 6.9
B407	CPA level 7.1

When selected, the program first ensures that the microcode file exists on the 2200. After this is confirmed, the checksum for the old microcode is read from the CPA's EEPROMs and compared to the checksum of the new microcode on disk.

It takes the CU approximately one minute to program the EEPROMs, during which time the red light on the unit blinks. The RESET button should *not* be pressed while the light is blinking.

Once the unit's red light stops blinking, you must press the RESET button because microcode execution is stopped after EEPROMs are programmed. A reminder is displayed on your screen, as shown in Figure 3-7.

```
Download Complete
Please allow about 1 minute for EEPROMs to be programmed.
    or until the RED error LED stops blinking.
Note: Control Unit must be reset before it can be
    brought online.
Begin Downloading Micro Code
```

Figure 3-7. Reminder Display for Tape/Disk EEPROM Programming

⑤ Set streaming mode only data transfer (default).

This choice sets the channel to streaming mode. The next section tells when to use this option. Streaming mode is the default data transfer mode. A message, as shown in Figure 3-8, is displayed.

```
Streaming mode data transfer only is set (non-400 type)
```

Figure 3-8. Set Streaming Mode Display

⑥ Set non-streaming mode data transfer (400 type).

This choice sets the channel to non-streaming mode. The next section tells when to use this option. This is for 2200/400 block multiplexer channels. A message, as shown in Figure 3-9, is displayed.

```
Allow non-streaming mode data transfer is set (400 type)
```

Figure 3-9. Streaming Mode Data Transfer Message

Setting CPA Streaming Mode

The 2200/400 channel type differs slightly from other channel types. It does not support non-streaming data transfers. This type of channel can also be found in some 2200/600 platforms. To allow non-streaming data transfers, the CPA memory location, X'3FFE3', must be set to X'20'. You can set and verify the correct value using a combination of menu options 2 and 8 from the main menu as follows:

1. Select option 2, "General Tests," to run a loopback test.
2. After entering your selection from the main menu, you are prompted for an iteration count:

```
Enter Iteration Count (0 - indefinite)
```

Enter an iteration count of at least 50 to provide a statistically acceptable sample size.
3. After entering the iteration count, the following message and prompt is displayed:

```
You may run wrap-mode in two ways:
  1) Using Source E-net Addr = Dest E-net Addr
  2) Wrap around prior to entering E-net (use SLM Wrap mode)
```

Enter your choice
Select option 2.

After entering your selection, the program starts the General Test. This test will cause a streaming overrun if the CPA setting is incompatible with the OS 2200 host setting. Refer to Appendix B in this guide for a brief explanation of the resulting statuses. Refer to the *System Processor and Storage Programming Reference Manual* from Unisys for a more complete explanation.

If the test is successful, no further tests are necessary. If the test fails, do the following:

1. Exit the current CPATST session.
2. Reset the control unit by pressing the ONLINE button at the front panel of the CPA. This takes the CPA offline; the ONLINE indicator light should turn off.
3. Press the RESET button. A series of LED values, beginning with ED, will display. When the LED display shows a value of 00, press the ONLINE button again to put the CPA online.
4. Run CPATST again. Select option 2, "General Tests," again, and repeat the loopback test described in Steps 1-3. If the test fails again, return to the main menu and select option 8, "Allow non-streaming mode data transfer."
5. Select option 2, "General Tests," again. You must reset the CPA externally, as described above in Step 5, each time between tests. Resetting forces the CPA to clear previous conditions (such as a status display of X'41' on the front panel), and allows the changes to take effect.
6. If none of the tests described above work, contact your Unisys CSE.

***CDI PING COMMAND**

The *CDI PING command lets you probe the server across the LAN to verify that it is reachable. The server must be identified in the CDI\$PARM element.

The server simply echoes the request with a timestamp. CDI will calculate the average response for the request.

If *CDI PING executes successfully, this verifies that both the hardware and software connectivity to the server are available and functional.

To PING the server, first verify that CDI is running. Next, enter the following command at the console:

```
*CDI PING Internet-address
```

where *Internet-address* is the server Internet address. For example:

```
*CDI PING 192.168.1.1
```

If *CDI PING responds with the message, “PING Timeout,” then the connection to the server is not available. Check the server Internet address configuration to verify that the addresses are correct.

If *CDI PING is successful, it responds with a message similar to:

```
PING RESPONSE TOOK .50 SECS, SEQ#12  
PING SUCCESSFUL.
```

The *SEQ#* displayed in the example message shows that there were a total of 12 PING commands issued since the start of CDI.

CHECKING YOUR HARDWARE CONFIGURATION

Use the following procedures to verify your hardware configuration:

1. Verify that the CPA is Powered On and Online

- 1.1. The LED status on the CPA should display two digits.
- 1.2. The online LED (normally green) should be on.

2. Ensure that the CPA is Configured Properly

- 2.1. Take the CPA offline by pressing the ON/OFFLINE button.
- 2.2. Press the RESET button.
- 2.3. During the reset process, the LED shows the internal configuration parameters.
- 2.4. Note the values of all of the LEDs, which follow the format:

ED 0E FE *xx xx yy* 00

where the values of *xx* are the last two digits of the internal Ethernet IDs, and *yy* is the configured control unit address. This is in the reverse order of the OS 2200 configuration subchannel address. Ignore the values for *xx xx*, but ensure that *yy* is correct.

For example, the value for *yy* should be 20 if the OS 2200 configuration of the subchannel address is 02, and 40 if the OS configuration of the subchannel address is 04. OS 2200 uses this subchannel address value to communicate with the CPA. If it is *not* correct, check the dip switch setting U4. Refer to the *Control Path Adaptor Technical Guide* for details on how to set the dip switches. Refer to the same document for further details if you want to do an online modification (using an ASCII terminal to reconfigure the EEPROM in the CPA).

- 2.5. Check all of the connections, including the transceiver cable connectors, and verify that they are securely connected.

CHECKING YOUR OS 2200 CONFIGURATION

To check the OS 2200 configuration, follow these steps:

1. Sign on to a demand terminal:
2. Enter:

```
@ASG,T T., *device
```

where **device* is one of the dev00,dev01,dev02 statements found in the CPA statement in CDI\$PARAM. Note that the asterisk (*) in front of the device name is required.

3. The OS 2200 configuration is correct if the assignment is successful or enters a facility hold state. The facility hold occurs when CDI is active and has exclusively obtained that facility.

CHECKING YOUR CDI NETWORK ADDRESSING

In the SYSS\$LIB\$*CSC-PARM.CDI\$PARAM element, you configure CDI by setting up parameters. CDI\$PARAM contains the definitions of the network connections to the server. Parameters in CDI\$PARAM specify the addresses of *all LAN nodes* (CPA or 802.3 physical connections). The following paragraphs describe the contents of CDI\$PARAM. Refer to the *CSC Installation Guide* for the exact syntax of these parameters.

CDI\$PARAM Configuration

When specifying parameters in SYSS\$LIB\$*CSC-PARM.CDI\$PARAM, the following rules apply:

1. Within a network, each CPA and Server LAN adaptor must have a unique Internet Protocol address. The address identifies the physical adaptor on the *network*, not the host.
2. In order for two adaptors on the same LAN to successfully communicate with each other, the first three parts of the two adaptor addresses must be equal (e.g., in the standard four-part Internet addressing scheme, 192.168.1 of addresses 192.168.1.1 and 192.168.1.3).
3. When specifying the addresses in CDI\$PARAM, the separator is a comma instead of the standard dot.

4. CDI\$PARAM must contain the following:

- the first statement in CDI\$PARAM must be CLIENT.
- the server requires at least one SERVER SGS.
- each CPA requires one CPA statement
- each CPA requires either a FORWARD ROUTE statement or a ROUTE statement.

The following is a sample CDI\$PARAM with one CPA and one SERVER:

```
CLIENT CPATMILO ''UNISYS-2200/423''  
CPA CPA0 ADDR 192,168,1,3 DEVICE CPA00,CPA01,CPA02  
SERVER IP ADDR 192,168,1,1  
ROUTE TO ADDR 192,168,1,1 VIA CPA0
```

Error Notification Displays

Following are descriptions of CDI error notification messages that display on the operator console. Each description includes problem-solving procedures for each error condition.

TCP CON # *msg* UNACKNOWLEDGED FOR # *min*.

This is a warning message that displays on the system console only when CDI Flag 7 is set (CDI flags are discussed in the next chapter). It reports that an acknowledgment to a TCP level message was not received within one minute. This condition is sensed by CSC as a timeout and CSC will resend the message. This is an information only message about a problem that is corrected automatically.

CPA# NOT RESPONDING, DEVICE DOWNED

Whenever CDI issues a channel write to the CPA, and receives a timeout instead of an I/O completion, this message will display on the console. This alerts the operator that the device is not responding to the request, and has been downed. This is a type and read message that the operator must answer. This condition is created by a failing CPA, or a when a DN command has been issued for a channel device to which the CPA is connected.

