Abstract

This manual is for programmers and describes the control-and-inquiry and Event Management Service (EMS) interfaces available to an application program for communicating with the HP NonStop™ TCP/IP process.

Product Version

NonStop TCP/IP G05

Supported Release Version Updates (RVUs)

This publication supports G05.00 and all subsequent G-series RVUs until otherwise indicated by its replacement publication.

<table>
<thead>
<tr>
<th>Part Number</th>
<th>Published</th>
</tr>
</thead>
<tbody>
<tr>
<td>529636-001</td>
<td>March 2005</td>
</tr>
<tr>
<td>Part Number</td>
<td>Product Version</td>
</tr>
<tr>
<td>-------------</td>
<td>-----------------------</td>
</tr>
<tr>
<td>31424</td>
<td>Tandem TCP/IP C20</td>
</tr>
<tr>
<td>111636</td>
<td>Tandem TCP/IP D30</td>
</tr>
<tr>
<td>114690</td>
<td>Tandem TCP/IP D30</td>
</tr>
<tr>
<td>137203</td>
<td>Tandem NonStop TCP/IP G05</td>
</tr>
<tr>
<td>529636-001</td>
<td>NonStop TCP/IP G05</td>
</tr>
</tbody>
</table>
What's New in This Manual vii
  Manual Information vii
  New and Changed Information viii
About This Manual ix
  Audience ix
  Organization of This Manual ix
  Where to Go for More Information xi
  Notation Conventions xi

1. Introduction
  Brief Overview of TCP/IP 1-1
  Objects Controlled by the Subsystem 1-1
    PROC Object 1-1
    SUBNET Object 1-2
    ROUTE Object 1-2
  Required Management Functions 1-3
  Relationship With Other Subsystems 1-4
    Logical Interface 1-5
    Filters 1-5
    Ethernet 4 ServerNet Adapter (E4SA) and Token Ring ServerNet Adapter (TRSA) 1-6
    Subsystem Architecture 1-6

2. Management Programming
  How a Management Application Fits Into the Architecture 2-1
  Commands 2-3
  Event Messages 2-6
  Communicating With the TCP/IP Subsystem 2-6

3. Elements of SPI Messages
  Definition Files 3-1
  Naming Rules and Guidelines for Applications 3-3
3. Elements of SPI Messages (continued)

Common Syntax Elements for the Subsystem 3-4
   Command Numbers 3-4
   Object Types 3-5
   Object Names 3-5
   Object-Name Templates 3-6
   Event Numbers 3-7
   Subjects of Event Messages 3-7
   Tokens 3-8
   Constructs Involving Multiple Tokens 3-9
   Components of a Command, a Response, or an Event Message 3-10

4. SPI Programming Considerations

   Building the Command Buffer 4-1
   Discontinuing a Command in Progress 4-2
   Receiving and Decoding the Response Buffer 4-2
   Error Handling 4-3
   Security 4-3
   Dynamic Route Switching 4-3
   Relationship to SLSA 4-3
   Relationship to X25AM LINE Object 4-3

5. Common Definitions

   SPI Standard Definitions 5-1
   EMS Standard Definitions 5-7
   Data Communications Standard Definitions 5-9
   Common Definitions for the TCP/IP Subsystem 5-14
      Buffer Declarations 5-15
      TCP/IP Private Token and Field Types 5-15
      TCP/IP Predefined Token and Field Values 5-16
      TCP/IP Simple Tokens 5-18
      TCP/IP Extensible Structured Tokens 5-18
      TCP/IP Tokens in Error Lists 5-19
      TCP/IP Tokens in Event Messages 5-19

6. Commands and Responses

   Commands by Object Type 6-1
   Sensitive and Nonsensitive Commands 6-2
   Command Descriptions 6-4
   ABORT Command 6-5
6. Commands and Responses (continued)

ADD Command 6-8
ALTER Command 6-16
DELETE Command 6-25
GETVERSION Command 6-28
INFO Command 6-31
LISTOBJECTS Command 6-42
LISTOPENS Command 6-45
RESETSTATS Command 6-50
START Command 6-70
STATISTICS Command 6-73
STATUS Command 6-100
STOP Command 6-107
TRACE Command 6-110

7. Event Management

Subsystem Processes That Report Events 7-1
Critical Events 7-2
Event-Management Programming 7-3
Event-Message Descriptions 7-3
1: ZTCI-EVT-NO-MEM 7-4
2: ZTCI-EVT-IOP-READ-ERR 7-6
3: ZTCI-EVT-IOP-WRITE-ERR 7-8
4: ZTCI-EVT-IOP-AWAITIO-ERR 7-10
5: ZTCI-EVT-SUMSTATE-CHG 7-12
6: ZTCI-EVT-INTL 7-14
7: ZTCI-EVT-DUP-IP-ADDR 7-16
8: ZTCI-EVT-BCAST-ENET-ADDR 7-18
9: ZTCI-EVT-MAX-DEV 7-20
10: ZTCI-EVT-CONN-ESTAB 7-22
11: ZTCI-EVT-CONN-TIMEOUT 7-23
12: ZTCI-EVT-DEV-NOT-AVAIL 7-25
13: ZTCI-EVT-SYN-ERR 7-28
16: ZTCI-EVT-CONF-FILE-ACC-ERR 7-29
17: ZTCI-EVT-CONF-FILE-SYN-ERR 7-32
19: ZTCI-EVT-NEW-PROC-ERR 7-34
20: ZTCI-EVT-XFER-ERR 7-36
21: ZTCI-EVT-PARAM-ERR 7-39
22: ZTCI-EVT-ADDR-ERR 7-42
7. Event Management (continued)

23: ZTCI-EVT-SOCK-ERR  7-44
24: ZTCI-EVT-PROTO-ERR  7-46
25: ZTCI-EVT-RECV-OPEN-FAIL  7-48
26: ZTCI-EVT-SSINT-FAILED  7-50
27: ZTCI-EVT-CANT-HANDLE-AF  7-52
28: ZTCI-EVT-OUT-OF-MBUF  7-54
29: ZTCI-EVT-TRACE-STOP-ERR  7-55
30: ZTCI-EVT-SSINIT-FAILED1  7-57
31: ZTCI-EVT-NEW-CL-MBUF  7-59
32: ZTCI-EVT-NON-ZERO-RUN-SOCK  7-60
33: ZTCI-EVT-NO-BACKUP  7-62
34: ZTCI-EVT-INVALID-CPU-NUM  7-64
35: ZTCI-EVT-CANT-START-BACKUP  7-66
36: ZTCI-EVT-BACKUP-IS DEAD  7-68
37: ZTCI-EVT-BACKUP-IS STARTED  7-70
38: ZTCI-EVT-CONN-ESTAB-GATE  7-71
39: ZTCI-EVT-CHKPT-FAILED  7-73
40: ZTCI-EVT-NO-BACKUP-CPU  7-75
41: ZTCI-EVT-IOP-QIO-ERR  7-76
42: ZTCI-EVT-QIOMODE-ON  7-78

A. Error Numbers and Error Lists

<table>
<thead>
<tr>
<th>Notation Used</th>
<th>A-1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Error Descriptions</td>
<td>A-1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>1: ZTCI-ERR-INV-FNAME</th>
<th>A-2</th>
</tr>
</thead>
<tbody>
<tr>
<td>5: ZTCI-ERR-ATTR-OUT-OF-RANGE</td>
<td>A-2</td>
</tr>
<tr>
<td>6: ZTCI-ERR-ENET-ADDR-NOT-AVAIL</td>
<td>A-3</td>
</tr>
<tr>
<td>7: ZTCI-ERR-DUP-ADDR</td>
<td>A-4</td>
</tr>
<tr>
<td>8: ZTCI-ERR-NET-UNREACH</td>
<td>A-5</td>
</tr>
<tr>
<td>10: ZTCI-ERR-SNAP-MTU-NOT-AVAIL</td>
<td>A-6</td>
</tr>
</tbody>
</table>

Glossary

Index
## Figures

- **Figure 1-1.** Relationship of TCP/IP Subsystem to SLSA and X25AM Subsystems 1-5
- **Figure 1-2.** TCP/IP Subsystem Architecture 1-7
- **Figure 2-1.** How an Application Fits Into the TCP/IP Subsystem Architecture 2-3

## Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Table 2-1.</td>
<td>TCP/IP Programmatic Commands</td>
<td>2-4</td>
</tr>
<tr>
<td>Table 2-2.</td>
<td>SCF Commands and Programmatic Commands</td>
<td>2-4</td>
</tr>
<tr>
<td>Table 5-1.</td>
<td>SPI Standard Definitions</td>
<td>5-1</td>
</tr>
<tr>
<td>Table 5-2.</td>
<td>EMS Standard Definitions</td>
<td>5-7</td>
</tr>
<tr>
<td>Table 5-3.</td>
<td>Data Communications Standard Definitions</td>
<td>5-10</td>
</tr>
<tr>
<td>Table 6-1.</td>
<td>Commands and Their Symbolic Names</td>
<td>6-3</td>
</tr>
<tr>
<td>Table 7-1.</td>
<td>TCP/IP Event Messages</td>
<td>7-1</td>
</tr>
</tbody>
</table>
What’s New in This Manual

Manual Information

Abstract

This manual is for programmers and describes the control-and-inquiry and Event Management Service (EMS) interfaces available to an application program for communicating with the HP NonStop™ TCP/IP process.

Product Version

NonStop TCP/IP G05

Supported Release Version Updates (RVUs)

This publication supports G05.00 and all subsequent G-series RVUs until otherwise indicated by its replacement publication.

<table>
<thead>
<tr>
<th>Part Number</th>
<th>Published</th>
</tr>
</thead>
<tbody>
<tr>
<td>529636-001</td>
<td>March 2005</td>
</tr>
</tbody>
</table>

Document History

<table>
<thead>
<tr>
<th>Part Number</th>
<th>Product Version</th>
<th>Published</th>
</tr>
</thead>
<tbody>
<tr>
<td>31424</td>
<td>Tandem TCP/IP C20</td>
<td>September 1990</td>
</tr>
<tr>
<td>111636</td>
<td>Tandem TCP/IP D30</td>
<td>January 1995</td>
</tr>
<tr>
<td>114690</td>
<td>Tandem TCP/IP D30</td>
<td>July 1995</td>
</tr>
<tr>
<td>137203</td>
<td>Tandem NonStop TCP/IP G05</td>
<td>February 1998</td>
</tr>
<tr>
<td>529636-001</td>
<td>NonStop TCP/IP G05</td>
<td>March 2005</td>
</tr>
</tbody>
</table>
New and Changed Information

In Section 5, Common Definitions, deleted the reference to the token ZTCI-DDL-MSG-MAXBUFFER in Buffer Declarations on page 5-15.
About This Manual

The HP NonStop TCP/IP subsystem provides a file-system interface to the TCP, User Datagram Protocol (UDP), and IP layers of the TCP/IP process structure. This manual provides subsystem-specific information for application programmers using the management-programming interfaces to manage the NonStop TCP/IP subsystem. The management-programming interfaces are based on the Subsystem Programmatic Interface (SPI) of Distributed Systems Management (DSM).

Audience

This manual describes the control-and-inquiry interface and the Event Management Service (EMS) interface and serves as a reference manual for the development of management applications. This manual is intended for application programmers in one of these categories:

- Transaction Application Language (TAL) and C programmers who are writing management applications that communicate with the TCP/IP subsystem
- COBOL programmers using ENTER TAL, who are writing management applications that communicate with the TCP/IP subsystem
- HP Tandem Advanced Command Language (TACL) users who are writing macros and routines to perform management functions on the TCP/IP subsystem

Note that, within this manual, whenever references are made to the TCP/IP subsystem, process, or commands, the reference is to the NonStop TCP/IP product.

Organization of This Manual

<table>
<thead>
<tr>
<th>Section</th>
<th>Description (page 1 of 2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section 1, Introduction</td>
<td>Introduces the TCP/IP subsystem and the objects controlled by the subsystem. It also describes the required management functions, the relationship of this subsystem to the ServerNet LAN Systems Access (SLSA) and X.25 Access Method (X25AM) subsystems, and the architecture of the TCP/IP subsystem.</td>
</tr>
<tr>
<td>Section 2, Management Programming</td>
<td>Describes the purpose of management programming and how applications fit into the DSM architecture. It also contains tables of TCP/IP programmatic commands and a comparison of programmatic and operator commands.</td>
</tr>
<tr>
<td>Section 3, Elements of SPI Messages</td>
<td>Describes the elements of SPI messages. It includes discussions of definition files, naming guidelines, and the various types of tokens that make up commands, responses, and event messages.</td>
</tr>
<tr>
<td>Section</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>-------------</td>
</tr>
<tr>
<td><strong>Section 4, SPI Programming Considerations</strong></td>
<td>Describes SPI programming considerations for the TCP/IP subsystem. This description includes information about building the command buffer, canceling commands, receiving and decoding response buffers, error handling, version compatibility, security, dynamic routing, and the relationship of the TCP/IP SUBNET object to the SLSA subsystem and the X25AM LINE object.</td>
</tr>
<tr>
<td><strong>Section 5, Common Definitions</strong></td>
<td>Summarizes standard SPI definitions, EMS definitions, and data communications definitions used by the subsystem in multiple commands, responses, or event messages. These definitions include token codes, token maps, token types, token values, field types, and field values.</td>
</tr>
<tr>
<td><strong>Section 6, Commands and Responses</strong></td>
<td>Describes all the TCP/IP subsystem commands and responses. Each description includes the syntax, object types, command tokens, response tokens, and operational notes associated with the command.</td>
</tr>
<tr>
<td><strong>Section 7, Event Management</strong></td>
<td>Describes event management for applications communicating with the TCP/IP subsystem. This section contains information on how to obtain event messages reported by the subsystem, a description of each event message that can be issued by the subsystem, and examples of filters that could be used by an application.</td>
</tr>
<tr>
<td><strong>Appendix A, Error Numbers and Error Lists</strong></td>
<td>Describes the error numbers and associated error lists defined by the TCP/IP subsystem.</td>
</tr>
<tr>
<td><strong>Glossary</strong></td>
<td>This manual also contains a glossary of technical terms and abbreviations used throughout the text.</td>
</tr>
</tbody>
</table>
Where to Go for More Information

To use this manual, you should be familiar with HP NonStop system architecture and DSM; have a reading knowledge of DDL; and have a working knowledge of the TCP, IP, UDP, Internet Control Message Protocol (ICMP), and the language in which you are working: TAL, C, COBOL, or TACL.

The *TCP/IP Configuration and Management Manual* is considered the introductory manual for all of the NonStop TCP/IP manuals. Before you begin to operate the TCP/IP subsystem, you must accomplish a number of configuration steps as explained in this prerequisite manual.

In addition to providing configuration instructions, the *TCP/IP Configuration and Management Manual* covers management issues related to NonStop TCP/IP software and includes an extensive list of outside references on the TCP/IP protocols.

Notation Conventions

General Syntax Notation

The following list summarizes the notation conventions for syntax presentation in this manual.

**UPPERCASE LETTERS.** Uppercase letters indicate keywords and reserved words; enter these items exactly as shown. Items not enclosed in brackets are required. For example:

```
MAXATTACH
```

**lowercase italic letters.** Lowercase italic letters indicate variable items that you supply. Items not enclosed in brackets are required. For example:

```
file-name
```

[ ] **Brackets.** Brackets enclose optional syntax items. For example:

```
TERM [\system-name.$]terminal-name
```

```
INT[ERRUPTS]
```

A group of items enclosed in brackets is a list from which you can choose one item or none. The items in the list may be arranged either vertically, with aligned brackets on each side of the list, or horizontally, enclosed in a pair of brackets and separated by vertical lines. For example:

```
LIGHTS [ ON ]
[ OFF ]
[ SMOOTH [ num ] ]
```

```
K [ X | D ] address-1
```
{ } **Braces.** A group of items enclosed in braces is a list from which you are required to choose one item. The items in the list may be arranged either vertically, with aligned braces on each side of the list, or horizontally, enclosed in a pair of braces and separated by vertical lines. For example:

```
LISTOPENS PROCESS { $appl-mgr-name } { $process-name }
```

```
ALLOWSU { ON | OFF }
```

| **Vertical Line.** A vertical line separates alternatives in a horizontal list that is enclosed in brackets or braces. For example:

```
INSPECT { OFF | ON | SAVEABEND }
```

| **Ellipsis.** An ellipsis immediately following a pair of brackets or braces indicates that you can repeat the enclosed sequence of syntax items any number of times. For example:

```
M address-1 [ , new-value ]...
```

```
[ - ] {0|1|2|3|4|5|6|7|8|9}...
```

An ellipsis immediately following a single syntax item indicates that you can repeat that syntax item any number of times. For example:

```
"s-char..."
```

**Punctuation.** Parentheses, commas, semicolons, and other symbols not previously described must be entered as shown. For example:

```
error := NEXTFILENAME ( file-name ) ;
```

```
LISTOPENS SU $process-name.#su-name
```

Quotation marks around a symbol such as a bracket or brace indicate the symbol is a required character that you must enter as shown. For example:

```
"[" repetition-constant-list "]"
```

**Item Spacing.** Spaces shown between items are required unless one of the items is a punctuation symbol such as a parenthesis or a comma. For example:

```
CALL STEPMOM ( process-id ) ;
```

If there is no space between two items, spaces are not permitted. In the following example, there are no spaces permitted between the period and any other items:

```
$process-name.#su-name
```

**Line Spacing.** If the syntax of a command is too long to fit on a single line, each continuation line is indented three spaces and is separated from the preceding line by
a blank line. This spacing distinguishes items in a continuation line from items in a vertical list of selections. For example:

```
ALTER [ / OUT file-spec / ] CONTROLLER

    [ , attribute-spec ]...
```

**!i and !o.** In procedure calls, the !i notation follows an input parameter (one that passes data to the called procedure); the !o notation follows an output parameter (one that returns data to the calling program). For example:

```
CALL CHECKRESIZESEGMENT ( segment-id !i
    , error ) ; !o
```

**!i,o.** In procedure calls, the !i,o notation follows an input/output parameter (one that both passes data to the called procedure and returns data to the calling program). For example:

```
error := COMPRESSEDIT ( filenum ) ; !i,o
```

**!i:i.** In procedure calls, the !i:i notation follows an input string parameter that has a corresponding parameter specifying the length of the string in bytes. For example:

```
error := FILENAME_COMPARE_ ( filename1:length !i:i
    , filename2:length ) ; !i:i
```

**!o:i.** In procedure calls, the !o:i notation follows an output buffer parameter that has a corresponding input parameter specifying the maximum length of the output buffer in bytes. For example:

```
error := FILE_GETINFO_ ( filenum !i
    , [ filename:maxlen ] ) ; !o:i
```

### Notation for Messages

The following list summarizes the notation conventions for the presentation of displayed messages in this manual.

**Nonitalic text.** Nonitalic letters, numbers, and punctuation indicate text that is displayed or returned exactly as shown. For example:

```
Backup Up.
```

**Lowercase italic letters.** Lowercase italic letters indicate variable items whose values are displayed or returned. For example:

```
p-register
process-name
```

**[ ] Brackets.** Brackets enclose items that are sometimes, but not always, displayed. For example:

```
Event number = number [ Subject = first-subject-value ]
```
A group of items enclosed in brackets is a list of all possible items that can be displayed, of which one or none might actually be displayed. The items in the list might be arranged either vertically, with aligned brackets on each side of the list, or horizontally, enclosed in a pair of brackets and separated by vertical lines. For example:

```
LDEV ldev [ CU %ccu | CU %... ] UP [ (cpu, chan, %ctlr, %unit) ]
```

{} Braces. A group of items enclosed in braces is a list of all possible items that can be displayed, of which one is actually displayed. The items in the list might be arranged either vertically, with aligned braces on each side of the list, or horizontally, enclosed in a pair of braces and separated by vertical lines. For example:

```
LBU { X | Y } POWER FAIL

process-name State changed from old-objstate to objstate
{ Operator Request. }
{ Unknown.     }
```

| Vertical Line. A vertical line separates alternatives in a horizontal list that is enclosed in brackets or braces. For example: |

```
Transfer status: { OK | Failed }
```

% Percent Sign. A percent sign precedes a number that is not in decimal notation. The % notation precedes an octal number. The % notation precedes a binary number. The %H notation precedes a hexadecimal number. For example:

```
%005400

P=%p-register E=%e-register
```

Notation for Management Programming Interfaces

UPPERCASE LETTERS. Uppercase letters indicate names from definition files; enter these names exactly as shown. For example:

```
ZCOM-TKN-SUBJ-SERV
```

lowercase letters. Words in lowercase letters are words that are part of the notation, including Data Definition Language (DDL) keywords. For example:

```
token-type
```

!r. The !r notation following a token or field name indicates that the token or field is required. For example:

```
ZCOM-TKN-OBJNAME token-type ZSPI-TYP-STRING. !r
```

!o. The !o notation following a token or field name indicates that the token or field is optional. For example:

```
ZSPI-TKN-MANAGER token-type ZSPI-TYP-FNAME32. !o
```
Change Bar Notation

Change bars are used to indicate substantive differences between this edition of the manual and the preceding edition. Change bars are vertical rules placed in the right margin of changed portions of text, figures, tables, examples, and so on. Change bars highlight new or revised information. For example:

The message types specified in the REPORT clause are different in the COBOL85 environment and the Common Run-Time Environment (CRE).

The CRE has many new message types and some new message type codes for old message types. In the CRE, the message type SYSTEM includes all messages except LOGICAL-CLOSE and LOGICAL-OPEN.
1 Introduction

This section introduces the TCP/IP subsystem and describes management concepts applicable to the TCP/IP subsystem. Some of the concepts discussed are the types of objects controlled by the TCP/IP subsystem and the monitoring and control functions available.

Brief Overview of TCP/IP

The TCP/IP protocols are a family of data communications protocols that allow communication between heterogeneous systems in a multinetwork environment known as the Internet. In the NonStop environment, this allows for communication between UNIX and NonStop systems.

The NonStop TCP/IP subsystem actually consists of a variety of products in the TCP/IP protocol family and provides services at the Network through Application Layers of the OSI Reference Model. The NonStop TCP/IP subsystem is the base subsystem for all the other components of the NonStop TCP/IP software. It provides a file-system interface to the TCP, UDP (User Datagram Protocol), and IP.

The TCP/IP subsystem runs as a single or dual process on the NonStop operating system and contains three I/O interfaces: an Ethernet interface, a Sub-network Access Protocol (SNAP) interface, and an X.25 interface. The Ethernet interface uses the ServerNet LAN Systems Access (SLSA) subsystem to provide access to Ethernet local area networks (LANs) using the Digital-Intel-Xerox (DIX) protocol. The SNAP interface uses the SLSA 802.2 LLC1 filter to provide access to Ethernet LANs using the IEEE 802.3 CSMA/CD protocol and token ring LANs using IEEE 802.5. The X.25 interface uses the HP X.25 Access Method (X25AM) to provide access to the Defense Data Network (DDN), X.25-based public data networks (PDNs), and other X.25-based networks. This X.25 interface is used to create virtual circuits to remote hosts, so that IP datagrams can be sent over these virtual circuits.

The TCP/IP subsystem does not currently run as a Nonstop process pair.

For further details about the components of the NonStop TCP/IP software and the services they provide, see the TCP/IP Configuration and Management Manual.

Objects Controlled by the Subsystem

You can manage the TCP/IP subsystem programmatically or interactively by sending commands that act on one or more DSM-related objects. The TCP/IP subsystem defines three types of objects: processes, subnets, and routes.

PROC Object

The process (PROC) object is the TCP/IP interface, which is accessed by an application program or the Subsystem Control Facility (SCF) by means of the
Subsystem Control Point (SCP). The PROC object must be a NonStop Kernel named process.

A NonStop system can support multiple TCP/IP processes. Each process has a unique IP address and is handled as a separate subsystem. Typically, a NonStop system attached to a TCP/IP network has only one TCP/IP process. However, when a NonStop system that contains multiple processes is attached to an IP network, each of the processes must appear as a separate host with a unique address on the IP network.

System configuration is not required for the TCP/IP process. Create the process with the RUN command.

An example of a command for creating a TCP/IP process:

```
1>RUN $SYSTEM.SYS03.TCPIP/NAME $ZTC0,NOWAIT,CPU 0/
```

Once the process is created, you can add subnets and routes. Adding routes is optional because the TCP/IP subsystem automatically adds a route when you add a subnet with the ADD command.

SUBNET Object

The SUBNET object defines the IP layer attachment point to either the SLSA subsystem or the X.25 I/O process (X25AM). It is through the SUBNET object that data can be sent to and received from either the Ethernet (DIX) LAN, IEEE 802.3 Ethernet LAN, IEEE 802.5 token ring LAN, or the X.25 network. For Ethernet and SNAP subnets, the SUBNET object in the TCP/IP subsystem is associated with a logical interface (LIF) in SLSA. For X.25 subnets, the SUBNET object in the TCP/IP subsystem is associated with a LINE object in X25AM. Subdevices on this X25AM LINE object are opened by the TCP/IP process.

You can add a maximum of 17 subnets to each TCP/IP process: 4 Ethernet subnets, 8 SNAP subnets (4 token ring and 4 Ethernet), 4 X.25 subnets, and 1 loopback driver. When adding subnets, follow the DSM naming conventions.

Note. For NonStop availability considerations, you must carefully plan before adding multiple subnets to a single TCP/IP process. For detailed information about NonStop availability considerations in this release, see the TCP/IP Configuration and Management Manual.

ROUTE Object

The ROUTE object specifies the path used to transmit IP datagrams between the TCP/IP subsystem and the destination. The route specifies both the destination and the next gateway or host to which a datagram must be sent to reach the destination.

The IP Layer at each host maintains a table of routes. The table lists the internetwork address of each accessible network or accessible host on a network. In addition, the table lists the gateway to be used to reach that network or host. A host’s routing table often specifies a default gateway to which datagrams can be sent if the routing table does not contain an entry for a particular network. If the default gateway is used, the
gateway that contains a more comprehensive routing table, determines the best path for the datagram.

A subnet has at least one route associated with it, but you can add as many routes as resources allow. In addition, routes can be dynamically redirected by the TCP/IP subsystem. For instance, when a SUBNET object enters the STOPPED summary state, any routes associated with that subnet are automatically redirected by the subsystem to another available subnet. When adding a ROUTE object, follow the DSM naming conventions. For more details on object names, see Object Names on page 3-5.

Required Management Functions

Management of a subsystem involves monitoring and controlling the objects defined in the subsystem, and monitoring and handling events occurring in the subsystem.

The ability to monitor the TCP/IP subsystem is significant. With the information made available to you through monitoring, you can make decisions needed to maintain a productive computing environment.

Monitoring the TCP/IP subsystem includes tasks such as:

- Obtaining the version level of the subsystem software
- Obtaining current attribute values for all the objects
- Obtaining the names of subnets and routes known to the subsystem
- Obtaining the current values of the statistics counters for all the objects
- Obtaining information about the current states of all the objects
- Obtaining information about the openers of the process
- Retrieving information about events occurring in the subsystem

Once you have gained this information, you can make informed decisions to control your subsystem. You can take action either programmatically or interactively using the available commands.

Controlling your subsystem includes tasks such as:

- Terminating the operation of subnets and routes
- Adding subnets and routes to the subsystem
- Changing certain attributes of a process or subnet
- Removing subnets and routes from the subsystem
- Resetting the statistics counters to the subsystem default values
- Initiating the operation of subnets and routes
- Tracing the activity of a process or subnet
Relationship With Other Subsystems

The TCP/IP subsystem provides a file-system interface to the TCP, UDP, and IP of the TCP/IP family. TCP or UDP provides services at the Transport Layer (Layer 4 in the OSI model), and IP provides services at the Network Layer (Layer 3). The TCP/IP subsystem relies on the SLSA subsystem for the Link and Physical Layers.

The SLSA subsystem provides the Link Layer (Layer 2) and Physical Layer (Layer 1) for the NonStop TCP/IP subsystem.

Figure 1-1 on page 1-5 shows the relationship between the NonStop TCP/IP and SLSA subsystems.

Note that Figure 1-1 on page 1-5 also includes the relationship between the TCP/IP subsystem and the X25AM subsystem. SLSA is used indirectly by SCF or an SPI application to add the #IP, #ARP, and #SNAP filters. X25AM is used to add the #TCPIP1 through #TCPIP32 subdevices on an X.25 line. It is through these subdevices that the TCP/IP subsystem can interface with an X.25 network. For further details on the relationships between these subsystems, see the TCP/IP Configuration and Management Manual.

The TCP/IP process interfaces to the Ethernet adapters (E4SAs, FESAs, and G4SAs) and token ring ServerNet adapters (TRSAs) through the SLSA subsystem and to the SLSA subsystem driver/interrupt handlers (DIHs) through the ServerNet fabrics of the system. As shown in Figure 1-1 on page 1-5, the SLSA subsystem provides logical interfaces (LIFs) through which packets pass to filters. Packets pass through the rest of the SLSA subsystem, which consists of ServerNet Addressable Adapters (SACs) and physical interfaces (PIFs), and then out to a LAN, WAN, or X.25 connection. A brief description of the SLSA components relevant to this discussion follows Figure 1-1 on page 1-5.

Note. Token ring is not supported in this release.
Logical Interface

The logical interface (LIF) is an object within the SLSA subsystem that provides an interface between the data filters and the PIF of an E4SA or a TRSA.

Filters

Filters provide a logical mechanism whereby frames received from the LAN can be sorted and delivered to a particular client such as NonStop TCP/IP. Filters replace the
PORT objects used in systems prior to the ServerNet architecture in the sense that filters are the final destination for data received from the LAN. The NonStop TCP/IP process in NonStop S-series systems registers ARP and IP filters through a LIF.

**Ethernet 4 ServerNet Adapter (E4SA) and Token Ring ServerNet Adapter (TRSA)**

The E4SA and TRSA are Customer replaceable units (CRUs) installed in an enclosure of a NonStop S-series system. An E4SA or TRSA contains one or more ServerNet addressable controllers (SACs) and two physical interfaces (PIFs).

**ServerNet Addressable Controller (SAC)**

A SAC is the ServerNet addressable entity on an adapter that allows a NonStop S-series system to communicate to a specific adapter across the X and Y fabrics of a ServerNet system area network (SAN) connected to a NonStop S-series system. A SAC can support up to two PIFs.

**Physical Interface (PIF)**

The PIF corresponds to the physical connection to the LAN, which is the industry-standard Ethernet port on an E4SA or token ring port on a TRSA.

**Subsystem Architecture**

The TCP/IP subsystem consists of a process controlling as many as 13 subnets. Each subnet can have several routes associated with it. The number of routes that can be added depends on system resources.

You can manage the TCP/IP subsystem through both interactive and programmatic interfaces.

The interactive interface for managing the TCP/IP subsystem actually consists of two interfaces:

- The Subsystem Control Facility (SCF), which can be used to monitor, configure, and control the subsystem. SCF provides an operator interface to the Subsystem Control Point (SCP) that provides an interface to the TCP/IP subsystem.
- The ViewPoint application, which can be used to retrieve event messages (previously sent to EMS by the subsystem) interactively.

For further details on the interactive interfaces, see the *TCP/IP Configuration and Management Manual* and the *ViewPoint Manual*.

The programmatic interface for managing the TCP/IP subsystem also consists of two interfaces:

- The control-and-inquiry interface, which provides control and inquiry capability to an application program. This interface (which is also provided by the SCP) allows...
an application program to send commands to the subsystem to control the subsystem or to obtain information.

- The Event Management Service (EMS) interface, which provides event-management capability to an application program. This interface allows an application program to retrieve event messages previously sent to EMS by the subsystem.

Figure 1-2 illustrates the architecture of the TCP/IP subsystem and the various interfaces to the subsystem.

**Figure 1-2. TCP/IP Subsystem Architecture**

Both the programmatic and interactive management interfaces use SPI message formats and conventions. To select the interfaces that will best serve your management needs, examine the features and capabilities of each and the environment in which it will be used.
You can manage the TCP/IP subsystem interactively or programmatically. For the TCP/IP subsystem, the SCF and ViewPoint provide interactive subsystem management, while the control-and-inquiry and Event Management Service (EMS) interfaces provide programmatic subsystem management.

The division of management tasks between operator control and programmatic control depends upon the subsystem configuration environment and your needs as a subsystem user.

To determine the most effective division of management tasks, you must be familiar with the functions available in both the interactive and programmatic interfaces.

Managing a subsystem programmatically can mean more efficient use of resources, like:

- If routine management tasks are performed programmatically instead of interactively, the operator can spend more time on tasks that require more judgement and creativity.
- Where appropriate, a management application can standardize responses to errors or other information obtained from the subsystem. This standardization allows you to establish procedures without having to train several operators about the procedures.
- A single management application can communicate with several subsystems.
- You have more latitude in responding to errors returned by the subsystem. A management application can be designed to take remedial action on selected reported errors instead of requiring operator intervention once the errors are reported.
- You can customize a management application to respond to the specific needs of your computer system or network.

How a Management Application Fits Into the Architecture

A management application uses SPI messages to communicate with the TCP/IP subsystem. These messages are passed between the management application and the TCP/IP subsystem by means of the SCP or the EMS interface. The function of an SPI message determines whether the message is passed through SCP or EMS. SCP is a management process that provides the control-and-inquiry interface. EMS is a collection of processes, tools, and interfaces that supports the reporting and retrieval of event information.

The control-and-inquiry interface differs from the EMS interface in key areas. Firstly, the control-and-inquiry interface requires a management application to initiate the communication with the subsystem by sending a command through SCP to the
subsystem. The subsystem then returns the appropriate response to the application through SCP. This means that the exchange of a command and response between your application and the subsystem is bidirectional and occurs synchronously. The commands available to you for managing the TCP/IP subsystem are listed in Table 2-1 on page 2-4.

The EMS interface requires the subsystem to initiate the communication. The subsystem reports event information when events happen, not in response to a request from a management application. The communication between EMS and your management application is unidirectional and occurs asynchronously. Using EMS, a management application can monitor events that occur in the TCP/IP subsystem, enabling the application to handle problems occurring in the subsystem or to track the current status of the subsystem. An application can then retrieve the event messages from the consumer distributor process. If you want your application to retrieve only specific information, you can write a filter. Figure 2-1 on page 2-3 illustrates how a management application fits into the TCP/IP subsystem architecture.
Figure 2-1. How an Application Fits Into the TCP/IP Subsystem Architecture

Legend
* Note: TCP in this instance indicates the ViewPoint Terminal Control Process.

Commands

Table 2-1 on page 2-4 lists and briefly describes the commands that a management application can send to the TCP/IP subsystem using the control-and-inquiry interface. For detailed descriptions of these commands, see Section 6, Commands and Responses. For a description of these commands as they relate to all data communications subsystems, or for information about using SPI messages to
communicate with data communications subsystems in general, see the *SPI Common Extensions Manual*.