To recover, first correct the hardware condition that created the problem. Then enter the following console keyin:

```
*CDI UP cpa#
```

CPA# I/O ERROR, DEVICE DOWNED

CDI contains code to handle most conditions reported by the CPA. If the CPA reports a condition that CDI cannot resolve, then CDI displays this message and downs the CPA. This message points to some kind of CPA hardware error.

Before this message is displayed, CDI dumps information from the arbitrary device packet into the CDI print file. This includes hardware related information that further defines the failure source. Use the *CDI BRKPT keyin to gain access to the CDI print file. Then correct the indicated hardware problem. Finally, return the CPA to service using the following console keyin:

*CDI UP *cpa#*

COMMON PROBLEMS

Common problems can be classified into the following groups:

- Group 1. Incorrect OS 2200 configuration.
- Group 2. CDI aborts on startup.
- Group 3. No successful TCP connections.

Group 1. Incorrect OS 2200 Configuration.

The CPA configuration on the OS 2200 NODE statements must agree with the CPA hardware settings. Verify that the channel addresses match the CPA internal channel settings as described earlier in this section.

Group 2. CDI Aborts on Startup.

The most common problems related to CDI aborting on startup are installation failures or configuration errors. Each of these problems produces an identifying message. The following sections tell the source of these problems.

CDI\$PARAM Element Not Found.

The CDI\$PARAM element containing the configuration parameter for the client and server LAN environment cannot be found.

Initial: Route Installation Failure.

This problem causes CDI to take an EABT instead of a graceful exit. It is caused by a configuration error in the CDI\$PARAM element. In this case, a ROUTE or FORWARD ROUTE statement is incorrect.

You can only define routes for hosts that have been defined in either the CPA SGS or SERVER SGS in the CDI\$PARAM element. See the *CSC Installation Guide* for further discussions of the ROUTE and FORWARD ROUTE SGSs.

Group 3. No successful TCP connections.

Once CSC is started (normally with an ST CSC keyin), it tries to establish connections with the server. After issuing the ST CSC keyin, wait at least three minutes and enter *CSC STATUS. If CSC does not show an ACTIVE status, do the following:

1. Make certain that CDI was started before CSC.
2. Ensure that CDI is up and running with the *CDI STATUS command. The display for this command also shows the status of each configured CPA. Make certain that the CPA addresses displayed have the correct Internet addresses, and that at least one CPA is UP.
3. Enter the *CDI TCP CON command at the system console. The responses show the status of each logical TCP connection.

Under normal circumstances, there will be one connection in the listening state. There may be additional connections for requests that are in progress.

4. If your CDI configuration includes multiple CPAs, issue a *CSC ACTIVATE command to switch the control path to another CPA.
5. Don't forget to verify that the server is up and running. Check with the server administrator to make sure that the server Internet addresses are configured properly.

This group of problems usually relates to incorrect configuration parameters in the CDI\$PARAM element and server configuration.

4. CDI TRACES

This chapter describes the communication trace points in the CSC product. It also describes the console commands used to control the associated traces. Tracing is included to aid in problem and performance analysis.

OVERVIEW

CSC and CDI each transform one level of request into one or more lower level requests. They also combine and process multiple lower level responses into a single higher level response. CSC transforms requests such as MOUNT TAPE into one or more server requests. It then translates these server requests into a series of high level communication requests. CDI accepts these high level communication requests and translates them into the required communications protocol messages. CSC and CDI both contain facilities to monitor their respective internal processing. The major types of tracing are illustrated in Figure 4-1.

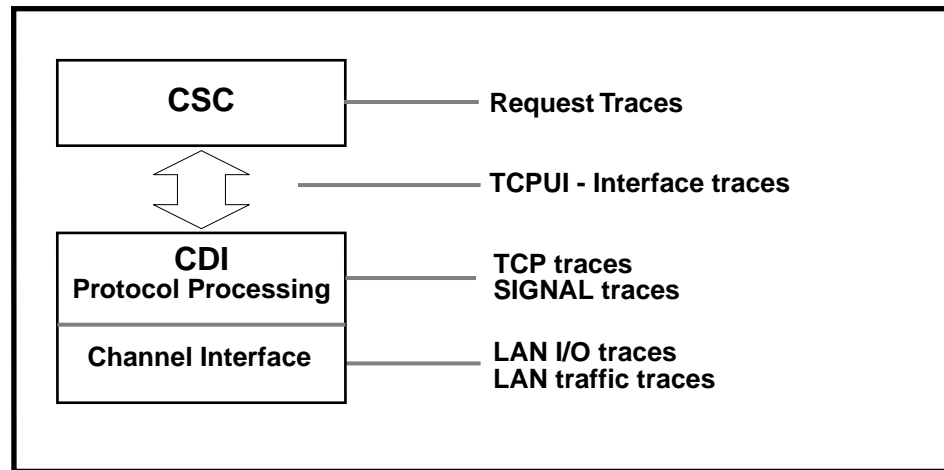


Figure 4-1. CDI Trace Points

CSC traces are controlled with the *CSC DEBUG keyin. The communication related traces in CSC consist of request completion reports. These are designed to analyze request processing. They are not sufficient to analyze communication events.

CDI tracing is provided for communication analysis. The following sections list the types of traces that exist in CDI and how they are controlled. Also included are general descriptions of the most useful types of traces. Finally, there is a sample of the CDI trace output. All of the figures in this chapter are taken from this output.

Some of the trace output from CDI is designed for line printer output. It uses up to 132 columns. The figures that accompany the general descriptions display long lines in one of the following ways.

- A single output line is shown as two lines with the second line right justified.
- The line is truncated. These lines end with "...". Since the sample trace at the end of the chapter shows all 132 output columns, you can find the remainder of the line there.

CDI TRACE FLAGS

All traces in CDI are controlled by the settings of on/off flags. Console commands are used to set these flags to the ON state or cleared them to the OFF state. These commands are *CDI SET and *CDI CLR respectively. Following is the format of these commands.

```
*CDI SET flag
*CDI CLR flag
*CDI SET ALL
*CDI CLR ALL
```

Where *flag* is the trace flag number. These are described in Table 4-1. The ALL option sets or clears all trace flags.

The following command displays the settings of all flags.

```
*CDI FLAGS
```

Table 4-1. CDI Trace Flags

Flag #	Name	Description of Traced Information
0	LAN traffic	This displays LAN data as written to or read from a CPA. See the LAN Traffic Trace section.
1	ARP	Address Resolution Protocol is used to find the physical network address associated with an Internet Protocol address. This address mapping is rather static throughout CDI execution.
2	ROUTE	Routing within CDI is limited to selecting a CPA through which each LAN frame is sent. This is rather static throughout CDI execution.
3	SIGNAL	Signals are internal messages within CDI used to coordinate the state of TCP connections. See the SIGNAL Trace section.
4	TCP	Transmission Control Protocol is the connection based messaging protocol used between CSC and the server. See the TCP Trace section.
5	ICMP	ICMP is a protocol for sending control and error messages between Internet Protocol hosts. CSC uses only the ICMP echo message to determine if any communication with the server is possible.
6	TCPUI	CDI treats its interface with CSC as a user interface. TCPUI traces show requests across this interface. See the TCPUI Trace section. NOTE: These traces appear in the CSC PRINT\$ file.
7	Warning msg	This flag enables additional warning messages within CDI.
8	LAN IO	This flag controls tracing of I/O request information as well as an abbreviated version of the LAN data. See the LAN I/O section.
9	STAT	This flag controls the collection and display of additional CPA operational statistics.

TCPUI TRACE

Code in the CDI common bank traces the interface calls between CSC and CDI. When enabled, the trace code is executed by the CDI caller and the output appears in the caller's PRINT\$ file. Since CSC is the caller, the TCPUI trace output is placed in the CSC PRINT\$ file.

Following are the commands to enable and disable the TCPUI traces.

```
*CDI SET 6
*CDI CLR 6
```

The general format of TCPUI traces is as follows:

```
<type> yyyyymmdd hh:mm:ss.mmm func fields
ER SNAP$ of the request packet
```

- Type is either TCPREQ or TCPRSP. These are for requests to CDI and responses from CDI respectively.
- Yymmdd hh:mm:ss.mmm is the date and time.
- Func is the requested function. Common functions are listed in Table 4-2.
- Fields are pairs of labels and values that tell what is in the request packet. The significant fields are listed in Table 4-2.

Figure 4-2 shows the two formats of TCPUI traces. The first form is used for open and status requests. The second is used for all other requests. Table 4-2 describes and discusses significant fields in these traces.

```
<TCPREQ> 19990710 10:28:50.765 Uc_Open STAT Ok Type 0 Dest 0 Sub 0 Cid 0
                                         ID 000217734517
+ Flg 0001(Tt_Push) ECODE 0 (EXOK) Event 0
+ DPORT 32780 DADDR [209.25.5.32] SPORT 0 SADDR [209.25.5.19]
                                         O_Flags 01 (To_Trace,To_Server,To_Active)
+ Timeout 10000 Rcv_Wnd_Siz 500 Snd_Wnd_Siz 1536
TCP PKT      666504      156602      10:28:50      ...
676140      000000000000      321031005023      010000023420      000000000000      0000...
676150      000000000000      321031005023      010000023420      000764003000      0000...
676160      040040040040      600043676257      0000000000030

<TCPRSP> 19990710 10:28:51.366 Uc_Recv STAT Ok Type 0 Dest 0 Sub 0 Cid 8
                                         ID 000217734517
+ Flg 2000(Tt_Open) ECODE 0 (EXOK) Event 0 ACW 000003342424
                                         (670505 + 0) Unfilled 1536 Num_Bytes 0
TCP PKT      666504      156326      10:28:51      ...
676140      0000000003722      321031005023      010000023420      000000000010      0000...
676150      0000000003722      321031005023      010000023420      001000003000      0000...
676160      040040040040      600043676257      0000000000030
```

Figure 4-2. TCPUI Trace Format

Table 4-2. Description of TCPUI trace output fields

Field	Description
function	Uc_Abort -- abort a connection Uc_Close -- close a connection Uc_Open -- open a connection Uc_Recv and Uc_Recv_NoWait -- receive data and/or status from a connection Uc_Send -- send data through a connection Uc_Status -- check the status of a connection
Flg	Bitmapped flags specifying options of the request or returned indications. The internal name for each set bit is also shown within parentheses.
STAT	Completion status for the request. The internal status description is also shown within parentheses.
Cid	The connection identifier is a value used to identify this connection. It is the same as the CON field in the output of the *CDI TCP CON console keyin.
ID	An identifier given to this connection by CSC. It is a timestamp in milliseconds past midnight.
ECODE	Error code returned to CSC. The internal descriptive name for this code is also shown within parentheses.
DADDR and DPORT	The destination (server) IP address and port number for this connection.
SPORT and SADDR	The source (CPA) IP address and port number for this connection.
O_Flgs	Open flags used by CSC to modify CDI processing of an open request. The internal name for each set flag is also shown within parentheses.
ACW	The address of the data to be transferred. This is a memory address (shifted left two bits) combined with a starting quarter word indicator (lower two bits).
Num_Bytes	Number of bytes being sent or being returned.

TCP TRACE

The TCP trace displays information about the state of a connection. This includes items from CDI data structures and a portion of the LAN data if applicable.

Following are the commands to enable and disable the TCP traces.

```
*CDI SET 4
*CDI CLR 4
```

Figure 4-3 shows typical output produced when TCP tracing is enabled.. Table 4-3 describes and discusses significant fields in these traces.

```
*** TCP Datagram Trace ***
Datagram Type : Inbound
State : SynSnt
Time : 19990710 10:28:50.970
Sequence Nbr : 4132442013
Ack Seq Nbr : 37730784
Tot Msg Len : 44
IP Header Len: 20
TCP Header Len: 20
Next Sequence Nbr : 37730784
Unacknowledged : 37730783
Send Window : 0
Receive Window : 512
Remote IP Address : [209.25.5.32]
Remote Port : 32780
Local IP Address : [209.25.5.19]
Local Port : 2002
Work-To-Be-Done : 000010
Flags Set (Octal) : 140022
                  : Syn
                  : Ack
Inbound Buffer : No Data Received
<<<< Transmission Control Block Dump (TCB) >>>>
TCB is active
Connection Type: 02 State: 03
WTBDF: 000010 Idx: 000010 SAddr: 321031005023 SPort: 2002
DAddr: 321031005040 DPort: 32780
RcvNxt: 0 RcvWnd: 512 RcvUp: 0 ISS: 37730783 SndUna: 37730783
SndNxt: 37730784 SndWin: 0 SndUP: 0 RtxSeq: 37730783
SndWL1: 0 SndWL2: 0 RtxTimeOut: 3000 TwaitTimeOut: 250
RtxTime: 2750 TwaitTime: 0 SegSeq: 0 SegAck: 0
SegWnd: 0 SegUp: 0 SegDat: 0 MaxSndWnd: 0 SegACW: 000000000000
RunID: 103010276405 NumRtx: 0 RtdSeq: 0 RtxStamp: 000000000000
RtdSample: 000000000000 RtdSmooth: 000000000000 UserID: 000217734517
Link: 000000000000
RcvWndSiz: 512 SndWndSiz: 1536 UsrFlag: 0000 Ecode: 00
UsrEvt: 0 UsrACW: 000000000000 UsrRoom: 1536 UsrNByte: 0 UsrBDI: 000000
IVB(D): 0 IQF(O): 000000 IQR(O): 000000
Push bit is not set on input
OPSH: 000000 OVB(D): 0 OQF: 000000 OQR: 000000 AckTimeOut: 15
*** TCP Datagram Trace End ***
```

Figure 4-3. TCP Trace Format

Table 4-3. IP Datagrams and TCP Segments Trace Data Descriptions

Field	Description	
Datagram Type	Either Inbound or Outbound.	
State	The TCP connection state at the time of the trace entry. These states are as described in RFC 793.	
	Value	State in RFC 793
	Closed	CLOSED
	CloseWt	CLOSE-WAIT
	Closing	CLOSING
	Establ	ESTABLISHED
	FinWt1	FIN-WAIT-1
	FinWt2	FIN-WAIT-2
	Lastack	LAST-ACK
	Listen	LISTEN
	SynRcv	SYN-RECEIVED
	SynSent	SYN-SENT
	TimeWt	TIME-WAIT
Time	Timestamp.	
Tot Msg Len	The total message length sent or received. The value includes the TCP and IP headers.	
IP Header Len	Length of the IP header.	
TCP Header Len	Length of the TCP header.	
Remote IP Address	The remote Internet address with which the connection is communicating.	
Remote Port	The port number with which the connection is communicating.	
Local IP Address	Also referred to as the source address. When combined with the port number, this refers to the local socket.	
Local Port	The port number for the local Internet host.	
Work-To-Be-Done	Flags indicating work waiting to be done for this connection.	
Flags Set (Octal)	This contains flag settings that should be included in the next frame sent for this connection.	

SIGNAL TRACE

CDI is a multi-activity program. When one activity needs the services of another, it passes the request in the form of an internal signal. The signal trace shows these signals and what is being requested.

Following are the commands to enable and disable the SIGNAL traces.

```
*CDI SET 3
*CDI CLR 3
```

Figure 4-4 shows the general format of the SIGNAL trace and a typical trace output. Table 4-4 describes and discusses fields in these traces.

```
<SIGNAL> yyyyymmdd hh:mm:ss.mmm type ID:cid what:spec Bytes:#1 Unfilled:#2
<SIGNAL> 19990710 10:28:51.052 TCP ID:734517 Event:Open Bytes:0 Unfilled:1536
```

Figure 4-4. SIGNAL Trace Format

Table 4-4. Description of SIGNAL trace output fields

Field	Description
yymmdd hh:mm:ss.mmm	Time when this signal was generated.
type	Type of signal. The TCP signals are significant..
cid	The low order 18 bits of the unique connection identifier that was assigned by CSC.
what	Tells what type of information is being signaled. This is either "Event" for an event that changes the TCP state or "FlagSet" to set a flag in the internal connection data structure.
spec	This identifies the events or flags being signaled.
#1 and #2	These are byte counts for events or flags that signal a data transfer

LAN I/O TRACE

CDI uses arbitrary device I/O to communicate with the CPA. The LAN I/O traces show the IOAID\$ packet and the beginning of any data that is transferred.

Following are the commands to enable and disable the LAN I/O traces.

```
*CDI SET 8
*CDI CLR 8
```

Each LAN I/O trace begins with a description of what is traced. This is one of the following:

- Post Attention Int Returned
- Read Channel Issued
- Read Channel Returned
- Write Channel Command

Except for Read Channel Issued, each of these is followed by a display of the information in the IOAID\$ packet. For requests that transfer data, this is followed by a display of the data.