### Table 2-1. TCP/IP Programmatic Commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABORT</td>
<td>Terminates the operation of subnets or routes immediately</td>
</tr>
<tr>
<td>ADD</td>
<td>Adds a subnet or route to the TCP/IP subsystem</td>
</tr>
<tr>
<td>ALTER</td>
<td>Changes the values of the attributes of a process or subnet</td>
</tr>
<tr>
<td>DELETE</td>
<td>Removes subnets or routes from the TCP/IP subsystem</td>
</tr>
<tr>
<td>GETVERSION</td>
<td>Returns the version level and version banner for the TCP/IP subsystem</td>
</tr>
<tr>
<td>INFO</td>
<td>Returns the current values of the attributes of processes, subnets, or routes</td>
</tr>
<tr>
<td>LISTOBJECTS</td>
<td>Returns the names of subnets and routes known to the TCP/IP subsystem</td>
</tr>
<tr>
<td>LISTOPENS</td>
<td>Returns information about the openers of the TCP/IP process</td>
</tr>
<tr>
<td>RESETSTATS</td>
<td>Returns statistics about the process, subnets, and routes, and then resets the statistics counters to the subsystem default values</td>
</tr>
<tr>
<td>START</td>
<td>Initiates the operation of subnets and routes</td>
</tr>
<tr>
<td>STATISTICS</td>
<td>Returns statistics about the process, subnets, and routes</td>
</tr>
<tr>
<td>STATUS</td>
<td>Returns information about the current state of the process, subnets, and routes</td>
</tr>
<tr>
<td>STOP</td>
<td>Terminates the operation of subnets and routes</td>
</tr>
<tr>
<td>TRACE</td>
<td>Traces the activity of a process or subnet</td>
</tr>
</tbody>
</table>

**Table 2-2** shows the correspondence between the SCF commands an operator issues to control and monitor the TCP/IP subsystem, and the programmatic commands a management application sends to the subsystem. For clarity, the table contains the command and object names in the form that you must use in applications. However, later in this manual, for example, in the text and headings for the command descriptions, the commands and object types appear in a shortened form (that is, without the ZCOM-CMD- and ZCOM-OBJ- prefixes).

### Table 2-2. SCF Commands and Programmatic Commands (page 1 of 2)

<table>
<thead>
<tr>
<th>SCF Command</th>
<th>Object Type</th>
<th>Programmatic Command</th>
<th>Object Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABORT</td>
<td>SUBNET</td>
<td>ZCOM-CMD-ABORT</td>
<td>ZCOM-OBJ-SUBNET</td>
</tr>
<tr>
<td></td>
<td>ROUTE</td>
<td>ZCOM-OBJ-ROUTE</td>
<td></td>
</tr>
<tr>
<td>ADD</td>
<td>SUBNET</td>
<td>ZCOM-CMD-ADD</td>
<td>ZCOM-OBJ-SUBNET</td>
</tr>
<tr>
<td></td>
<td>ROUTE</td>
<td>ZCOM-OBJ-ROUTE</td>
<td></td>
</tr>
<tr>
<td>ALTER</td>
<td>PROCESS</td>
<td>ZCOM-CMD-ALTER</td>
<td>ZCOM-OBJ-PROC</td>
</tr>
<tr>
<td></td>
<td>SUBNET</td>
<td>ZCOM-OBJ-SUBNET</td>
<td></td>
</tr>
<tr>
<td>DELETE</td>
<td>SUBNET</td>
<td>ZCOM-CMD-DELETE</td>
<td>ZCOM-OBJ-SUBNET</td>
</tr>
<tr>
<td>SCF Command</td>
<td>Object Type</td>
<td>Programmatic Command</td>
<td>Object Type</td>
</tr>
<tr>
<td>-------------</td>
<td>-------------</td>
<td>----------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>INFO</td>
<td>PROCESS</td>
<td>ZCOM-CMD-INFO</td>
<td>ZCOM-OBJ-PROC</td>
</tr>
<tr>
<td></td>
<td>SUBNET</td>
<td></td>
<td>ZCOM-OBJ-SUBNET</td>
</tr>
<tr>
<td></td>
<td>ROUTE</td>
<td></td>
<td>ZCOM-OBJ-ROUTE</td>
</tr>
<tr>
<td>LISTOPENS</td>
<td>PROCESS</td>
<td>ZCOM-CMD-LISTOPENS</td>
<td>ZCOM-OBJ-PROC</td>
</tr>
<tr>
<td>NAMES</td>
<td>SUBNET</td>
<td>ZCOM-CMD-LISTOBJECTS</td>
<td>ZCOM-OBJ-SUBNET</td>
</tr>
<tr>
<td></td>
<td>ROUTE</td>
<td></td>
<td>ZCOM-OBJ-ROUTE</td>
</tr>
<tr>
<td>RESETSTATS</td>
<td>PROCESS</td>
<td>ZCOM-CMD-RESETSTATS</td>
<td>ZCOM-OBJ-PROC</td>
</tr>
<tr>
<td></td>
<td>SUBNET</td>
<td></td>
<td>ZCOM-OBJ-SUBNET</td>
</tr>
<tr>
<td></td>
<td>ROUTE</td>
<td></td>
<td>ZCOM-OBJ-ROUTE</td>
</tr>
<tr>
<td>START</td>
<td>SUBNET</td>
<td>ZCOM-CMD-START</td>
<td>ZCOM-OBJ-SUBNET</td>
</tr>
<tr>
<td></td>
<td>ROUTE</td>
<td></td>
<td>ZCOM-OBJ-ROUTE</td>
</tr>
<tr>
<td>STATS</td>
<td>PROCESS</td>
<td>ZCOM-CMD-STATISTICS</td>
<td>ZCOM-OBJ-PROC</td>
</tr>
<tr>
<td></td>
<td>SUBNET</td>
<td></td>
<td>ZCOM-OBJ-SUBNET</td>
</tr>
<tr>
<td></td>
<td>ROUTE</td>
<td></td>
<td>ZCOM-OBJ-ROUTE</td>
</tr>
<tr>
<td>STATUS</td>
<td>PROCESS</td>
<td>ZCOM-CMD-STATUS</td>
<td>ZCOM-OBJ-PROC</td>
</tr>
<tr>
<td></td>
<td>SUBNET</td>
<td></td>
<td>ZCOM-OBJ-SUBNET</td>
</tr>
<tr>
<td></td>
<td>ROUTE</td>
<td></td>
<td>ZCOM-OBJ-ROUTE</td>
</tr>
<tr>
<td>STOP</td>
<td>SUBNET</td>
<td>ZCOM-CMD-STOP</td>
<td>ZCOM-OBJ-SUBNET</td>
</tr>
<tr>
<td></td>
<td>ROUTE</td>
<td></td>
<td>ZCOM-OBJ-ROUTE</td>
</tr>
<tr>
<td>TRACE</td>
<td>PROCESS</td>
<td>ZCOM-CMD-TRACE</td>
<td>ZCOM-OBJ-PROC</td>
</tr>
<tr>
<td></td>
<td>SUBNET</td>
<td></td>
<td>ZCOM-OBJ-SUBNET</td>
</tr>
<tr>
<td>VERSION</td>
<td>PROCESS</td>
<td>ZCOM-CMD-GETVERSION</td>
<td>ZCOM-OBJ-PROC</td>
</tr>
</tbody>
</table>

For detailed descriptions of the NonStop TCP/IP SCF commands, see the *TCP/IP Configuration and Management Manual*. For a detailed description of SCF and general descriptions of the SCF commands, see the *SCF Reference Manual for G-Series RVUs*. 
Event Messages

An event message describes an anomaly that has occurred in a subsystem, such as a hardware or software error. All event messages are in an SPI message format.

All event messages issued by the TCP/IP subsystem are sent to the EMS collector process ($0). These event messages are then placed into the user-specified log files. Once placed into the log files, the EMS consumer distributor process can read the event messages and retrieve the selected events for the management application.

A management application retrieves event messages by sending commands in an SPI message format to an EMS consumer distributor. To select event messages, you can write a filter that is specific to your needs. If no filter is used, all event messages in the log file are passed to the management application by the consumer distributor.

Figure 2-1 on page 2-3 illustrates the flow of data between the TCP/IP subsystem and a management application in the EMS environment. Note that event messages can also be retrieved interactively through the ViewPoint console application. For further details on the ViewPoint application, see the ViewPoint Manual.

For detailed descriptions of event messages and event management as they relate to the TCP/IP subsystem, see Section 7, Event Management. For a description of event messages and event management in general, see the EMS Manual.

Communicating With the TCP/IP Subsystem

A management application communicates with the TCP/IP subsystem by means of the SCP or the EMS consumer distributor process. When communicating with the SCP or the EMS, an application calls SPI procedures to build messages to be sent to the subsystem and to retrieve messages sent from the subsystem. However, to send and receive SPI messages, and to establish and terminate communication with the SCP or the EMS distributor process, an application must use the NonStop operating system.

How you use the file system depends on the language in which your application is written. If your application is written in TAL or C, you use the file-system procedures directly to send and receive the SPI messages. If your application is written in COBOL (COBOL 74 does not support SPI), you use the file-system procedures indirectly by invoking them through COBOL85 verbs. If your application is written in TACL, you use the file-system procedures indirectly by invoking them through TACL built-ins.

Regardless of the language you are using, the tasks an application must perform when communicating with either the SCP or the EMS distributor process are:

1. Establish communication with the SCP or the EMS distributor process.
2. Call SPI procedures to build a message.
3. Send the message to the SCP or the EMS distributor process.
4. Receive the response.
5. Check for error codes and handle any errors.

6. Decode the response.

7. Take appropriate action and return to Step 2, if necessary.

8. Terminate communication with the SCP or the EMS distributor process.

All data communications subsystems must use the same message protocol for communicating with SCP. For details on this protocol, see the *SPI Common Extensions Manual* and the *SPI Programming Manual*. For more information about communicating with the EMS distributor process, see the *EMS Manual*. 
3 Elements of SPI Messages

The SPI procedures facilitate communication between a management application and the TCP/IP subsystem. These procedures allow a management application to build commands in an SPI message format to be sent through SCP to the subsystem. These procedures also allow a management application to decode the responses from the subsystem that have been sent through SCP in an SPI message format. However, NonStop OS procedures, are used to transport SPI-formatted commands and responses between a management application and SCP, and between SCP and a subsystem. For details on message transport, see the SPI Programming Manual.

Note. SCP is responsible for the communication between SCP and the TCP/IP subsystem. Your management application needs to be concerned only with the steps enumerated in Section 2, Management Programming.

SPI procedures are also used to obtain complete event messages from the EMS consumer distributor process. However, the EMS procedure EMSGET is used to extract tokens from event messages, and the EMS procedure EMSTEXT is used to obtain text versions of event messages. Like commands and responses sent through SCP, event messages are sent and received in an SPI message format.

The commands sent to the subsystem and the responses and event messages received from the subsystem are made up of tokens. A token can be a single value or a structure made up of several values. The concept of tokens is described in the SPI Programming Manual. For an explanation of the tokens present in multiple commands, responses, or event messages for the TCP/IP subsystem, see Section 5, Common Definitions.

This manual does not attempt to give a complete explanation of tokens; it provides subsystem-specific information about the tokens used to communicate with the TCP/IP subsystem. General information about tokens can be found in the SPI Programming Manual, and information about tokens common to all data communications subsystems can be found in the SPI Common Extensions Manual.

Definition Files

Definition files supplied by HP provide declarations of commonly needed tokens and other variables, in addition to structures and values. The declaration names in these files have a standard form that identifies the definition file they come from and what the declaration defines. In addition, each source of definitions, such as the TCP/IP subsystem or SPI, has associated with it a set of five definition files: one in TAL, one in COBOL, one in TACL, one in C, and one in DDL. The definition files in TAL, COBOL, C, and TACL are derived from the definition file in DDL.
To include the definition files in your management application, use the mechanism that is appropriate for the programming language in which the application is written. The mechanisms are:

- In TAL, use the $SOURCE compiler directive to include the definition files in your program.
- In C, use the #INCLUDE compiler directive to include the definition files in your program.
- In COBOL, use COPY statements to copy the definition files into your program.
- In TACL, use the #LOAD commands to load the definition files into your program.

Note that the definitions in a COBOL definition file are grouped into sections to enable COBOL programmers to declare multiple copies of structures in the definition file, whereas, when programming in TAL, C, or TACL, the entire definition file is always included or loaded. For further information on how a management application accesses definition files, see the *SPI Programming Manual*.

A management application that communicates with the TCP/IP subsystem must include these definition files:

- The SPI definition file, ZSPIDEF.ZSPI.xxx. Note that in this and each of the definition filenames that follow, xxx represents one of five specific abbreviations for COBOL, TAL, TACL, C, or DDL and corresponds to the programming language in which the management application is written.
- The EMS definition file, ZSPIDEF.ZEMS.xxx. This definition file is needed only if the application retrieves event messages or controls the EMS.
- The data communications definition file, ZSPIDEF.ZCOM.xxx.
- The TCP/IP definition file, ZSPIDEF.ZTCI.xxx.
- If the application communicates with other subsystems, the definition file ZSPIDEF.Z_subsys (where subsys represents a three-character subsystem identifier) for each of the other subsystems.

An example of the definition files needed by a TCP/IP application written in TAL that retrieves event messages is:

```
ZSPIDEF.ZSPITAL       SPI definitions
ZSPIDEF.ZEMSTAL        EMS definitions
ZSPIDEF.ZCOMTAL        Data communications definitions
ZSPIDEF.ZTCITAL        TCP/IP definitions
```
An example of the definition files needed by a TCP/IP application written in C that retrieves event messages is:

- ZSPIDEF.ZSPIC: SPI definitions
- ZSPIDEF.ZEMSC: EMS definitions
- ZSPIDEF.ZCOMC: Data communications definitions
- ZSPIDEF.ZTCIC: TCP/IP definitions

An example of the definition files needed by a TCP/IP application written in COBOL that retrieves event messages is:

- ZSPIDEF.ZSPICOB: SPI definitions
- ZSPIDEF.ZEMSCOB: EMS definitions
- ZSPIDEF.ZCOMCOB: Data communications definitions
- ZSPIDEF.ZTCICOB: TCP/IP definitions

An example of the definition files needed by a TCP/IP application written in TACL that retrieves event messages is:

- ZSPIDEF.ZSPITACL: SPI definitions
- ZSPIDEF.ZEMSTACL: EMS definitions
- ZSPIDEF.ZCOMTACL: Data communications definitions
- ZSPIDEF.ZTCITACL: TCP/IP definitions

These definition files are located on the disk volume chosen by your site. The default disk volume used by the INSTALL program is $SYSTEM.

Section 5, Common Definitions, lists the SPI, EMS, and data communications standard definitions used by the TCP/IP subsystem. For detailed descriptions of these standard definitions, see the SPI Programming Manual, the EMS Manual, and the SPI Common Extensions Manual, respectively. The declarations in the TCP/IP definition files are described in Section 5, Common Definitions.

Note. This manual uses DDL to describe all tokens. For a brief explanation of DDL, refer to the SPI Programming Manual. If you are programming in TAL or TACL, substitute a circumflex (^) for each hyphen in a definition name as it appears in this manual. If you are programming in C, substitute an underscore (_) for each hyphen in a definition name as it appears in this manual. For example, ZCOM-OBJ-SUBNET becomes ZCOM^OBJ^SUBNET in TAL and TACL and ZCOM_OBJ_SUBNET in C.

Naming Rules and Guidelines for Applications

All items in definition files are assigned symbolic names in addition to numbers. Use the symbolic names to prevent transcription errors and to improve readability. HP uses names beginning with the letter Z for all definitions and all component fields of structures in its definition files. To avoid present and future conflicts with names defined...
by HP, do not begin any name you define in your application with uppercase or lowercase Z.

Common Syntax Elements for the Subsystem

The remainder of this section contains subsystem-specific information about syntax elements of commands, responses, and event messages. These elements are in token form and are referred to by their symbolic names. An advantage of using the symbolic name is that you are not required to know the address or position of the element within the buffer.

For more complete information about these elements as they relate to data communications subsystems, see the SPI Common Extensions Manual.

Command Numbers

Command numbers specify a command to the TCP/IP subsystem or a response to a management application. The management application uses a symbolic name to express the command number to the TCP/IP subsystem, and the subsystem uses the symbolic name to identify the corresponding response. The symbolic name is of the form ZCOM-CMD-name, where name identifies the command. For example, ZCOM-CMD-ADD identifies the ADD command in all data communications subsystems, including the TCP/IP subsystem.

The command number specified by the management application is stored in the header of the command message. The subsystem and management application can read the command number from the header using the token ZSPI-TKN-COMMAND in the SSGET SPI procedure.

Management applications should not use the token ZSPI-TKN-COMMAND to place the command number into the message header, since the command number to be specified is already known. Instead, to place the command number into the message header, your management application should use the command's symbolic name, ZCOM-CMD-name (which is the value of the token ZSPI-TKN-COMMAND), in the SSPUT SPI procedure.

Although all data communications subsystems must use symbolic names of the form ZCOM-CMD-name, each subsystem defines which commands it supports. The TCP/IP subsystem supports these commands:

- ABORT
- INFO
- STATISTICS
- ADD
- LISTOBJECTS
- STATUS
- ALTER
- LISTOPENS
- STOP
- DELETE
- RESETSTATS
- TRACE
- GETVERSION
- START

These commands are described in Section 6, Commands and Responses.
Object Types

Object-type numbers represent the object type (or class) of an object. The value of an object-type number is subsystem-specific and has a symbolic name. The symbolic name is of the form ZCOM-OBJ-\textit{type}, where \textit{type} identifies the object type. For example, ZCOM-OBJ-SUBNET identifies the subnet object type in all data communications subsystems, including the TCP/IP subsystem.

Note that object-type numbers are part of the header of each command and response. The header token ZSPI-TKN-OBJECT-TYPE contains the object-type number. However, if there are multiple objects in a response or an object is returned within a data list, the object-type number is contained in the ZCOM-TKN-OBJTYPE token. The symbolic name of the value is also of the form ZCOM-OBJ-\textit{type}, where \textit{type} identifies the object type.

The TCP/IP subsystem supports these object types:

- NULL
- PROC
- ROUTE
- SUBNET

The objects defined by the TCP/IP subsystem are organized in a hierarchy by object type. The SUBNET and ROUTE object types are peer object types, subordinate to the PROC object type. The NULL object type is not within the hierarchy and applies to only two commands: GETVERSION and LISTOBJECTS. The hierarchy is important when issuing commands to the TCP/IP subsystem for processing. For example, because the SUBNET and ROUTE object types are subordinate to the PROC object type, any commands pertaining to a SUBNET or ROUTE object can be issued only when the PROC object is in the STARTED summary state.

Object Names

An object name identifies a specific object. The object specified by an object name for the TCP/IP subsystem must be one of three object types, PROC, SUBNET, or ROUTE. The NULL object type does not have an object name associated with it.

As with any NonStop subsystem, the TCP/IP subsystem must have a NonStop OS process name. When you assign an object name to a process, you must assign a unique name. The name must be preceded by a dollar sign ($) and can have a maximum of four alphanumeric characters, not including the dollar sign. The first character following the dollar sign must always be a letter. The recommended form is $\textit{ZTC}x$, where \textit{x} is an alphanumeric character. An example of a valid object name for a
Elements of SPI Messages

Object-Name Templates

process is $ZTC0. If the process name is more than four characters long, the process cannot be accessed from remote systems.

Note. In the TCP/IP subsystem, a TCP/IP process can have more than one IP address associated with it. However, each process must have a NonStop OS process name, and each IP address must be unique within the network. (For more information about IP addresses, see the TCP/IP Configuration and Management Manual.)

When you assign an object name to a subnet, you must assign a unique name to each subnet on a single process. The name must be preceded by a pound sign (#) and can have a maximum of seven alphanumeric characters, not including the pound sign. The first character following the pound sign must always be a letter.

Subnets on separate processes can have identical object names; therefore, the process name and a period must precede the subnet name to reference a specific subnet. It is recommended that you use the letters SN followed by a subnet number to identify a subnet. An example of a valid object name for a subnet is $ZTC0.#SN1.

When you assign an object name to a route, you must assign a unique name to each route associated with a process. The name must be preceded by a pound sign (#) and can have a maximum of seven alphanumeric characters, not including the pound sign. The first character following the pound sign must always be a letter.

Routes on separate processes can have identical object names; therefore, the process name and a period must precede the route name to reference a specific route. It is recommended that you use the letters ROU followed by a route number to identify a route. An example of a valid object name for a route is $ZTC0.#ROU2.

Object-Name Templates

Some TCP/IP commands can affect multiple objects by using an object-name template. An object-name template is a character string that can include one or more wild-card characters and is specified in the ZCOM-TKN-OBJNAME token in the command. The TCP/IP subsystem supports these wild-card characters within an object-name template:

* An asterisk represents a character string of undefined length. An asterisk can be used to represent these:
  * A whole separated name, for example, $ZTCI.*. This selects all subnets and routes associated with process $ZTCI.
  * A trailing string, for example, $ZTCI.#SN*. This selects all subnets whose names begin with the letters SN and are associated with the process $ZTCI.
  * An undefined number of characters, for example, $ZTC*. This selects all TCP/IP processes whose names begin with $ZTC.

? A question mark represents one unknown character in a specific position. An example of its use is $ZTCI.#ROU1?. This selects all routes whose names begin with ROU1, are followed by only one character, and are associated with the process $ZTCI.
You can use wild-card characters in any combination.

You can use object-name templates in these commands:

- ABORT
- DELETE
- INFO
- LISTOBJECTS
- RESETSTATS
- START
- STATISTICS
- STATUS
- STOP

Note that object-name templates cannot be specified for the PROC object type. There is only one PROC object, and it must be fully specified.

**Event Numbers**

Event numbers identify event messages issued by the TCP/IP subsystem. Each event message that can be reported by the TCP/IP subsystem has a different event number assigned to it. The TCP/IP event numbers and their associated event messages are described in [Section 7, Event Management](#).

An event number is part of the header of an event message. The header token ZEMS-TKN-EVENTNUMBER contains the event number. The value of the event number is subsystem-specific and also has a symbolic name. Because event numbers represent unique events for a particular subsystem, you must specify both the event number and the subsystem ID of the subsystem that created the event to identify a particular event message. The symbolic names of event numbers are of the form $Z_{subsys}$-EVT-$event$, where $subsys$ is a three-character abbreviation for the subsystem and $event$ identifies the event. The TCP/IP subsystem is represented by the abbreviation TCI.

**Subjects of Event Messages**

The subject of an event message is always preceded by the ZEMS-TKN-SUBJECT-MARK token and can be one of two tokens for the TCP/IP subsystem: either ZCOM-TKN-SUBJ-PROC or ZCOM-TKN-SUBJ-SUBNET. The subject is a string that contains the name of the TCP/IP process or subnet that is the subject of the event message.

An application obtains the subject of an event message by passing the ZEMS-TKN-SUBJECT token to the EMSGET procedure. For details on obtaining subjects of event messages, see the **EMS Manual**.
Tokens

The TCP/IP subsystem uses tokens for parameters and responses to commands, and for elements of event messages. Each token has a token code and token value associated with it. The token code is the identifying tag that programs use to refer to the token by name. The token value is the actual data item that is assigned to the token code. The token code has two parts: a token type that specifies the data type and length of the token value, and a token number that uniquely identifies the token for the subsystem. For a list of the token types used in the TCP/IP subsystem, see Section 5, Common Definitions.

The actual token value can be in one of these forms:

- A single field, such as an integer or a character string
- A fixed structure, which consists of a fixed number of data fields that are always present
- An extensible structure, which consists of data fields whose number can be extended from release to release and may contain fields with null (unassigned) values

Tokens that have a token value that is a single field or a fixed structure are called simple tokens. Tokens that have a token value that is an extensible structure are called extensible structured tokens.

Simple Tokens

Simple tokens are tokens whose values are single fields or fixed structures. They are identified by token codes. They are considered simple tokens because you do not need a map to build and decode their internal structure. The token codes for the simple tokens used by the TCP/IP subsystem are represented by symbolic names of the form Z\text{subsys}-TKN-\text{name}, where \text{subsys} is a three-character subsystem abbreviation and \text{name} identifies the token code. For example, the token code ZEMS-TKN-LDEV identifies a simple token that contains a logical device number.

All the token codes defined by the TCP/IP subsystem (token codes with names of the form ZTCI-TKN-\text{name}) can be found in Section 5, Common Definitions.

Extensible Structured Tokens

Extensible structured tokens are tokens whose values are extensible structures and are identified by token maps. The token maps for the extensible structured tokens used by the TCP/IP subsystem are represented by symbolic names of the form Z\text{subsys}-MAP-\text{name}, where \text{subsys} is a three-character subsystem abbreviation and \text{name} identifies the token map. For example, the token map ZTCI-MAP-INFO-ROUTE identifies an extensible structured token that contains all the dynamic parameters for the specified route.

Extensible structured tokens are declared in the DDL files for each subsystem. Within an extensible structured token, you may find fields that are identified by predefined
value names. For more information on the form of these predefined value names, see **Predefined Value Names** on page 3-9. Note that extensible structured tokens do not appear in any event message issued by the TCP/IP subsystem.

For more information on all the token maps defined by the TCP/IP subsystem, see **Section 5, Common Definitions**.

**Token Types**

The token types used by the TCP/IP subsystem are represented by symbolic names of the form $Z_{\text{subsys}}$-$TYP$-$name$, where $subsys$ is a three-character subsystem abbreviation and $name$ identifies the token type.

For more information on all the token types used by the TCP/IP subsystem, see **Section 5, Common Definitions**.

**Predefined Value Names**

The predefined value names used by the TCP/IP subsystem are represented by symbolic names of the form $Z_{\text{subsys}}$-$VAL$-$name$, where $subsys$ is a three-character subsystem abbreviation and $name$ identifies the predefined value. For example, the predefined value ZTCI-$VAL$-$SSID$ represents the subsystem ID of the TCP/IP subsystem that is present in all commands and responses associated with the TCP/IP subsystem. For more information on all the predefined value names used by the TCP/IP subsystem, see **Section 5, Common Definitions**.

**Field Types**

The field types used by the TCP/IP subsystem are represented by symbolic names of the form $Z_{\text{subsys}}$-$DDL$-$ftype$, where $subsys$ is a three-character subsystem abbreviation and $ftype$ identifies the field type. For example, the field type ZSPI-$DDL$-$BOOLEAN$ represents the Boolean data type. Although most field types are defined by SPI, private field types can be defined by a subsystem. The TCP/IP subsystem does not define any private field types. For more information on all the field types used by the TCP/IP subsystem, see **Section 5, Common Definitions**.

**Constructs Involving Multiple Tokens**

The TCP/IP subsystem supports two kinds of constructs that involve multiple tokens: data lists and error lists. For the TCP/IP subsystem, these constructs are found only in a response buffer from the subsystem. A data list begins with the token ZSPI-$TKN$-$DATALIST$, while an error list begins with the token ZSPI-$TKN$-$ERRLIST$; both end with the token ZSPI-$TKN$-$ENDLIST$.

A data list is used in a response buffer to enclose information about one object or, in some cases, a related group of objects.

An error list is used in a response buffer to enclose information about an error or warning.
For details on these two constructs, see the *SPI Programming Manual*.

**Components of a Command, a Response, or an Event Message**

For general information about the components of a command, a response, or an event message, see the *SPI Common Extensions Manual*.

For information about the tokens associated with each TCP/IP command, response, and event message, see Section 6, Commands and Responses, and Section 7, Event Management. For information about considerations for building and retrieving SPI buffers, see Section 4, SPI Programming Considerations.
4

SPI Programming Considerations

The SPI Common Extensions Manual discusses general programming considerations for writing applications that use the SPI to communicate with data communications subsystems like the TCP/IP subsystem. This section discusses SPI programming considerations specific to the TCP/IP subsystem.

Building the Command Buffer

The following programming considerations apply when an application builds the command buffer for a TCP/IP command.

The recommended buffer length (in bytes) for a command is a predefined value with the symbolic name ZTCI-VAL-BUFLEN.

The subsystem does not support multiple commands in a single request message. However, many of the commands either accept more than one object name in one command or can contain an object-name template. Therefore, one command can affect multiple objects. All the objects whose names appear in a command must be of the same object type. The subsystem does not support continuation of a command across multiple messages, but it does support the special case of receiving multiple reply messages using the context token, as described in the SPI Programming Manual.

The command buffers for all commands can contain extensible structured tokens. An extensible structured token that is to appear in a command should always be initialized by the SPI procedure SSNULL to ensure that every field of the token is initialized to its null value. This action is important because an operation will be performed if a field contains a value other than its null value. For example, in the ALTER SUBNET command, if a field of the extensible structured token that describes the subnet attributes contains a value other than its null value, that value is assigned to the attribute corresponding to that field. Using SSNULL is important even when every field of an extensible structured token will have a value assigned to it: that way, the application will continue to work if it is compiled with new definition files that add new fields to the token.

The only tokens that can appear more than once in a command are ZSPI-TKN-COMMENT and ZCOM-TKN-TRACE-OPT. The comment token, ZSPI-TKN-COMMENT is described in the SPI Programming Manual.

ZCOM-TKN-TRACE-OPT, a token used in the TRACE command, is described under TRACE Command on page 6-110.

If any of the other tokens in a command appear more than once, an error is returned.
Discontinuing a Command in Progress

General programming considerations for discontinuing a command in progress are discussed in the *SPI Common Extensions Manual*.

The TCP/IP subsystem supports the use of the token ZSPI-TKN-ALLOW-TYPE, which allows an application to specify, in a command operating on multiple objects, whether this subsystem should continue immediately to the next object if an operation failed on the previous object. The only value supported for this token is ZSPI-VAL-NORM-ONLY. This means that the TCP/IP subsystem continues to the next object only if no errors or warnings occurred on the previous object.

When the TCP/IP subsystem discontinues a command because of an error or a warning, it immediately sends a reply message to the application. The reply message contains a context token. The application can then either resend or abandon the command. If the application resends the command, it uses the context token to inform the TCP/IP subsystem to proceed to the next object in the set.

Receiving and Decoding the Response Buffer

The following programming considerations apply when an application receives and decodes the response buffer.

Response records to TCP/IP commands that operate on objects always contain information for the operation of the command on a single object.

The subsystem supports multiple response records in a single reply message. The number of response records returned in one reply message is controlled by the value of the header token ZSPI-TKN-MAXRESP in the command. The supported values of the token ZSPI-TKN-MAXRESP are 0, -1, and $n$ (any integer value greater than 0).

If multiple response records are to be returned in one reply message, the recommended buffer size (in bytes) for the response is a predefined value with the symbolic name ZTCI-VAL-MAX-BUFLEN. If a single response record is to be returned in each reply message, the recommended buffer size (in bytes) for the response is the predefined value ZTCI-VAL-BUFLEN.

The subsystem supports multiple response records over multiple messages (continuation) by returning a context token when a response is continued.

The responses to the INFO, LISTOPENS, RESETSTATS, STATISTICS, and STATUS commands each contain an extensible structured token that returns the requested information.

The object-name token ZCOM-TKN-OBJNAME can appear more than once in a response record returned by the LISTOBJECTS command. This is the only case in which a token might appear multiple times in a response record.
Error Handling

General programming considerations for handling errors that occur when using the management-programming interfaces are discussed in the *SPI Common Extensions Manual*.

In the programmatic control-and-inquiry interface to the TCP/IP subsystem, there are two values for the return token (ZSPI-TKN-RETCODE) that indicate that the response record does not contain an error list. One value, ZCOM-ERR-OK, means that no error or warning occurred. The other is a ZCOM error number that signifies the “empty” response record. This value, ZCOM-ERR-EMPT-RSP, indicates there are no more responses to be returned.

Error-handling considerations for specific commands are described in Section 6, Commands and Responses.

Security

Security for the subsystem is handled by SCP, as described in the *SPI Common Extensions Manual*.

Some of the commands are sensitive commands, meaning that they can be issued only by certain users. When a user is not allowed to issue sensitive commands, attempts to issue one, SCP returns an error. For more information on sensitive commands, see Sensitive and Nonsensitive Commands on page 6-2.

Dynamic Route Switching

The TCP/IP subsystem provides dynamic route switching. This redetermines the subnet with which the route is associated. This can occur when a subnet is deleted or system resources need to be allocated differently. Monitoring subnet routing might be one of the tasks performed by an application responsible for managing the TCP/IP subsystem.

Relationship to SLSA

For Ethernet subnets, the SUBNET object of the TCP/IP subsystem is associated with a SLSA LIF. The TYPE attribute specifies Ethernet, SNAP, or X.25.

Relationship to X25AM LINE Object

For X.25 subnets, the SUBNET object of the TCP/IP subsystem is associated with the LINE object, namely the #TCPIP1 through #TCPIP32 subdevices, on an X25AM process. These X25AM subdevices must be added prior to adding the X.25-type subnet. For more information on the X25AM LINE object, see the *X25AM Management Programming Manual*. 
5 Common Definitions

The management-programming interfaces to the TCP/IP subsystem use definitions from four sources: SPI standard definitions, EMS standard definitions, data communications standard definitions, and TCP/IP subsystem-specific definitions. Many of the definitions are used in multiple commands, responses, event messages, or error lists.

This section contains general information that applies to all uses of the common definitions in the interfaces to the TCP/IP subsystem. Information about a definition that is specific to a particular command, response, event message, or error list is given in the description for that item.

All definitions in this section are represented in DDL. Structures, both fixed and extensible, are defined by using DDL DEF statements. For a brief explanation of DDL as it applies to SPI, see the SPI Programming Manual.

Definitions are listed in alphabetic order within each category.

SPI Standard Definitions

Definitions whose names begin with ZSPI- are SPI standard definitions. These definitions are available to all subsystems that support SPI procedures.

Table 5-1 lists only those SPI standard definitions that are used in the management-programming interfaces to the TCP/IP subsystem. It does not list SPI error numbers or definitions used only in error lists associated with those error numbers. For a description of all SPI standard definitions, see the SPI Programming Manual.

The SPI definitions applicable to the TCP/IP subsystem consists of:

- Header tokens
- Simple tokens
- Token and field values
- Token types
- Fixed structures

Table 5-1. SPI Standard Definitions (page 1 of 2)

<table>
<thead>
<tr>
<th>Header Tokens</th>
<th>Other Simple Tokens</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZSPI-TKN-CHECKSUM</td>
<td>ZSPI-TKN-ALLOW-TYPE</td>
</tr>
<tr>
<td>ZSPI-TKN-COMMAND</td>
<td>ZSPI-TKN-COMMENT</td>
</tr>
<tr>
<td>ZSPI-TKN-HDRTYPE</td>
<td>ZSPI-TKN-CONTEXT</td>
</tr>
<tr>
<td>ZSPI-TKN-MAX-FIELD-VERSION</td>
<td>ZSPI-TKN-DATALIST</td>
</tr>
<tr>
<td>ZSPI-TKN-MAXRESP</td>
<td>ZSPI-TKN-ENDLIST</td>
</tr>
<tr>
<td>ZSPI-TKN-OBJECT-TYPE</td>
<td>ZSPI-TKN-ERRLIST</td>
</tr>
</tbody>
</table>
### SPI Standard Definitions

<table>
<thead>
<tr>
<th>Token Types</th>
<th>Fixed Structures</th>
<th>Value Names</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZSPI-TKN-SERVER-VERSION</td>
<td>ZSPI-TKN-ERROR</td>
<td>ZSPI-SSN-ZTCI</td>
</tr>
<tr>
<td>ZSPI-TKN-SSID</td>
<td>ZSPI-TKN-RETCODE</td>
<td>ZSPI-VAL-CMDHDR</td>
</tr>
<tr>
<td>ZSPI-TKN-USEDLEN</td>
<td>ZSPI-TKN-SERVER-BANNER</td>
<td>ZSPI-VAL-EVTHDR</td>
</tr>
</tbody>
</table>

SPI definitions that are specific to the TCP/IP subsystem are described below.

**ZSPI-SSN-ZTCI**

specifies the subsystem number assigned to the TCP/IP subsystem. This value name is defined as a constant in the SPI DDL file, ZSPIDEF.ZSPIDDL.
**ZSPI-TKN-ALLOW-TYPE**

within a command buffer indicates the conditions under which processing of a set of objects will continue. This token applies to only those commands that accept object-name templates.

The ALLOW-TYPE token controls whether the subsystem continues to the next object of a set when there is an error or a warning during processing of an object. It does not control the response when the tokens within a command message are incorrect or missing. The TCP/IP subsystem supports only this value:

**ZSPI-VAL-NORM-ONLY**

indicates that the TCP/IP subsystem continues to process the next object in the set if the command is completed successfully on the previous object. There cannot be an error list in the response buffer. This is the only supported value.

**ZSPI-TKN-CHECKSUM**

provides checksum protection against accidental corruption of the SPI buffer between calls to the SPI procedures. The possible values are:

**ZSPI-VAL-FALSE**

indicates that there is no checksum protection.

**ZSPI-VAL-TRUE**

indicates that there is checksum protection.

**ZSPI-TKN-COMMAND**

specifies the command number of a TCP/IP command. The value of this token is always ZCOM-CMD-name, where name specifies the command to be performed. The commands supported by TCP/IP are described in [Section 6, Commands and Responses](#).

**ZSPI-TKN-COMMENT**

allows the requester to include comment information, which is not acted upon by the server or returned by the server. The requester can include as many as 80 bytes of comment in every command. If the comment exceeds this number of bytes and causes a command to be too large for the server’s read buffer, the server is not responsible for recovery. The TCP/IP subsystem guarantees only 80 bytes.

**ZSPI-TKN-CONTEXT**

within the response buffer indicates to the requester whether there are more objects to process. This token applies only when a command references more than one object and the information to be returned cannot fit into one response buffer. For the TCP/IP subsystem, this occurs only when an object-name template has
been specified for a command or the NULL object has been specified for the LISTOBJECTS command.

The server places information in this token, which the server requires to continue execution of the command at the next object. When a requester receives a response containing the context token, the requester copies the context token into a duplicate of the original command message and sends this command message to the server. The server then processes the next object in the set. This sequence is repeated until all the objects in the set have been processed.

Once all the objects in the set are processed, the ZSPI-TKN-CONTEXT token is not placed in the response by the server. Its absence indicates that all objects of the set have been processed. If the server receives a command with the context token and there are no more objects to be processed, the server returns the empty response value (ZCOM-ERR-EMPT-RSP) in the return token (ZSPI-TKN-RETCODE).

The contents of ZSPI-TKN-CONTEXT should not be processed or altered by the requester; they should only be echoed to the server in a command. Only one context token can be present in a command.

ZSPI-TKN-DATALIST
indicates the beginning of a data list within a response. The presence of this token depends on the value of ZSPI-TKN-MAXRESP. If the value of ZSPI-TKN-MAXRESP is nonzero, the response tokens, excluding header tokens and the ZSPI-TKN-CONTEXT token, are enclosed in one or more data lists. Although there can be multiple data lists within a single response buffer, each data list must contain information about only one object type.

ZSPI-TKN-ENDLIST
indicates the end of a data list or an error list within a response.

ZSPI-TKN-ERRLIST
indicates the beginning of an error list. All information about warnings or errors, except ZSPI-TKN-RETCODE, must be enclosed in an error list. An error list is present only if the return code is nonzero. Note that the empty response does not contain an error list.

The contents of an error list are owned by the server returning the error list. The exception is a nested error list, which may have a different owner.

Multiple error lists can be returned in a single response buffer or even in a single data list.

ZSPI-TKN-ERROR
is the standard SPI error token described in the SPI Programming Manual. Its value consists of the TCP/IP subsystem ID and one of the TCP/IP subsystem-
specific error numbers described in Appendix A, Error Numbers and Error Lists. This token is always present in an error list.

ZSPI-TKN-HDRTYPE

indicates whether the SPI buffer is for an event message or a command. If the buffer is for a command, the value of this token is ZSPI-VAL-CMDHDR; if the buffer is for an event, the value of this token is ZSPI-VAL-EVTHDR.

ZSPI-TKN-MAX-FIELD-VERSION

contains the latest version associated with the non-null fields within all extensible structured tokens in a command. This value is compared with the server version. If the server version is earlier than the value of ZSPI-TKN-MAX-FIELD-VERSION, the command is rejected, because the server needs to be a more recent version.

ZSPI-TKN-MAXRESP

within a command buffer indicates the maximum number of response records a requester can accept in a response buffer. The MAXRESP token default value is provided when the requester uses the SSINIT procedure. However, if the requester chooses to specify a value other than the default value for the MAXRESP token, the requester must specify a value using the SSPUT procedure. The TCP/IP interface provides full support for the MAXRESP token; that is, the requester can specify any of these values:

-1 indicates as many response records as will fit in the response buffer can be returned. Each response record will be enclosed in a data list.

0 indicates only one response record will be returned in the response buffer. The response record will not be enclosed in a data list. This is the default action.

n indicates as many as n response records, where n is an integer greater than 0, will be returned in the response buffer. Each response record will be enclosed in a data list.

ZSPI-TKN-OBJECT-TYPE

specifies the object-type number of a TCP/IP object. The value of this token for the TCP/IP subsystem is always ZCOM-OBJ-type, where type specifies the object. The valid object types for this subsystem are PROC, SUBNET, ROUTE, and NULL. The object types associated with a command vary with each command. For information on which objects are valid for a specific command, see Section 6, Commands and Responses.

ZSPI-TKN-RETCODE

is the return token, which specifies whether a command has been completed successfully. If a command has been completed successfully, including when a command is completed with a warning, the value of this token is ZCOM-ERR-OK.
Note that if the empty response (ZCOM-ERR-EMPT-RSP) is returned, the value of this token is nonzero, although the command has been completed successfully.

If the command was not completed successfully, the value of this token indicates the reason the command failed. The meaning of each error number can be subsystem-specific and possibly command-specific. Although the TCP/IP subsystem does not define any subsystem-specific errors, it does return some of the data communications and SPI-defined errors. For descriptions of all common data communications errors, see the SPI Common Extensions Manual. For descriptions of all common SPI errors, see the SPI Programming Manual.

**ZSPI-TKN-SERVER-BANNER**

contains the server-banner string for the TCP/IP subsystem. The server-banner string reflects the product name, product number, release date, and the date the TCP/IP code was compiled. This token is returned only for the GETVERSION command. The server banner for TCP/IP is:

TCP/IP T9551D41 - release date - compile date

**ZSPI-TKN-SERVER-VERSION**

specifies the server version of the TCP/IP subsystem. This token is required in all the commands and responses. The SSINIT procedure initializes the command buffer with this token, using the value defined in the subsystem's definition file. The value used for TCP/IP is ZTCI-VAL-VERSION. The subsystem provides the value of this token for comparison. The value provided by the subsystem must be at least as current as the value provided by SSINIT. If it is not as current, the subsystem rejects the command.

**ZSPI-TKN-SSID**

specifies the subsystem ID. This token is required in all commands and responses. The SSINIT procedure initializes the command buffer with this token, using the value defined in the subsystem's definition file. The value used for TCP/IP is ZTCI-VAL-SSID. The contents of the token are:

```
def ZTCI-VAL-SSID tacl ssid.
  02 Z-FILLER type character 8
    value is ZSPI-VAL-TANDEM.
  02 Z-OWNER redefines Z-FILLER
    type ZSPI-DDL-CHAR8.
  02 Z-NUMBER type ZSPI-DDL-INT
    value is ZSPI-SSN-ZTCI.
  02 Z-VERSION type ZSPI-DDL-UINT
    value is ZTCI-VAL-VERSION.
end.
```

**ZSPI-TKN-USEDLEN**

specifies the length of the used portion of a command message. The value of this token is used to verify that the entire message has been read by the server. The
value is compared with the count-read value supplied by the file system. If the value of ZSPI-TKN-USEDLEN is greater than the count-read value, then the TCP/IP subsystem rejects the command because the command was too long.

**ZSPI-VAL-CMDHDR**

indicates that the SPI buffer is for a command. It is one of the two possible values of the ZSPI-TKN-HDRTYPE token.

**ZSPI-VAL-EVTHDR**

indicates that the SPI buffer is for an event message. It is one of the two possible values of the ZSPI-TKN-HDRTYPE token.

**ZSPI-VAL-NORM-ONLY**

is the only supported value of the ZSPI-TKN-ALLOW-TYPE token for the TCP/IP subsystem. This value indicates that the TCP/IP subsystem should continue to process the next object in a set only if the command is completed successfully on the previous object.

**ZSPI-VAL-TANDEM**

indicates that the destination subsystem is on a NonStop system. The requester must assign this value to the owner field of the ZSPI-TKN-SSID token before sending a command to a NonStop system.

---

### EMS Standard Definitions

Definitions whose names begin with ZEMS- are Event Management Service (EMS) standard definitions. These definitions are available to all subsystems that support EMS. They can be found in the EMS DDL file ZSPIDEF.ZEMSDDL and in the definition files for other languages.

The EMS definitions used by the TCP/IP subsystem consists of:

- Header tokens
- Simple tokens

Table 5-2 lists only those EMS standard definitions that are used in the management-programming interfaces to the TCP/IP subsystem. It does not list EMS error numbers or definitions used only in error lists associated with those error numbers. For a description of all EMS standard definitions, see the *EMS Manual*.

<table>
<thead>
<tr>
<th>Header Tokens</th>
<th>Other Simple Tokens</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZEMS-TKN-CONSOLE-PRINT</td>
<td>ZEMS-TKN-SUBJECT-MARK</td>
</tr>
<tr>
<td>ZEMS-TKN-CPU</td>
<td></td>
</tr>
</tbody>
</table>
EMS standard definitions specific to the TCP/IP subsystem are described below.

**ZEMS-TKN-CONSOLE-PRINT**

specifies whether the event message in which it appears is processed by the compatibility distributor. The value of this token for the TCP/IP subsystem is ZSPI-VAL-TRUE. This indicates that TCP/IP event messages are displayed at the operator console.

**ZEMS-TKN-CPU**

specifies the processor of the subsystem.

**ZEMS-TKN-CRTPID**

specifies the process ID of the subsystem reporting the event.

**ZEMS-TKN-EMPHASIS**

specifies whether the event message is considered critical. Critical events are errors and warnings for which the consequences could be serious, such as a hardware failure or a software error from which there is no automatic recovery. Noncritical events are often expected events that are informational, such as a summary state change. If the value of the token is ZSPI-VAL-TRUE, the event is considered critical; if the value is ZSPI-VAL-FALSE, the event is not considered critical.

**ZEMS-TKN-EVENTNUMBER**

specifies the event number of a TCP/IP event message. These event numbers and their associated event messages are described in Section 7, Event Management.

**ZEMS-TKN-GENTIME**

is the time, in Greenwich Mean Time (GMT), that the subsystem created the event message.
ZEMS-TKN-LOGTIME  
is the time, in GMT, that the collector wrote the event message to its log files.

ZEMS-TKN-PIN  
is the process identification number (PIN) of the subsystem that reported the event.

ZEMS-TKN-SUBJECT-MARK  
indicates that the token that follows is the subject token. The TCP/IP subsystem uses two subject tokens in event messages: ZCOM-TKN-SUBJ-PROC and ZCOM-TKN-SUBJ-SUBNET. For more details, see Section 7, Event Management.

ZEMS-TKN-SYSTEM  
is the Expand system number of the system (node) on which the event was reported.

ZEMS-TKN-USERID  
specifies the user ID of the TCP/IP subsystem process.

Data Communications Standard Definitions

Definitions whose names begin with ZCOM- are data communications standard definitions. These definitions are available to all data communications subsystems, such as TCP/IP, and are found in the ZSPIDEF.ZCOMDDL definition file and in the files for other languages. For more details, see Definition Files on page 3-1.

The data communications definitions used by the TCP/IP subsystem consist of:

- Simple tokens
- Extensible structured tokens
- Object types
- Fixed structures
- Command numbers
- Token and field values

Table 5-3 on page 5-10 lists only those data communications standard definitions used in the management-programming interfaces to the TCP/IP subsystem. It does not list data communications error numbers or definitions used only in error lists associated with those error numbers. For a description of all data communications standard definitions and error numbers, see the SPI Common Extensions Manual.
Data communications standard definitions specific to the TCP/IP subsystem are described below.

**ZCOM-MAP-REQID** provides security checking in addition to the sensitive/nonsensitive command implementation. This token contains the NonStop OS process ID (PID) and the process access ID (PAID) of the application issuing the command. The ZCOM-MAP-REQID token is valid only if the requester has the SUPER.SUPER user ID. If the user does not have the SUPER.SUPER user ID and the requester either specifies ZCOM-MAP-REQID in the command or duplicates the token in the command buffer, the command is rejected with a security violation. If

### Table 5-3. Data Communications Standard Definitions

<table>
<thead>
<tr>
<th>Simple Tokens</th>
<th>Command Numbers</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZCOM-TKN-CAUSE-SUMSTATE-CHG</td>
<td>ZCOM-CMD-ABORT</td>
</tr>
<tr>
<td>ZCOM-TKN-OBJNAME</td>
<td>ZCOM-CMD-ADD</td>
</tr>
<tr>
<td>ZCOM-TKN-OBJSTATE</td>
<td>ZCOM-CMD-ALTER</td>
</tr>
<tr>
<td>ZCOM-TKN-OBJTYPE</td>
<td>ZCOM-CMD-DELETE</td>
</tr>
<tr>
<td>ZCOM-TKN-OLD-OBJSTATE</td>
<td>ZCOM-CMD-GETVERSION</td>
</tr>
<tr>
<td>ZCOM-TKN-SUBJ-PROC</td>
<td>ZCOM-CMD-INFO</td>
</tr>
<tr>
<td>ZCOM-TKN-SUBJ-SUBNET</td>
<td>ZCOM-CMD-LISTOBJECTS</td>
</tr>
<tr>
<td>ZCOM-TKN-TRACE-OPT</td>
<td>ZCOM-CMD-LISTOPENS</td>
</tr>
<tr>
<td></td>
<td>ZCOM-CMD-RESETSTATS</td>
</tr>
<tr>
<td>Extensible</td>
<td></td>
</tr>
<tr>
<td>Structured Tokens</td>
<td>Extensible</td>
</tr>
<tr>
<td>ZCOM-MAP-REQID</td>
<td>ZCOM-CMD-START</td>
</tr>
<tr>
<td>ZCOM-MAP-TRACE-MODIF</td>
<td>ZCOM-CMD-STATISTICS</td>
</tr>
<tr>
<td></td>
<td>ZCOM-CMD-STATUS</td>
</tr>
<tr>
<td></td>
<td>ZCOM-CMD-STOP</td>
</tr>
<tr>
<td></td>
<td>ZCOM-CMD-TRACE</td>
</tr>
<tr>
<td>Fixed Structures</td>
<td>Object Types</td>
</tr>
<tr>
<td>ZCOM-DDL-REQID</td>
<td>ZCOM-OBJ-NULL</td>
</tr>
<tr>
<td>ZCOM-DDL-TRACE-MODIF</td>
<td>ZCOM-OBJ-PROC</td>
</tr>
<tr>
<td>Value Names</td>
<td>ZCOM-OBJ-ROUTE</td>
</tr>
<tr>
<td>ZCOM-VAL-BUFLEN</td>
<td>ZCOM-OBJ-SUBNET</td>
</tr>
<tr>
<td>ZCOM-VAL-MAXBUFLEN</td>
<td></td>
</tr>
<tr>
<td>ZCOM-VAL-SUMSTATE-STARTED</td>
<td></td>
</tr>
<tr>
<td>ZCOM-VAL-SUMSTATE-STOPPED</td>
<td></td>
</tr>
</tbody>
</table>

Data communications standard definitions specific to the TCP/IP subsystem are described below.

**ZCOM-MAP-REQID** provides security checking in addition to the sensitive/nonsensitive command implementation. This token contains the NonStop OS process ID (PID) and the process access ID (PAID) of the application issuing the command. The ZCOM-MAP-REQID token is valid only if the requester has the SUPER.SUPER user ID. If the user does not have the SUPER.SUPER user ID and the requester either specifies ZCOM-MAP-REQID in the command or duplicates the token in the command buffer, the command is rejected with a security violation. If
ZCOM-MAP-REQID is legally specified, SCP performs a security check based on the user ID included in the token. The contents of the token are:

ZCOM-MAP-REQID

def ZCOM-MAP-REQID.
  02 ZCRTPID type ZSPI-DDL-CRTPID.
  02 ZPAID   type ZSPI-DDL-INT.
end.

ZCOM-MAP-TRACE-MODIF

is the extensible structured token sent in the TRACE command by the requester. It contains the trace parameters that control the initiation of the trace file, the selection of the trace environment, and the termination of trace mode. The initiation and selection are handled in one ZCOM-MAP-TRACE-MODIF token. When the trace is to be terminated, the requester sends another TRACE command with another ZCOM-MAP-TRACE-MODIF token with a ZSTOP field containing the value ZSPI-VAL-TRUE. The contents of the token are:

ZCOM-MAP-TRACE-MODIF

def ZCOM-DDL-TRACE-MODIF. !All fields are required.
  02 ZSTOP   type ZSPI-DDL-BOOLEAN.
  02 ZPRIM   type ZSPI-DDL-BOOLEAN.
  02 ZREC-SIZE type ZSPI-DDL-INT.
  02 ZCNT    type ZSPI-DDL-INT2.
  02 ZXDS-PAGES type ZSPI-DDL-INT.
  02 ZCOLL   type ZSPI-DDL-BOOLEAN.
  02 ZFNAME  type ZSPI-DDL-FNAME32.
  02 ZWRAP   type ZSPI-DDL-BOOLEAN.
end.

ZCOM-OBJ-NULL

does not represent a controllable TCP/IP object. It is used as the object type in commands that do not require an object-name token. Only two TCP/IP commands use the NULL object: GETVERSION and LISTOBJECTS.

ZCOM-OBJ-PROC

is the TCP/IP process running on a NonStop system. The TCP/IP process is created using the RUN command. The only valid summary state for this object is the STARTED summary state.

ZCOM-OBJ-ROUTE

is the path used for sending IP datagrams. Each route is associated with a subnet. There can be several routes associated with a single subnet. The valid summary states are STARTED and STOPPED.
**ZCOM-OBJ-SUBNET**

is the point of attachment for the Internet Protocol (IP) Layer through which the IP Layer can send or receive data. The valid summary states are STARTED and STOPPED.

**ZCOM-TKN-CAUSE-SUMSTATE-CHG**

is an enumerated value that indicates the reason the summary state of an object has changed. This token is returned in the event message ZTCI-EVT-SUMSTATE-CHG. For details about the event message, see Section 7, Event Management. The possible values are subsystem-specific and are:

- **ZTCI-VAL-CAUSE-ERR**
  - indicates that the change occurred because of an error on the subnet interface.

- **ZTCI-VAL-CAUSE-OP**
  - indicates that the change occurred because of a command.

- **ZTCI-VAL-CAUSE-RECOV**
  - indicates that the change occurred because of a recovery operation.

**ZCOM-TKN-OBJNAME**

contains the name of a TCP/IP object. The object name must follow the naming conventions described in Section 3, Elements of SPI Messages.

This token is always present in the command, unless the NULL object has been specified, and can be present in every response.

In a command, the object name indicates the object on which the command is to operate. In addition, in some commands the requester can specify object-name templates. Using an object-name template allows a command to affect multiple objects.

In a response, the object name indicates the object on which the command has been performed. This token is always in the response, unless an error prevented the command from being attempted at all. This token can also be present in data lists and error lists.

If an object-name template has been specified, a separate object-name token is returned for each object that meets the template requirements.

The object name of a ROUTE object is the name of a specific route associated with a subnet. A route can be dynamically added or it can be added using the ADD command. The names are assigned when the route is added and are unique within a subnet.

The object name of a SUBNET object is the name of a specific subnet. A subnet is added using the ADD command. The name is assigned when the subnet is added.
and is unique within a process. As many as eight subnets can be added to one TCP/IP process.

The NULL object has no object name.

The object name of the PROC object is the object name of the TCP/IP process.

ZCOM-TKN-OBJSTATE

is the summary state of a subnet after a summary-state change has taken place. This token is returned in the event message ZTCI-EVT-SUMSTATE-CHG. For details on this event message, see Section 7, Event Management. The possible values of this token are:

ZCOM-VAL-SUMSTATE-STARTED

indicates that the subnet is in the STARTED summary state.

ZCOM-VAL-SUMSTATE-STopped

indicates that the subnet is in the STOPPED summary state.

ZCOM-TKN-OBJTYPE

contains the object type of a TCP/IP object. This token is present in every response. The value within a command can be ZCOM-OBJ-ROUTE, ZCOM-OBJ-NULL, ZCOM-OBJ-SUBNET, or ZCOM-OBJ-PROC. This token is the same as the object type specified in the command header token ZSPI-TKN-OBJECT-TYPE.

The ZCOM-TKN-OBJTYPE token is also present in data lists and error lists. It indicates the object type for which information is being returned or the object type that experienced the error.

ZCOM-TKN-OLD-OBJSTATE

is the summary state of the object before a summary-state change of the object took place. This token is returned in the event ZTCI-EVT-SUMSTATE-CHG. For details on the event message, see Section 7, Event Management. The possible values of this token are:

ZCOM-VAL-SUMSTATE-STARTED

indicates that the object was in the STARTED summary state.

ZCOM-VAL-SUMSTATE-STopped

indicates that the object was in the STOPPED summary state.

ZCOM-TKN-SUBJ-PROC

indicates the name of the TCP/IP process that is the subject of an event message.
ZCOM-TKN-SUBJ-SUBNET

indicates the name of the TCP/IP subnet that is the subject of an event message.

ZCOM-TKN-TRACE-OPT

indicates what items are to be traced by the TRACE command. For information on the possible values of this token, see the TRACE Command on page 6-110. The SCP creates a bit mask from the trace options specified and sends it to the TCP/IP subsystem.

ZCOM-VAL-BUFLEN

is the recommended buffer length for command messages and for responses in which each response message will contain a single response record.

ZCOM-VAL-MAXBUFLEN

is the maximum buffer length allowed for SPI message buffers. It is the recommended buffer length for responses in which each response message will contain multiple response records.

ZCOM-VAL-SUMSTATE-STARTED

indicates that a TCP/IP object is in the STARTED summary state. It is a value of the state field within the extensible structured tokens ZTCI-MAP-STATUS-PROC, ZTCI-MAP-STATUS-SUBNET, and ZTCI-MAP-STATUS-ROUTE. It is also a value of the ZCOM-TKN-OBJSTATE and ZCOM-TKN-OLD-OBJSTATE tokens in the ZTCI-EVT-SUMSTATE-CHG event message. The process is always in this summary state. For more information, see the STATUS Command on page 6-100.

ZCOM-VAL-SUMSTATE-STOPPED

indicates that a TCP/IP subnet or route is in the STOPPED summary state. It is a value of the state field within the extensible structured tokens ZTCI-MAP-STATUS-SUBNET and ZTCI-MAP-STATUS-ROUTE. It is also a value of the ZCOM-TKN-OBJSTATE and ZCOM-TKN-OLD-OBJSTATE tokens in the ZTCI-EVT-SUMSTATE-CHG event message. The PROC object cannot be in this summary state. If the PROC object is not started, it is undefined. For more information, see the STATUS Command on page 6-100.

Common Definitions for the TCP/IP Subsystem

Definitions whose names begin with ZTCI- are TCP/IP definitions. The definitions that appear in several commands or responses are described in this section. Information that is specific to particular commands or responses is discussed in Section 6, Commands and Responses.
Buffer Declarations

The TCP/IP subsystem provides the following buffer declarations for message buffers that can be used with the SPI procedures. All applications should use one of these declarations for a buffer of the recommended size.

ZTCI-DDL-MSG-BUFFER

is a message buffer whose size is the size recommended for command messages and for responses in which each response message contains a single response record.

DEF zTCI-ddl-msg-buffer.
  02 z-msgcode TYPE zspi-ddl-int.
  02 z-buflen TYPE zspi-ddl-uint.
  02 z-occurs TYPE zspi-ddl-uint.
  02 z-filler TYPE zspi-ddl-byte
      OCCURS 0 to zcom-val-buflen TIMES
      DEPENDING ON z-occurs.
END.

TCP/IP Private Token and Field Types

Private tokens are those tokens used only by your subsystem. A token type defines the data type and length of a token. A field type defines the data type and length of a field within a structured token. The TCP/IP subsystem defines these private token types:

ZTCI-TYP-CAUSE
ZTCI-TYP-SEV
ZTCI-TYP-SUBNET-TYPE
ZTCI-TYP-TCP-STATE
ZTCI-TYP-TRACE
ZTCI-TYP-XFER

ZTCI-TYP-CAUSE

defines the token type of the values of the ZCOM-TKN-CAUSE-SUMSTATE-CHG token as enumerated values.

ZTCI-TYP-SEV

defines the token type of the values of the ZTCI-TKN-SEV token as enumerated values.

ZTCI-TYP-SUBNET-TYPE

defines the token type of the values of the ZTYPE field in the ZTCI-MAP-ADD-SUBNET and ZTCI-MAP-INFO-SUBNET extensible structured tokens as enumerated values.

ZTCI-TYP-TCP-STATE

defines the token type of the values of the structured token as enumerated values.
ZTCI-TYP-TRACE

defines the token type of the values of the ZCOM-TKN-TRACE-OPT token (the TRACE command options) as enumerated values.

ZTCI-TYP-XFER

defines the token type of the values of the ZTCI-TKN-XFER-UOW token as enumerated values.

TCP/IP Predefined Token and Field Values

The TCP/IP subsystem defines these token and field values:

<table>
<thead>
<tr>
<th>ZTCI-VAL-BUFLEN</th>
<th>ZTCI-VSL-TCP-STATE-SYNC-SENT</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZTCI-VAL-CAUSE-ERR</td>
<td>ZTCI-VAL-TCP-STATE-TIME-WAIT</td>
</tr>
<tr>
<td>ZTCI-VAL-CAUSE-OP</td>
<td>ZTCI-VAL-TRACE-IP-IN</td>
</tr>
<tr>
<td>ZTCI-VAL-CAUSE-RECOV</td>
<td>ZTCI-VAL-TRACE-MBUF-ALLOC</td>
</tr>
<tr>
<td>ZTCI-VAL-MAX-BUFLEN</td>
<td>ZTCI-VAL-TRACE-ROUTE</td>
</tr>
<tr>
<td>ZTCI-VAL-MAX-CAUSE</td>
<td>ZTCI-VAL-TRACE-SOCKET-CMDS</td>
</tr>
<tr>
<td>ZTCI-VAL-MAX-SEV</td>
<td>ZTCI-VAL-TRACE-SOCKET-CREATE</td>
</tr>
<tr>
<td>ZTCI-VAL-MAX-TRACE</td>
<td>ZTCI-VAL-TRACE-TCP</td>
</tr>
<tr>
<td>ZTCI-VAL-MAX-XFER</td>
<td>ZTCI-VAL-TRACE-UDP-DETL-IN</td>
</tr>
<tr>
<td>ZTCI-VAL-MIN-CAUSE</td>
<td>ZTCI-VAL-TRACE-UDP-DETL-OUT</td>
</tr>
<tr>
<td>ZTCI-VAL-MIN-SEV</td>
<td>ZTCI-VAL-UDP-IN</td>
</tr>
<tr>
<td>ZTCI-VAL-MIN-TRACE</td>
<td>ZTCI-VAL-UDP-USER-REQ</td>
</tr>
<tr>
<td>ZTCI-VAL-MIN-XFER</td>
<td>ZTCI-VAL-VERSION</td>
</tr>
<tr>
<td>ZTCI-VAL-SEV-FATAL</td>
<td>ZTCI-VAL-XFER-ADD-ITEM-REC</td>
</tr>
<tr>
<td>ZTCI-VAL-SEV-INFO</td>
<td>ZTCI-VAL-XFER-ADD-RECIP</td>
</tr>
<tr>
<td>ZTCI-VAL-SEV-WARN</td>
<td>ZTCI-VAL-XFER-ATTACH-COMPNT</td>
</tr>
<tr>
<td>ZTCI-VAL-SSID</td>
<td>ZTCI-VAL-XFER-CREATE-FOLDER</td>
</tr>
<tr>
<td>ZTCI-VAL-SUBNET-TYPE-ENET</td>
<td>ZTCI-VAL-XFER-CREATE-ITEM</td>
</tr>
<tr>
<td>ZTCI-VAL-SUBNET-TYPE-LOOP</td>
<td>ZTCI-VAL-XFER-DETACH-COMPNT</td>
</tr>
<tr>
<td>ZTCI-VAL-SUBNET-TYPE-SNAP</td>
<td>ZTCI-VAL-XFER-END-SESS</td>
</tr>
<tr>
<td>ZTCI-VAL-SUBNET-TYPE-X25</td>
<td>ZTCI-VAL-XFER-GET-ITEM-COMPNT</td>
</tr>
<tr>
<td>ZTCI-VAL-TCP-STATE-CLOSE-WAIT</td>
<td>ZTCI-VAL-XFER-GET-ITEM-DESCR</td>
</tr>
<tr>
<td>ZTCI-VAL-TCP-STATE-CLOSING</td>
<td>ZTCI-VAL-XFER-GET-ITEM-REC</td>
</tr>
<tr>
<td>ZTCI-VAL-TCP-STATE-ESTAB</td>
<td>ZTCI-VAL-XFER-GET-RECIPI-REC</td>
</tr>
<tr>
<td>ZTCI-VAL-TCP-STATE-FIN-WAIT-1</td>
<td>ZTCI-VAL-XFER-SAVE-ITEM</td>
</tr>
<tr>
<td>ZTCI-VAL-TCP-STATE-FIN-WAIT-2</td>
<td>ZTCI-VAL-XFER-SCAN-FOLDER</td>
</tr>
<tr>
<td>ZTCI-VAL-TCP-STATE-LAST-ACK</td>
<td>ZTCI-VAL-XFER-START-SESS</td>
</tr>
<tr>
<td>ZTCI-VAL-TCP-STATE-LAST-ACK</td>
<td>ZTCI-VAL-XFER-SUBMIT-PKG</td>
</tr>
<tr>
<td>ZTCI-VAL-TCP-STATE-LISTEN</td>
<td>ZTCI-VAL-XFER-UNSAVE-ITEM</td>
</tr>
<tr>
<td>ZTCI-VAL-TCP-STATE-SYNC-REC</td>
<td></td>
</tr>
</tbody>
</table>

ZTCI-VAL-BUFLEN

is the recommended buffer size (in bytes) for command messages sent to the TCP/IP subsystem and for responses in which each response message contains a single response record.
(ZTCI-VAL-CAUSE-) ERR, OP, and RECOV

are the possible values of the token ZCOM-TKN-CAUSE-SUMSTATE-CHG in the
event message ZTCI-EVT-SUMSTATE-CHG. These values indicate the cause of
the summary state change for an object. For more information, see Section 7,
Event Management.

ZTCI-VAL-MAX-BUFLEN

is the maximum buffer size (in bytes) for command messages sent to the TCP/IP
subsystem and the recommended buffer size for responses in which each
response message contains multiple response records.

(ZTCI-VAL-MAX-) CAUSE, SEV, SUBNET-TYPE, TCP-STATE, TRACE, and
XFER

specify the upper limit of the supported values for the associated tokens.

(ZTCI-VAL-MIN-) CAUSE, SEV, SUBNET-TYPE, TCP-STATE, TRACE, and
XFER

specify the lower limit of the supported values for the associated tokens.

(ZTCI-VAL-SEV-) FATAL, INFO, and WARN

are the possible values of the token ZTCI-TKN-SEV. The ZTCI-TKN-SEV token
appears in all the event messages issued by the TCP/IP subsystem. For more
details, see Section 7, Event Management.

ZTCI-VAL-SSID

is the subsystem ID for the TCP/IP subsystem. Its structure definition is given
earlier in this section under SPI Standard Definitions on page 5-1.

For more information about ZTCI-VAL-SSID, see the discussion of the subsystem
ID in the SPI Programming Manual.

(ZTCI-VAL-SUBNET-TYPE-) ENET, LOOP, SNAP, and X25

are the values of the ZTYPE field in the ZTCI-MAP-ADD-SUBNET and
ZTCI-MAP-INFO-SUBNET extensible structured tokens. The value ENET indicates
that the subnet is associated with a SLSA LIF using an Ethernet type filter. The
value LOOP indicates that the subnet is associated with the loopback driver for
testing purposes. The value SNAP indicates that the subnet is associated with a
SLSA LIF using an LLC1 type filter. The value X25 indicates that the subnet is
associated with an X25AM device.
TCP/IP Simple Tokens

The only simple tokens defined by the TCP/IP subsystem are returned in event messages. For a listing of the TCP/IP-defined event-message tokens, see TCP/IP Tokens in Event Messages on page 5-19.

TCP/IP Extensible Structured Tokens

The extensible structured tokens defined by the TCP/IP subsystem are:

- ZTCI-MAP-ADD-ROUTE
- ZTCI-MAP-ADD-SUBNET
- ZTCI-MAP-ALTER-PROC
- ZTCI-MAP-ALTER-SUBNET
- ZTCI-MAP-ICMP-STATS
- ZTCI-MAP-INFO-PROC
- ZTCI-MAP-INFO-ROUTE
- ZTCI-MAP-INFO-SUBNET
- ZTCI-MAP-IP-STATS
- ZTCI-MAP-OPENS-INET
- ZTCI-MAP-ROUTE-STATS
- ZTCI-MAP-STATS-ROUTE
- ZTCI-MAP-STATS-SUBNET
- ZTCI-MAP-STATUS-PROC
- ZTCI-MAP-STATUS-PROTO
- ZTCI-MAP-STATUS-ROUTE
- ZTCI-MAP-STATUS-SUBNET
- ZTCI-MAP-TCP-STATS
- ZTCI-MAP-UDP-STATS
These tokens are described with the commands and responses in which they appear. For more information, see Section 7, Event Management.