Figure 4-5 shows output from a typical LAN I/O trace.

```

Write Channel Command      : 19990710 10:28:50.818
IOAID: 062475  BRAVOB  I/O Stat: 07  S/W Code: 06
    Comp: 000001 CSWF
    CSWs:000502021020/402053342550/000000000001/000000000014
ADI: Device 0x0C<CE DE>  Res: 4  Subch:
Write Buffer (Hex): 08 00 00 00 00 2C 63 08 00 00 00 00 86 7A 00 00
Write Buffer (Hex): D1 19 05 20 08 00 20 1B 1E D2 00 00 00 00 00 00
Write Buffer (Hex): 45 00 00 2C 00 4C 00 00 14 06 FA 1A D1 19 05 13
...
Write Buffer (Oct): 010 000 000 000 000 054 143 410 000 000 000 000 206 ...
Write Buffer (Oct): 321 031 005 040 010 000 040 033 036 322 000 000 000 ...
Write Buffer (Oct): 105 000 000 054 000 114 000 000 024 006 372 032 321 ...
...
IOAID Packet (Oct): 072706332407 050505050505 000000000000 070101000033
IOAID Packet (Oct): 000001000000 000003062666 000000000001 000006000000
IOAID Packet (Oct): 000000062672 041457114546 041457114554 000502021020
IOAID Packet (Oct): 402053342550 000000000001 000000000014 000000000000
IOAID Packet (Oct): 000000000000 000000000000 400000000000 000000000000
IOAID Packet (Oct): 000000000000 000000000000 000000000000 000000000000

Read Channel Issued : 19990710 10:28:50.919 BRAVOA
Read Channel Returned : 19990710 10:28:50.956
IOAID: 062442  BRAVOA  I/O Stat: 07  S/W Code: 06
    Comp: 000001 CSWF
    CSWs:000502021000/402053311140/0400000000204/000000000014
ADI: Device 0x0C<CE DE>  Res: 530  Subch:

Channel Data Read :

>>>>> IP Header <<<<<
45 00 00 2C 8A 26 40 00 FF 06 45 3F D1 19 05 20 D1 19 05 13

>>>>> TCP Header <<<<<
80 0C 07 D2 F6 50 0F 9D 02 3F B9 E0 60 12 23 98 81 C8 00 00

>>>>> User Data (HEX) <<<<<
02 04 02 18 FF FF 00 00 00 00 00 00 00 00 00 00

```

Figure 4-5. LAN I/O Trace Format

LAN TRAFFIC TRACE

The LAN traffic trace displays the entire frame written to or read from a CPA. This is displayed as a stream of hexadecimal digits.

Following are the commands to enable and disable the LAN traffic traces.

```
*CDI SET 0
*CDI CLR 0
```

Figure 4-6 shows typical output produced when LAN Traffic tracing is enabled. Output for IP frames begins with "C>" and non-IP frames begin with "#>". The first line of each trace entry includes a direction indicator and the time. The direction indicator is RD for read from CPA or WR for write to CPA.

```
C>WR10:28:50.814
C>4500002C004C00001406FA1AD1190513D119052007D2800C023FB9DF000000006002020 ...
C>RD10:28:50.952
C>4500002C8A264000FF06453FD1190520D1190513800C07D2F6500F9D023FB9E06012239 ...
C>WR10:28:51.057
C>45000028004D00001406FA1DD1190513D119052007D2800C023FB9E0F6500F9E5010020 ...
```

Figure 4-6. LAN Traffic Trace Format

TRACE FILE EXAMPLE

This is a portion of a CDI print file with all trace flags turned ON. This shows the exchanges to open a connection for sending a request to the server.

```
C>WR10:28:50.814
C>4500002C004C00001406FA1AD1190513D119052007D2800C023FB9DF0000000060020200A95F000002040218
  Write Channel Command      : 19990710 10:28:50.818
IOAID: 062475 BRAVOB I/O Stat: 07 S/W Code: 06
  Comp: 000001 CSWF
  CSWs:000502021020/402053342550/000000000001/000000000014
ADI: Device 0x0C<CE DE> Res: 4 Subch:
Write Buffer (Hex): 08 00 00 00 00 2C 63 08 00 00 00 00 86 7A 00 00
Write Buffer (Hex): D1 19 05 20 08 00 20 1B 1E D2 00 00 00 00 00 00
Write Buffer (Hex): 45 00 00 2C 00 4C 00 00 14 06 FA 1A D1 19 05 13
Write Buffer (Hex): D1 19 05 20 07 D2 80 0C 02 3F B9 DF 00 00 00 00
Write Buffer (Hex): 60 02 02 00 A9 5F 00 00 02 04 02 18 00 00 FF 00
Write Buffer (Hex): 00 00 00 02 00 02 00 00 00 00 00 00 00 00 00 00
Write Buffer (Hex): 00 00 00 00 00 00 00 00 00 00 02 00 00 C0 A8 E1 00
Write Buffer (Hex): 00 00 00 00 00 00 00 00 00 00 00 02 00 02 00 00
Write Buffer (Hex): C7 CB 44 00 00 00 00 00 00 00 00 00 00 00 00 05
Write Buffer (Hex): 00 02 00 00 C0 A8 64 00 00 00 00 00 00 00 00 00
Write Buffer (Oct): 010 000 000 000 000 054 143 410 000 000 000 000 206 172 000 000
Write Buffer (Oct): 321 031 005 040 010 000 040 033 036 322 000 000 000 000 000 000
Write Buffer (Oct): 105 000 000 054 000 114 000 000 024 006 372 032 321 031 005 023
Write Buffer (Oct): 321 031 005 040 007 322 200 014 002 077 271 337 000 000 000 000
Write Buffer (Oct): 140 002 002 000 251 137 000 000 002 004 002 030 000 000 777 000
Write Buffer (Oct): 000 000 000 002 000 002 000 000 000 000 000 000 000 000 000 000
Write Buffer (Oct): 000 000 000 000 000 000 000 000 000 002 000 000 300 250 341 000
Write Buffer (Oct): 000 000 000 000 000 000 000 000 000 000 000 002 000 002 000 000
Write Buffer (Oct): 307 313 104 000 000 000 000 000 000 000 000 000 000 000 000 005
Write Buffer (Oct): 000 002 000 000 300 250 144 000 000 000 000 000 000 000 000 000
IOAID Packet (Oct): 072706332407 050505050505 000000000000 070101000033
IOAID Packet (Oct): 000001000000 000003062666 000000000001 000006000000
IOAID Packet (Oct): 000000062672 041457114546 041457114554 000502021020
IOAID Packet (Oct): 402053342550 000000000001 000000000014 000000000000
```