TCP/IP Tokens in Error Lists

The TCP/IP subsystem returns standard error lists and does not define any subsystem-specific tokens that are found in error lists. The error numbers defined by the NonStop TCP/IP subsystem are described in Appendix A, Error Numbers and Error Lists.

For general information on the return token, error tokens, and error lists, see the SPI Common Extensions Manual. For command-specific information about the errors returned, see Command Descriptions on page 6-4.

TCP/IP Tokens in Event Messages

The TCP/IP subsystem defines these tokens that appear in event messages:

ZTCI-TKN-ENET-ADDR       ZTCI-TKN-ERR-DEV
ZTCI-TKN-ERR-DEV          ZTCI-TKN-SEV
ZTCI-TKN-FNAME            ZTCI-TKN-SIZE
ZTCI-TKN-IOPNAME          ZTCI-TKN-TEXT
ZTCI-TKN-XFER-DEV         ZTCI-TKN-XFER-DETL
ZTCI-TKN-NEW-PROC-ERR     ZTCI-TKN-XFER-UOW

These tokens are described with the event messages in which they appear in Section 7, Event Management.
This section describes each of the commands and responses in the programmatic control-and-inquiry interface to the TCP/IP subsystem. Each description includes the syntax, object types, command tokens, response tokens, and operational notes associated with the commands. In this section, command numbers, types, tokens, and values are represented in DDL. For a brief explanation of DDL as it applies to SPI, see the SPI Programming Manual.

The commands available in the programmatic interface allow an application to:

- Change the attributes of a TCP/IP process and subnets
- Start and stop the operation of TCP/IP subnets and routes
- Add or remove a subnet or route to or from the TCP/IP subsystem
- Obtain attribute values
- Obtain statistics and status information
- Obtain information about the object names known to the TCP/IP subsystem
- Trace the activity of a TCP/IP process or subnet
- Change the system resources used by a TCP/IP process
- Obtain the version of the TCP/IP subsystem

Table 6-1 on page 6-3 lists and gives the symbolic names of the commands available in the programmatic control-and-inquiry interface to the TCP/IP subsystem.

Commands by Object Type

The TCP/IP subsystem can recognize four object types in a control-and-inquiry command. The object types recognized vary with each command. Three of the object types: PROC, SUBNET, and ROUTE, have a hierarchical relationship, where SUBNET and ROUTE are peer objects that are subordinate to the PROC object. This hierarchy means that the PROC object type controls the SUBNET and ROUTE object types.

The fourth object type recognized by the TCP/IP subsystem is the NULL object. The NULL object type is not included in the object-type hierarchy and is used only when the object type is unnecessary, as in the GETVERSION command.
The TCP/IP subsystem accepts the PROC object type only in these control-and-inquiry interface commands:

ALTER        RESETSTATS
GETVERSION    STATISTICS
INFO          STATUS
LISTOPENS     TRACE

The TCP/IP subsystem accepts the SUBNET object type only in these control-and-inquiry interface commands:

ABORT         RESETSTATS
ADD            START
ALTER         STATISTICS
DELETE        STATUS
INFO          STOP
LISTOBJECTS   TRACE

The TCP/IP subsystem accepts the ROUTE object type only in these control-and-inquiry interface commands:

ABORT         RESETSTATS
ADD            START
DELETE        STATISTICS
INFO          STATUS
LISTOBJECTS   STOP

The TCP/IP subsystem accepts the NULL object type only in these control-and-inquiry interface commands:

GETVERSION    LISTOBJECTS

**Sensitive and Nonsensitive Commands**

Commands can be divided into two categories: sensitive and nonsensitive. Sensitive commands can change the state of the specified object; nonsensitive commands cannot. In addition, sensitive commands can be issued only by users in the SUPER user group or users that are in the same group as the owner of the process with which the target object is associated. Nonsensitive commands can be issued by any valid user on the system.
The sensitive commands available to a TCP/IP subsystem user are:

ABORT  RESETSTATS
ADD     START
ALTER   STOP
DELETE  TRACE

The nonsensitive commands available to a TCP/IP subsystem user are:

GETVERSION LISTOPENS
INFO    STATISTICS
LISTOBJECTS STATUS

<table>
<thead>
<tr>
<th>Command</th>
<th>Symbolic Name of Command</th>
<th>Symbolic Name of Object</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABORT</td>
<td>ZCOM-CMD-ABORT</td>
<td>ZCOM-OBJ-ROUTE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ZCOM-OBJ-SUBNET</td>
</tr>
<tr>
<td>ADD</td>
<td>ZCOM-CMD-ADD</td>
<td>ZCOM-OBJ-ADDRMAP</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ZCOM-OBJ-ROUTE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ZCOM-OBJ-SUBNET</td>
</tr>
<tr>
<td>ALTER</td>
<td>ZCOM-CMD-ALTER</td>
<td>ZCOM-OBJ-PROC</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ZCOM-OBJ-SUBNET</td>
</tr>
<tr>
<td>DELETE</td>
<td>ZCOM-CMD-DELETE</td>
<td>ZCOM-OBJ-ROUTE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ZCOM-OBJ-SUBNET</td>
</tr>
<tr>
<td>GETVERSION</td>
<td>ZCOM-CMD-GETVERSION</td>
<td>ZCOM-OBJ-NULL</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ZCOM-OBJ-PROC</td>
</tr>
<tr>
<td>INFO</td>
<td>ZCOM-CMD-INFO</td>
<td>ZCOM-OBJ-PROC</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ZCOM-OBJ-ROUTE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ZCOM-OBJ-SUBNET</td>
</tr>
<tr>
<td>LISTOBJECTS</td>
<td>ZCOM-CMD-LISTOBJECTS</td>
<td>ZCOM-OBJ-NULL</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ZCOM-OBJ-ROUTE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ZCOM-OBJ-SUBNET</td>
</tr>
<tr>
<td>LISTOPENS</td>
<td>ZCOM-CMD-LISTOPENS</td>
<td>ZCOM-OBJ-PROCESS</td>
</tr>
<tr>
<td>RESETSTATS</td>
<td>ZCOM-CMD-RESETSTATS</td>
<td>ZCOM-OBJ-PROCESS</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ZCOM-OBJ-ROUTE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ZCOM-OBJ-SUBNET</td>
</tr>
<tr>
<td>START</td>
<td>ZCOM-CMD-START</td>
<td>ZCOM-OBJ-ROUTE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ZCOM-OBJ-SUBNET</td>
</tr>
</tbody>
</table>
Command Descriptions

This section describes each command and its response in detail. The command descriptions appear in alphabetic order.

Within each command description, there is a box that contains the symbolic name of the command, the symbolic names of all object types accepted by the command, a list of tokens that can be present in the command buffer, and a list of tokens that can be present in the response buffer.

With the exception of the tokens ZSPI-TKN-DATALIST, ZSPI-TKN-ERRLIST, and ZSPI-TKN-ENDLIST, the order of the tokens listed in the command buffer and response buffer is not necessarily the order in which they will actually appear in a command or response. The token ZSPI-TKN-DATALIST, if present in a response, always appears at the beginning of a response record. The token ZSPI-TKN-ERRLIST, if present in a response, always appears at the beginning of an error list. The token ZSPI-TKN-ENDLIST always appears at the end of a response record started by the token ZSPI-TKN-DATALIST, or at the end of an error list started by the token ZSPI-TKN-ERRLIST.

The list of tokens in the response buffer, except for the context token (ZSPI-TKN-CONTEXT), represents the tokens that can be present in one response record. For more information on response records, and for information on the context token, see the SPI Programming Manual.

The notation used in the box for simple tokens is a shorthand version of the essential information given in the DDL TOKEN-CODE statement. Extensible structured tokens are defined by using DDL DEF statements.

For each token in the command or response buffer, and for each field of an extensible structured token in the command or response buffer, the box contains an indication of

<table>
<thead>
<tr>
<th>Command</th>
<th>Symbolic Name of Command</th>
<th>Symbolic Name of Object</th>
</tr>
</thead>
<tbody>
<tr>
<td>STATISTICS</td>
<td>ZCOM-CMD-STATISTICS</td>
<td>ZCOM-OBJ-PROC</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ZCOM-OBJ-ROUTE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ZCOM-OBJ-SUBNET</td>
</tr>
<tr>
<td>STATUS</td>
<td>ZCOM-CMD-STATUS</td>
<td>ZCOM-OBJ-PROC</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ZCOM-OBJ-ROUTE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ZCOM-OBJ-SUBNET</td>
</tr>
<tr>
<td>STOP</td>
<td>ZCOM-CMD-STOP</td>
<td>ZCOM-OBJ-ROUTE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ZCOM-OBJ-SUBNET</td>
</tr>
<tr>
<td>TRACE</td>
<td>ZCOM-CMD-TRACE</td>
<td>ZCOM-OBJ-PROC</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ZCOM-OBJ-SUBNET</td>
</tr>
</tbody>
</table>
whether or not the token or field must be or is always present in the command, and other dependencies.

For a description of the notation scheme, see Notation Conventions on page xi. Error lists are indicated in the box by following the token ZSPI-TKN-ERRLIST by an ellipsis (...) and the token ZSPI-TKN-ENDLIST. Tokens that appear in a response record, including the token ZSPI-TKN-ERRLIST, are indented following the token ZSPI-TKN-DATALIST to show that they are contained within the data list.

The value of the token ZSPI-TKN-MAXRESP determines whether a data list is used in a response. For more information on data lists and error lists, see the SPI Programming Manual.

Following the box, the tokens in the command buffer and the tokens in the response buffer are described in the same order as they appear in the box.

**ABORT Command**

The ABORT command terminates the operation of the specified object and clears any existing connection to it. When the subsystem finishes processing the command, the specified object is placed in the STOPPED summary state. You can abort only a subnet or a route.

You can use an object-name template to abort multiple subnets or multiple routes. For more details, see Object-Name Templates on page 3-6.

When using the ABORT command, note that the termination is done without regard to processing currently in progress for the specified object(s). All activities being performed by the object(s) are halted, and files and listings might be left in an inconsistent or incomplete state.

<table>
<thead>
<tr>
<th>Command</th>
<th>ZCOM-CMD-ABORT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object Type</td>
<td>ZCOM-OBJ-SUBNET or ZCOM-OBJ-ROUTE</td>
</tr>
<tr>
<td>Tokens in Command Buffer</td>
<td></td>
</tr>
<tr>
<td>ZSPI-TKN-SSID</td>
<td>token-type ZSPI-TYP-SSID.</td>
</tr>
<tr>
<td>ZCOM-TKN-OBJNAME</td>
<td>token-type ZSPI-TYP-STRING.</td>
</tr>
<tr>
<td>ZSPI-TKN-MAXRESP</td>
<td>token-type ZSPI-TYP-INT.</td>
</tr>
<tr>
<td>ZSPI-TKN-CONTEXT</td>
<td>token-type ZSPI-TYP-BYTESTRING.</td>
</tr>
<tr>
<td>ZSPI-TKN-ALLOW-TYPE</td>
<td>token-type ZSPI-TYP-ENUM.</td>
</tr>
<tr>
<td>ZSPI-TKN-COMMENT</td>
<td>token-type ZSPI-TYP-STRING.</td>
</tr>
</tbody>
</table>
**Tokens in Command Buffer**

**ZSPI-TKN-SSID**

specifies the subsystem ID, which your management application must pass to SSINIT. This token is described in [SPI Standard Definitions](#) on page 5-1, and in the [SPI Programming Manual](#).

**ZCOM-TKN-OBJNAME**

specifies the subnet or route that is to be terminated. Your application must provide this token in the command. The object name can include wild-card characters. Using wild-card characters allows the command to affect multiple objects. This token is described more fully in [Data Communications Standard Definitions](#) on page 5-9.

**(ZSPI-TKN-) MAXRESP, CONTEXT, ALLOW-TYPE, and COMMENT**

are standard SPI tokens that your application can provide. These tokens are described in [SPI Standard Definitions](#) on page 5-1.

**Tokens in Response Buffer**

**ZSPI-TKN-SSID**

is an echo of the subsystem ID value provided in the command by your application. For more details, see [SPI Standard Definitions](#) on page 5-1.

**ZSPI-TKN-SERVER-VERSION**

indicates the software release version of the TCP/IP subsystem. For more details, see [SPI Standard Definitions](#) on page 5-1.
ZSPI-TKN-DATALIST

indicates the beginning of a data list. The ZSPI-TKN-MAXRESP token must have a value other than 0 for data lists to be included in the response. For more details, see SPI Standard Definitions on page 5-1.

ZCOM-TKN-OBJNAME

specifies the name of a TCP/IP subnet or route terminated by the command. Unless there is an error that prevents the ABORT command from being attempted at all, this token is present in every response record. If you have used an object-name template, a separate ZCOM-TKN-OBJNAME token is returned for each subnet or route that meets the template requirements. ZCOM-TKN-OBJNAME is described more fully in Data Communications Standard Definitions on page 5-9.

ZCOM-TKN-OBJTYPE

specifies the type of object terminated. The possible values are ZCOM-OBJ-SUBNET and ZCOM-OBJ-ROUTE. This token is present in every response record. The value of the ZCOM-TKN-OBJTYPE token is the same as the value of the ZSPI-TKN-OBJECT-TYPE token in the header. For more details, see Data Communications Standard Definitions on page 5-9.

ZSPI-TKN-RETCODE

specifies whether the command has been completed successfully. The value of this token remains zero (ZCOM-ERR-OK) when the command is completed successfully with or without a warning. Note that if the empty response (ZCOM-ERR-EMPT-RSP) is returned, the value of this token is nonzero, although the command is completed successfully.

If the command is not completed successfully, the value of this token indicates the reason the command failed. Always check the contents of the error list to determine the token or value in error. For details on the ZCOM return codes and associated error lists, refer to the SPI Common Extensions Manual. The possible nonzero values of RETCODE in response to the ABORT command are:

- ZCOM-ERR-TKN-REQ (-29)
- ZCOM-ERR-TKN-DUP (-27)
- ZCOM-ERR-TKN-CODE-INV (-26)
- ZCOM-ERR-SSID-INV (-24)
- ZCOM-ERR-SPI-ERR (-23)
- ZCOM-ERR-SECUR-VIOL (-22)
- ZCOM-ERR-OBJ-NOT-SUPP (-18)
- ZCOM-ERR-OBJ-NOT-FOUND (-17)
- ZCOM-ERR-OBJ-ALRDY-IN-SUMSTATE (-16)
ZCOM-ERR-CMD-TOO-LARGE (-8)
ZCOM-ERR-CMD-NOT-SUPP (-5)

ZSPI-TKN-ERRLIST
indicates the beginning of an error list. Within the error list, the object that experienced the error is listed. For more details, see SPI Standard Definitions on page 5-1.

ZSPI-TKN-ENDLIST
indicates the end of a data list or an error list. For more details, see SPI Standard Definitions on page 5-1.

ZSPI-TKN-CONTEXT
specifies whether there are more response messages to be received for a response. If this token is present, the response continues to the next response message. If this token is not present, this response is the last to be received. For more details, see SPI Standard Definitions on page 5-1.

Operational Notes
• ABORT is a sensitive command.
• When the ABORT command is completed, the subnet or route is placed in the STOPPED summary state.
• For more controlled termination of a subnet or route, use the STOP command.
• To reinitiate the operation of a subnet or route, use the START command.
• Use the STATUS command to determine the state of an object.

ADD Command
The ADD command adds a subnet, an address map, or a route to the TCP/IP subsystem. This command is not valid for the PROC object. The PROC object must be in the STARTED summary state; that is, it must exist when adding either a subnet or a route. If the process is not in the STARTED state, use the RUN command to create it. When the subsystem finishes processing the command (without errors), the subnet or route is placed in the STOPPED summary state.

Object-name templates do not apply to this command.

When your management application issues the ADD command, the subsystem checks the configurable attributes. These attributes are defined in the ZTCI-MAP-ADD-SUBNET, ZTCI-MAP-ADD-ADDRMAP, or TCI-MAP-ADD-ROUTE extensible structured tokens. If the attributes are valid, the subsystem verifies that the specified object is not already defined and that the number of objects that you add
does not exceed the limit defined for each object type. If either of these checks fails, the TCP/IP subsystem returns an error.

The maximum number of subnets you can add is 17 (4 Ethernet, 8 SNAP (4 token ring and 4 Ethernet), 4 X.25, and 1 loopback). Adding routes is optional because the TCP/IP subsystem automatically adds a route when you add a subnet. The maximum number of routes you can add depends on the resources currently in use.

**Note.** For information on NonStop availability considerations, see the *TCP/IP Configuration and Management Manual.*

TCP/IP subsystem automatically adds a route when you add a subnet. The maximum number of routes you can add depends on the resources currently in use.

<table>
<thead>
<tr>
<th>Command</th>
<th>ZCOM-CMD-ADD</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Object Type</strong></td>
<td>ZCOM-OBJ-SUBNET or ZCOM-OBJ-ROUTE or ZCOM-OBJ-ADDRMAP</td>
</tr>
</tbody>
</table>
Tokens in Command Buffer

<table>
<thead>
<tr>
<th>Token Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZSPI-TKN-SSID</td>
<td>token-type ZSPI-TYP-SSID.</td>
</tr>
<tr>
<td>ZTCI-MAP-ADD-SUBNET</td>
<td>!{A}</td>
</tr>
<tr>
<td>ZTCI-DDL-ADD-SUBNET</td>
<td>!OBJ-SUBNET</td>
</tr>
<tr>
<td>ZTCI-DDL-ADD-SUBNET</td>
<td>def ZTCI-DDL-ADD-SUBNET.</td>
</tr>
<tr>
<td>ZTYPE</td>
<td>type ZSPI-DDL-ENUM.</td>
</tr>
<tr>
<td>ZIP-ADDR</td>
<td>type ZSPI-DDL-INT2.</td>
</tr>
<tr>
<td>ZIOPNAME</td>
<td>type ZSPI-DDL-FNAME32.</td>
</tr>
<tr>
<td>ZSU-NAME-IP</td>
<td>type ZSPI-DDL-BOOLEAN</td>
</tr>
<tr>
<td>ZSU-NAME</td>
<td>type ZSPI-DDL-CHAR8</td>
</tr>
<tr>
<td>ZFORCE-QIO-IP</td>
<td>type ZSPI-DDL-BOOLEAN</td>
</tr>
<tr>
<td>ZFORCE-QIO</td>
<td>type ZSPI-DDL-BOOLEAN</td>
</tr>
<tr>
<td>ZUSE-IRDP-IP</td>
<td>type ZSPI-DDL-BOOLEAN</td>
</tr>
<tr>
<td>ZUSE-IRDP</td>
<td>type ZSPI-DDL-BOOLEAN</td>
</tr>
<tr>
<td>ZTCI-DDL-ADD-ROUTE</td>
<td>!{A}</td>
</tr>
<tr>
<td>ZTCI-DDL-ADD-ROUTE</td>
<td>def ZTCI-DDL-ADD-ROUTE.</td>
</tr>
<tr>
<td>ZDEST-IP-ADDR</td>
<td>type ZSPI-DDL-INT2.</td>
</tr>
<tr>
<td>ZGW-IP-ADDR</td>
<td>type ZSPI-DDL-INT2.</td>
</tr>
<tr>
<td>ZIF-HOST</td>
<td>type ZSPI-DDL-BOOLEAN</td>
</tr>
<tr>
<td>ZTCI-DDL-ADD-ADDRMAP</td>
<td>!{A}</td>
</tr>
<tr>
<td>ZTCI-DDL-ADD-ADDRMAP</td>
<td>def ZTCI-DDL-ADD-ADDRMAP.</td>
</tr>
<tr>
<td>ZIP-ADDR</td>
<td>type ZSPI-DDL-INT2.</td>
</tr>
<tr>
<td>ZX121-ADDR</td>
<td>type ZSPI-DDL-CHAR16</td>
</tr>
<tr>
<td>ZCOM-TKN-OBJNAME</td>
<td>token-type ZSPI-TYP-STRING.</td>
</tr>
<tr>
<td>ZSPI-TKN-MAXRESP</td>
<td>token-type ZSPI-TYP-INT.</td>
</tr>
<tr>
<td>ZSPI-TKN-COMMENT</td>
<td>token-type ZSPI-TYP-STRING.</td>
</tr>
</tbody>
</table>

Tokens in Response Buffer

<table>
<thead>
<tr>
<th>Token Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZSPI-TKN-SSID</td>
<td>token-type ZSPI-TYP-SSID.</td>
</tr>
<tr>
<td>ZSPI-TKN-SERVER-VERSION</td>
<td>token-type ZSPI-TYP-UINT.</td>
</tr>
<tr>
<td>ZSPI-TKN-DATA LIST</td>
<td>token-type ZSPI-TYP-LIST.</td>
</tr>
<tr>
<td>ZCOM-TKN-OBJNAME</td>
<td>token-type ZSPI-TYP-STRING.</td>
</tr>
<tr>
<td>ZCOM-TKN-OBJTYPE</td>
<td>token-type ZSPI-TYP-ENUM.</td>
</tr>
<tr>
<td>ZSPI-TKN-RETCODE</td>
<td>token-type ZSPI-TYP-ENUM.</td>
</tr>
<tr>
<td>ZSPI-TKN-ERRLIST</td>
<td>token-type ZSPI-TYP-LIST.</td>
</tr>
<tr>
<td>ZSPI-TKN-ENDLIST</td>
<td>token-type ZSPI-TYP-SSCTL.</td>
</tr>
<tr>
<td>ZSPI-TKN-ENDLIST</td>
<td>token-type ZSPI-TYP-SSCTL.</td>
</tr>
</tbody>
</table>
Tokens in Command Buffer

ZSPI-TKN-SSID

specifies the subsystem ID, which your management application must pass to SSINIT. This token is described in SPI Standard Definitions on page 5-1, and in the SPI Programming Manual.

ZTCI-MAP-ADD-SUBNET

is the extensible structured token that defines the configurable attributes for a subnet to be added. It has the token number ZTCI-TNM-ADD-SUBNET and the DDL definition ZTCI-DDL-ADD-SUBNET. This token must be in the command if the ZCOM-OBJ-SUBNET value has been specified. Duplicate occurrences are not allowed.

Each attribute is defined in a separate field. All fields are required.

The fields in the ZTCI-MAP-ADD-SUBNET token are:

ZTYPE

is an enumerated value that specifies the type of subnet to be added. The possible subnet type values (ZTCI-VAL-SUBNET-TYPE-) are:

ENET

indicates that the subnet to be added is associated with a SLSA LIF using an Ethernet filter.

LOOP

indicates that the subnet to be added is associated with the loopback driver and is for testing purposes.

SNAP

indicates that the subnet to be added is associated with a SLSA LIF using an LLC1 filter.

X25

indicates that the subnet to be added is associated with an X25AM device.

ZIP-ADDR

is a 32-bit integer that specifies the address of the subnet or the loopback address (which is used for testing purposes). Either value is specified as an IP address. The conventional loopback address is 127.1.
ADD Command

ZIOPNAME

is a 32-byte internal file name that specifies the name of the SLSA LIF or the X25AM device with which the subnet interfaces.

The TCP/IP subsystem automatically adds a route with a destination address equal to the value specified in the network portion of the ZIP-ADDR field converted to its broadcast form, and a gateway address equal to the value specified as the IP address of the subnet in the ZIP-ADDR field whenever a subnet is added.

**Note.** The SLSA LIF’s format is not that of a real internal file name. If you are using the SCF DETAIL CMDBUFFER ON command and DETAIL RSPBUFFER ON commands to examine this token, it will not display correctly for the SLSA LIF.

ZSU-NAME

is the subdevice name. The SUNAME attribute is valid only for TYPE X25 subnets.

ZFORCE-QIO

is a Boolean variable that specifies whether the SUBNET object should only use the QIO direct interface to the controller. If the value is ZSPI-VAL-TRUE, the SUBNET object uses the QIO direct interface to the controller. If the value is ZSPI-VAL-FALSE, the SUBNET object does not use the QIO direct interface to the controller.

ZUSE-IRDP

is a Boolean variable that specifies whether the SUBNET object should enable its router discovery interface. If the value is ZSPI-VAL-TRUE, the SUBNET object uses the router discovery interface. If the value is ZSPI-VAL-FALSE, the SUBNET object does not use the router discovery interface. The default is disabled.

ZTCI-MAP-ADD-ROUTE

is the extensible structured token that defines the configurable attributes for a route to be added. It has the token number ZTCI-TNM-ADD-ROUTE and the DDL definition ZTCI-DDL-ADD-ROUTE. This token must be in the command if the ZCOM-OBJ-ROUTE value has been specified. Duplicate occurrences are not allowed.

Each attribute is defined in a separate field. All fields are required.

The fields in the ZTCI-MAP-ADD-ROUTE token are:

ZDEST-IP-ADDR

is a 32-bit integer that specifies the Internet address of the destination to be associated with this route. Depending on the value specified, this address can
indicate either a single host that can be reached through the gateway specified in the ZGW-IP-ADDR field, an entire network that can be reached through the gateway specified in the ZGW-IP-ADDR field, or that the ZGW-IP-ADDR field that follows is the default gateway. To specify a host, specify the complete IP address. To specify a network, assign the local portion of the IP address a value of zero. (This zero value acts as a wild-card character, representing all hosts on the network specified in the network portion of the address.) To specify the default gateway, assign a destination address of 0 (0.0.0.0). The default gateway is the gateway to which all packets with addresses for which routes cannot be determined are sent.

ZGW-IP-ADDR

is a 32-bit integer that specifies the Internet address of the gateway through which the destination specified in ZDEST-IP-ADDR can be reached. When a destination address of 0 has been assigned to the preceding ZDEST-IP-ADDR field, this value specifies the Internet address of the default gateway.

ZIF-HOST

is a Boolean variable that specifies whether the route connects to a host. If the value is ZSPI-VAL-TRUE, the route connects to a host; that is, it is a point-to-point connection. If the value is ZSPI-VAL-FALSE, the route does not connect to a host but is broadcast instead.

ZCOM-TKN-OBJNAME

specifies the subnet or route to be added. Your application must provide this token in the command. The object name cannot be an object-name template. This token is described more fully in Data Communications Standard Definitions on page 5-9.

(ZSPI-TKN-) MAXRESP and COMMENT

are standard SPI tokens that your application can provide. These tokens are described in SPI Standard Definitions on page 5-1.

ZTCI-MAP-ADD-ADDRMAP

is the extensible structured token that defines the configurable attributes for an address map to be added. The address map specifies the name for the entry in the TCP/IP X.25 address table; that is, an internal table mapping IP addresses to X.121 addresses. It has the token number ZTCI-TNM-ADD-ADDRMAP and the DDL definition ZTCI-DDL-ADD-ADDRMAP. This token must be in the command if the ZCOM-OBJ-ADDRMAP value has been specified. Duplicate occurrences are not allowed. The name must start with a pound sign (#) and can be up to seven alphanumeric characters. The first character following the pound sign must be a letter. You must assign a unique address name to each address map associated with a particular process. Names beginning with #ZADR are reserved for use by the process when creating dynamic entries.

Each attribute is defined in a separate field. All fields are required.
The fields in the ZTCI-MAP-ADD-ADDRMAP token are:

**ZIP-ADDR**

is an integer that specifies the Internet address in dotted decimal format. An Internet address is a 4-octet (32-bit) numeric value identifying a particular network (network address portion) and a local host on that network (local address portion) as defined in RFC1010. 25 address table.

**ZX121-ADDR**

is a 16-character string that specifies the X.25 address.

### Tokens in Response Buffer

**ZSPI-TKN-SSID**

is an echo of the subsystem ID value provided in the command by your application. For more details, see [SPI Standard Definitions](#) on page 5-1.

**ZSPI-TKN-SERVER-VERSION**

indicates the software release version of the TCP/IP subsystem. For more details, see [SPI Standard Definitions](#) on page 5-1.

**ZSPI-TKN-DATALIST**

indicates the beginning of a data list. The ZSPI-TKN-MAXRESP token must have a value other than 0 for data lists to be included in the response. For more details, see [SPI Standard Definitions](#) on page 5-1.

**ZCOM-TKN-OBJNAME**

specifies the name of the TCP/IP subnet or route added to the subsystem by the command. Unless there is an error that prevents the ADD command from being attempted at all, this token is present in every response record. ZCOM-TKN-OBJNAME is described more fully in [Data Communications Standard Definitions](#) on page 5-9.

**ZCOM-TKN-OBJTYPE**

specifies the type of object added. You can add a subnet or a route; therefore, the value of this token can be ZCOM-OBJ-SUBNET or ZCOM-OBJ-ROUTE. If there is a data list with an error list in it, the ZCOM-TKN-OBJTYPE token is present in both lists. Note that the header token ZSPI-TKN-OBJECT-TYPE has the same value as ZCOM-TKN-OBJTYPE. For more details, see [Data Communications Standard Definitions](#) on page 5-9.
ZSPI-TKN-RETCODE

specifies whether the command has been completed successfully. The value of this token remains zero (ZCOM-ERR-OK) when the command is completed successfully with or without a warning.

If the command is not completed successfully, the value of this token indicates the reason the command failed. Always check the contents of the error list to determine the token or value in error. For details on the ZCOM return codes and associated error lists, see the SPI Common Extensions Manual. For details on the ZTCI return codes and associated error lists, see Appendix A, Error Numbers and Error Lists. The possible nonzero values of RETCODE in response to the ADD command are:

- ZCOM-ERR-TKN-REQ (-29)
- ZCOM-ERR-TKN-DUP (-27)
- ZCOM-ERR-TKN-CODE-INV (-26)
- ZCOM-ERR-SSID-INV (-24)
- ZCOM-ERR-SPI-ERR (-23)
- ZCOM-ERR-SECUR-VIOL (-22)
- ZCOM-ERR-OBJNAME-INV (-19)
- ZCOM-ERR-OBJ-NOT-SUPP (-18)
- ZCOM-ERR-OBJ-ALRDY-DEF (-15)
- ZCOM-ERR-NO-MEM-SPACE (-14)
- ZCOM-ERR-FS-ERR (-10)
- ZCOM-ERR-CMD-TOO-LARGE (-8)
- ZCOM-ERR-CMD-NOT-SUPP (-5)
- ZTCI-ERR-INV-FNAME (1)
- ZTCI-ERR-ENET-ADDR-NOT-AVAIL (6)
- ZTCI-ERR-DUP-ADDR (7)
- ZTCI-ERR-FORCE-QIO-TOO-MANY (15)
- ZTCI-ERR-FORCE-QIO-IOP-WRG-CFG (16)
- ZTCI-ERR-FORCE-QIO-IOP-WRG-CPU (17)
- ZTCI-ERR-FORCE-QIO-OPEN-ERR (18)
ALTER Command

The ALTER command changes attribute values associated with the specified object. You can change attributes for a process and a subnet, but not for a route.

The specified object must be in the STOPPED summary state before the ALTER command is issued. Object-name templates do not apply to this command.

<table>
<thead>
<tr>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZCOM-CMD-ALTER</td>
</tr>
</tbody>
</table>
Object Type

ZCOM-OBJ-PROC or ZCOM-OBJ-SUBNET or ZCOM-OBJ-ADDRMAP

Tokens in Command Buffer

ZSPI-TKN-SSID token-type ZSPI-TYP-SSID. !{}
ZTCI-MAP-ALTER-ADDRMAP !{A}

def ZTCI-DDL-ALTER-ADDRMAP !Specify at least one set of field values.
  02 ZIP-ADDR-IP type ZSPI-DDL-BOOLEAN.
  02 ZIP-ADDR type ZSPI-DDL-INT2.
  02 ZX121-ADDR-IP type ZSPI-DDL-BOOLEAN.
  02 ZX121-ADDR type ZSPI-DDL-CHAR16.
end.

!OBJ-PROC

def ZTCI-DDL-ALTER-PROC. !Specify at least one set of field values.
  02 ZTCP-SEND-SPACE-IP type ZSPI-DDL-BOOLEAN.
  02 ZTCP-SEND-SPACE type ZSPI-DDL-INT.
  02 ZTCP-RECV-SPACE-IP type ZSPI-DDL-BOOLEAN.
  02 ZTCP-RECV-SPACE type ZSPI-DDL-INT.
  02 ZUDP-SEND-SPACE-IP type ZSPI-DDL-BOOLEAN.
  02 ZUDP-SEND-SPACE type ZSPI-DDL-INT.
  02 ZUDP-RECV-SPACE-IP type ZSPI-DDL-BOOLEAN.
  02 ZUDP-RECV-SPACE type ZSPI-DDL-INT.
  02 ZDELAY-ACKS-IP type ZSPI-DDL-BOOLEAN.
  02 ZDELAY-ACKS type ZSPI-DDL-BOOLEAN.
  02 ZDELAY-ACKS-TIME-IP type ZSPI-DDL-BOOLEAN.
  02 ZDELAY-ACKS-TIME type ZSPI-DDL-INT.
  02 ZHOSTID-IP type ZSPI-DDL-BOOLEAN.
  02 ZHOSTID type ZSPI-DDL-INT2.
  02 ZHOSTNAME-IP type ZSPI-DDL-BOOLEAN.
  02 ZHOSTNAME type ZSPI-DDL-CHAR50.
  02 ZTCP-KEEPIDLE-IP type ZSPI-DDL-BOOLEAN.
  02 ZTCP-KEEPIDLE type ZSPI-DDL-INT.
  02 ZTCP-KEEPINTVL-IP type ZSPI-DDL-BOOLEAN.
  02 ZTCP-KEEPINTVL type ZSPI-DDL-INT.
  02 ZTCP-KEEPCNT-IP type ZSPI-DDL-BOOLEAN.
  02 ZTCP-KEEPCNT type ZSPI-DDL-INT.
  02 ZDEBUG-ENABLE-IP type ZSPI-DDL-BOOLEAN.
  02 ZDEBUG-ENABLE type ZSPI-DDL-BOOLEAN.
  02 ZFULL-DUMP-IP type ZSPI-DDL-BOOLEAN.
  02 ZFULL-DUMP type ZSPI-DDL-BOOLEAN.
end.
ALTER Command

**Tokens in Command Buffer**

ZSPI-TKN-SSID

specifies the subsystem ID, which your management application must pass to SSINIT. This token is described in SPI Standard Definitions on page 5-1, and in the SPI Programming Manual.

ZTCI-MAP-ALTER-PROC

is the extensible structured token that defines the configurable attributes to be altered for the process. It has the token number ZTCI-TNM-ALTER-PROC and the DDL definition ZTCI-DDL-ALTER-PROC. This token must be in the command when the specified object is ZCOM-OBJ-PROC. Duplicate occurrences of these fields are not allowed.

Each attribute is defined in a separate field and, if specified, must have the corresponding IS-PRESENT (-IP) field defined with the value ZSPI-VAL-TRUE. At least one set of fields, that is, the -IP field with the corresponding attribute field

```plaintext
ZTCI-MAP-ALTER-SUBNET
!{A}
OBJ-SUBNET

def ZTCI-DDL-ALTER-SUBNET. !Specify at least one set of field values.
  02 ZIP-ADDR-IP       type ZSPI-DDL-BOOLEAN.
  02 ZIP-ADDR          type ZSPI-DDL-INT2.
  02 ZSUBNET-MASK-IP   type ZSPI-DDL-BOOLEAN.
  02 ZSUBNET-MASK      type ZSPI-DDL-INT2.
  02 ZSU-NAME-IP       type ZSPI-DDL-BOOLEAN.
  02 ZSU-NAME          type ZSPI-DDL-CHAR8.
  02 ZUSE-IRDP-IP      type ZSPI-DDL-BOOLEAN.
  02 ZUSE-IRDP         type ZSPI-DDL-BOOLEAN.

end.

ZCOM-TKN-OBJNAME          token-type ZSPI-TYP-STRING. !{}
ZSPI-TKN-MAXBRESP         token-type ZSPI-TYP-INT.
ZSPI-TKN-COMMENT          token-type ZSPI-TYP-STRING.

Tokens in Response Buffer

ZSPI-TKN-SSID             token-type ZSPI-TYP-SSID. !{}
ZSPI-TKN-SERVER-VERSION   token-type ZSPI-TYP-UINT. !{}
ZSPI-TKN-DATALIST         token-type ZSPI-TYP-LIST.
  ZCOM-TKN-OBJNAME        token-type ZSPI-TYP-STRING. !{}
  ZCOM-TKN-OBJTYPE        token-type ZSPI-TYP-ENUM. !{}
  ZSPI-TKN-RETCODE        token-type ZSPI-TYP-ENUM. !{}
  ZSPI-TKN-ERRLIST        token-type ZSPI-TYP-LIST. !...
  ZSPI-TKN-ENDLIST        token-type ZSPI-TYP-SSCTL.
  ZSPI-TKN-ENDLIST        token-type ZSPI-TYP-SSCTL.
```
must be specified if the ALTER command is to be effective. The fields in the ZTCI-MAP-ALTER-PROC extensible structured token are:

**Zxxx-IP**

is a Boolean variable that specifies whether the succeeding field is to be altered, where *xxx* specifies the attribute field. The possible values are:

**ZSPI-VAL-TRUE**

indicates that the succeeding attribute field is to be altered.

**ZSPI-VAL-FALSE**

indicates that the succeeding attribute field is not to be altered.

**ZTCP-SEND-SPACE**

is an integer that specifies the amount of data that can be buffered in the Transmission Control Protocol (TCP) Layer when sending data to a remote site. The size must be in the range 512 bytes through 12K bytes. The default window size is 8K bytes.

**ZTCP-RECV-SPACE**

is an integer that specifies the amount of data that can be buffered in the TCP Layer when receiving data from a remote site. The size must be in the range 512 bytes through 12K bytes. The default is 8K bytes. Although some applications may use values in the lower range, for general use this value should not be set less than 2K.

**ZUDP-SEND-SPACE**

is an integer that specifies the amount of data that can be buffered in the User Datagram Protocol (UDP) Layer when sending data to a remote site. The size must be in the range 512 bytes through 12K bytes. The default window size is 8K bytes. ZUDP-RECV-SPACE is an integer that specifies the amount of data that can be buffered in the UDP Layer when receiving data from a remote site. This is used when there have been no user receive requests. The size must be in the range 512 bytes through 12K bytes. The default is 8K bytes. Although some applications may use values in the lower range, for general use this value should not be set less than 2K.

**ZDELAY-ACKS**

is a Boolean variable that specifies whether an acknowledgment (ACK) is delayed when a TCP packet is received from a remote site. The possible values are:
ZSPI-VAL-TRUE

specifies that an ACK for a TCP packet should be delayed until the TCP receive window is completely filled, or one interval as defined by ZDELAY-ACKS-TIME has expired, whichever occurs first.

By delaying the ACK, more than one packet can be acknowledged with a single ACK, thereby reducing network traffic. This is the default. If this value is set, see the ZDELAY-ACKS-TIME field for the time interval between ACKs.

ZSPI-VAL-FALSE

specifies that an ACK should be returned as soon as a TCP packet is received.

ZDELAY-ACKS-TIME

is an integer that specifies the time, in units of .01 seconds, that the TCP/IP interface waits before sending an ACK for a TCP packet received from a remote site. This field is only meaningful if the ZDELAY-ACKS field has a value of ZSPI-VAL-TRUE. The value can be in the range of 1 through 100; however, it is recommended that the value not exceed 20 (200 milliseconds). The default value is 5 (50 milliseconds).

ZHOSTID

is a 32-bit integer that identifies the host. Typically, this value is the host number portion of the IP address that has been assigned to the host. The default is 0D. Note that this value is only used to support the gethostid socket call.

ZHOSTNAME

is a character string of no more than 50 characters that identifies the host on which the TCP/IP process is running. The default is null. Note that this value is only used to support the gethostname socket call.

ZTCP-KEEPIDLE

is an integer that specifies the amount of time in seconds before TCP will issue a keep-alive packet on sockets that have enabled this option. The default is 45 seconds. The range is 10 through 7200 seconds.

ZTCP-KEEPCNT

is an integer that specifies the number of times a keep-alive packet will be sent without receiving an acknowledgment, after which the TCP connection will be dropped. The default is 8. The range is 1 through 20.
**ZTCP-KEEPINTVL**

is an integer that specifies the time interval in seconds between retransmissions of unacknowledged keep-alive packets. The default is 45 seconds. The range is 10 through 1260 seconds.

**ZDEBUG-ENABLE**

is a Boolean variable that specifies whether the debug option will be enabled. This option is used internally by HP Support and Development personnel to enable the display of more TCP internal information for debugging purposes.

**ZSPI-VAL-TRUE**

indicates that the debug option is to be enabled.

**ZSPI-VAL-FALSE**

indicates that the debug option is not to be enabled.

**ZFULL-DUMP**

is a Boolean variable that specifies whether the QIO segment is saved when the TCP/IP process abends. The default is ON. When disabled, the TCP/IP process will save only its stack when it abends. This will conserve disk space over a full dump. The preferred setting for this parameter is ON.

**ZSPI-VAL-TRUE**

indicates that the QIO segment is to be saved when the TCP/IP process abends.

**ZSPI-VAL-FALSE**

indicates that the QIO segment is not to be saved when the TCP/IP process abends.

**ZTCI-MAP-ALTER-SUBNET**

is the extensible structured token that defines the configurable attributes to be altered for the subnet. It has the token number ZTCI-TNM-ALTER-SUBNET and the DDL definition ZTCI-DDL-ALTER-SUBNET. This extensible structured token must be in the command when the specified object is ZCOM-OBJ-SUBNET. Duplicate occurrences of the fields are not allowed.

Each attribute is defined in a separate field and, if specified, must have the corresponding IS-PRESENT (-IP) field defined with the value ZSPI-VAL-TRUE. At least one set of fields, that is, the IP field with the corresponding attribute field must be specified if the ALTER command is to be effective. The fields in the ZTCI-MAP-ALTER-SUBNET extensible structured token are:
**Zxxx-IP**

is a Boolean variable that specifies whether the succeeding field is to be altered, where \( xxx \) specifies the attribute field. The possible values are:

**ZSPI-VAL-TRUE**

indicates that the succeeding attribute field is to be altered.

**ZSPI-VAL-FALSE**

indicates that the succeeding attribute field is not to be altered.

**ZIP-ADDR**

is a 32-bit integer that specifies the address of the subnet or the loopback address. Either value is specified as an IP address. The conventional loopback address (which is used for testing) is 127.1.

**ZSUBNET-MASK**

is a 32-bit integer that specifies the subnet mask to be used with this subnet. A subnet mask identifies which part of the IP local address represents the subnet number and which part represents the host ID. If bits in the subnet mask are set to 1, the corresponding bits in the IP address are to be part of the network (and subnet) address. If bits in the subnet mask are set to 0, the corresponding bits in the IP address are to be part of the host ID (that is, that portion of the local address masked with 1s identifies the subnet, and the remainder of the local address uniquely identifies a host connected to the subnet). For further details on subnet addressing, subnet masking, and the various classes of IP addresses, see the *TCP/IP Configuration and Management Manual*.

**ZSU-NAME**

is the subdevice name. Previously, the SUNAME had to be set using a TACL PARAM command before the subnet could be added using SCF. With this RVU, you cannot use PARAM commands in the TACL environment to specify SUNAME or X25ADDR attributes for TCP/IP. The SUNAME attribute is valid only for TYPE X25 subnets.

**ZUSE-IRDP**

is a Boolean variable that specifies whether the subnet should enable its router discovery interface. The default is OFF. Possible values are:

**ZSPI-VAL-TRUE**

indicates that the router discovery interface is to be enabled.

**ZSPI-VAL-FALSE**

indicates that the router discovery interface is not to be enabled.
ZCOM-TKN-OBJNAME

specifies the process or subnet to be altered. Your application must provide this
token in the command. The object name cannot be an object-name template. This
token is described more fully in Data Communications Standard Definitions on
page 5-9.

(ZSPI-TKN-) MAXRESP and COMMENT

are standard SPI tokens that your application can provide. These tokens are
described in SPI Standard Definitions on page 5-1.

Tokens in Response Buffer

ZSPI-TKN-SSID

is an echo of the subsystem ID value provided in the command by your application.
For more details, see SPI Standard Definitions on page 5-1.

ZSPI-TKN-SERVER-VERSION

indicates the software release version of the TCP/IP subsystem. For more details,
see SPI Standard Definitions on page 5-1.

ZSPI-TKN-DATALIST

indicates the beginning of a data list. The ZSPI-TKN-MAXRESP token must have a
value other than 0 for data lists to be included in the response. For more details,
see SPI Standard Definitions on page 5-1.

ZCOM-TKN-OBJNAME

specifies the name of the TCP/IP process or subnet altered by the command.
Unless there is an error that prevents the ALTER command from being attempted
at all, this token is present in every response record. ZCOM-TKN-OBJNAME is
described more fully in Data Communications Standard Definitions on page 5-9.

ZCOM-TKN-OBJTYPE

specifies the type of object altered. This token has the value ZCOM-OBJ-PROC or
ZCOM-OBJ-SUBNET. For more details, see Data Communications Standard
Definitions on page 5-9.

ZSPI-TKN-RETCODE

specifies whether the command has been completed successfully. The value of
this token remains zero (ZCOM-ERR-OK) when the command is completed
successfully with or without a warning. Note that if the empty response (ZCOM-
ERR-EMPT-RSP) is returned, the value of this token is nonzero although the
command is completed successfully.
If the command is not completed successfully, the value of this token indicates the reason the command failed. Always check the contents of the error list to determine the token or value in error. For details on the ZCOM return codes, see the *SPI Common Extensions Manual*. For details on the ZTCI return codes, see *Appendix A, Error Numbers and Error Lists*. The possible nonzero values of RETCODE in response to the ALTER command are:

- ZCOM-ERR-TKN-REQ (-29)
- ZCOM-ERR-TKN-DUP (-27)
- ZCOM-ERR-TKN-CODE-INV (-26)
- ZCOM-ERR-SSID-INV (-24)
- ZCOM-ERR-SPI-ERR (-23)
- ZCOM-ERR-SECUR-VIOL (-22)
- ZCOM-ERR-OBJ-NOT-SUPP (-18)
- ZCOM-ERR-OBJ-NOT-FOUND (-17)
- ZCOM-ERR-CMD-TOO-LARGE (-8)
- ZCOM-ERR-CMD-NOT-SUPP (-5)
- ZCOM-ERR-CMD-INV-IN-SUMSTATE (-3)
- ZTCI-ERR-ATTR-OUT-OF-RANGE (5)
- ZTCI-ERR-DUP-ADDR (7)

**ZSPI-TKN-ERRLIST**

indicates the beginning of an error list. For more details, see *SPI Standard Definitions* on page 5-1.

**ZSPI-TKN-ENDLIST**

indicates the end of a data list or an error list. For more details, see *SPI Standard Definitions* on page 5-1.

### Operational Notes

- ALTER is a sensitive command.
- The object must be in the STOPPED summary state when the ALTER command is issued.
- When the ALTER command is completed, the object remains in the same state as before the command was executed.
- To obtain attributes of a process or subnet, use the INFO command.
DELETE Command

The DELETE command removes subnets and routes from the TCP/IP subsystem. You cannot delete a process.

When using the DELETE command, the subnet(s) or route(s) to be deleted must be in the STOPPED summary state. In addition, when your management application issues the DELETE command, the TCP/IP subsystem checks for object-name templates. For more information on using object-name templates, see Object-Name Templates on page 3-6.

<table>
<thead>
<tr>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZCOM-CMD-DELETE</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Object Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZCOM-OBJ-SUBNET or ZCOM-OBJ-ROUTE</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Tokens in Command Buffer</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZSPI-TKN-SSID</td>
</tr>
<tr>
<td>ZCOM-TKN-OBJNAME</td>
</tr>
<tr>
<td>ZSPI-TKN-MAXRESP</td>
</tr>
<tr>
<td>ZSPI-TKN-CONTEXT</td>
</tr>
<tr>
<td>ZSPI-TKN-ALLOW-TYPE</td>
</tr>
<tr>
<td>ZSPI-TKN-COMMENT</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Tokens in Response Buffer</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZSPI-TKN-SSID</td>
</tr>
<tr>
<td>ZSPI-TKN-SERVER-VERSION</td>
</tr>
<tr>
<td>ZSPI-TKN-DATA LIST</td>
</tr>
<tr>
<td>ZCOM-TKN-OBJNAME</td>
</tr>
<tr>
<td>ZSPI-TKN-OBJTYPE</td>
</tr>
<tr>
<td>ZSPI-TKN-RETCODE</td>
</tr>
<tr>
<td>ZSPI-TKN-ERRLIST</td>
</tr>
<tr>
<td>ZSPI-TKN-ENDLIST</td>
</tr>
<tr>
<td>ZSPI-TKN-CONTEXT</td>
</tr>
</tbody>
</table>

Tokens in Command Buffer

ZSPI-TKN-SSID

specifies the subsystem ID, which your management application must pass to SSINIT. This token is described in SPI Standard Definitions on page 5-1, and in the SPI Programming Manual.
DETERM-TKN-OBJNAME

specifies the subnet or route to be deleted. Your application must provide this
token in the command. The object name can include wild-card characters. Using
wild-card characters allows the command to affect multiple objects. This token is
described more fully in Data Communications Standard Definitions on page 5-9.

(ZSPI-TKN-) MAXRESP, CONTEXT, ALLOW-TYPE, and COMMENT

are standard SPI tokens that your application can provide. These tokens are
described in SPI Standard Definitions on page 5-1.

Tokens in Response Buffer

ZSPI-TKN-SSID

is an echo of the subsystem ID value provided in the command by your application.
For more details, see SPI Standard Definitions on page 5-1.

ZSPI-TKN-SERVER-VERSION

indicates the software release version of the TCP/IP subsystem. For more details,
see SPI Standard Definitions on page 5-1.

ZSPI-TKN-DATALIST

indicates the beginning of a data list. The ZSPI-TKN-MAXRESP token must have a
value other than 0 for data lists to be included in the response. For more details,
see SPI Standard Definitions on page 5-1.

ZCOM-TKN-OBJNAME

specifies the name of a TCP/IP subnet or route deleted. Unless there is an error
that prevents the DELETE command from being attempted at all, this token is
present in every response record. ZCOM-TKN-OBJNAME is described more fully
in Data Communications Standard Definitions on page 5-9.

ZCOM-TKN-OBJTYPE

specifies the type of object deleted. This token can have the value ZCOM-OBJ-
SUBNET or ZCOM-OBJ-ROUTE. For more details, see Data Communications
Standard Definitions on page 5-9.

ZSPI-TKN-RETCODE

specifies whether the command has been completed successfully. The value of
this token remains zero (ZCOM-ERR-OK) when the command is completed
successfully with or without a warning. Note that if the empty response (ZCOM-
ERR-EMPT-RSP) is returned, the value of this token is nonzero, although the
command is completed successfully.
If the command was not completed successfully, the value of this token indicates the reason the command failed. Always check the contents of the error list to determine the token or value in error. For details on the return codes, see the SPI Common Extensions Manual. The possible nonzero values of RETCODE in response to the DELETE command are:

- ZCOM-ERR-TKN-REQ (-29)
- ZCOM-ERR-TKN-DUP (-27)
- ZCOM-ERR-TKN-CODE-INV (-26)
- ZCOM-ERR-SSID-INV (-24)
- ZCOM-ERR-SPI-ERR (-23)
- ZCOM-ERR-SECUR-VIOL (-22)
- ZCOM-ERR-OBJ-NOT-SUPP (-18)
- ZCOM-ERR-OBJ-NOT-FOUND (-17)
- ZCOM-ERR-CMD-TOO-LARGE (-8)
- ZCOM-ERR-CMD-NOT-SUPP (-5)
- ZCOM-ERR-CMD-INV-IN-SUMSTATE (-3)

ZSPI-TKN-ERRLIST

indicates the beginning of an error list. For more details, see SPI Standard Definitions on page 5-1.

ZSPI-TKN-ENDLIST

indicates the end of a data list or an error list. For more details, see SPI Standard Definitions on page 5-1.

ZSPI-TKN-CONTEXT

specifies whether there are more response messages to be received for a response. If this token is present, the response continues to the next response message. If this token is not present, this response message is the last to be received. This token is present only if an object-name template has been specified. For more details, see SPI Standard Definitions on page 5-1.

Operational Notes

- DELETE is a sensitive command.
- When the DELETE command is completed, the subnet or route is removed from the subsystem.
- The object must be in the STOPPED summary state when the DELETE command is issued.
• To add subnets or routes, use the ADD command.

**GETVERSION Command**

The GETVERSION command causes the TCP/IP subsystem to return its server version and server-banner string. The server-banner string reflects the product name, product number, release date, and the date the TCP/IP code was compiled. The string can be a maximum of 50 bytes in length.

Your management application can issue the GETVERSION command with the NULL or PROC object type. If your management application specifies any other object type in the command, the TCP/IP subsystem returns an error.

Any user on the system can issue the GETVERSION command.

<table>
<thead>
<tr>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZCOM−CMD−GETVERSION</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Object Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZCOM−OBJ−PROC or ZCOM−OBJ−NULL</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Tokens in Command Buffer</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZSPI−TKN−SSID token-type ZSPI−TYP−SSID. !{}</td>
</tr>
<tr>
<td>ZCOM−TKN−OBJNAME token-type ZSPI−TYP−STRING. !{}</td>
</tr>
<tr>
<td>ZSPI−TKN−MAXRESP token-type ZSPI−TYP−INT. !{}</td>
</tr>
<tr>
<td>ZSPI−TKN−COMMENT token-type ZSPI−TYP−STRING. !{}</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Tokens in Response Buffer</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZSPI−TKN−SSID token-type ZSPI−TYP−SSID. !{}</td>
</tr>
<tr>
<td>ZSPI−TKN−SERVER−VERSION token-type ZSPI−TYP−UINT. !{}</td>
</tr>
<tr>
<td>ZSPI−TKN−DATALIST token-type ZSPI−TYP−LIST. !{}</td>
</tr>
<tr>
<td>ZSPI−TKN−SERVER−BANNER token-type ZSPI−TYP−CHAR50. !{}</td>
</tr>
<tr>
<td>ZCOM−TKN−OBJNAME token-type ZSPI−TYP−STRING. !{}</td>
</tr>
<tr>
<td>ZCOM−TKN−OBJTYPE token-type ZSPI−TYP−ENUM. !{}</td>
</tr>
<tr>
<td>ZSPI−TKN−RETCODE token-type ZSPI−TYP−ENUM. !{}</td>
</tr>
<tr>
<td>ZSPI−TKN−ERRLIST token-type ZSPI−TYP−LIST. !{}</td>
</tr>
<tr>
<td>ZSPI−TKN−ENDLIST token-type ZSPI−TYP−SSCTL.</td>
</tr>
<tr>
<td>ZSPI−TKN−ENDLIST token-type ZSPI−TYP−SSCTL.</td>
</tr>
<tr>
<td>...</td>
</tr>
<tr>
<td>...</td>
</tr>
</tbody>
</table>
Tokens in Command Buffer

ZSPI-TKN-SSID

specifies the subsystem ID, which your management application must pass to SSINIT. This token is described in SPI Standard Definitions on page 5-1, and in the SPI Programming Manual.

ZCOM-TKN-OBJNAME

specifies the name of the TCP/IP process to return the version information. Your application must provide this token only if the value ZCOM-OBJ-PROC has been specified. Your application cannot provide this token if the value ZCOM-OBJ-NONE has been specified, as the NULL object type does not have an object name associated with it.

(ZSPI-TKN-) MAXRESP and COMMENT

are standard SPI tokens that your application can provide. These tokens are described in SPI Standard Definitions on page 5-1.

Tokens in Response Buffer

ZSPI-TKN-SSID

is an echo of the subsystem ID value provided in the command by your application. For more details, see SPI Standard Definitions on page 5-1.

ZSPI-TKN-SERVER-VERSION

indicates the software release version of the TCP/IP subsystem. For more details, see SPI Standard Definitions on page 5-1.

ZSPI-TKN-DATALIST

indicates the beginning of a data list. The ZSPI-TKN-MAXRESP token must have a value other than 0 for data lists to be included in the response. For more details, see SPI Standard Definitions on page 5-1.

ZSPI-TKN-SERVER-BANNER

specifies the 50-byte character string that contains the server product name, product number, and release date.

ZCOM-TKN-OBJNAME

specifies the name of the TCP/IP process to which the command was sent. Unless there is an error that prevents the GETVERSION command from being attempted at all, this token is present in every response record. ZCOM-TKN-OBJNAME is described more fully in Data Communications Standard Definitions on page 5-9.
ZCOM-TKN-OBJTYPE

specifies the type of object for which information was requested. The value of this
token can be ZCOM-OBJ-PROC or ZCOM-OBJ-NUL. For more details, see Data
Communications Standard Definitions on page 5-9.

ZSPI-TKN-RETCODE

specifies whether the command has been completed successfully. The value of
this token remains zero (ZCOM-ERR-OK) when the command is completed
successfully with or without a warning.

If the command is not completed successfully, the value of this token indicates the
reason the command failed. Always check the error list to determine the token or
value in error. For details on the ZCOM return codes, see the SPI Common
Extensions Manual.

The possible nonzero values of RETCODE in response to the GETVERSION
command are:

ZCOM-ERR-TKN-REQ (-29)
ZCOM-ERR-TKN-DUP (-27)
ZCOM-ERR-TKN-CODE-INV (-26)
ZCOM-ERR-SSID-INV (-24)
ZCOM-ERR-SPI-ERR (-23)
ZCOM-ERR-OBJ-NOT-SUPP (-18)
ZCOM-ERR-OBJ-NOT-FOUND (-17)
ZCOM-ERR-CMD-TOO-LARGE (-8)
ZCOM-ERR-CMD-MISMATCH (-4)

ZSPI-TKN-ERRLIST

indicates the beginning of an error list. For more details, see SPI Standard
Definitions on page 5-1.

ZSPI-TKN-ENDLIST

indicates the end of a data list or an error list. For more details, see SPI Standard
Definitions on page 5-1.

Operational Notes

- GETVERSION is a nonsensitive command. All users on the system can issue this
command.

- When the GETVERSION command is completed, the summary state of the object
remains the same as it was before the command was executed.
The GETVERSION command is valid only for the PROC or NULL objects.

INFO Command

The INFO command causes the TCP/IP subsystem to return the values of the configurable attributes of the specified object(s). These values are returned in the ZTCI-MAP-INFO-PROC, ZTCI-MAP-INFO-SUBNET, and ZTCI-MAP-INFO-ROUTE extensible structured tokens.

When your management application issues the INFO command for a subnet or route, the subsystem checks for an object-name template. If one is present, information is returned for all objects of that type defined in the TCP/IP subsystem. For more details, see Object-Name Templates on page 3-6.

If your management application issues the INFO command without specifying an object, the subsystem returns an error.

The current summary state of the objects has no effect on the information returned by this command. For information on summary-state, see STATUS Command on page 6-100.

<table>
<thead>
<tr>
<th>Command</th>
<th>Object Type</th>
<th>Tokens in Command Buffer</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZCOM-CMD-INFO</td>
<td>ZCOM-OBJ-PROC or ZCOM-OBJ-SUBNET or ZCOM-OBJ-ROUTE or ZCOM-OBJ-ADDRMAP</td>
<td>ZSPI-TKN-SSID token-type ZSPI-TYP-SSID. !{}</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ZCOM-TKN-OBJNAME token-type ZSPI-TYP-STRING. !{}</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ZSPI-TKN-MAXRESP token-type ZSPI-TYP-INT.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ZSPI-TKN-CONTEXT token-type ZSPI-TYP-BYTESTRING.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ZSPI-TKN-ALLOW-TYPE token-type ZSPI-TYP-ENUM.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ZSPI-TKN-COMMENT token-type ZSPI-TYP-STRING.</td>
</tr>
</tbody>
</table>
### Tokens in Response Buffer

<table>
<thead>
<tr>
<th>Token Type</th>
<th>Token Description</th>
<th>Value Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZSPI-TKN-SSID</td>
<td>token-type ZSPI-TYP-SSID.</td>
<td>!{}</td>
</tr>
<tr>
<td>ZSPI-TKN-SERVER-VERSION</td>
<td>token-type ZSPI-TYP-UINT.</td>
<td>!{}</td>
</tr>
<tr>
<td>ZSPI-TKN-DATALIST</td>
<td>token-type ZSPI-TYP-LIST.</td>
<td>...</td>
</tr>
<tr>
<td>ZTCI-MAP-INFO-PROC</td>
<td></td>
<td>!{A}</td>
</tr>
<tr>
<td></td>
<td></td>
<td>OBJ-PROC</td>
</tr>
<tr>
<td>def ZTCI-DDL-INFO-PROC.</td>
<td></td>
<td>!All fields are present.</td>
</tr>
<tr>
<td>02 ZTCP-SEND-SPACE type ZSPI-DDL-INT.</td>
<td>02 ZTCP-RECV-SPACE type ZSPI-DDL-INT.</td>
<td>02 ZUDP-SEND-SPACE type ZSPI-DDL-INT.</td>
</tr>
<tr>
<td></td>
<td>def ZTCI-DDL-INFO-ADDRMAP.</td>
<td>!A}</td>
</tr>
<tr>
<td></td>
<td></td>
<td>OBJ-ADDRMAP</td>
</tr>
<tr>
<td>02 ZIP-ADDR-REMOTE type ZSPI-DDL-INT2.</td>
<td>02 ZX121-ADDR-REMOTE type ZSPI-DDL-CHAR16.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>......</td>
<td>ZINFO-ADDRMAP-LAST-USED type ZSPI-DDL-TIMESTAMP.</td>
</tr>
<tr>
<td>02 ZINFO-LAST-LDEV-USED type ZSPI-DDL-CHAR50.</td>
<td>end.</td>
<td></td>
</tr>
</tbody>
</table>
INFO Command

Tokens in Command Buffer

**ZSPI-TKN-SSID**

specifies the subsystem ID, which your management application must pass to SSINIT. This token is described in SPI Standard Definitions on page 5-1, and in the SPI Programming Manual.

**ZCOM-TKN-OBJNAME**

specifies the process, subnet, or route to be queried. Your application must provide this token in the command. The object name can include wild-card characters. Using wild-card characters allows the command to affect multiple objects. This token is described in Data Communications Standard Definitions on page 5-9.

**(ZSPI-TKN-)** MAXRESP, CONTEXT, ALLOW-TYPE, and COMMENT

are standard SPI tokens that your application can provide. These tokens are described in SPI Standard Definitions on page 5-1.
Tokens in Response Buffer

**ZSPI-TKN-SSID**

is an echo of the subsystem ID value provided in the command by your application. For more details, see [SPI Standard Definitions](#) on page 5-1.

**ZSPI-TKN-SERVER-VERSION**

indicates the software release version of the TCP/IP subsystem. For more details, see [SPI Standard Definitions](#) on page 5-1.

**ZSPI-TKN-DATALIST**

indicates the beginning of a data list. The ZSPI-TKN-MAXRESP token must have a value other than zerol for data lists to be included in the response. For more details, see [SPI Standard Definitions](#) on page 5-1.

**ZTCI-MAP-INFO-PROC**

is the extensible structured token that lists the configurable attributes for the process. It has the token number ZTCI-TNM-INFO-PROC and the DDL definition ZTCI-DDL-INFO-PROC. This token always appears in the response when the specified object is ZCOM-OBJ-PROC.

The fields in the ZTCI-MAP-INFO-PROC extensible structured token are:

**ZTCP-SEND-SPACE**

is an integer that specifies the amount of data that can be buffered in the TCP Layer when sending data to a remote site.

**ZTCP-RECV-SPACE**

is an integer that specifies the amount of data that can be buffered in the TCP Layer when receiving data from a remote site.

**ZUDP-SEND-SPACE**

is an integer that specifies the amount of data that can be buffered in the UDP Layer when sending data to a remote site.

**ZUDP-RECV-SPACE**

is an integer that specifies the amount of data that can be buffered in the UDP Layer when receiving data from a remote site.

**ZDELAY-ACKS**

is a Boolean variable that specifies whether an acknowledgment (ACK) is to be delayed when a TCP packet is received from a remote site. The possible values are:
INFO Command

ZSPI-VAL-TRUE

specifies that an ACK for a TCP packet should be delayed until the TCP receive window is completely filled, or one interval as defined by ZDELAY-ACKS-TIME has expired, whichever occurs first.

By delaying the ACK, more than one packet can be acknowledged with a single ACK, thereby reducing network traffic. This is the default. If this value is set, see the ZDELAY-ACKS-TIME field for the time interval between ACKs.

ZSPI-VAL-FALSE

specifies that an ACK should be returned as soon as a TCP packet is received.

ZDELAY-ACKS-TIME

is an integer that specifies the time, in units of .01 seconds, that the TCP/IP subsystem waits before sending an ACK for a TCP packet received from a remote site. This field is only meaningful if the ZDELAY-ACKS field has the value ZSPI-VAL-TRUE.

ZHOSTID

is a 32-bit integer that identifies the host. Typically, this value is the host number portion of the IP address assigned to the host.

ZHOSTNAME

is a character string that identifies the host on which the TCP/IP process is running.

ZBKUP-CPU

is an integer that specifies the number of the processor in which the backup TCP/IP process is running. This field is reserved for future use.

ZBKUP-PIN

is an integer that specifies the process identification number (PIN) of the backup TCP/IP process. This field is reserved for future use.

ZPRIM-CPU

is an integer that specifies the number of the processor in which the primary TCP/IP process is running.

ZPRIM-PIN

is an integer that specifies the PIN of the primary TCP/IP process.
ZQIO-LIMIT

is the value of the QIOLIMIT TCP/IP process startup parameter. Valid values are 20 to 100 percent. The default is 100 percent.

ZTCP-KEEPIDLE

is an integer that specifies the amount of time, in seconds, before TCP will issue a keep-alive packet on sockets that have enabled this option. The default is 45 seconds. The range is 10 through 7200 seconds.

ZTCP-KEEPINTVL

is an integer that specifies the time interval, in seconds, between retransmissions of unacknowledged keep-alive packets. The default is 45 seconds. The range is 10 through 1260 seconds.

ZTCP-KEEPCNT

is an integer that specifies the number of times a keep-alive packet will be sent without receiving an acknowledgment, after which the TCP connection will be dropped. The default is 8. The range is 1 through 20.

ZPROG-NAME

is a 32-byte internal file name that specifies the name of the file currently being executed for the TCP/IP process.

ZDEBUG-ENABLE

is a Boolean variable that specifies whether the debug option is enabled. This option is used internally by the HP Support and Development personnel to enable the display of more TCP internal information for debugging purposes.

ZSPI-VAL-TRUE

indicates that the debug option is to be enabled.

ZSPI-VAL-FALSE

indicates that the debug option is not to be enabled.

ZFULL-DUMP

is a Boolean variable that specifies whether the QIO segment is saved when the TCP/IP process abends. The default is ON. When disabled, the TCP/IP process will only save its stack when it abends. This conserves disk space over a full dump. The preferred setting for this parameter is ON.

ZSPI-VAL-TRUE

indicates that the QIO segment is to be saved when the TCP/IP process abends.
ZSPI-VAL-FALSE

indicates that the QIO segment is not to be saved when the TCP/IP process abends.

ZALLNETSARELOCAL

is a Boolean variable; ON causes TCP to use the interface MTU as a base for determining the TCP Maximum Segment Size (MSS) for each non-local TCP connection. A non-local TCP connection is one that goes to another network (not just another subnetwork). OFF causes TCP to conform to RFC-specified behavior and use 512 bytes as the default MSS for non-local segments. For example, for Ethernet, when ALLNETSARELOCAL is ON, the non-local MSS will be 1460; setting ALLNETSARELOCAL to ON can improve performance. The default is ON.

ZTCP-COMPAT-42

is the flag used to set the TCP/IP process compatible with BSD 4.2 versions as follows: if the flag is ON, then the original ACK minus 1 is sent in the keepalive packet; if the flag is OFF, the original ACK is sent in the keepalive packet. The default value of this flag is ON.

ZTCI-MAP-INFO-SUBNET

is an extensible structured token that lists the configurable attributes for a subnet. It has the token number ZTCI-TNM-INFO-SUBNET and the DDL definition ZTCI-DDLINFOSUBNET. This extensible structured token always appears in the response when the specified object is ZCOM-OBJ-SUBNET. Duplicate occurrences of the fields are not allowed.

The fields in the ZTCI-MAP-INFO-SUBNET extensible structured token are:

ZIP-ADDR

is a 32-bit integer that specifies the address of the subnet or the loopback address. Either value is specified as an IP address. The conventional loopback address (which is used for testing purposes) is 127.1.

ZSUBNET-MASK

is a 32-bit integer that specifies the subnet mask to be used with this subnet. A subnet mask identifies which part of the IP local address represents the subnet number and which part represents the host ID. If bits in the subnet mask are set to 1, the corresponding bits in the IP address are to be part of the network (and subnet) address. If bits in the subnet mask are set to 0, the corresponding bits in the IP address are to be part of the host ID (that is, that portion of the local address masked with 1s identifies the subnet, and the remainder of the local address uniquely identifies a host connected to the subnet). For further details on subnet addressing, subnet masking, and the various classes of IP addresses, see the TCP/IP Configuration and Management Manual.
ZIOPNAME

is a 32-byte internal file name that specifies the name of the SLSA LIF or the X25AM device with which the subnet interfaces.

The TCP/IP subsystem automatically adds a route with a destination address equal to the value specified in the network portion of the ZIP-ADDR field converted to its broadcast form, and a gateway address equal to the value specified as the IP address of the subnet in the ZIP-ADDR field, whenever a subnet is added.

**Note.** The SLSA LIF’s format is not that of a real internal file name. If you are using the SCF DETAIL CMDBUFFER ON command and DETAIL RSPBUFFER ON commands to examine this token, it will not display correctly for the SLSA LIF.

ZTYPE

is an enumerated value that specifies the type of subnet. The possible subnet type values (ZTCI-VAL-SUBNET-TYPE-) are:

ENET

indicates that the subnet is associated with a SLSA LIF using an Ethernet filter.