IOAID Packet (Oct): 000000000000 000000000000 400000000000 000000000000
IOAID Packet (Oct): 000000000000 000000000000 000000000000 000000000000

```
*** TCP Datagram Trace ***
Datagram Type : Outbound
State : SynSnt
Time : 19990710 10:28:50.881
Sequence Nbr : 37730783
Ack Seq Nbr : 0
Tot Msg Len : 44
IP Header Len: 20
TCP Header Len: 20
Next Sequence Nbr : 37730784
Unacknowledged : 37730783
Send Window : 0
Receive Window : 512
Remote IP Address : [209.25.5.32]
Remote Port : 32780
Local IP Address : [209.25.5.19]
Local Port : 2002
Work-To-Be-Done : 000014
: Rtx
: Syn
Flags Set (Octal) : 140002
: Syn
Inbound Buffer : No Data Received
<<<< Transmission Control Block Dump (TCB) >>>>
TCB is active
Connection Type: 02 State: 03
WTBDF: 000014 Idx: 000010 SAddr: 321031005023 SPort: 2002 DAddr:321031005040 DPort:32780
RcvNxt: 0 RcvWnd: 512 RcvUp: 0 ISS: 37730783 SndUna: 37730783
SndNxt: 37730784 SndWin: 0 SndUP: 0 RtxSeq: 0
SndWL1: 0 SndWL2: 0 RtxTimeOut: 3000 TwaitTimeOut: 250
RtxTime:0 TwaitTime: 0 SegSeq: 0 SegAck: 0
SegWnd: 0 SegUp: 0 SegDat: 0 MaxSndWnd: 0 SegACW: 000000000000
RunID: 103010276405 NumRtx: 0 RtdSeq: 0 RtxStamp: 000000000000
RtdSample: 000000000000 RtdSmooth: 000000000000 UserID: 000217734517 Link: 000000000000
RcvWndSiz: 512 SndWndSiz: 1536 UsrFlag: 0000 Ecode: 00
UsrEvt: 0 UsrACW: 000000000000 UsrRoom: 1536 UsrNByte: 0 UsrBDI: 000000
IVB(D): 0 IQF(O): 000000 IQR(O): 000000
Push bit is not set on input
OPSH: 000000 OVB(D): 0 OQF: 000000 OQR: 000000 AckTimeOut: 15
*** TCP Datagram Trace End ***
Post Attention Interrupts : 19990710 10:28:50.916
IOAID: 062442 BRAVOA I/O Stat: 00 S/W Code: 00
Comp: 000001 CSWF
CSWs:000502021000/402053403676/000020000004/000000000200
ADI: Device 0x80<Attn> Res: 16 Subch:
Read Channel Issued : 19990710 10:28:50.919 BRAVOA
C>RD10:28:50.952
C>4500002C8A264000FF06453FD1190520D1190513800C07D2F6500F9D023FB9E06012239881C8000002040218
Read Channel Returned : 19990710 10:28:50.956
IOAID: 062442 BRAVOA I/O Stat: 07 S/W Code: 06
Comp: 000001 CSWF
CSWs:000502021000/402053311140/040000000204/000000000014
ADI: Device 0x0C<CE DE> Res: 530 Subch:

Channel Data Read :

>>>> IP Header <<<<<
45 00 00 2C 8A 26 40 00 FF 06 45 3F D1 19 05 20 D1 19 05 13

>>>> TCP Header <<<<<
80 0C 07 D2 F6 50 0F 9D 02 3F B9 E0 60 12 23 98 81 C8 00 00

>>>> User Data (HEX) <<<<<
02 04 02 18 FF FF 00 00 00 00 00 00 00 00 00 00

*** TCP Datagram Trace ***
Datagram Type : Inbound
State : SynSnt
Time : 19990710 10:28:50.970
Sequence Nbr : 4132442013
Ack Seq Nbr : 37730784
Tot Msg Len : 44
IP Header Len: 20
TCP Header Len: 20
Next Sequence Nbr : 37730784
Unacknowledged : 37730783
Send Window : 0
Receive Window : 512
```

```

Remote IP Address : [209.25.5.32]
Remote Port      : 32780
Local IP Address  : [209.25.5.19]
Local Port       : 2002
Work-To-Be-Done  : 000010
: Syn
Flags Set (Octal) : 140022
: Syn
: Ack
Inbound Buffer    : No Data Received
<<<< Transmission Control Block Dump (TCB) >>>>
TCB is active
Connection Type: 02 State: 03
WTBDF: 000010 Idx: 000010 SAddr: 321031005023 SPort: 2002 DAddr: 321031005040 DPort: 32780
RcvNxt: 0 RcvWnd: 512 RcvUp: 0 ISS: 37730783 SndUna: 37730783
SndNxt: 37730784 SndWin: 0 SndUP: 0 RtxSeq: 37730783
SndWL1: 0 SndWL2: 0 RtxTimeOut: 3000 TwaitTimeOut: 250
RtxTime: 2750 TwaitTime: 0 SegSeq: 0 SegAck: 0
SegWnd: 0 SegUp: 0 SegDat: 0 MaxSndWnd: 0 SegACW: 000000000000
RunID: 103010276405 NumRtx: 0 RtdSeq: 0 RtxStamp: 000000000000
RtdSample: 000000000000 RtdSmooth: 000000000000 UserID: 000217734517 Link: 000000000000
RcvWndSiz: 512 SndWndSiz: 1536 UsrFlag: 0000 Ecode: 00
UsrEvt: 0 UsrACW: 000000000000 UsrRoom: 1536 UsrNByte: 0 UsrBDI: 000000
IVB(D): 0 IQF(O): 000000 IQR(O): 000000
Push bit is not set on input
OPSH: 000000 OVB(D): 0 OQF: 000000 OQR: 000000 AckTimeOut: 15
*** TCP Datagram Trace End ***

<SIGNAL> 19990710 10:28:51.052 TCP ID:734517 Event:Open Bytes:0 Unfilled:1536

Read Channel Issued : 19990710 10:28:51.055 BRAVOA
C>WR10:28:51.057
C>45000028004D000001406FA1DD1190513D119052007D2800C023FB9E0F6500F9E50100200B7810000
Write Channel Command : 19990710 10:28:51.062
IOAID: 062475 BRAVOB I/O Stat: 07 S/W Code: 06
Comp: 000001 CSWF
CSWs: 000502021020/402053136442/000000000001/0000000000014
ADI: Device 0x0C<CE DE> Res: 4 Subch:
Write Buffer (Hex): 08 00 00 00 00 28 63 08 00 00 00 00 87 EE 00 00
Write Buffer (Hex): D1 19 05 20 08 00 20 1B 1E D2 00 00 00 00 00 00
Write Buffer (Hex): 45 00 00 28 00 4D 00 00 14 06 FA 1D D1 19 05 13
Write Buffer (Hex): D1 19 05 20 07 D2 80 0C 02 3F B9 E0 F6 50 0F 9E
Write Buffer (Hex): 50 10 02 00 B7 81 00 00 00 00 00 00 00 00 00 FF 00
Write Buffer (Hex): 00 00 00 00 00 00 00 FF FF FF FF FF FF FF 09
Write Buffer (Hex): 70 16 24 02 00 34 1D 2F 24 00 00 00 00 02 13 24
Write Buffer (Hex): A7 06 2C 02 00 34 1D 2F 24 01 00 00 00 02 0D 09
Write Buffer (Hex): 76 0D 90 02 00 34 1D 2F 24 01 01 00 00 00 64 E9
Write Buffer (Hex): 46 1C D7 02 00 34 1D 2F 24 01 02 00 00 02 0B 4F
Write Buffer (Oct): 010 000 000 000 000 050 143 410 000 000 000 000 207 356 000 000
Write Buffer (Oct): 321 031 005 040 010 000 040 033 036 322 000 000 000 000 000 000
Write Buffer (Oct): 105 000 000 050 000 115 000 000 024 006 372 035 321 031 005 023
Write Buffer (Oct): 321 031 005 040 007 322 200 014 002 077 271 340 366 120 017 236
Write Buffer (Oct): 120 020 002 000 267 201 000 000 000 000 000 000 000 000 777 000
Write Buffer (Oct): 000 000 000 000 000 000 000 377 377 377 377 377 377 011
Write Buffer (Oct): 160 026 044 002 000 064 035 057 044 000 000 000 000 002 023 044
Write Buffer (Oct): 247 006 054 002 000 064 035 057 044 001 000 000 000 002 015 011
Write Buffer (Oct): 166 015 220 002 000 064 035 057 044 001 001 000 000 000 144 351
Write Buffer (Oct): 106 034 327 002 000 064 035 057 044 001 002 000 000 002 013 117
IOAID Packet (Oct): 072706332407 050505050505 000000000000 070101000033
IOAID Packet (Oct): 000001000000 000003062666 000000000001 000006000000
IOAID Packet (Oct): 000000062672 041457117250 041457117256 000502021020
IOAID Packet (Oct): 402053136442 000000000001 000000000014 000000000000
IOAID Packet (Oct): 000000000000 000000000000 400000000000 000000000000
IOAID Packet (Oct): 000000000000 000000000000 000000000000 000000000000

*** TCP Datagram Trace ***
Datagram Type : Outbound
State : Establ
Time : 19990710 10:28:51.135
Sequence Nbr : 37730784
Ack Seq Nbr : 4132442014
Tot Msg Len : 40
IP Header Len: 20
TCP Header Len: 20
Next Sequence Nbr : 37730784
Unacknowledged : 37730784
Send Window : 9112
Receive Window : 512
Remote IP Address : [209.25.5.32]
Remote Port : 32780
Local IP Address : [209.25.5.19]

```

```

Local Port      : 2002
Work-To-Be-Done : 000000
Flags Set (Octal) : 120020
: Ack
Inbound Buffer   : No Data Received
<<<< Transmission Control Block Dump (TCB) >>>>
TCB is active
Connection Type: 02   State: 05
WTBDF: 000000 Idx: 000010   SAddr: 321031005023   SPort: 2002   DAddr:321031005040   DPort:32780
RcvNxt: 4132442014   RcvWnd: 512   RcvUp: 0   ISS: 37730783   SndUna: 37730784
SndNxt: 37730784   SndWin: 9112   SndUP: 0   RtxSeq: 37730783
SndWL1: 0   SndWL2: 0   RtxTimeOut: 3000   TwaitTimeOut: 250
RtxTime:0   TwaitTime: 0   SegSeq: 4132442013   SegAck: 37730784
SegWnd: 9112   SegUp: 0   SegDat: 0   MaxSndWnd: 0   SegACW: 000001125110
RunID: 103010276405   NumRtx: 0   RtdSeq: 0   RtxStamp: 000000000000
RtdSample: 000000000000   RtdSmooth: 000000000000   UserID: 000217734517   Link: 000000000000
RcvWndSiz: 512   SndWndSiz: 1536   UsrFlag: 2000   Ecode: 00
UsrEvt: 0   UsrACW: 000000000000   UsrRoom: 1536   UsrNByte: 0   UsrBDI: 000000
IVB(D): 0   IQF(O): 000000   IQR(O): 000000
Push bit is not set on input
OPSH: 000000   OVB(D): 0   OQF: 000000   OQR: 000000   AckTimeOut: 15
*** TCP Datagram Trace End ***

```

APPENDIX A. ERROR CODES

This appendix describes errors you might encounter while executing CDI. Some are configuration errors, while others are internal errors.