LOOP

indicates that the subnet is associated with the loopback driver and is for testing.

SNAP

indicates that the subnet is associated with a SLSA LIF using an LLC1 filter.

X25

indicates that the subnet is associated with an X25AM device.

ZSU-NAME

is an eight-character string that indicates the name of X25AM.

ZFORCE-QIO

is a Boolean variable that specifies whether the SUBNET object should only use the QIO direct interface to the controller.

ZSPI-VAL-TRUE

indicates that the SUBNET object is to use the QIO direct interface to the controller.
**INFO Command**

**ZSPI-VAL-False**
indicates that the SUBNET object is not to use the QIO direct interface to the controller.

**ZUSE-IRDP**
is a Boolean variable that specifies whether the SUBNET object should enable its router discovery interface. The default is disabled.

**ZSPI-VAL-True**
indicates that the SUBNET object is to use the router discovery interface.

**ZSPI-VAL-False**
indicates that the SUBNET object is to not use the router discovery interface.

**ZTCI-MAP-INFO-ROUTE**
is the extensible structured token that lists the configurable attributes for a route. It has the token number ZTCI-TNM-INFO-ROUTE and the DDL definition ZTCI-DDL-INFO-ROUTE. This token must be in the command if the ZCOM-OBJ-ROUTE value has been specified. Duplicate occurrences are not allowed. All fields are required.

The fields in the ZTCI-MAP-INFO-ROUTE token are:

**ZDEST-IP-ADDR**
is a 32-bit integer that specifies the destination address.

**ZGW-IP-ADDR**
is a 32-bit integer that specifies the gateway address.

**ZIF-HOST**
is a Boolean variable that specifies whether the route connects to a host. The default is ZSPI-VAL-False.

**ZSPI-VAL-True**
indicates that the route connects to a host; that is, it is a point-to-point connection.

**ZSPI-VAL-False**
indicates that the route does not connect to a host but is broadcast instead.

**ZIF-GW**
is a Boolean variable that specifies whether the route goes through a gateway.
ZSPI-VAL-TRUE
indicates that the route connects to the destination address by means of a gateway.

ZSPI-VAL-FALSE
indicates that the route does not go through a gateway.

ZSUBNET
is an eight-character field that specifies the name of the subnet interface used by the specific route.

ZIF-IRDP
is a variable that specifies whether ICMP Router Discovery Protocol (IRDP) has been enabled on the subnet. The default is OFF. The possible values are:

ZSPI-VAL-TRUE
indicates that the IRDP has been enabled on the subnet.

ZSPI-VAL-FALSE
indicates that the IRDP has been disabled on the subnet.

ZCOM-TKN-OBJNAME
specifies the name of the TCP/IP process, subnet, or route queried. Unless there is an error that prevents the INFO command from being attempted at all, this token is present in every response record. ZCOM-TKN-OBJNAME is described in Data Communications Standard Definitions on page 5-9.

ZCOM-TKN-OBJTYPE
specifies the type of object queried. This token has the value ZCOM-OBJ-PROC, ZCOM-OBJ-SUBNET, or ZCOM-OBJ-ROUTE. For more details, see Data Communications Standard Definitions on page 5-9.

ZSPI-TKN-RETCODE
specifies whether the command has been completed successfully. The value of this token remains zero (ZCOM-ERR-OK) when the command is completed successfully with or without a warning. Note that if the empty response (ZCOM-ERR-EMPT-RSP) is returned, the value of this token is nonzero, although the command is completed successfully.

If the command is not completed successfully, the value of this token indicates the reason the command failed. Always check the contents of the error list to determine the token or value in error. For details on the ZCOM return codes and associated error lists, see the SPI Common Extensions Manual.
The possible nonzero values of RETCODE in response to the INFO command are:
ZCOM-ERR-TKN-REQ (-29)
ZCOM-ERR-TKN-DUP (-27)
ZCOM-ERR-TKN-CODE-INV (-26)
ZCOM-ERR-SSID-INV (-24)
ZCOM-ERR-SPI-ERR (-23)
ZCOM-ERR-OBJ-NOT-SUPP (-18)
ZCOM-ERR-OBJ-NOT-FOUND (-17)
ZCOM-ERR-CMD-TOO-LARGE (-8)
ZCOM-ERR-CMD-NOT-SUPP (-5)

ZSPI-TKN-ERRLIST
indicates the beginning of an error list. For more details, see SPI Standard Definitions on page 5-1.

ZSPI-TKN-ENDLIST
indicates the end of a data list or an error list. For more details, see SPI Standard Definitions on page 5-1.

ZSPI-TKN-CONTEXT
specifies whether the response continues with more response messages. If this token is present, the response continues. If this token is not present, this response message is the last of the response. For more details, see SPI Standard Definitions on page 5-1.

Operational Notes

• INFO is a nonsensitive command; all valid users on the system can issue the INFO command.
• The summary state of an object does not affect obtaining information for the object.
• When the INFO command is completed, the object remains in the same summary state as before the command was executed.
• To change the values of the attributes, use the ALTER command.
• To obtain information about the current summary state of an object, use the STATUS command.
• To obtain statistics about an object, use the STATISTICS command.
LISTOBJECTS Command

The LISTOBJECTS command lists the names of objects known to the TCP/IP subsystem. The objects listed depend on the value of the ZSPI-TKN-OBJECT-TYPE and ZCOM-TKN-OBJNAME tokens.

When your management application specifies the NULL object, all object types and object names known to the TCP/IP subsystem are listed. In this case, the OBJNAME token must either have a zero length or be absent. If these conditions are violated, the command is rejected and the TCP/IP subsystem returns an error.

When your management application specifies the SUBNET or ROUTE object, the object types and object names listed vary, based on the value of the OBJNAME token. In this case, the OBJNAME token can contain a specific object name, an object-name template, a zero length, or it can be absent. If the OBJNAME token contains a specific object name, only that object is listed. If the OBJNAME token contains an object-name template, only those objects that meet the template requirements are listed. If the token has a zero length or is absent, all objects of the specified type known to the TCP/IP subsystem are listed.

Although multiple object names of the same object type can be returned within a single data list, one data list cannot contain information for more than one object type. When the objects to be returned consist of more than one object type or the number of objects to be returned cannot fit in one data list, additional data lists can be returned by using the context token.

All users on the system can issue this command.

<table>
<thead>
<tr>
<th>Command</th>
<th>ZCOM-CMD-LISTOBJECTS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object Type</td>
<td>ZCOM-OBJ-NULL or ZCOM-OBJ-SUBNET or ZCOM-OBJ-ROUTE</td>
</tr>
<tr>
<td>Tokens in Command Buffer</td>
<td></td>
</tr>
<tr>
<td>ZSPI-TKN-SSID</td>
<td>token-type ZSPI-TYP-SSID. !{}</td>
</tr>
<tr>
<td>ZCOM-TKN-OBJNAME</td>
<td>token-type ZSPI-TYP-STRING.</td>
</tr>
<tr>
<td>ZSPI-TKN-MAXRESP</td>
<td>token-type ZSPI-TYP-INT.</td>
</tr>
<tr>
<td>ZSPI-TKN-CONTEXT</td>
<td>token-type ZSPI-TYP-BYTESTRING.</td>
</tr>
<tr>
<td>ZSPI-TKN-ALLOW-TYPE</td>
<td>token-type ZSPI-TYP-ENUM.</td>
</tr>
<tr>
<td>ZSPI-TKN-COMMENT</td>
<td>token-type ZSPI-TYP-STRING.</td>
</tr>
</tbody>
</table>
**Tokens in Command Buffer**

**ZSPI-TKN-SSID**

specifies the subsystem ID, which your management application must pass to SSINIT. This token is described in [SPI Standard Definitions](#) on page 5-1, and in the *SPI Programming Manual*.

**ZCOM-TKN-OBJNAME**

specifies the name of the object to be listed. This is an optional token. The object names of the ROUTE and SUBNET objects can include wild-card characters. Using wild-card characters allows the command to affect multiple objects. This token is described in [Data Communications Standard Definitions](#) on page 5-9.

(ZSPI-TKN-) **MAXRESP, CONTEXT, ALLOW-TYPE, and COMMENT**

are standard SPI tokens that your application can provide. These tokens are described in [SPI Standard Definitions](#) on page 5-1.

**Tokens in Response Buffer**

**ZSPI-TKN-SSID**

is an echo of the subsystem ID value provided in the command by your application. For more details, see [SPI Standard Definitions](#) on page 5-1.

**ZSPI-TKN-SERVER-VERSION**

indicates the software release version of the TCP/IP subsystem. For more details, see [SPI Standard Definitions](#) on page 5-1.

**ZSPI-TKN-DATALIST**

indicates the beginning of a data list. The ZSPI-TKN-MAXRESP token must have a value other than zero for data lists to be included in the response. For more details, see [SPI Standard Definitions](#) on page 5-1.
**ZCOM-TKN-OBJNAME**

specifies the name of a TCP/IP object being listed by the command. Unless there is an error that prevents the LISTOBJECTS command from being attempted at all, this token is present in every response record. There can be multiple ZCOM-TKN-OBJNAME tokens in a single response record. ZCOM-TKN-OBJNAME is described in Data Communications Standard Definitions on page 5-9.

**ZCOM-TKN-OBJTYPE**

specifies the type of object listed. This token can have the value ZCOM-OBJ-SUBNET, ZCOM-OBJ-ROUTE, or ZCOM-OBJ-NULL. For more details, see Data Communications Standard Definitions on page 5-9.

**ZSPI-TKN-RETCODE**

specifies whether the command has been completed successfully. The value of this token remains zero (ZCOM-ERR-OK) when the command is completed successfully with or without a warning.

If the command is not completed successfully, the value of this token indicates the reason the command failed. Always check the contents of the error list to determine the token or value in error. For details on the ZCOM return codes, see the SPI Common Extensions Manual. The possible nonzero values of RETCODE in response to the LISTOBJECTS command are:

- ZCOM-ERR-TKN-REQ (-29)
- ZCOM-ERR-TKN-DUP (-27)
- ZCOM-ERR-TKN-CODE-INV (-26)
- ZCOM-ERR-SSID-INV (-24)
- ZCOM-ERR-SPI-ERR (-23)
- ZCOM-ERR-OBJ-NOT-SUPP (-18)
- ZCOM-ERR-OBJ-NOT-FOUND (-17)
- ZCOM-ERR-CMD-TOO-LARGE (-8)
- ZCOM-ERR-CMD-NOT-SUPP (-5)

**ZSPI-TKN-ERRLIST**

indicates the beginning of an error list. For more details, see SPI Standard Definitions on page 5-1.

**ZSPI-TKN-ENDLIST**

indicates the end of a data list or an error list. For more details, see SPI Standard Definitions on page 5-1.
LISTOPENS Command

The LISTOPENS command returns information about the openers of the TCP/IP process. The extensible structured token ZTCI-MAP-OPENS-INET, which contains the opener information, can be returned multiple times, once for each open. Note that each open corresponds to a successful socket call. For details on the socket function, see the TCP/IP Programming Manual.

**Command**

ZCOM-CMD-LISTOPENS

**Object Type**

ZCOM-OBJ-PROC

**Tokens in Command Buffer**

<table>
<thead>
<tr>
<th>Token</th>
<th>Token Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZSPI-TKN-SSID</td>
<td>token-type ZSPI-TYP-SSID.</td>
</tr>
<tr>
<td>ZCOM-TKN-OBJNAME</td>
<td>token-type ZSPI-TYP-STRING.</td>
</tr>
<tr>
<td>ZSPI-TKN-MAXRESP</td>
<td>token-type ZSPI-TYP-INT.</td>
</tr>
<tr>
<td>ZSPI-TKN-CONTEXT</td>
<td>token-type ZSPI-TYP-BYTESTRING.</td>
</tr>
<tr>
<td>ZSPI-TKN-ALLOW-TYPE</td>
<td>token-type ZSPI-TYP-ENUM.</td>
</tr>
<tr>
<td>ZSPI-TKN-COMMENT</td>
<td>token-type ZSPI-TYP-STRING.</td>
</tr>
</tbody>
</table>
Tokens in Command Buffer

ZSPI-TKN-SSID

specifies the subsystem ID, which your management application must pass to SSINIT. This token is described in SPI Standard Definitions on page 5-1, and in the SPI Programming Manual.

ZCOM-TKN-OBJNAME

specifies the process for which openers are to be listed. Your application must provide this token in the command. This token is described in Data Communications Standard Definitions on page 5-9.

(ZSPI-TKN-) MAXRESP, ALLOW-TYPE, CONTEXT, and COMMENT

are standard SPI tokens that your application can provide. These tokens are described in SPI Standard Definitions on page 5-1.
Tokens in Response Buffer

ZSPI-TKN-SSID

is an echo of the subsystem ID value provided in the command by your application. For more details, see SPI Standard Definitions on page 5-1.

ZSPI-TKN-SERVER-VERSION

indicates the software release version of the TCP/IP subsystem. For more details, see SPI Standard Definitions on page 5-1.

ZSPI-TKN-DATALIST

indicates the beginning of a data list. The ZSPI-TKN-MAXRESP token must have a value other than zero for data lists to be included in the response. For more details, see SPI Standard Definitions on page 5-1.

ZTCI-MAP-OPENS-INET

is the extensible structured token that lists information about the openers of the TCP/IP process. It has the token number ZTCI-TNM-OPENS-INET and the DDL definition ZTCI-DDL-OPENS-INET. This token is always in the response record. Note that there is a separate extensible structured token for each opener and each extensible structured token is contained in a separate data list. Duplicate occurrences of the fields within a single extensible structured token are not allowed.

The fields in the ZTCI-MAP-OPENS-INET extensible structured token are:

ZBKUP-CPU

is an integer that specifies the number of the processor in which the backup process of the opener is running.

ZBKUP-PIN

is an integer that specifies the process identification number (PIN) of the backup process of the opener.

ZBKUP-FNUM

is an integer that specifies the backup file number assigned to the opener when the OPEN procedure was called. This field is reserved for future use.

ZPRIM-CPU

is an integer that specifies the number of the processor in which the primary process of the opener is running.

ZPRIM-PIN

is an integer that specifies the PIN of the primary process of the opener.
ZPRIM-FNUM

is an integer that specifies the primary file number assigned to the opener when the OPEN procedure was called. Note that the file number is equivalent to the socket number.

ZPROTO

is an integer that specifies which protocol module has been opened. The possible values are:

0  indicates the #ZSPI module has been opened.
6  indicates the TCP protocol module has been opened.
1  indicates the UDP protocol module has been opened.

ZPROCNAME

is a null-terminated string that specifies the system name of the opener.

ZLPORT

is the local port number.

ZLOC-ADDR

is the local Internet address.

ZLOC-FPORT

is the foreign port number.

ZFORGN-ADDR

is the foreign Internet address.

ZSTATE

is the state of the connection. This is related only to TCP.

ZSEND

is the number of bytes of data currently in the send queue for a socket.

ZRECV

is the number of bytes of data currently in the receive queue for a socket.

ZCOM-TKN-OBJNAME

specifies the name of the TCP/IP process specified in the command. Unless there is an error that prevents the LISTOPENS command from being attempted at all,
this token is present in every response record. ZCOM-TKN-OBJNAME is described in Data Communications Standard Definitions on page 5-9.

ZCOM-TKN-OBJTYPE

specifies the type of object queried. This token has the value ZCOM-OBJ-PROC. For more details, see Data Communications Standard Definitions on page 5-9.

ZSPI-TKN-RETCODE

specifies whether the command has been completed successfully. The value of this token remains zero (ZCOM-ERR-OK) when the command is completed successfully with or without a warning. Note that if the empty response (ZCOM-ERR-EMPT-RSP) is returned, the value of this token is nonzero although the command is completed successfully.

If the command is not completed successfully, the value of this token indicates the reason the command failed. Always check the contents of the error list to determine the token or value in error. For details on the ZCOM return codes, see the SPI Common Extensions Manual.

The possible nonzero values of RETCODE in response to the LISTOPENS command are:

- ZCOM-ERR-TKN-REQ (-29)
- ZCOM-ERR-TKN-DUP (-27)
- ZCOM-ERR-TKN-CODE-INV (-26)
- ZCOM-ERR-SSID-INV (-24)
- ZCOM-ERR-SPI-ERR (-23)
- ZCOM-ERR-OBJ-NOT-SUPP (-18)
- ZCOM-ERR-OBJ-NOT-FOUND (-17)
- ZCOM-ERR-CMD-TOO-LARGE (-8)
- ZCOM-ERR-CMD-MISMATCH (-4)

ZSPI-TKN-ERRLIST

indicates the beginning of an error list. For more details, see SPI Standard Definitions on page 5-1.

ZSPI-TKN-ENDLIST

indicates the end of a data list or an error list. For more details, see SPI Standard Definitions on page 5-1.
ZSPI-TKN-CONTEXT

specifies whether the response continues with more response messages. If this token is present, the response continues. If this token is not present, this response message is the last of the response. For more details, see SPI Standard Definitions on page 5-1.

Operational Notes

- LISTOPENS is a nonsensitive command.
- When the LISTOPENS command is completed, the PROC object remains in the same summary state as before the command was executed.

RESETSTATS Command

The RESETSTATS command returns the execution history statistics for the specified object(s). The type of information returned by this command is the same as that returned by the STATISTICS command, but this command initializes the statistics counters after the information is placed in a response message. The RESETSTATS command is a sensitive command.

Various statistics counters are maintained during the sample period defined by the RESET-TIME and CURR-TIME fields.

You can issue this command for a process, subnet, or route. You can use an object-name template with the SUBNET and ROUTE objects to return statistics about multiple subnets or routes. For more details, see Object-Name Templates on page 3-6.

The summary state of the object is not checked by the TCP/IP subsystem when you issue the RESETSTATS command.

<table>
<thead>
<tr>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZCOM-CMD-RESETSTATS</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Object Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZOM-OBJ-ADDRMAP or ZCOM-OBJ-PROC or ZCOM-OBJ-SUBNET or ZCOM-OBJ-ROUTE</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Tokens in Command Buffer</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZSPI-TKN-SSID</td>
</tr>
<tr>
<td>ZCOM-TKN-OBJNAME</td>
</tr>
<tr>
<td>ZSPI-TKN-MAXRESP</td>
</tr>
<tr>
<td>ZSPI-TKN-CONTEXT</td>
</tr>
<tr>
<td>ZSPI-TKN-ALLOW-TYPE</td>
</tr>
<tr>
<td>ZSPI-TKN-COMMENT</td>
</tr>
</tbody>
</table>
### Tokens in Response Buffer

<table>
<thead>
<tr>
<th>Token Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZSPI-TKN-SSID</td>
<td>token-type ZSPI-TYP-SSID</td>
</tr>
<tr>
<td>ZSPI-TKN-SERVER-VERSION</td>
<td>token-type ZSPI-TYP-UINT</td>
</tr>
<tr>
<td>ZSPI-TKN-DATA LIST</td>
<td>token-type ZSPI-TYP-LIST</td>
</tr>
<tr>
<td>ZTCI-MAP-TCP-STATS</td>
<td>!OBJ-PROC</td>
</tr>
</tbody>
</table>

```plaintext
def ZTCI-DDL-TCP-STATS. !All fields are present.
02 ZSTATS-TCP-BAD-XSUM       type ZSPI-DDL-INT2.
02 ZSTATS-TCP-BAD-OFF        type ZSPI-DDL-INT2.
02 ZSTATS-TCP-BAD-HDR-LEN    type ZSPI-DDL-INT2.
02 ZSTATS-TCP-BAD-SEGS       type ZSPI-DDL-INT2.
02 ZSTATS-TCP-NOT-ACK        type ZSPI-DDL-INT2.
02 ZSTATS-TCP-REXMIT         type ZSPI-DDL-INT2.
02 ZSTATS-TCP-TIMEOUT        type ZSPI-DDL-INT2.
02 ZSTATS-TCP-IN-CONN        type ZSPI-DDL-INT2.
02 ZSTATS-TCP-OUT-CONN       type ZSPI-DDL-INT2.
02 ZSTATS-TCP-IN-PKTS        type ZSPI-DDL-INT2.
02 ZSTATS-TCP-OUT-PKTS       type ZSPI-DDL-INT2.
02 ZSTATS-TCP-NO-PORT        type ZSPI-DDL-INT2.
02 ZSTATS-TCP-URGENT         type ZSPI-DDL-INT2.
02 ZSTATS-TCP-CURR-TIME      type SPI-DDL-TIMESTAMP.
02 ZSTATS-TCP-KEEP-TIMEOUT   type SPI-DDL-INT2.
02 ZSTATS-TCP-KEEP-DROPS     type SPI-DDL-INT2.
02 ZSTATS-TCP-RCV-PCK        type SPI-DDL-INT2.
02 ZSTATS-TCP-RCV-BYTE       type SPI-DDL-INT2.
02 ZSTATS-TCP-RCV-DUP-PACK   type SPI-DDL-INT2.
02 ZSTATS-TCP-RCV-DUP-BYTE   type SPI-DDL-INT2.
```
<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>02 ZSTATS-TCP-RCV-OUT-BYTE</td>
<td>SPI-DDL-INT2</td>
</tr>
<tr>
<td>02 ZSTATS-TCP-RCV-WIN-PACK</td>
<td>SPI-DDL-INT2</td>
</tr>
<tr>
<td>02 ZSTATS-TCP-RCV-WIN-BYTE</td>
<td>SPI-DDL-INT2</td>
</tr>
<tr>
<td>02 ZSTATS-TCP-RCV-AFTER-CLOSE</td>
<td>SPI-DDL-INT2</td>
</tr>
<tr>
<td>02 ZSTATS-TCP-RCV-WIN-PROBE</td>
<td>SPI-DDL-INT2</td>
</tr>
<tr>
<td>02 ZSTATS-TCP-RCV-DUP-ACK</td>
<td>SPI-DDL-INT2</td>
</tr>
<tr>
<td>02 ZSTATS-TCP-RCV-ACK-TOO-MUCH</td>
<td>SPI-DDL-INT2</td>
</tr>
<tr>
<td>02 ZSTATS-TCP-RCV-ACK-PACK</td>
<td>SPI-DDL-INT2</td>
</tr>
<tr>
<td>02 ZSTATS-TCP-RCV-ACK-BYTE</td>
<td>SPI-DDL-INT2</td>
</tr>
<tr>
<td>02 ZSTATS-TCP-RCV-WIN-UPDATE</td>
<td>SPI-DDL-INT2</td>
</tr>
</tbody>
</table>

end.

ZTCI-MAP-UDP-STATS

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>02 ZSTATS-UDP-BAD-HDR-LEN</td>
<td>ZSPI-DDL-INT2</td>
</tr>
<tr>
<td>02 ZSTATS-UDP-BAD-XSUM</td>
<td>ZSPI-DDL-INT2</td>
</tr>
<tr>
<td>02 ZSTATS-UDP-BAD-LEN</td>
<td>ZSPI-DDL-INT2</td>
</tr>
<tr>
<td>02 ZSTATS-UDP-IN-PKTS</td>
<td>ZSPI-DDL-INT2</td>
</tr>
<tr>
<td>02 ZSTATS-UDP-OUT-PKTS</td>
<td>ZSPI-DDL-INT2</td>
</tr>
<tr>
<td>02 ZSTATS-UDP-CURR-TIME</td>
<td>ZSPI-DDL-TIMESTAMP</td>
</tr>
<tr>
<td>02 ZSTATS-UDP-RESET-TIME</td>
<td>ZSPI-DDL-TIMESTAMP</td>
</tr>
</tbody>
</table>

end.

ZTCI-MAP-IP-STATS

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>02 ZSTATS-IP-TOTAL</td>
<td>ZSPI-DDL-INT2</td>
</tr>
<tr>
<td>02 ZSTATS-IP-BAD-XSUM</td>
<td>ZSPI-DDL-INT2</td>
</tr>
<tr>
<td>02 ZSTATS-IP-TOO-SHORT</td>
<td>ZSPI-DDL-INT2</td>
</tr>
<tr>
<td>02 ZSTATS-IP-TOO-SMALL</td>
<td>ZSPI-DDL-INT2</td>
</tr>
<tr>
<td>02 ZSTATS-IP-BAD-HDR-LEN</td>
<td>ZSPI-DDL-INT2</td>
</tr>
<tr>
<td>02 ZSTATS-IP-BAD-LEN</td>
<td>ZSPI-DDL-INT2</td>
</tr>
<tr>
<td>02 ZSTATS-IP-FRAG</td>
<td>ZSPI-DDL-INT2</td>
</tr>
<tr>
<td>02 ZSTATS-IP-FRAG-DROP</td>
<td>ZSPI-DDL-INT2</td>
</tr>
<tr>
<td>02 ZSTATS-IP-FRAG-TIMEOUT</td>
<td>ZSPI-DDL-INT2</td>
</tr>
<tr>
<td>02 ZSTATS-IP-FWD</td>
<td>ZSPI-DDL-INT2</td>
</tr>
<tr>
<td>02 ZSTATS-IP-CANT-FWD</td>
<td>ZSPI-DDL-INT2</td>
</tr>
<tr>
<td>02 ZSTATS-IP-REDIR-SENT</td>
<td>ZSPI-DDL-INT2</td>
</tr>
<tr>
<td>02 ZSTATS-IP-OUT-PKTS</td>
<td>ZSPI-DDL-INT2</td>
</tr>
<tr>
<td>02 ZSTATS-IP-CURR-TIME</td>
<td>ZSPI-DDL-TIMESTAMP</td>
</tr>
<tr>
<td>02 ZSTATS-IP-RESET-TIME</td>
<td>ZSPI-DDL-TIMESTAMP</td>
</tr>
</tbody>
</table>
**RESETSTATS Command**

```
ZTCI-MAP-ROUTE-STATS       !{}
                           !OBJ-PROC
   def ZTCI-DDL-ROUTE-STATS.   !All fields are present.
      02 ZSTATS-ROUTE-BAD-REDIR  type ZSPI-DDL-INT2.
      02 ZSTATS-ROUTE-DYN        type ZSPI-DDL-INT2.
      02 ZSTATS-ROUTE-NEW-GW     type ZSPI-DDL-INT2.
      02 ZSTATS-ROUTE-NOT-REACH  type ZSPI-DDL-INT2.
      02 ZSTATS-ROUTE-WILD-CARD  type ZSPI-DDL-INT2.
      02 ZSTATS-ROUTE-CURR-TIME  type ZSPI-DDL-TIMESTAMP.
      02 ZSTATS-ROUTE-RESET-TIME type ZSPI-DDL-TIMESTAMP.
   end.

ZTCI-MAP-ICMP-STATS        !{}
                           !OBJ-PROC
   def ZTCI-DDL-ICMP-STATS.   !All fields are present.
      02 ZSTATS-ICMP-ERR         type ZSPI-DDL-INT2.
      02 ZSTATS-ICMP-OLD-SHORT   type ZSPI-DDL-INT2.
      02 ZSTATS-ICMP-OLD-ICMP    type ZSPI-DDL-INT2.
      02 ZSTATS-ICMP-OUT-HIST    type ZSPI-DDL-INT2
                    occurs 19 times.
      02 ZSTATS-ICMP-BADCODE     type ZSPI-DDL-INT2.
      02 ZSTATS-ICMP-TOO-SHORT   type ZSPI-DDL-INT2.
      02 ZSTATS-ICMP-BAD-XSUM    type ZSPI-DDL-INT2.
      02 ZSTATS-ICMP-BAD-LEN     type ZSPI-DDL-INT2.
      02 ZSTATS-ICMP-REFLECT     type ZSPI-DDL-INT2.
      02 ZSTATS-ICMP-IN-HIST     type ZSPI-DDL-INT2
                    occurs 19 times.
      02 ZSTATS-ICMP-CURR-TIME   type ZSPI-DDL-TIMESTAMP.
      02 ZSTATS-ICMP-RESET-TIME  type ZSPI-DDL-TIMESTAMP.
      02 ZSTATS-ICMP-XSUM        type ZSPI-DDL-INT2.
      02 ZSTATS-ICMP-BAD-RAP-SUBCODE
                    type ZSPI-DDL-INT2.
      02 ZSTATS-ICMP-BAD-RAP-ADDRLIST
                    type ZSPI-DDL-INT2.
      02 ZSTATS-ICMP-BAD-RAP-WPA
                    type ZSPI-DDL-INT2.
      02 ZSTATS-ICMP-BAD-RAP-RDISC-NRECORDED
                    type ZSPI-DDL-INT2.
```
ZTCI-MAP-SMEM-STATS  !{ }
!OBJ-STATS

def ZTCI-MAP-SMEM-STATS  .  !All fields are present.
  02 ZSTATS-SMEM-DATA-MDS  type ZSPI-DDL-INT2.
  02 ZSTATS-SMEM-MAX-DATA-MDS  type ZSPI-DDL-INT2.
  02 ZSTATS-SMEM-DUP-MDS  type ZSPI-DDL-INT2.
  02 ZSTATS-SMEM-MAX-DUP-MDS  type ZSPI-DDL-INT2.
  02 ZSTATS-SMEM-NO-DATA-MDS  type ZSPI-DDL-INT2.
  02 ZSTATS-SMEM-NO-DUP-MDS  type ZSPI-DDL-INT2.
  02 ZSTATS-SMEM-RES-RECOV-RUN  type ZSPI-DDL-INT2.
  02 ZSTATS-SMEM-QIO-LIMIT-WARNS  type ZSPI-DDL-INT2.
  02 ZSTATS-SMEM-QIO-DRIVER-ERRS  type ZSPI-DDL-INT2.
  02 ZSTATS-SMEM-CURR-POOL-SPACE  type ZSPI-DDL-INT2.
  02 ZSTATS-SMEM-MAX-POOL-SPACE  type ZSPI-DDL-INT2.
  02 ZSTATS-SMEM-POOL-FAILS  type ZSPI-DDL-INT2.
  02 ZSTATS-SMEM-TOTAL-MBUFS  type ZSPI-DDL-INT2.
  02 ZSTATS-SMEM-CURR-MBUFS-INUSE  type ZSPI-DDL-INT2.
  02 ZSTATS-SMEM-MAX-MBUFS-INUSE  type ZSPI-DDL-INT2.
  02 ZSTATS-SMEM-MBUF-FAILS  type ZSPI-DDL-INT2.
  02 ZSTATS-SMEM-DUP-DRIV-MDS  type ZSPI-DDL-INT2.
  02 ZSTATS-SMEM-MAX-DUP-DRIV-MDS  type ZSPI-DDL-INT2.
  02 ZSTATS-SMEM-NO-DATA-MDS  type ZSPI-DDL-INT2.
  02 ZSTATS-SMEM-DUP-MDS  type ZSPI-DDL-INT2.
  02 ZSTATS-SMEM-NO-DUP-MDS  type ZSPI-DDL-INT2.
  02 ZSTATS-SMEM-RES-RECOV-RUN  type ZSPI-DDL-INT2.
  02 ZSTATS-SMEM-QIO-LIMIT-WARNS  type ZSPI-DDL-INT2.
  02 ZSTATS-SMEM-QIO-DRIVER-ERRS  type ZSPI-DDL-INT2.
  02 ZSTATS-SMEM-CURR-POOL-SPACE  type ZSPI-DDL-INT2.
  02 ZSTATS-SMEM-MAX-POOL-SPACE  type ZSPI-DDL-INT2.
  02 ZSTATS-SMEM-POOL-FAILS  type ZSPI-DDL-INT2.
  02 ZSTATS-SMEM-TOTAL-MBUFS  type ZSPI-DDL-INT2.
  02 ZSTATS-SMEM-CURR-MBUFS-INUSE  type ZSPI-DDL-INT2.
  02 ZSTATS-SMEM-MAX-MBUFS-INUSE  type ZSPI-DDL-INT2.
  02 ZSTATS-SMEM-MBUF-FAILS  type ZSPI-DDL-INT2.
  02 ZSTATS-SMEM-DUP-DRIV-MDS  type ZSPI-DDL-INT2.
  02 ZSTATS-SMEM-MAX-DUP-DRIV-MDS  type ZSPI-DDL-INT2.
  02 ZSTATS-SMEM-SIZE-128  type ZSPI-DDL-INT2.
  02 ZSTATS-SMEM-SIZE-256  type ZSPI-DDL-INT2.
  02 ZSTATS-SMEM-SIZE-512  type ZSPI-DDL-INT2.
  02 ZSTATS-SMEM-SIZE-1024  type ZSPI-DDL-INT2.
  02 ZSTATS-SMEM-SIZE-2048  type ZSPI-DDL-INT2.
  02 ZSTATS-SMEM-SIZE-4096  type ZSPI-DDL-INT2.
  02 ZSTATS-SMEM-SIZE-8192  type ZSPI-DDL-INT2.
  02 ZSTATS-SMEM-SIZE-12277  type ZSPI-DDL-INT2.
  02 ZSTATS-SMEM-SIZE-16384  type ZSPI-DDL-INT2.
  02 ZSTATS-SMEM-SIZE-32768  type ZSPI-DDL-INT2.

ZTCI-MAP-STATS-ADDRMAP  !{ }
!OBJ-ADDRMAP

def ZTCI-DDL-STATS-ADDRMAP.  !All fields are present.
  02 ZTCI-DLL-STATS-ADDRMAP  type ZSPI-DDL-INT2.
  02 ZIP-ADDR-LOCAL  type ZSPI-DDL-CHAR16.
  02 ZX121-ADDR-LOCAL  type ZSPI-DDL-INT2.
  02 ZSTATS-ADDRMAP-IN-CALLS  type ZSPI-DDL-INT2.
  02 ZSTATS-ADDRMAP-OUT-CALLS  type ZSPI-DDL-INT2.
  02 ZSTATS-ADDRMAP-LAST-USED  type ZSPI-DDL-TIMESTAMP.
  02 ZSTATS-ADDRMAP-CURR-TIME  type ZSPI-DDL-TIMESTAMP.
  02 ZSTATS-ADDRMAP-RESET-TIME  type ZSPI-DDL-TIMESTAMP.
end.
Tokens in Command Buffer

ZSPI-TKN-SSID

specifies the subsystem ID, which your management application must pass to SSINIT. This token is described in SPI Standard Definitions on page 5-1, and in the SPI Programming Manual.

ZCOM-TKN-OBJNAME

specifies the process, subnet, or route for which the subsystem is to return statistics and reset counters. Your application must provide this token in the command. The object names of the ROUTE and SUBNET objects can include wild-card characters. Using wild-card characters allows the command to affect multiple objects. This token is described in Data Communications Standard Definitions on page 5-9.

(ZSPI-TKN-) MAXRESP, CONTEXT, ALLOW-TYPE, and COMMENT

are standard SPI tokens that your application can provide. These tokens are described in SPI Standard Definitions on page 5-1.

---

ZTCI-MAP-STATS-SUBNET

<table>
<thead>
<tr>
<th>def ZTCI-DDL-STATS-SUBNET.</th>
<th>!All fields are present.</th>
</tr>
</thead>
<tbody>
<tr>
<td>02 ZSTATS-SUBNET-OUT-PKTS</td>
<td>type ZSPI-DDL-INT2.</td>
</tr>
<tr>
<td>02 ZSTATS-SUBNET-IN-PKTS</td>
<td>type ZSPI-DDL-INT2.</td>
</tr>
<tr>
<td>02 ZSTATS-SUBNET-OUT-ERRS</td>
<td>type ZSPI-DDL-INT2.</td>
</tr>
<tr>
<td>02 ZSTATS-SUBNET-IN-ERRS</td>
<td>type ZSPI-DDL-INT2.</td>
</tr>
<tr>
<td>02 ZSTATS-SUBNET-CURR-TIME</td>
<td>type ZSPI-DDL-TIMESTAMP.</td>
</tr>
</tbody>
</table>

end.

ZTCI-MAP-STATS-ROUTE

<table>
<thead>
<tr>
<th>def ZTCI-DDL-STATS-ROUTE.</th>
<th>!All fields are present.</th>
</tr>
</thead>
<tbody>
<tr>
<td>02 ZUSAGE</td>
<td>type ZSPI-DDL-INT2.</td>
</tr>
<tr>
<td>02 ZSTATS-ROUTE-CURR-TIME</td>
<td>type ZSPI-DDL-TIMESTAMP.</td>
</tr>
<tr>
<td>02 ZSTATS-ROUTE-RESET-TIME</td>
<td>type ZSPI-DDL-TIMESTAMP.</td>
</tr>
</tbody>
</table>

end.

ZCOM-TKN-OBJNAME  token-type ZSPI-TYP-STRING.  !{}
ZCOM-TKN-OBJTYPE  token-type ZSPI-TYP-ENUM.    !{}
ZSPI-TKN-RETCODE  token-type ZSPI-TYP-ENUM.    !{}
ZSPI-TKN-ERRLIST  token-type ZSPI-TYP-LIST.    !...
...                ...                        ...
ZSPI-TKN-ENDLIST  token-type ZSPI-TYP-SSCTL.
ZSPI-TKN-ENDLIST  token-type ZSPI-TYP-SSCTL.
ZSPI-TKN-CONTEXT  token-type ZSPI-TYP-BYTESTRING.
**Tokens in Response Buffer**

**ZSPI-TKN-SSID**

is an echo of the subsystem ID value provided in the command by your application. For more details, see SPI Standard Definitions on page 5-1.

**ZSPI-TKN-SERVER-VERSION**

is the software release version of the TCP/IP subsystem. For more details, see SPI Standard Definitions on page 5-1.

**ZSPI-TKN-DATALIST**

indicates the beginning of the data list. The ZSPI-TKN-MAXRESP token must have a value other than zero for data lists to be included in the response. For more details, see SPI Standard Definitions on page 5-1.

**ZTCI-MAP-TCP-STATS**

is the extensible structured token that lists the statistics counters maintained for the TCP Layer during the sample period. These statistics are returned only for the PROC object. This token has the token number ZTCI-TNM-TCP-STATS and the DDL definition ZTCI-DDL-TCP-STATS.

Each counter is defined in a separate field. The fields in the ZTCI-MAP-TCP-STATS token are:

**ZSTATS-TCP-BAD-XSUM**

is the number of packets with an invalid checksum value received by the TCP Layer of the TCP/IP subsystem during the sample period. An invalid checksum is usually caused by a noisy link.

**ZSTATS-TCP-BAD-OFF**

is the number of packets with an invalid data offset in the TCP header received by the TCP Layer of the TCP/IP subsystem during the sample period. An invalid offset usually indicates that the sender of the packet made an internal error in generating the packet or that the receiver of the packet had a byte-swapping problem. This error is very rare; it is usually seen only during protocol development.

**ZSTATS-TCP-BAD-HDR-LEN**

is the number of packets with an invalid header size received by the TCP Layer of the TCP/IP subsystem during the sample period. This error is rare; it usually indicates a problem between IP and TCP.
ZSTATS-TCP-BAD-SEGS

is the number of packets with an invalid segment size received by the TCP Layer of the TCP/IP subsystem during the sample period. This field is reserved for future use.

ZSTATS-TCP-NOT-ACK

is the number of packets, received by the TCP Layer of the TCP/IP subsystem during the sample period, that the subsystem has not acknowledged. This field is reserved for future use.

ZSTATS-TCP-REXMIT

is the number of packets retransmitted during the sample period. Packets are retransmitted when they are not acknowledged within a certain time period. The retransmission of packets can be caused by any of these conditions: the network is overloaded, the other end of the connection is overloaded (so that the appropriate acknowledgments cannot be received and/or sent), or a packet is received that has been corrupted (that is, a packet with an invalid checksum is received).

ZSTATS-TCP-TIMEOUT

is the number of TCP Layer connection timeouts that occurred during the sample period. A connection timeout is recorded each time the keep-alive timer or the retransmission timer expires.

The keep-alive timer expires when the connection is inactive for a designated period of time. This inactivity can be caused by the other end of the connection going down or by network congestion.

The retransmission timer expires when a packet is not acknowledged within a certain time period.

ZSTATS-TCP-IN-CONN

is the number of incoming connection requests received by the TCP Layer of the TCP/IP subsystem during the sample period.

ZSTATS-TCP-OUT-CONN

is the number of outgoing connection requests sent by the TCP Layer of the TCP/IP subsystem during the sample period.

ZSTATS-TCP-IN-PKTS

is the number of incoming packets received by the TCP Layer of the TCP/IP subsystem during the sample period.
ZSTATS-TCP-OUT-PKTS

is the number of outgoing packets sent by the TCP Layer of TCP/IP subsystem during the sample period.

ZSTATS-TCP-NO-PORT

is the number of packets received by the TCP Layer of the TCP/IP subsystem during the sample period for a connection that has been closed or does not exist. This event can be a normal occurrence or it can be caused by a TCP/IP implementation that does not conform to the TCP/IP state table.

ZSTATS-TCP-URGENT

is the number of incoming packets, received by the TCP Layer of the TCP/IP subsystem during the sample period, that had the urgent bit set.

ZSTATS-TCP-CURR-TIME

is the time at which the TCP/IP process read the TCP Layer statistics counters. This time indicates the end of the sample period.

ZSTATS-TCP-RESET-TIME

is the time at which the TCP Layer statistics counters were last reset. This time indicates the beginning of the sample period.

ZSTATS-TCP-CONN-ESTA

the number of connections established.

ZSTATS-TCP-DROPS

the number of connections dropped.

ZSTATS-TCP-CONN-DROPS

the number of embryonic connections dropped.

ZSTATS-TCP-CLOSED

the number of connections closed (including dropped).

ZSTATS-TCP-SEGS-TIMED

the number of segments for which round trip time was measured.

ZSTATS-TCP-RTT-UPDATED

the number of round trip times updated.

ZSTATS-TCP-DEL-ACK

the number of delayed ACKs sent.
ZSTATS-TCP-TIMEOUT-DROP
    the number of connections dropped in a retransmit timeout.

ZSTATS-TCP-REXMT-TIMEOUT
    the number of retransmit timeouts.

ZSTATS-TCP-PERSIST-TIMEOUT
    the number of persistent timeouts.

ZSTATS-TCP-KEEP-TIMEOUT
    the number of keep-alive timeouts.

ZSTATS-TCP-KEEP-PROBE
    the number of keep-alive probes sent.

ZSTATS-TCP-KEEP-DROPS
    the number of connections dropped in keep-alive timeout.

ZSTATS-TCP-SND-PACK
    the total number of data packets sent.

ZSTATS-TCP-SND-BYTE
    the total number of data bytes sent.

ZSTATS-TCP-SND-REXMIT-BYTE
    the number of bytes retransmitted.

ZSTATS-TCP-SND-ACKS
    the number of ACK packets sent.

ZSTATS-TCP-SND-PROBE
    the number of window probes sent.

ZSTATS-TCP-SND-URG
    the number of packets sent with the urgent bit set.

ZSTATS-TCP-SND-WIN-UPDATE
    the number of window update packets sent.

ZSTATS-TCP-SND-CTRL
    the number of control (SYN|FIN|RST) packets sent.
ZSTATS-TCP-RCV-PCK
the number of packets received in sequence.

ZSTATS-TCP-RCV-BYTE
the number of bytes received in sequence.

ZSTATS-TCP-RCV-DUP-PACK
the number of duplicate packets received.

ZSTATS-TCP-RCV-DUP-BYTE
the number of duplicate bytes received.

ZSTATS-TCP-RCV-PART-DUP-PACK
the number of packets received with some duplicate data.

ZSTATS-TCP-RCV-PART-DUP-BYTE
the number of duplicate bytes received in partially duplicate packets.

ZSTATS-TCP-RCV-OUT-PACK
the number of out-of-order packets received.

ZSTATS-TCP-RCV-OUT-BYTE
the number of out-of-order bytes received.

ZSTATS-TCP-RCV-WIN-PACK
the number of packets received over the window boundary.

ZSTATS-TCP-RCV-WIN-BYTE
the number of bytes received over the window boundary.

ZSTATS-TCP-RCV-AFTER-CLOSE
the number of packets received after the close.

ZSTATS-TCP-RCV-WIN-PROBE
the number of window probe packets received.

ZSTATS-TCP-RCV-DUP-ACK
the number of duplicate ACK packets received.

ZSTATS-TCP-RCV-ACK-TOO-MUCH
the number of duplicate ACK packets received for unsent data.
ZSTATS-TCP-RCV-ACK-PACK
    the number of ACK packets received.

ZSTATS-TCP-RCV-ACK-BYTE
    the number of ACK bytes acknowledged by received ACKs.

ZSTATS-TCP-RCV-WIN-UPDATE
    the number of window update packets received.

ZTCI-MAP-UDP-STATS
    is the extensible structured token that lists the statistics counters maintained for the
    UDP Layer during the sample period. These statistics are returned only for the
    PROC object. This token has the token number ZTCI-TNM-UDP-STATS and the
    DDL definition ZTCI-DDL-UDP-STATS.

    Each counter is defined in a separate field. The fields in the
    ZTCI-MAP-UDP-STATS token are:

ZSTATS-UDP-BAD-HDR-LEN
    is the number of packets with an invalid header size received by the UDP
    Layer of the TCP/IP subsystem during the sample period. This error is very
    rare; it usually indicates a problem between IP and UDP.

ZSTATS-UDP-BAD-XSUM
    is the number of packets with an invalid checksum value received by the UDP
    Layer of the TCP/IP subsystem during the sample period. An invalid checksum
    is usually caused by a noisy link.

ZSTATS-UDP-BAD-LEN
    is the number of packets received by the UDP Layer of the TCP/IP subsystem
    during the sample period that contains either more or less data than has been
    specified in their header. This error is rare; it usually indicates that the sender
    has a protocol error or that the receiver has a byte-ordering problem.

ZSTATS-UDP-IN-PKTS
    is the number of incoming packets received by the UDP Layer of the TCP/IP
    subsystem during the sample period.

ZSTATS-UDP-OUT-PKTS
    is the number of outgoing packets sent by the UDP Layer of the TCP/IP
    subsystem during the sample period.
ZSTATS-UDP-CURR-TIME
is the time at which the TCP/IP process read the UDP Layer statistics counters. This time indicates the end of the sample period.

ZSTATS-UDP-RESET-TIME
is the time at which the UDP Layer statistics counters were last reset. This time indicates the beginning of the sample period.

ZTCI-MAP-IP-STATS
is the extensible structured token that lists the statistics counters maintained for the IP Layer during the sample period. These statistics are returned only for the PROC object. This token has the token number ZTCI-TNM-IP-STATS and the DDL definition ZTCI-DDL-IP-STATS.

Each counter is defined in a separate field. The fields in the ZTCI-MAP-IP-STATS token are:

ZSTATS-IP-TOTAL
is the number of incoming packets received by the IP Layer of TCP/IP subsystem during the sample period.

ZSTATS-IP-BAD-XSUM
is the number of packets with an invalid checksum value received by the IP Layer of the TCP/IP subsystem during the sample period. An invalid checksum is usually caused by a noisy link.

ZSTATS-IP-TOO-SHORT
is the number of packets, received by the IP Layer of the TCP/IP subsystem during the sample period, that contained less data than was specified in their header. This can be caused by noisy links, a protocol error by the sender of the packet, or a byte-swapping problem on the receiver.

ZSTATS-IP-TOO-SMALL
is the number of packets, received by the IP Layer of the TCP/IP subsystem during the sample period, that contained less data than expected when read into the local buffers. This error is rare; it usually indicates a problem with the local machine’s buffering scheme.

ZSTATS-IP-BAD-HDR-LEN
is the number of packets received by the IP Layer of the TCP/IP subsystem during the sample period with a header size that is larger than the header length provided in the packet. This error is rare and indicates either a problem with the sender of the packet, or a problem in reading the data from the link controller to IP.
RESETSTATS Command

ZSTATS-IP-BAD-LEN

is the number of packets received by the IP Layer of the TCP/IP subsystem during the sample period with a packet length that was shorter than expected. This error is very similar to ZSTATS-IP-BAD-HDR-LEN and is usually caused by similar conditions.

ZSTATS-IP-FRAG

is the number of packet fragments received by the IP Layer of the TCP/IP subsystem during the sample period. Usually, a packet is fragmented when the size of the packet is too large for a particular gateway or network. This statistic might indicate that the sender’s maximum segment size is too large for the connection.

ZSTATS-IP-FRAG-DROP

is the number of packet fragments dropped by the IP Layer of the TCP/IP subsystem during the sample period. A fragment is dropped when memory cannot be allocated for the fragment, or the fragment is a duplicate of a fragment that has already been received.

ZSTATS-IP-FRAG-TIMEOUT

is the number of packet fragments, received by the IP Layer of the TCP/IP subsystem during the sample period, that have timed out before the whole packet was received. This is usually caused by congestion, noisy links, or some event that prevents one of the fragments from being received with the rest of the fragments.

ZSTATS-IP-FWD

is the number of packets destined for another host that have been forwarded by the IP Layer of the TCP/IP subsystem during the sample period.

ZSTATS-IP-CANT-FWD

is the number of packets destined for another host that were received but not forwarded by the IP Layer of the TCP/IP subsystem during the sample period. The packets could not be forwarded because either the local host is not configured as a gateway or there is no route available to the specified destination.

ZSTATS-IP-REDIR-SENT

is the number of ICMP Redirect messages sent by the IP Layer of the TCP/IP subsystem during the sample period. Redirect messages are sent to the source host to indicate that there is a shorter path to the destination. The source host should either send the packet directly to the destination host or send the packet to another gateway.
ZSTATS-IP-OUT-PKTS
  is the number of outgoing packets sent by the IP Layer of the TCP/IP subsystem during the sample period.

ZSTATS-IP-CURR-TIME
  is the time at which the TCP/IP process read the IP Layer statistics counters. This time indicates the end of the sample period.

ZSTATS-IP-RESET-TIME
  is the time at which the IP Layer statistics were last reset. This time indicates the beginning of the sample period.

ZTCI-MAP-ROUTE-STATS
  is the extensible structured token that lists the routing statistics counters maintained for the IP Layer during the sample period. These statistics are returned only for the PROC object. This token has the token number ZTCI-TNM-ROUTE-STATS and the DDL definition ZTCI-DDL-ROUTE-STATS.

Each counter is defined in a separate field. The fields in the ZTCI-MAP-ROUTE-STATS token are:

ZSTATS-ROUTE-BAD-REDIR
  is the number of Redirect messages received by the IP Layer of the TCP/IP subsystem during the sample period.

ZSTATS-ROUTE-DYN
  is the number of dynamic route messages received by the IP Layer of the TCP/IP subsystem during the sample period. These messages indicate where the TCP/IP subsystem should route messages for a specific destination.

ZSTATS-ROUTE-NEW-GW
  is the number of messages received by the IP Layer of the TCP/IP subsystem during the sample period that established a route for a new or unknown gateway.

ZSTATS-ROUTE-NOT-REACH
  is the number of messages, received by the IP Layer of the TCP/IP subsystem during the sample period, that indicate the specified destination is unreachable.

ZSTATS-ROUTE-WILD-CARD
  is the number of wild-card matches found during the sample period.
**ZSTATS-ROUTE-CURR-TIME**

is the time at which the TCP/IP process read the routing statistics counters. This time indicates the end of the sample period.

**ZSTATS-ROUTE-RESET-TIME**

is the time at which the routing statistics counters were last reset. This time indicates the beginning of the sample period.

**ZTCI-MAP-ICMP-STATS**

is the extensible structured token that lists the statistics counters maintained for the ICMP Layer during the sample period. These statistics are returned only for the PROC object. This token has the token number ZTCI-TNM-ICMP-STATS and the DDL definition ZTCI-DDL-ICMP-STATS.

Each counter is defined in a separate field. The fields in the ZTCI-MAP-ICMP-STATS token are:

**ZSTATS-ICMP-ERR**

is the number of times an ICMP error has been generated during the sample period. Note that Redirect messages are not included in the total. ICMP errors can be caused by any of these reasons: invalid IP options, problems in forwarding IP packets, or a UDP server crash.

**ZSTATS-ICMP-OLD-SHORT**

is the number of packets, received by the ICMP Layer of the TCP/IP subsystem during the sample period, that were too short. This field is reserved for future use.

**ZSTATS-ICMP-OLD-ICMP**

is the number of invalid ICMP packets that have been received during the sample period.

**ZSTATS-ICMP-OUT-HIST**

is the output histogram of ICMP messages recorded during the sample period. This histogram indicates the number of each type of ICMP message that the TCP/IP subsystem sent to the network. Statistics are maintained in this histogram for these ICMP messages:

- Type 0  Echo Reply
- Type 3  Destination Unreachable
- Type 4  Source Quench
- Type 5  Redirect
- Type 8  Echo
Type 11  Time Exceeded
Type 12  Parameter Problem
Type 13  Timestamp
Type 14  Timestamp Reply
Type 15  Information Request
Type 16  Information Reply

For more detailed descriptions of these messages, see the STATISTICS command. This field is reserved for future use.

ZSTATS-ICMP-BADCODE

is the number of packets, received by the ICMP Layer of the TCP/IP subsystem during the sample period, that contain an invalid ICMP packet type code in the header. For a description of the packet type codes currently implemented, see the description of the ZSTATS-ICMP-IN-HIST field.

ZSTATS-ICMP-TOO-SHORT

is the number of packets, received by the ICMP Layer of the TCP/IP subsystem during the sample period, that were shorter than the minimum length allowed for an ICMP packet. Short packets are usually caused by a noisy link.

ZSTATS-ICMP-BAD-XSUM

is the number of packets with an invalid checksum received by the ICMP Layer of the TCP/IP subsystem during the sample period. An invalid checksum is usually caused by a noisy link.

ZSTATS-ICMP-BAD-LEN

is the number of packets received by the ICMP Layer of the TCP/IP subsystem during the sample period with a length that is shorter than the length field specified in the header. This error, usually caused by a noisy link, is rarely reported because the checksum routine also detects this problem.

ZSTATS-ICMP-REFLECT

is the number of ICMP packets received by the ICMP Layer of the TCP/IP subsystem during the sample period that have been sent a response. Note that not all ICMP packets require a response.

ZSTATS-ICMP-IN-HIST

is the input histogram of ICMP messages recorded during the sample period. This histogram indicates the number of each type of ICMP message that the
TCP/IP subsystem has received from the network. Statistics are maintained in this histogram for these messages:

Type 0   Echo Reply  
Type 3   Destination Unreachable  
Type 4   Source Quench  
Type 5   Redirect  
Type 8   Echo  
Type 11  Time Exceeded  
Type 12  Parameter Problem  
Type 13  Timestamp  
Type 14  Timestamp Reply  
Type 15  Information Request  
Type 16  Information Reply

For detailed descriptions of these messages, see the STATISTICS Command on page 6-73.

ZSTATS-ICMP-CURR-TIME

is the time at which the TCP/IP process read the ICMP Layer statistics counters. This time indicates the end of the sample period.

ZSTATS-ICMP-RESET-TIME

is the time at which the ICMP Layer statistics counters were last reset. This time indicates the beginning of the sample period.

ZTCI-MAP-STATS-SUBNET

is the extensible structured token that lists the statistics counters maintained for the subnet interface during the sample period. These statistics are returned only for the SUBNET object. This token has the token number ZTCI-TNM-STATS-SUBNET and the DDL definition ZTCI-DDL-STATS-SUBNET.

Each counter is defined in a separate field. The fields in the ZTCI-MAP-STATS-SUBNET token are:

ZSTATS-SUBNET-OUT-PKTS

is the number of packets sent by the subnet interface during the sample period.

ZSTATS-SUBNET-IN-PKTS

is the number of packets received by the subnet interface during the sample period.
ZSTATS-SUBNET-OUT-ERRS

is the number of errors that occurred while the subnet interface was sending packets during the sample period. Each output error also generates one of these event messages: ZTCI-EVT-IOP-READ-ERR, ZTCI-EVT-IOP-WRITE-ERR, or ZTCI-EVT-IOP-AWAITIO-ERR.

ZSTATS-SUBNET-IN-ERRS

is the number of errors that occurred while the subnet interface was receiving packets during the sample period. Each input error also generates one of these event messages: ZTCI-EVT-IOP-READ-ERR, ZTCI-EVT-IOP-WRITE-ERR, or ZTCI-EVT-IOP-AWAITIO-ERR.

ZSTATS-SUBNET-CURR-TIME

is the time at which the TCP/IP process read the subnet statistics counters. This time indicates the end of the sample period.

ZSTATS-SUBNET-RESET-TIME

is the time at which the subnet statistics counters were last reset. This time indicates the beginning of the sample period.

ZTCI-MAP-STATS-ROUTE

is the extensible structured token that lists the statistics counters maintained for the route during the sample period. These statistics are returned only for the ROUTE object. This token has the token number ZTCI-TNM-STATS-ROUTE and the DDL definition ZTCI-DDL-STATS-ROUTE.

The fields in the ZTCI-MAP-STATS-ROUTE token are:

ZUSAGE

indicates the number of times the specified route has been used to send IP datagrams.

ZSTATS-ROUTE-CURR-TIME

is the time at which the TCP/IP process read the route statistics counters. This time indicates the end of the sample period.

ZSTATS-ROUTE-RESET-TIME

is the time at which the route statistics counters were last reset. This time indicates the beginning of the sample period.

ZCOM-TKN-OBJNAME

specifies the name of a TCP/IP object for which statistics information was requested. Unless there is an error that prevents the RESETSTATS command from being attempted at all, this token is present in every response record.
ZCOM-TKN-OBJNAME is described in Data Communications Standard Definitions on page 5-9.

ZCOM-TKN-OBJTYPE

specifies the type of object for which information was requested. For more details, see Data Communications Standard Definitions on page 5-9.

ZSPI-TKN-RETCODE

specifies whether the command has been completed successfully. The value of this token remains zero (ZCOM-ERR-OK) when the command is completed successfully with or without a warning. Note that if the empty response (ZCOM-ERR-EMPT-RSP) is returned, the value of this token is nonzero, although the command is completed successfully.

If the command is not completed successfully, the value of this token indicates the reason the command failed. Always check the contents of the error list to determine the token or value in error. For details on the ZCOM return codes, see the SPI Common Extensions Manual.

The possible nonzero values of RETCODE in response to the RESETSTATS command are:

- ZCOM-ERR-TKN-REQ (-29)
- ZCOM-ERR-TKN-DUP (-27)
- ZCOM-ERR-TKN-CODE-INV (-26)
- ZCOM-ERR-SSID-INV (-24)
- ZCOM-ERR-SPI-ERR (-23)
- ZCOM-ERR-SECUR-VIOL (-22)
- ZCOM-ERR-OBJ-NOT-SUPP (-18)
- ZCOM-ERR-OBJ-NOT-FOUND (-17)
- ZCOM-ERR-CMD-TOO-LARGE (-8)
- ZCOM-ERR-CMD-NOT-SUPP (-5)

ZSPI-TKN-ERRLIST

indicates the beginning of an error list. For more details, see SPI Standard Definitions on page 5-1.

ZSPI-TKN-ENDLIST

indicates the end of a data list or an error list. For more details, see SPI Standard Definitions on page 5-1.
ZSPI-TKN-CONTEXT

specifies whether the response continues with more response messages. If this token is present, the response continues. If this token is not present, this response message is the last of the response. For more details, see SPI Standard Definitions on page 5-1.

Operational Notes

• RESETSTATS is a sensitive command.

• When the RESETSTATS command is completed, the object remains in the same summary state as before the command was executed.

• The summary state of the object is not checked when the RESETSTATS command is issued.

• To obtain statistics without initializing the counters, use the STATISTICS command.

START Command

The START command initiates the operation of subnets and routes. When the subsystem finishes processing this command, the specified object is placed in the STARTED summary state. You can start a subnet or a route, but not a process.

You can use an object-name template to start multiple subnets or routes. For more details, see Object-Name Templates on page 3-6.

When you use the START command, the object must be in the STOPPED summary state. If the object is in the STARTED summary state and you issue this command, a warning is returned.

Command

ZCOM-CMD-START

Object Type

ZCOM-OBJ-SUBNET or ZCOM-OBJ-ROUTE

Tokens in Command Buffer

| ZSPI-TKN-SSID  | token-type ZSPI-TYP-SSID. | !{} |
| ZCOM-TKN-OBJNAME | token-type ZSPI-TYP-STRING. | !{} |
| ZSPI-TKN-MAXRESP | token-type ZSPI-TYP-INT. |
| ZSPI-TKN-CONTEXT | token-type ZSPI-TYP-BYTESTRING. |
| ZSPI-TKN-ALLOW-TYPE | token-type ZSPI-TYP-ENUM. |
| ZSPI-TKN-COMMENT | token-type ZSPI-TYP-STRING. |
**Tokens in Command Buffer**

**ZSPI-TKN-SSID**

specifies the subsystem ID, which your management application must pass to SSINIT. This token is described in [SPI Standard Definitions](#) on page 5-1, and in the [SPI Programming Manual](#).

**ZCOM-TKN-OBJNAME**

specifies the subnet or route to be started. Your application must provide this token in the command. The object name can include wild-card characters. Using wild-card characters allows the command to affect multiple objects. This token is described more fully in [Data Communications Standard Definitions](#) on page 5-9.

(ZSPI-TKN-) MAXRESP, CONTEXT, ALLOW-TYPE, and COMMENT

are standard SPI tokens that your application can provide. These tokens are described in [SPI Standard Definitions](#) on page 5-1.

**Tokens in Response Buffer**

**ZSPI-TKN-SSID**

is an echo of the subsystem ID value provided in the command by your application. For more details, see [SPI Standard Definitions](#) on page 5-1.

**ZSPI-TKN-SERVER-VERSION**

indicates the software release version of the TCP/IP subsystem. For more details, see [SPI Standard Definitions](#) on page 5-1.

**ZSPI-TKN-DATALIST**

indicates the beginning of a data list. The ZSPI-TKN-MAXRESP token must have a value other than zero for data lists to be included in the response. For more details, see [SPI Standard Definitions](#) on page 5-1.
ZCOM-TKN-OBJNAME

specifies the name of the TCP/IP subnet or route started by the command. Unless there is an error that prevents the START command from being attempted at all, this token is present in every response record. If you have used an object-name template, a separate ZCOM-TKN-OBJNAME token is returned for each subnet or route that meets the template requirements. ZCOM-TKN-OBJNAME is described in Data Communications Standard Definitions on page 5-9.

ZCOM-TKN-OBJTYPE

specifies the type of object started. The possible values are ZCOM-OBJ-SUBNET and ZCOM-OBJ-ROUTE. This token is present in every response record. The value of the ZCOM-TKN-OBJTYPE token is the same as the value of the ZSPI-TKN-OBJECT-TYPE token in the header. For more details, see Data Communications Standard Definitions on page 5-9.

ZSPI-TKN-RETCODE

specifies whether the command has been completed successfully. The value of this token remains zero (ZCOM-ERR-OK) when the command is completed successfully with or without a warning. Note that if the empty response (ZCOM-ERR-EMPT-RSP) is returned, the value of this token is nonzero, although the command is completed successfully.

If the command is not completed successfully, the value of this token indicates the reason the command failed. Always check the contents of the error list to determine the token or value in error. For details on the ZCOM return codes and associated error lists, see the SPI Common Extensions Manual. The possible nonzero values of RETCODE in response to the START command are:

- ZCOM-ERR-TKN-REQ (-29)
- ZCOM-ERR-TKN-DUP (-27)
- ZCOM-ERR-TKN-CODE-INV (-26)
- ZCOM-ERR-SSID-INV (-24)
- ZCOM-ERR-SPI-ERR (-23)
- ZCOM-ERR-SECUR-VIOL (-22)
- ZCOM-ERR-OBJ-NOT-SUPP (-18)
- ZCOM-ERR-OBJ-NOT-FOUND (-17)
- ZCOM-ERR-OBJ-ALRDY-IN-SUMSTATE (-16)
- ZCOM-ERR-CMD-TOO-LARGE (-8)
- ZCOM-ERR-CMD-NOT-SUPP (-5)
ZSPI-TKN-ERRLIST

indicates the beginning of an error list. For more details, see SPI Standard Definitions on page 5-1.

ZSPI-TKN-ENDLIST

indicates the end of a data list or an error list. For more details, see SPI Standard Definitions on page 5-1.

ZSPI-TKN-CONTEXT

specifies whether the response continues with more response messages. If this token is present, the response continues. If this token is not present, this response message is the last of the response. For more details, see SPI Standard Definitions on page 5-1.

Operational Notes

- START is a sensitive command.
- When the START command is completed, the object is placed in the STARTED summary state.
- The object must be in the STOPPED summary state when the START command is issued. If the object is in the STARTED summary state, an error is returned.
- To terminate the operation of subnets and routes, use the ABORT or STOP command.
- The #ARP and #IP ports must be in the STARTED state when the START command is issued.

STATISTICS Command

The STATISTICS command returns statistics about the execution history for the specified object(s). The type of information returned by this command is the same as that returned by the RESETSTATS command, but this command does not initialize the statistics counters after the TCP/IP subsystem places the information in a response message. This is a nonsensitive command.

Various statistics are maintained during the sample period defined by the RESET-TIME and CURR-TIME fields.

You can issue this command for a process, subnet, or route. You can use an object-name template for the subnet and route objects to return information about multiple objects. The summary state of the object is not checked by the TCP/IP subsystem when you issue the STATISTICS command.
**Command**

ZCOM-CMD-STATISTICS

**Object Type**

ZCOM-OBJ-PROC or ZCOM-OBJ-SUBNET or ZCOM-OBJ-ROUTE or ZCOM-OBJ-ADDRMAP

**Tokens in Command Buffer**

<table>
<thead>
<tr>
<th>Token Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZSPI-TKN-SSID</td>
<td>token-type ZSPI-TYP-SSID.</td>
</tr>
<tr>
<td>ZCOM-TKN-OBJNAME</td>
<td>token-type ZSPI-TYP-STRING.</td>
</tr>
<tr>
<td>ZSPI-TKN-MAXRESP</td>
<td>token-type ZSPI-TYP-INT.</td>
</tr>
<tr>
<td>ZSPI-TKN-CONTEXT</td>
<td>token-type ZSPI-TYP-BYTESTRING.</td>
</tr>
<tr>
<td>ZSPI-TKN-ALLOW-TYPE</td>
<td>token-type ZSPI-TYP-ENUM.</td>
</tr>
<tr>
<td>ZSPI-TKN-COMMENT</td>
<td>token-type ZSPI-TYP-STRING.</td>
</tr>
</tbody>
</table>
Tokens in Response Buffer

ZSPI-TKN-SSID token-type ZSPI-TYP-SSID. !{}
ZSPI-TKN-SERVER-VERSION token-type ZSPI-TYP-UINT. !{}
ZSPI-TKN-DATALIST token-type ZSPI-TYP-LIST. !...