The most common errors are described in the first section, “Common CDI Error Codes.” Each error code is followed by a brief description.

A complete listing of internal errors is in the second section, “Summary of Internal Error Messages.” If you encounter an internal error, please send the PRINT\$ output to Unisys for further analysis.

Finally, the appendix concludes with a “Summary of Print Messages.”

COMMON CDI ERROR CODES

Main Errors

0101 - CDI Error : DUMP\$ file asgn error.

(Runstream error) CDI is unable to assign the DUMP\$ file. The external name is SYSS\$*NETWORKDUMP. Check the run listing for facility errors encountered.

0110 - No message : hardware feature mode conflict.

(Internal error) This conflict check is placed in CDI to trap error in setting hardware features. This is an internal error.

0110 - CDI Error : MEMPOOL too small.

(Internal error) MEMPOOL is a configuration tag in the CDI generation. This tag is not available to the customer. Usually, this error is caused by extremely heavy traffic across the LAN. One way to obtain more information is to restart CDI, key in '*CDI buffers' every 10 min. for a period of time. This key-in will display the total number of memory configured, maximum and minimum used.

0111 - No message : IOP not found in table.

(OS configuration) CDI looks for the exec ID and matches up the arbitrary device channel format. CDI supports all 1100s and 2200s. This error might occur if CDI does not know the CPU (Unisys has a new processor) or the system does not use a standard processor ID. Contact your Unisys CSE for further information.

0200 - CDI Error : Incorrect HDWFEATURE set.

(Internal error) This error indicates that there are conflicts in the hardware feature set and the actual processor found by CDI. This is caused by an erroneous configuration parameter in the CDI system generation and will happen only when CDI is running on an old EXEC (Level 38 or prior). The customer has no access to this configuration tag.

TCP Errors

0101 - No message : CPU\$BYTE not matching.

(TCPSUB, Internal error) CPU\$BYTE is used for specifying the instruction set to be used.

TCP\$IP\$: User bank overlaps TCP\$IP\$ bank. (Error).

(TCPUI, User program error) TCPUI is used for user programs to access the CDI bank directly (TCP\$IP\$BANK). Note that the external name is TCP\$IP\$. CSC uses the TCPUI programmatic interfaces to communicate with CDI. This and the following errors pertain to the CSC function calling errors. If they occur, contact your Unisys CSE immediately.

TCP\$IP\$: Bad PLUS stack addressing. (Error).

(TCPUI, User program error) CSC/CDI interface error. An invalid PLUS stack address is being used.

TCP\$IP\$: Bad PLUS workspace addressing. (Error).

(TCPUI, User program error) CSC/CDI interface error. An invalid PLUS workspace address is being used.

TCP\$IP\$: Bad function code passed. (Error).

(TCPUI, User program error) The valid function codes are from 0 through 16. This error indicates an invalid function code is passed to CDI for processing.

TCP\$IP\$: Bad packet addressing. (Error).

(TCPUI, User program error) An invalid packet address is being used that CDI cannot reach.

TCP\$IP\$: Quarter word mode required. (Error).

(TCPUI, User program error) The user program is restricted to use quarter word mode addressing.

TCP\$IP\$: Character addressing mode not allowed. (Error).

(TCPUI, User program error) Character addressing mode not allowed.

TCP\$IP\$: Welcome; you've made it into the TCP\$IP\$ common bank.

(TCPUI, Comment for user programs) This is a test function code to simply flash a success message to the caller. In this case, the caller is CSC.

TCP\$IP\$: PLUS stack overflow

(TCPUI, User program error) The user program did not reserve sufficient PLUS stack space for CDI.

Utilities, Internal Errors

67 - No free entry in the ACB (Activity Control Block).
0101 - Number of Activity Control Block is zero.
68 - Attempting to access ACB not in use.
69 - Switch List not found.
0105 - PLS workspace pointer diff from ACB
05 - Incorrect buffer queue
070 - Buffer not queued

IIACT PLS Internal Errors

0600 - ?Signal: XXXXXX Unknown signal received.
0101 - Panic dump error, file asgn failure.

Configuration Errors

0101 - CDI Error : CDI\$\$TABLE @use error

(INITIAL, Internal error) CDI obtained an error while trying to assign a USE statement to the server configuration table 'CDI\$\$TABLE'.

0101 - CDI Error : CDI\$\$TABLE @asg error

(INITIAL, Configuration error) CDI is trying to assign the CDI\$\$TABLE file internally. In a normal run, this is a temporary file created in the CDI runstream generation skeleton. This error indicates that the run skeleton is not correct and therefore did not create the CDI\$\$TABLE temporary file. Check the CDI\$PARAM element in the SYS\$LIB\$*CSC-PARM file.

0101 - CDI Error : C\$HOSTFILE @use error

(INITIAL, Internal error) This condition is similar to the CDI\$\$TABLE error above.

0101 - CDI Error : C\$HOSTFILE @asg error

(INITIAL, Configuration error) CDI is trying to assign the C\$HOSTFILE file internally. In a normal run, this is a temporary file created in the CDI runstream generation skeleton. This error indicates that the run skeleton is not correct and therefore did not create the C\$HOSTFILE temporary file. Check the correctness of the CDI\$PARAM element in SYS\$LIB\$*CSC-PARM file.

0101 - No Message. Error call DUM link bank

(INITIAL, Internal error) Error in running a special DUM processor. This is an internal processor. This error indicates a fatal error and should be reported to your Unisys CSE.

0300 - CDI Error : Host name not found in hostfile

(INITIAL, Configuration error) C\$HOSTFILE contains all valid hosts configured for CDI. Remember both clients and servers are considered as hosts in the context of a TCP/IP network. Also note that hosts in TCP/IP are 'PHYSICAL CONNECTIONS'.

For example, a client with a dual-CPA configuration constitutes two hosts and requires two Internet addresses. When the host name in the CDI program execution does not match one of the host names in the CDI\$PARAM element, this error will occur. Normally, the host name found in the CDI program execution is generated by the skeleton and is not user configurable. Make sure that the CDI\$PARAM is proper, and instruct the user to always use the standard runstream to start CDI.

INITIAL: invalid subnetmask specified on processor call.

(INITIAL, Configuration error) The subnetmask is the mask provided to CDI for the purpose of separating the different subnetworks in the TCP/IP addressing scheme. Resolve this error in the same manner as the above error condition.

0101 - CDI Error : CDI\$\$TABLE @free error

(INITIAL, File error) This error should not occur.

0101 - No Message. Host name is NULL

(INITIAL, Configuration error) If the host name is not found in the CDI program execution call line, this message will appear. Check the CDI\$PARAM element and make sure all host names are in place. Remember that the host name parameter in the CDI program call statement is automatically generated by the standard skeleton.

This error usually means that either the user changed the generated runstream instead of following the standard procedure, or the CDI\$PARAM element is incorrect.

0102 - CDI Error : Route install fail

(INITIAL, Configuration error) This error arises when a duplicate server address or a duplicate CLIENT SGS tag is found. Make sure that the following is true in the CDI\$PARAM element:

- One, and only one, CLIENT statement is specified.
- At least one CPA statement is specified.
- At least one PATH statement is specified.
- The PATH statement relates to one of the CPAs.
- The addressing scheme is correct. Refer to the section “Checking Your CDI Network Addressing” in this guide for more information.

0103 - INITIAL: route/arp installation failed!.

(INITIAL, Configuration error) Same as above.

0777 - The CPA EEPROM is not set for Attention Mode operation.

Run the CPATST program described in Chapter 3 of this guide to set the proper memory bytes and restart CDI.

SUMMARY OF INTERNAL ERROR MESSAGES

Table A-1 summarizes the CDI internal error messages.

Table A-1. CDI Internal Error Messages

Error Code	Module Name	Description
01	BPOOL	Bad parameter - priority must be > 0.
100	NIFK200	Packet leader contains '377' and packet type is IP.
00100	TIMER	Timer_Queue_Init(): If Free_Timer_Head not equal to null.
0101	CONTINGENCY	ER CRTN\$ executed in non-contingency mode.
0101	IIACT-PLS	Signal '0' received.
0101	IIACT-PLS	Route command bad request.
0101	IIACT-PLS	Unexpected error from Route_Ctl.
0101	IIACT-PLS	arp-ctl bad request received.
0101	INITIAL	Assign a file (used for name and address table lookup) failure.
0101	INITIAL	Error while reading INFOR\$TABL.
0101	INITIAL	Demand mode ATREAD failure.
0101	IPPLUS	IP_Input - null packet received.
0101	MAIN	DUMP\$ file assign failure. Host name input request.
0101	NETIF	Netif_Signal - packet type other than IP.
0101	NETIF	Nif_Enq_Pkt : output queue and queue length inconsistent.
0101	NETARP	ARP packet hardware type is invalid (not Ethernet).
0101	NIFK200	Build_CCW(): IOP type invalid (not 1100 60/70/80/90).
0101	NIFK200	K200-IF_UP(): NIF-K200 interface is not initialized.
0101	NIFK200	K200_IF_Int(): shouldn't receive an Ethernet interrupt.

Error Code	Module Name	Description
0101	NIFK200	K200_IF_Start(): packet to be transmitted doesn't contain Ethernet address.
0101	NIFLOOP	Loop_IF_UP(): interface is not initialized.
0101	NIFLOOP	Loop_IF-Init(): multiple calls received.
0101	NIFLOOP	rtrequest() returns a bad request type.
0101	NIFLOOP	Loop_IF_DN(): rtrequest() returns an unexpected error.
0101	NIFLOOP	Loop_IF_Int: shouldn't receive an interrupt for the loopback interface.
0101	PLS-UTIL	ER BANK\$ failure for KEYIN\$ packet.
0101	ROUTE	rtfree() : an attempt to decrement reference count which is already '0'.
0101	ROUTE	rtfree() : route table entry pointer is null.
0101	ROUTE	rtrequest_internal() : route request is not an add route or delete route.
0101	ROUTE	rtinstdyn() : rtrequest_internal() to add a dynamic route entry failure.
0101	ROUTE	rt_alloc_internal(): route structure is null.
0101	TCPUTILPLUS	Copy_Seq_tran(): If available bytes become negative.
0101	TCPUTILPLUS	Copy_Seq_Retran(): If number of bytes to be copied is negative.
0101	TIMER	Timer_Tick_Tock(): if Time left is less than or equal to Tick_Counter.
0010	TIMER	Make_Timer(): if Timer_Queue_Head equal to null (No more timer queue entries)
0102	IPPLUS	IP_Input - packet type is not IP.
0010	TIMER	Make_Timer(): if Free_Timer_Head entry is in use.
0102	TIMER	Start_Timer(): If time interval specified is negative.
0103	NETARP	ARP packet hardware type is invalid (not Ethernet, NETEX).
00103	TIMER	Make_Timer(): if Free_Timer_head entry backward link is not null.

Error Code	Module Name	Description
0104	NIFK200	K200_IF_Start(): packet type is unknown.
00104	TIMER	Delete_Timer(): An attempt to delete unused entry.
0105	NETARP	ARPSSEND: ARP packet hardware type is invalid (not Ethernet, NETEX).
00105	TIMER	Start_Timer(): Start_Timer entry not in use.
00106	TIMER	Start_Timer(): If number of Entries_Seen is greater than total number of entries (loop somewhere).
00107	TIMER	Start_Timer(): If Timer_Queue time left is negative.
0110	MAIN	Not running in Basic Mode.
00110	TIMER	Start_Timer(): If Timer_Queue time left is negative.
00111	TIMER	Remove_Timer(): If back link of an entry to be removed from active timer queue is null and entry is not Timer_Queue_Head.
00112	TIMER	Remove_Time(): An entry not found in Timer_List.
00113	TIMER	Timer_Tick_Tock(): If first entry is not in use or entry is not active or time left equal to zero.
00114	TIMER	Timer_Tick_Tock(): If Bget() returns a signal with value null.
0130	NIFK200	ADI_Status(): IOP type invalid (not 1100 60/70/80/90).
0150	IIACT-PLS	Interactive command line processing for case !,abort,die console mode is not < 4.
02	BPOOL	Bad parameter - priority must be < 4.
20	RECV	Received message is not of correct type ('NETSTA').
0201	ROUTE	rtfree(): route table entry pointer is null.
0202	ROUTE	rtfree(): route table entry points to null route structure.
0202	NIFK200	Mapname(): device name is not an @use name.
03	BPOOL	Buffer identification mismatch (acquire).

Error Code	Module Name	Description
0300	NIFK200	K200_IF_Init(): unit type is not proper (should be <= C_N_K200).
300	TCPPLUS	Sa_Closed(): Tcp_Send_Int returns with value > 3.
300	TCPPLUS	Sa_Listen(): Tcp_Send_Int returns with value > 3.
301	TCPPLUS	Tcp_check_work(): If request is Delete_Tcb and TCB_Ctype is not equal to CC_User.
0301	TCPSRV	Server_process(): User_Port is not operator, echo, sink or tty test.
0301	TCPSRV	Handle_Ttytst(), Handle_Echo(), Handle_Sink(), Handle_Oper():
0301	TCPSRV	Handle_Ttytst(), Handle_Echo(), Handle_Sink(), Handle_oper(): Invalid event received.
0301	TCPUSER	Tcp_Send(): Invalid Tcb_State.
0302	TCPUSER	Tcp_Close(): If Tcb_State is St_Est, St_Finwt1, St_Finwt2 and Tcb_Usr_ACW is not equal to zero.
0302	NIFK200	K200_IF_Init(): While constructing name if last character is not space or all characters are spaces.
0303	TCPUTILPLUS	Send_Data(): If number of bytes to be transmitted is negative.
0303	TCPUTILPLUS	Copy_Seg_trans(), Copy_Seg_Retran(): If request count is negative.
0303	TCPUTILPLUS	Copy_Seg_tran(), Move_NACW(), Copy_Seg_Retran(): Broken pointers.
0307	TCPUTILPLUS	Seg_Copy(): No buffers are available for chaining of data.
0311	TCPUSER	Tcp_Send(), Tcp_Close(), Tcp_Receive(), Tcp_Abort(): If connection ID index is less than 1 or greater than the addition of number of TCP connections and number of user connections.
0333	TCPUTILPLUS	Remove_Queue(): If number of bytes to be removed from the retransmission queue is greater than Tcb_Ovb.

Error Code	Module Name	Description
0346	TCPUI-PLS	Copy_To_User(): If number of bytes to be copied into base bank is less than zero.
0377	TCPUSER	Tcp_Receive(), Tcp_Abort(): Invalid Tcb_State.
0377	TCPUTILPLUS	Free-Tcb(): If TCB_Ctype is invalid.
04	BPOOL	No buffer available.
0400	NETIF	Subnet mask is not proper.
0402	FIBALLOC	Buffer of requested size not available.
0403	FIBALLOC	Invalid buffer size.
0404	FIBALLOC	An attempt to free unallocated buffer.
05	BPOOL	Buffer identification mismatch (release).
053	TCPUI	TCP\$IP\$: PLUS stack overflow.
06	BPOOL	An attempt to release unallocated buffer.
067	GLOBALDEFS	An attempt to use TCS for cleared cell.
07	BPOOL	Release buffer - BHDRLAST pointer incorrect.
0701	NIFK200	K200_Reset(): ADI_status() invalid (not 0,6,7).
0702	NIFK200	K200_Reset(): ADI_status() invalid (not 0,6,7).
08	BPOOL	Release buffer - release more buffers than in use. BHDRCNT is negative.
09	BPOOL	Release buffer - pad word mismatch.
10	BPOOL	Release buffer - index out of range.

APPENDIX B. CHANNEL STATUS WORD FORMATS

This appendix contains a diagram and table that describes the format used in the trace data for the Channel Device Interface. The table uses the following abbreviations:

- CCW Channel Control Word
- CSW Channel Status Word
- DSF Device Status Field
- MSU Main Storage Unit
- RBC Residual Byte Count
- TSW Tabled Status Word
- UPI Universal Processor Interface

CHANNEL STATUS WORD FORMATS

Figure B-1 shows the CSW/TSW formats in a block multiplexer:

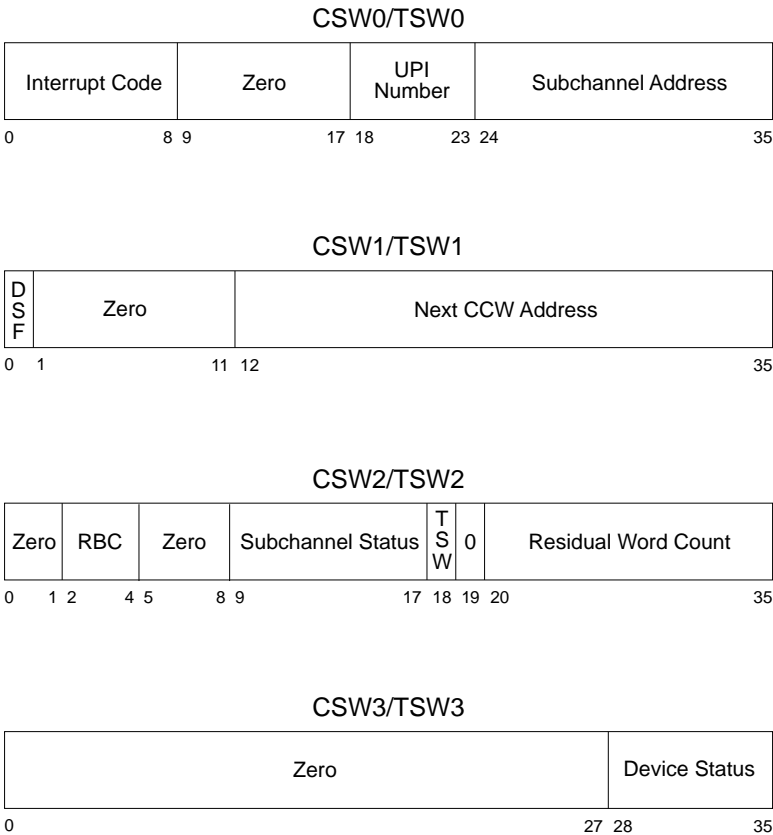


Figure B-1. Block Multiplexer CSW/TSW Formats

Table B-1 shows the CSW/TSW formats.

Table B-1. CSW/TSW Formats

CSW/TSW	Name	Description	
CSW0/TSW0	Interrupt Code (Bits 0-8)	If set to one, specifies that CSW was written in the UPI and an interrupt generated due to the termination of an I/O operation or the presentation of unsolicited device status.	
	UPI Number	Contains the UPI number 60-63 or the I/O processor presenting the interrupt.	
	Subchannel Number (Bits 24-35)	Contains the address of the subchannel presenting status information.	
CSW1/TSW1	DSF (Bit 0)	Implies that the Device Status Field in bits 28-35 of CSW/TSW 3 contains status presented by the device or control unit.	
	Next CCW Address (Bits 12-35)	Contains the address of the next CCW at the time the status information is stored.	
CSW2/TSW2	RBC (Bits 2-4)	Contains the number of bytes not transferred from the last data word during an output operation. Also contains the number of unfilled bytes in the last data word transferred to the MSU during an input operation.	
	Subchannel Status (Bits 9-17)	The subchannel status field bits are described below:	
		Device Not Available (Bit 9)	Indicates that the addressed control unit or device was off-line or powered down.
		Incorrect Length (Bit 10)	Indicates that the number of bytes specified in the CCWs for an I/O operation did not exactly equal the number of bytes requested or offered by the device. The suppress length indication CCW flag disables the reporting of an incorrect length condition.

CSW/TSW	Name	Description
		<p>Program Check (Bit 11)</p> <p>The program check subchannel status bit is set whenever one of the following software errors are detected.</p> <ul style="list-style-type: none"> • The CAW or TIC did not specify a CCW address on a double word boundary. • A CCW specified an invalid command. • A CCW with a command other than TIC specified a word count of 0. • Two successive CCWs specified the TIC command. • A CCW specified a data address that is not available. • A channel program directed the I/O processor to read a CCW from not available storage. • A block multiplexer CCW specified either multiple formats or no format.
		<p>MSU ACK Fault (Bit 12)</p> <p>Indicates one or more of the following:</p> <ul style="list-style-type: none"> • The MSU detected a multiple uncorrectable error. • The I/O processor detected a read data parity. • An internal check was detected by the MSU on the addressed word.
		<p>Unsolicited Status (Bit 13)</p> <p>Indicates that the subchannel mode was idle when the device presented status.</p>
		<p>MSU Interface Fault (Bit 14)</p> <p>Indicates that the fault has been detected on the MSU I/O processor interface.</p>
		<p>Deferred Condition Code (Bit 15)</p> <p>Implies that an SIOF previously reported a condition code of 0 when it was placed in the SIOF stack. When the SIOF was read from the stack, a status condition was detected prior to transferring the first byte of data to the control unit.</p>

CSW/TSW	Name	Description	
		Internal I/O Processor Fault (Bit 16)	Indicates that the operation for the addressed subchannel encountered an internal hardware fault.
		Six Word CSW/TSW (Bit 17)	Indicates that two channel fault words were written along with the four word (CSW/TSW).
	TSW (Bit 18)	Used for TSW only. The TSW toggle bit is set to one for the first pass through the status table and is toggled during each consecutive pass.	
	Residual Word Count (Bits 20-35)	Contains the number of words not transferred from the current CCW.	
	Device Status (Bits 28-35)	Contains a device-generated or control-unit-generated status byte if certain conditions are detected by the device or control unit.	

INDEX

- *CDI CLR Command, 4-3
- *CDI FS Command, 2-2
- *CDI PING Command, 3-10
- *CDI SET Command, 4-3
- *CDI TCP CON Command, 2-4
- *CSC STATUS PATH Command, 2-6

A

Architecture
 CSC/CDI environment, 1-2

C

CDI Aborts on Startup, 3-14
 CDI\$PARAM element not found, 3-15
 initial route installation failure, 3-15

CDI Commands

- *CDI CLR, 4-3
- *CDI FS, 2-2
- *CDI PING, 3-10
- *CDI SET, 4-3
- *CDI TCP CON, 2-4
- *CSC STATUS PATH, 2-6

CDI Error Codes, A-1

CDI Error Notification Displays, 3-13

CDI Trace Flags, 4-3

CDI\$PARAM Element, 3-12, A-7

Channel Device Interface Trace

 channel status word formats, B-1

Channel Status Word Formats, B-1

Checking Your CDI Networking Addressing, 3-12
 CDI\$PARAM configuration, 3-12
 error notification displays, 3-13

Checking Your Hardware Configuration, 3-11

Checking Your OS 2200 Configuration, 3-12

Command Syntax Notation, xi

Common Problems

- CDI aborts on startup, 3-14
- incorrect OS 2200 configuration, 3-14
- no successful TCP connections, 3-15

CPATST Utility, 3-2

- allowing non-streaming data transfers, 3-9
- displaying EEPROM parameters, 3-3
- downloading ELC2 microcode from disk, 3-7
- downloading ELCU microcode from disk, 3-7
- running general tests from menu, 3-5
- setting non-streaming mode data transfer, 3-8
- setting streaming mode data transfer, 3-8

CSC Error Notification Displays, 3-13

D

Displaying EEPROM Parameters, 3-3

Displaying Status Information

- of a network interface, 2-2
- of active TCP connections, 2-4

Documentation

- related, xi
- usage guidelines, x

Downloading Microcode to CPA, 3-2

E

Error Codes

- configuration errors, A-5
- IIACT PLS internal errors, A-5
- internal utility errors, A-5
- main errors, A-2
- summary of internal error messages, A-8
- TCP errors, A-3

Error Notification Displays, 3-13

I

Incorrect OS 2200 Configuration, 3-14

Internal Error Messages

- summary, A-8

Introduction to CDI, 1-1

L

LAN I/O trace, 4-9

LAN Traffic trace, 4-11

Listening to Ethernet Traffic on LAN, 3-2

O

Operations

- monitoring startup process, 1-4

P

Probing a Host on the LAN, 3-10

R

Running CPATST, 3-2

Running General Tests with CPATST, 3-4

S

Setting Correct Channel Type, 3-2

SIGNAL trace, 4-9

Software Configuration Problems

- CDI aborts on startup, 3-14
- no successful TCP connections, 3-15
- OS 2200 configuration, 3-14

ST CDI Command, 1-4

Starting CDI, 1-4

Status Commands, 2-1

- *CDI FS, 2-2
- *CDI TCP CON, 2-4
- *CSC STATUS PATH, 2-6

Syntax

- notation guidelines, xi

T

TCP trace, 4-7

TCPUI trace, 4-5

Trace file example, 4-11

Traces

- CDI trace flags, 4-3
- LAN I/O, 4-9
- LAN Traffic, 4-11
- overview, 4-2
- SIGNAL, 4-9
- TCP, 4-7
- TCPUI, 4-5
- trace file example, 4-11

Troubleshooting Procedures

- checking your CDI network addressing, 3-12
- checking your hardware configuration, 3-11
- checking your OS 2200 configuration, 3-12
- common problems, 3-14
- detecting and isolating problems, 1-4

Troubleshooting Tools

- *CDI PING, 3-10
- CPATST, 3-2

V

Verifying a Destination Host is Reachable, 3-10

Verifying CPA Configuration, 3-11

Verifying CPA is Powered On, 3-11

EFFECTIVE PAGES

Manual:

CDI Troubleshooting Guide 313471701

Issue Date:

July 2001

Reissue Date:

This document has 76 total pages, including:

Title			
Copyrights, Disclaimers & Trademarks			
Contents, Tables & Figures			
Preface	ix	thru	xii
Chapter 1:	1-1	thru	1-4
Chapter 2:	2-1	thru	2-8
Chapter 3:	3-1	thru	3-16
Chapter 4:	4-1	thru	4-14
Appendix A:	A-1	thru	A-12
Appendix B:	B-1	thru	B-6
Index	Index-1	thru	Index-2

Effective Pages

Reader's Comment Form

Business Reply Mailer