ZTCI-MAP-TCP-STATS !{}

!OBJ-PROC
def ZTCI-DDL-TCP-STATS. !All fields are present.
02 ZSTATS-TCP-BAD-XSUM type ZSPI-DDL-INT2.
02 ZSTATS-TCP-BAD-OFF type ZSPI-DDL-INT2.
02 ZSTATS-TCP-BAD-HDR-LEN type ZSPI-DDL-INT2.
02 ZSTATS-TCP-BAD-SEGS type ZSPI-DDL-INT2.
02 ZSTATS-TCP-NOT-ACK type ZSPI-DDL-INT2.
02 ZSTATS-TCP-REXMIT type ZSPI-DDL-INT2.
02 ZSTATS-TCP-TIMEOUT type ZSPI-DDL-INT2.
02 ZSTATS-TCP-IN-CONN type ZSPI-DDL-INT2.
02 ZSTATS-TCP-OUT-CONN type ZSPI-DDL-INT2.
02 ZSTATS-TCP-IN-PKTS type ZSPI-DDL-INT2.
02 ZSTATS-TCP-OUT-PKTS type ZSPI-DDL-INT2.
02 ZSTATS-TCP-NO-PORT type ZSPI-DDL-INT2.
02 ZSTATS-TCP-URGENT type ZSPI-DDL-INT2.
02 ZSTATS-TCP-CURR-TIME type ZSPI-DDL-TIMESTAMP.
02 ZSTATS-TCP-RESET-TIME type ZSPI-DDL-TIMESTAMP.
02 ZSTATS-TCP-CONN-ESTA type ZSPI-DDL-INT2
02 ZSTATS-TCP-DROPS type ZSPI-DDL-INT2
02 ZSTATS-TCP-CONN-DROPS type ZSPI-DDL-INT2
02 ZSTATS-TCP-CLOSED type ZSPI-DDL-INT2
02 ZSTATS-TCP-SEGMENTS-TIMED type ZSPI-DDL-INT2
02 ZSTATS-TCP-RTT-UPDATED type ZSPI-DDL-INT2
02 ZSTATS-TCP-DEL-ACK type ZSPI-DDL-INT2
02 ZSTATS-TCP-TIMEOUT-DROP type ZSPI-DDL-INT2
02 ZSTATS-TCP-REXMT-TIMEOUT type ZSPI-DDL-INT2
02 ZSTATS-TCP-PERSIST-TIMEOUT type ZSPI-DDL-INT2
02 ZSTATS-TCP-KEEP-TIMEOUT type ZSPI-DDL-INT2
02 ZSTATS-TCP-KEEP-PROBE type ZSPI-DDL-INT2
02 ZSTATS-TCP-KEEP-DROPS type ZSPI-DDL-INT2
02 ZSTATS-TCP-SND-PACK type ZSPI-DDL-INT2
02 ZSTATS-TCP-SND-BYTE type ZSPI-DDL-INT2
02 ZSTATS-TCP-SND-REXMIT-BYTE type ZSPI-DDL-INT2
02 ZSTATS-TCP-SND-ACKS type ZSPI-DDL-INT2
02 ZSTATS-TCP-SND-PROBE type ZSPI-DDL-INT2
02 ZSTATS-TCP-SND-URG type ZSPI-DDL-INT2
02 ZSTATS-TCP-SND-WIN-UPDATE type ZSPI-DDL-INT2
02 ZSTATS-TCP-SND-CNTRL type ZSPI-DDL-INT2
02 ZSTATS-TCP-RCV-PACK type ZSPI-DDL-INT2
02 ZSTATS-TCP-RCV-BYTE type ZSPI-DDL-INT2
02 ZSTATS-TCP-RCV-DUP-PACK type ZSPI-DDL-INT2
02 ZSTATS-TCP-RCV-DUP-BYTE type ZSPI-DDL-INT2
02 ZSTATS-TCP-RCV-PART-DUP-PACK type ZSPI-DDL-INT2
02 ZSTATS-TCP-RCV-PART-DUP-BYTE type ZSPI-DDL-INT2
02 ZSTATS-TCP-RCV-OUT-PACK type ZSPI-DDL-INT2
02 ZSTATS-TCP-RCV-OUT-BYTE type ZSPI-DDL-INT2
### STATISTICS Command

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZSTATS-TCP-RCV-WIN-PACK</td>
<td>ZSPI-DDL-INT2</td>
</tr>
<tr>
<td>ZSTATS-TCP-RCV-WIN-BYTE</td>
<td>ZSPI-DDL-INT2</td>
</tr>
<tr>
<td>ZSTATS-TCP-RCV-AFTER-CLOSE</td>
<td>ZSPI-DDL-INT2</td>
</tr>
<tr>
<td>ZSTATS-TCP-RCV-WIN-PROBE</td>
<td>ZSPI-DDL-INT2</td>
</tr>
<tr>
<td>ZSTATS-TCP-RCV-DUP-ACK</td>
<td>ZSPI-DDL-INT2</td>
</tr>
<tr>
<td>ZSTATS-TCP-RCV-ACK-TOO-MUCH</td>
<td>ZSPI-DDL-INT2</td>
</tr>
<tr>
<td>ZSTATS-TCP-RCV-ACK-PACK</td>
<td>ZSPI-DDL-INT2</td>
</tr>
<tr>
<td>ZSTATS-TCP-RCV-ACK-BYTE</td>
<td>ZSPI-DDL-INT2</td>
</tr>
<tr>
<td>ZSTATS-TCP-RCV-WIN-UPDATE</td>
<td>ZSPI-DDL-INT2</td>
</tr>
<tr>
<td>ZSTATS-UDP-BAD-HDR-LEN</td>
<td>ZSPI-DDL-INT2</td>
</tr>
<tr>
<td>ZSTATS-UDP-BAD-XSUM</td>
<td>ZSPI-DDL-INT2</td>
</tr>
<tr>
<td>ZSTATS-UDP-BAD-LEN</td>
<td>ZSPI-DDL-INT2</td>
</tr>
<tr>
<td>ZSTATS-UDP-IN-PKTS</td>
<td>ZSPI-DDL-INT2</td>
</tr>
<tr>
<td>ZSTATS-UDP-OUT-PKTS</td>
<td>ZSPI-DDL-INT2</td>
</tr>
<tr>
<td>ZSTATS-UDP-CURR-TIME</td>
<td>ZSPI-DDL-TIMESTAMP</td>
</tr>
<tr>
<td>ZSTATS-UDP-RESET-TIME</td>
<td>ZSPI-DDL-TIMESTAMP</td>
</tr>
<tr>
<td>ZSTATS-IP-TOTAL</td>
<td>ZSPI-DDL-INT2</td>
</tr>
<tr>
<td>ZSTATS-IP-BAD-XSUM</td>
<td>ZSPI-DDL-INT2</td>
</tr>
<tr>
<td>ZSTATS-IP-TOO-SHORT</td>
<td>ZSPI-DDL-INT2</td>
</tr>
<tr>
<td>ZSTATS-IP-TOO-SMALL</td>
<td>ZSPI-DDL-INT2</td>
</tr>
<tr>
<td>ZSTATS-IP-BAD-HDR-LEN</td>
<td>ZSPI-DDL-INT2</td>
</tr>
<tr>
<td>ZSTATS-IP-BAD-LEN</td>
<td>ZSPI-DDL-INT2</td>
</tr>
<tr>
<td>ZSTATS-IP-FRAG</td>
<td>ZSPI-DDL-INT2</td>
</tr>
<tr>
<td>ZSTATS-IP-FRAG-DROP</td>
<td>ZSPI-DDL-INT2</td>
</tr>
<tr>
<td>ZSTATS-IP-FRAG-TIMEOUT</td>
<td>ZSPI-DDL-INT2</td>
</tr>
<tr>
<td>ZSTATS-IP-FWD</td>
<td>ZSPI-DDL-INT2</td>
</tr>
<tr>
<td>ZSTATS-IP-CANT-FWD</td>
<td>ZSPI-DDL-INT2</td>
</tr>
<tr>
<td>ZSTATS-IP-REDIR-SENT</td>
<td>ZSPI-DDL-INT2</td>
</tr>
<tr>
<td>ZSTATS-IP-OUT-PKTS</td>
<td>ZSPI-DDL-INT2</td>
</tr>
<tr>
<td>ZSTATS-IP-CURR-TIME</td>
<td>ZSPI-DDL-TIMESTAMP</td>
</tr>
<tr>
<td>ZSTATS-IP-RESET-TIME</td>
<td>ZSPI-DDL-TIMESTAMP</td>
</tr>
</tbody>
</table>

{ZTCI-MAP-UDP-STATS} !{} !OBJ-PROC
def ZTCI-DDL-UDP-STATS. !All fields are present.
  02 ZSTATS-UDP-BAD-HDR-LEN type ZSPI-DDL-INT2.
  02 ZSTATS-UDP-BAD-XSUM type ZSPI-DDL-INT2.
  02 ZSTATS-UDP-BAD-LEN type ZSPI-DDL-INT2.
  02 ZSTATS-UDP-IN-PKTS type ZSPI-DDL-INT2.
  02 ZSTATS-UDP-OUT-PKTS type ZSPI-DDL-INT2.
  02 ZSTATS-UDP-CURR-TIME type ZSPI-DDL-TIMESTAMP.
  02 ZSTATS-UDP-RESET-TIME type ZSPI-DDL-TIMESTAMP.
end.

{ZTCI-MAP-IP-STATS} !{} !OBJ-PROC
def ZTCI-DDL-IP-STATS. !All fields are present.
  02 ZSTATS-IP-TOTAL type ZSPI-DDL-INT2.
  02 ZSTATS-IP-BAD-XSUM type ZSPI-DDL-INT2.
  02 ZSTATS-IP-TOO-SHORT type ZSPI-DDL-INT2.
  02 ZSTATS-IP-TOO-SMALL type ZSPI-DDL-INT2.
  02 ZSTATS-IP-BAD-HDR-LEN type ZSPI-DDL-INT2.
  02 ZSTATS-IP-BAD-LEN type ZSPI-DDL-INT2.
  02 ZSTATS-IP-FRAG type ZSPI-DDL-INT2.
  02 ZSTATS-IP-FRAG-DROP type ZSPI-DDL-INT2.
  02 ZSTATS-IP-FRAG-TIMEOUT type ZSPI-DDL-INT2.
  02 ZSTATS-IP-FWD type ZSPI-DDL-INT2.
  02 ZSTATS-IP-CANT-FWD type ZSPI-DDL-INT2.
  02 ZSTATS-IP-REDIR-SENT type ZSPI-DDL-INT2.
  02 ZSTATS-IP-OUT-PKTS type ZSPI-DDL-INT2.
  02 ZSTATS-IP-CURR-TIME type ZSPI-DDL-TIMESTAMP.
  02 ZSTATS-IP-RESET-TIME type ZSPI-DDL-TIMESTAMP.
end.
**ZTCI-MAP-SMEM-STATS**

```
!{}!

!OBJ-PROC

def ZTCI-DDL-SMEM-STATS. !All fields are present.
02 ZSTATS-SMEM-DATA-MDS TYPE ZSPI-DDL-INT2
02 ZSTATS-SMEM-MAX-DATA-MDS TYPE ZSPI-DDL-INT2
02 ZSTATS-SMEM-DUP-MDS TYPE ZSPI-DDL-INT2
02 ZSTATS-SMEM-MAX-DUP-MDS TYPE ZSPI-DDL-INT2
02 ZSTATS-SMEM-NO-DATA-MDS TYPE ZSPI-DDL-INT2
02 ZSTATS-SMEM-NO-DUP-MDS TYPE ZSPI-DDL-INT2
02 ZSTATS-SMEM-RES-RECOV-RUN TYPE ZSPI-DDL-INT2
02 ZSTATS-SMEM-QIO-LIMIT-WARNs TYPE ZSPI-DDL-INT2
02 ZSTATS-SMEM-QIO-DRIVER-ERRs TYPE ZSPI-DDL-INT2
02 ZSTATS-SMEM-CURR-POOL-SPACE TYPE ZSPI-DDL-INT2
02 ZSTATS-SMEM-MAX-POOL-SPACE TYPE ZSPI-DDL-INT2
02 ZSTATS-SMEM-POOL-FAILS TYPE ZSPI-DDL-INT2
02 ZSTATS-SMEM-TOTAL-MBUFS TYPE ZSPI-DDL-INT2
02 ZSTATS-SMEM-CURR-MBUF-INVUSE TYPE ZSPI-DDL-INT2
02 ZSTATS-SMEM-MAX-MBUFS-INVUSE TYPE ZSPI-DDL-INT2
02 ZSTATS-SMEM-MBUF-FAILS TYPE ZSPI-DDL-INT2
02 ZSTATS-SMEM-MAX-DUP-DRIV-MDS TYPE ZSPI-DDL-INT2
02 ZSTATS-SMEM-NO-DUP-DRIV-MDS TYPE ZSPI-DDL-INT2
02 ZSTATS-SMEM-SIZE-128 TYPE ZSPI-DDL-INT2
02 ZSTATS-SMEM-SIZE-256 TYPE ZSPI-DDL-INT2
02 ZSTATS-SMEM-SIZE-1024 TYPE ZSPI-DDL-INT2
02 ZSTATS-SMEM-SIZE-2048 TYPE ZSPI-DDL-INT2
02 ZSTATS-SMEM-SIZE-4096 TYPE ZSPI-DDL-INT2
02 ZSTATS-SMEM-SIZE-8192 TYPE ZSPI-DDL-INT2
02 ZSTATS-SMEM-SIZE-12288 TYPE ZSPI-DDL-INT2
02 ZSTATS-SMEM-SIZE-16384 TYPE ZSPI-DDL-INT2
02 ZSTATS-SMEM-SIZE-32768 TYPE ZSPI-DDL-INT2
02 ZSTATS-SMEM-CURR-TIME TYPE ZSPI-DDL-TIMESTAMP
02 ZSTATS-SMEM-RESET-TIME TYPE ZSPI-DDL-TIMESTAMP

end.
```

**ZTCI-MAP-ROUTE-STATS**

```
!{}!

!OBJ-PROC

def ZTCI-DDL-ROUTE-STATS. !All fields are present.
02 ZSTATS-ROUTE-BAD-REDIR type ZSPI-DDL-INT2.
02 ZSTATS-ROUTE-DYN type ZSPI-DDL-INT2.
02 ZSTATS-ROUTE-NEW-GW type ZSPI-DDL-INT2.
02 ZSTATS-ROUTE-NOT-REACH type ZSPI-DDL-INT2.
02 ZSTATS-ROUTE-WILD-CARD type ZSPI-DDL-INT2.
02 ZSTATS-ROUTE-CURR-TIME type ZSPI-DDL-TIMESTAMP.
02 ZSTATS-ROUTE-RESET-TIME type ZSPI-DDL-TIMESTAMP.

end.
```
ZTCI-MAP-ICMP-STATS

def ZTCI-DDL-ICMP-STATS.  !All fields are present.
  02 ZSTATS-ICMP-ERR           type ZSPI-DDL-INT2.
  02 ZSTATS-ICMP-OLD-SHORT     type ZSPI-DDL-INT2.
  02 ZSTATS-ICMP-OLD-ICMP      type ZSPI-DDL-INT2.
  02 ZSTATS-ICMP-OUT-HIST      type ZSPI-DDL-INT2
      occurs 17 times.
  02 ZSTATS-ICMP-BADCODE       type ZSPI-DDL-INT2.
  02 ZSTATS-ICMP-TOO-SHORT     type ZSPI-DDL-INT2.
  02 ZSTATS-ICMP-XSUM          type ZSPI-DDL-INT2.
  02 ZSTATS-ICMP-BAD-LEN       type ZSPI-DDL-INT2.
  02 ZSTATS-ICMP-REFLECT       type ZSPI-DDL-INT2.
  02 ZSTATS-ICMP-IN-HIST       type ZSPI-DDL-INT2
      occurs 17 times.
  02 ZSTATS-ICMP-CURR-TIME     type ZSPI-DDL-TIMESTAMP.
  02 ZSTATS-ICMP-RESET-TIME    type ZSPI-DDL-TIMESTAMP.
  02 ZSTATS-ICMP-BAD-RAP-SUBCODE type ZSPI-DDL-INT2
  02 ZSTATS-ICMP-BAD-ADDRLIST  type ZSPI-DDL-INT2
  02 ZSTATS-ICMP-BAD-WPA        type ZSPI-DDL-INT2
  02 ZSTATS-ICMP-RDOSC=MRECPRDED type ZSPI-DDL-INT2
  02 ZSTATS-ICMP-BAD-
end.

ZTCI-MAP-STATS-SUBNET

def ZTCI-DDL-STATS-SUBNET.  !All fields are present.
  02 ZSTATS-SUBNET-OUT-PKTS    type ZSPI-DDL-INT2.
  02 ZSTATS-SUBNET-IN-PKTS     type ZSPI-DDL-INT2.
  02 ZSTATS-SUBNET-OUT-ERRS    type ZSPI-DDL-INT2.
  02 ZSTATS-SUBNET-IN-ERRS     type ZSPI-DDL-INT2.
  02 ZSTATS-SUBNET-CURR-TIME   type ZSPI-DDL-TIMESTAMP.
  02 ZSTATS-SUBNET-RESET-TIME  type ZSPI-DDL-TIMESTAMP.
end.

ZTCI-MAP-STATS-ROUTE

def ZTCI-DDL-STATS-ROUTE.  !All fields are present.
  02 ZUSAGE                   type ZSPI-DDL-INT2.
  02 ZSTATS-ROUTE-CURR-TIME   type ZSPI-DDL-TIMESTAMP.
  02 ZSTATS-ROUTE-RESET-TIME  type ZSPI-DDL-TIMESTAMP.
end.
Commands and Responses

ST A TISTICS Command

Tokens in Command Buffer

ZSPI-TKN-SSID

specifies the subsystem ID, which your management application must pass to SSINIT. This token is described in SPI Standard Definitions on page 5-1, and in the SPI Programming Manual.

ZCOM-TKN-OBJNAME

specifies the object for which the subsystem is to return statistical information. Your application must provide this token in the command. You can use an object-name template for the SUBNET and ROUTE objects. This token is described in Data Communications Standard Definitions on page 5-9.

(ZSPI-TKN-) MAXRESP, CONTEXT, ALLOW-TYPE, and COMMENT

are standard SPI tokens that your application can provide. These tokens are described in SPI Standard Definitions on page 5-1.

Tokens in Response Buffer

ZSPI-TKN-SSID

is an echo of the subsystem ID value provided in the command by your application. For more details, see SPI Standard Definitions on page 5-1.

ZSPI-TKN-SERVER-VERSION

is the software release version of the TCP/IP subsystem. For more details, see SPI Standard Definitions on page 5-1.

ZTCI-DDL-STATS-ADDRMAP

<table>
<thead>
<tr>
<th>Token</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZIP-ADDR-LOCAL</td>
<td>ZSPI-DDL-INT2</td>
</tr>
<tr>
<td>ZX121-ADDR-LOCAL</td>
<td>ZSPI-DDL-CHAR16</td>
</tr>
<tr>
<td>ZSTATS-ADDRMAP-IN-CALLS</td>
<td>ZSPI-DDL-INT2</td>
</tr>
<tr>
<td>ZSTATS-ADDRMAP-OUT-CALLS</td>
<td>ZSPI-DDL-INT2</td>
</tr>
<tr>
<td>ZSTATS-ADDRMAP-LAST-USED</td>
<td>ZSPI-DDL-TIMESTAMP</td>
</tr>
<tr>
<td>ZSTATS-ADDRMAP-CURR-TIME</td>
<td>ZSPI-DDL-TIMESTAMP</td>
</tr>
<tr>
<td>ZSTATS-ADDRMAP-RESET-TIME</td>
<td>ZSPI-DDL-TIMESTAMP</td>
</tr>
</tbody>
</table>

end.

ZCOM-TKN-OBJNAME token-type ZSPI-TYP-STRING.
ZCOM-TKN-OBJTYPE token-type ZSPI-TYP-ENUM.
ZSPI-TKN-RETCODE token-type ZSPI-TYP-ENUM.
ZSPI-TKN-ERRLIST token-type ZSPI-TYP-LIST.
ZSPI-TKN-ENDLIST token-type ZSPI-TYP-SSCTL.
ZSPI-TKN-ENDLIST token-type ZSPI-TYP-SSCTL.
ZSPI-TKN-CONTEXT token-type ZSPI-TYP-BYTESTRING.
ZSPI-TKN-DATALIST

indicates the beginning of a data list. The ZSPI-TKN-MAXRESP token must have a value other than zero for data lists to be included in the response. For more details, see SPI Standard Definitions on page 5-1.

ZTCI-MAP-TCP-STATS

is the extensible structured token that lists the statistics counters maintained for the TCP Layer during the sample period. These statistics are returned only for the PROC object. This token has the token number ZTCI-TNM-TCP-STATS and the DDL definition ZTCI-DDL-TCP-STATS.

Each counter is defined in a separate field. The fields in the ZTCI-MAP-TCP-STATS token are:

ZSTATS-TCP-BAD-XSUM

is the number of packets with an invalid checksum value received by the TCP Layer of the TCP/IP subsystem during the sample period. An invalid checksum is usually caused by a noisy link.

ZSTATS-TCP-BAD-OFF

is the number of packets received by the TCP Layer of the TCP/IP subsystem during the sample period with an invalid data offset in the TCP header. An invalid data offset usually indicates that either the sender of the packet made an internal error in generating the packet or that the receiver of the packet had a byte-swapping problem. This error is very rare; it is usually seen only during protocol development.

ZSTATS-TCP-BAD-HDR-LEN

is the number of packets with an invalid header size received by the TCP Layer of the TCP/IP subsystem during the sample period. This error is very rare; it usually indicates a problem between IP and TCP.

ZSTATS-TCP-BAD-SEGS

is the number of packets with an invalid segment size received by the TCP Layer of the TCP/IP subsystem during the sample period. This field is reserved for future use.

ZSTATS-TCP-NOT-ACK

is the number of packets, received by the TCP Layer of the TCP/IP subsystem during the sample period, that the subsystem has not acknowledged. This field is reserved for future use.
ZSTATS-TCP-REXMIT

is the number of packets retransmitted during the sample period. Packets are retransmitted when a packet is not acknowledged within a certain time interval. The retransmission of packets can be caused by any of these conditions: the network is overloaded, the other end of the connection is overloaded (so that appropriate acknowledgments cannot be received or sent), or a packet is received that has been corrupted (that is, a packet with an invalid checksum is received).

ZSTATS-TCP-TIMEOUT

is the number of connection timeouts that occurred at the TCP Layer of the TCP/IP subsystem during the sample period. A connection timeout is recorded each time the keep-alive timer or the retransmission timer expires.

The keep-alive timer expires when the connection is inactive for a certain period of time. This inactivity can be caused by the other end of the connection going down or by network congestion.

The retransmission timer expires when a packet is not acknowledged within a certain time period. The retransmission of packets can be caused by any of these conditions: the network is overloaded, the other end of the connection is overloaded (so that appropriate acknowledgments cannot be received and/or sent), or a packet is received that has been corrupted (that is, a packet with an invalid checksum is received).

ZSTATS-TCP-IN-CONN

is the number of incoming connection requests received by the TCP Layer of the TCP/IP subsystem during the sample period.

ZSTATS-TCP-OUT-CONN

is the number of outgoing connection requests sent by the TCP Layer of the TCP/IP subsystem during the sample period.

ZSTATS-TCP-IN-PKTS

is the number of incoming packets received by the TCP Layer of the TCP/IP subsystem during the sample period.

ZSTATS-TCP-OUT-PKTS

is the number of outgoing packets sent by the TCP Layer of TCP/IP subsystem during the sample period.

ZSTATS-TCP-NO-PORT

is the number of packets received by the TCP Layer of the TCP/IP subsystem during the sample period for a connection that has been closed or does not
exist. This event can be a normal occurrence or it can be caused by a TCP/IP implementation that does not conform to the TCP/IP specification’s state table.

\textbf{ZSTATS-TCP-URGENT}

is the number of incoming packets with the urgent bit set received by the TCP Layer of the TCP/IP subsystem during the sample period.

\textbf{ZSTATS-TCP-CURR-TIME}

is the time at which the TCP/IP process read the TCP Layer statistics counters. This time indicates the end of the sample period.

\textbf{ZSTATS-TCP-RESET-TIME}

is the time at which the TCP Layer statistics counters were last reset. This time indicates the beginning of the sample period.

\textbf{ZSTATS-TCP-CONN-ESTA}

the number of connections established.

\textbf{ZSTATS-TCP-DROPS}

the number of connections dropped.

\textbf{ZSTATS-TCP-CONN-DROPS}

the number of embryonic connections dropped.

\textbf{ZSTATS-TCP-CLOSED}

the number of connections closed (including dropped).

\textbf{ZSTATS-TCP-SEGS-TIMED}

the number of segments for which round-trip time was measured.

\textbf{ZSTATS-TCP-RTT-UPDATED}

the number of round-trip times updated.

\textbf{ZSTATS-TCP-DEL-ACK}

the number of delayed ACKs sent.

\textbf{ZSTATS-TCP-TIMEOUT-DROP}

the number of connections dropped in a retransmit timeout.

\textbf{ZSTATS-TCP-REXMT-TIMEOUT}

the number of retransmit timeouts.
ZSTATS-TCP-PERSIST-TIMEOUT
  the number of persistent timeouts.

ZSTATS-TCP-KEEP-TIMEOUT
  the number of keep-alive timeouts.

ZSTATS-TCP-KEEP-PROBE
  the number of keep-alive probes sent.

ZSTATS-TCP-KEEP-DROPS
  the number of connections dropped in keep-alive timeouts.

ZSTATS-TCP-SND-PACK
  the total number of data packets sent.

ZSTATS-TCP-SND-BYTE
  the total number of data bytes sent.

ZSTATS-TCP-SND-REXMIT-BYTE
  the number of bytes retransmitted.

ZSTATS-TCP-SND-ACKS
  the number of ACK packets sent.

ZSTATS-TCP-SND-PROBE
  the number of window probes sent.

ZSTATS-TCP-SND-URG
  the number of packets sent with the urgent bit set.

ZSTATS-TCP-SND-WIN-UPDATE
  the number of window update packets sent.

ZSTATS-TCP-SND-CTRL
  the number of control (SYN|FIN|RST) packets sent.

ZSTATS-TCP-RCV-PCK
  the number of packets received in sequence.

ZSTATS-TCP-RCV-BYTE
  the number of bytes received in sequence.
ZSTATS-TCP-RCV-DUP-PACK
    the number of duplicate packets received.

ZSTATS-TCP-RCV-DUP-BYTE
    the number of duplicate bytes received.

ZSTATS-TCP-RCV-PART-DUP-PACK
    the number of packets received with some duplicate data.

ZSTATS-TCP-RCV-PART-DUP-BYTE
    the number of duplicate bytes received in partially duplicate packets.

ZSTATS-TCP-RCV-OUT-PACK
    the number of out-of-order packets received.

ZSTATS-TCP-RCV-OUT-BYTE
    the number of out-of-order bytes received.

ZSTATS-TCP-RCV-WIN-PACK
    the number of packets received over the window boundary.

ZSTATS-TCP-RCV-WIN-BYTE
    the number of bytes received over the window boundary.

ZSTATS-TCP-RCV-AFTER-CLOSE
    the number of packets received after the close.

ZSTATS-TCP-RCV-WIN-PROBE
    the number of window probe packets received.

ZSTATS-TCP-RCV-DUP-ACK
    the number of duplicate ACK packets received.

ZSTATS-TCP-RCV-ACK TOO MUCH
    the number of duplicate ACK packets received for unsent data.

ZSTATS-TCP-RCV-ACK-PACK
    the number of ACK packets received.

ZSTATS-TCP-RCV-ACK-BYTE
    the number of ACK bytes acknowledged by received ACKs.
ZSTATS-TCP-RCV-WIN-UPDATE

the number of window update packets received.

ZTCI-MAP-UDP-STATS

is the extensible structured token that lists the statistics counters maintained for the UDP Layer during the sample period. These statistics are returned only for the PROC object. This token has the token number ZTCI-TNM-UDP-STATS and the DDL definition ZTCI-DDL-UDP-STATS.

Each counter is defined in a separate field. The fields in the ZTCI-MAP-UDP-STATS token are:

ZSTATS-UDP-BAD-HDR-LEN

is the number of packets with an invalid header size received by the UDP Layer of the TCP/IP subsystem during the sample period. This error is very rare; it usually indicates a problem between IP and UDP.

ZSTATS-UDP-BAD-XSUM

is the number of packets with an invalid checksum value received by the UDP Layer of the TCP/IP subsystem during the sample period. An invalid checksum is usually caused by a noisy link.

ZSTATS-UDP-BAD-LEN

is the number of packets received by the UDP Layer of the TCP/IP subsystem during the sample period that contain either more or less data than has been specified in their header. This error is rare; it usually indicates that the sender has a protocol error or that the receiver has a byte-ordering problem.

ZSTATS-UDP-IN-PKTS

is the number of incoming packets received by the UDP Layer of the TCP/IP subsystem during the sample period.

ZSTATS-UDP-OUT-PKTS

is the number of outgoing packets sent by the UDP Layer of the TCP/IP subsystem during the sample period.

ZSTATS-UDP-CURR-TIME

is the time at which the TCP/IP process read the UDP Layer statistics counters. This time indicates the end of the sample period.

ZSTATS-UDP-RESET-TIME

is the time at which the UDP Layer statistics counters were last reset. This indicates the beginning of the sample period.
**ZTCI-MAP-IP-STATS**

is the extensible structured token that lists the statistics counters, which have been maintained for the IP Layer during the sample period. These statistics are returned only for the PROC object. This token has the token number ZTCI-TNM-IP-STATS and the DDL definition ZTCI-DDL-IP-STATS.

Each counter is defined in a separate field. The fields in the ZTCI-MAP-IP-STATS token are:

**ZSTATS-IP-TOTAL**

is the number of incoming packets received by the IP Layer of the TCP/IP subsystem during the sample period.

**ZSTATS-IP-BAD-XSUM**

is the number of packets with an invalid checksum value received by the IP Layer of the TCP/IP subsystem during the sample period. An invalid checksum is usually caused by a noisy link.

**ZSTATS-IP-TOO-SHORT**

is the number of packets, received by the IP Layer of the TCP/IP subsystem during the sample period, that contained less data than was specified in the header. This can be caused by noisy links, a protocol error by the sender of the packet, or a byte-swapping problem on the receiver.

**ZSTATS-IP-TOO-SMALL**

is the number of packets, received by the IP Layer of the TCP/IP subsystem during the sample period, that contained less data than expected when read into the local buffers. This error is rare; it usually indicates a problem with the local machine’s buffering scheme.

**ZSTATS-IP-BAD-HDR-LEN**

is the number of packets received by the IP Layer of the TCP/IP subsystem during the sample period with a header size that is larger than the header length provided in the packet. This error is rare; it indicates either a problem with the sender of the packet or a problem in reading the data from the link controller to the IP Layer.

**ZSTATS-IP-BAD-LEN**

is the number of packets received by the IP Layer of the TCP/IP subsystem during the sample period with a packet length that was shorter than expected. This error is very similar to ZSTATS-IP-BAD-HDR-LEN and is usually caused by similar conditions.
**Commands and Responses**

**STATISTICS Command**

**ZSTATS-IP-FRAG**

is the number of packet fragments received by the IP Layer of the TCP/IP subsystem during the sample period. Usually, a packet is fragmented when the size of the packet is too large for a particular gateway or network. This statistic might indicate that the sender’s maximum segment size is too large for the connection.

**ZSTATS-IP-FRAG-DROP**

is the number of packet fragments dropped by the IP Layer of the TCP/IP subsystem during the sample period. A fragment is dropped when memory cannot be allocated for the fragment or the fragment is a duplicate of a fragment already received.

**ZSTATS-IP-FRAG-TIMEOUT**

is the number of packet fragments, received by the IP Layer of the TCP/IP subsystem during the sample period, that timed out before the whole packet was received. This is usually caused by congestion, noisy links, or some event that prevents one of the fragments from being received with the rest of the fragments.

**ZSTATS-IP-FWD**

is the number of packets destined for another host that have been forwarded by the IP Layer of the TCP/IP subsystem during the sample period.

**ZSTATS-IP-CANT-FWD**

is the number of packets destined for another host that were received but not forwarded by the IP Layer of the TCP/IP subsystem during the sample period. The packets could not be forwarded because either the local host is not configured as a gateway or there is no route available to the specified destination.

**ZSTATS-IP-REDIR-SENT**

is the number of ICMP Redirect messages sent by the IP Layer of the TCP/IP subsystem during the sample period. Redirect messages are sent to the source host to indicate that there is a shorter path to the destination. The source host should either send the packet directly to the destination host or send the packet to another gateway.

**ZSTATS-IP-OUT-PKTS**

is the number of outgoing packets sent by the IP Layer of the TCP/IP subsystem during the sample period.
ZSTATS-IP-CURR-TIME

is the time at which the TCP/IP process read the IP Layer statistics counters. This time indicates the end of the sample period.

ZSTATS-IP-RESET-TIME

is the time at which the IP Layer statistics counters were last reset. This time indicates the beginning of the sample period.

ZTCI-MAP-ROUTE-STATS

is the extensible structured token that lists the routing statistics counters maintained for the IP Layer during the sample period. These statistics are returned only for the PROC object. This token has the token number ZTCI-TNM-ROUTE-STATS and the DDL definition ZTCI-DDL-ROUTE-STATS.

Each counter is defined in a separate field. The fields in the ZTCI-MAP-ROUTE-STATS token are:

ZSTATS-ROUTE-BAD-REDIR

is the number of Redirect messages received by the IP Layer of the TCP/IP subsystem during the sample period.

ZSTATS-ROUTE-DYN

is the number of dynamic route messages received by the IP Layer of the TCP/IP subsystem during the sample period. These messages indicate where the TCP/IP subsystem should route messages for a specific destination.

ZSTATS-ROUTE-NEW-GW

is the number of messages received by the IP Layer of the TCP/IP subsystem during the sample period that established a route for a new or unknown gateway.

ZSTATS-ROUTE-NOT-REACH

is the number of messages received by the IP Layer of the TCP/IP subsystem during the sample period that indicate that the specified destination is unreachable.

ZSTATS-ROUTE-WILD-CARD

is the number of wild-card matches found during the sample period.

ZSTATS-ROUTE-CURR-TIME

is the time at which the TCP/IP process read the routing statistics counters. This time indicates the end of the sample period.
**ZSTATS-ROUTE-RESET-TIME**

is the time at which the routing statistics counters were last reset. This time indicates the beginning of the sample period.

**ZTCI-MAP-ICMP-STATS**

is the extensible structured token that lists the statistics counters maintained for the ICMP Layer during the sample period. These statistics are returned only for the PROC object. This token has the token number ZTCI-TNM-ICMP-STATS and the DDL definition ZTCI-DDL-ICMP-STATS.

Each counter is defined in a separate field. The fields in the ZTCI-MAP-ICMP-STATS token are:

- **ZSTATS-ICMP-ERR**
  
is the number of times an ICMP error has been generated during the sample period. Note that Redirect messages are not included in the total. ICMP errors can be caused by any of these reasons: invalid IP options, problems in forwarding IP packets, or a UDP server crash.

- **ZSTATS-ICMP-OLD-SHORT**
  
is the number of packets, received by the ICMP Layer of the TCP/IP subsystem during the sample period, that were too short. This field is reserved for future use.

- **ZSTATS-ICMP-OLD-ICMP**
  
is the number of invalid ICMP packets received during the sample period.

- **ZSTATS-ICMP-OUT-HIST**
  
is the output histogram of ICMP messages recorded during the sample period. This histogram indicates the number of each type of ICMP message that the TCP/IP subsystem sent to the network. Statistics are maintained in this histogram for these ICMP messages:

  - **Type 0** is the Echo Reply message.
    
    This ICMP message is the reply to the Echo (8) message. Essentially, an Echo Reply message is just the original Echo message with the type changed from eight to zero and the destination and source addresses reversed. The data returned in the Echo Reply message is the same as that received in the Echo message. The transmission of an Echo Reply message informs the remote host that the local host is still alive. The data returned also gives the remote host a means of testing the integrity of the link.

  - **Type 3** is the Destination Unreachable message.
A Destination Unreachable message is sent to the source host by the TCP/IP subsystem (functioning as a host) when the destination host or port is unreachable. This message is sent by the TCP/IP subsystem (functioning as a gateway) when the destination network or host is unreachable, fragmentation is needed and the Don’t Fragment flag is set, or the source route failed.

Type 4 is the Source Quench message.

The TCP/IP subsystem sends this message to the source host to indicate that the TCP/IP subsystem is receiving datagrams more quickly than it can process them. When the source host receives this message, it should reduce the rate at which it is sending the datagrams.

Type 5 is the Redirect message.

The TCP/IP subsystem (functioning as a gateway) sends this message to the source host to indicate that there is a shorter path to the destination through another gateway.

Type 8 is the Echo message.

This ICMP message is sent from the source address to the destination address. An Echo Reply message containing the same data is expected from the destination address.

Type 11 is the Time Exceeded message.

The TCP/IP subsystem (functioning as a gateway) sends this message to notify the source host that the time-to-live field is zero and the gateway has discarded the datagram. The TCP/IP subsystem (functioning as a destination host) sends this message if it cannot reassemble a fragmented datagram within the time limit because fragments are missing. The TCP/IP subsystem discards the datagram.

Type 12 is the Parameter Problem message.

The TCP/IP subsystem (functioning as a host or gateway) sends this message to notify the source host that one of its datagrams has been discarded because the header parameters are incorrect.

Type 13 is the Timestamp message.

The TCP/IP subsystem (functioning as a host or gateway) sends this message to indicate the time it last handled the message before sending it.

Type 14 is the Timestamp Reply message.
The TCP/IP subsystem (functioning as a host or gateway) sends this message in reply to a Timestamp message. This message indicates the time in the original Timestamp message and the time at which the message was received by the destination. The Timestamp facility is used to obtain the network time. Special applications can be written to use this facility.

**Type 15** is the Information Request message.

The TCP/IP subsystem (functioning as a host or gateway) can send this message with the network portion of the source address and destination address set to zero, to determine the number of the network on which it is running. Any host on the local network can respond to this request with an Information Reply message.

**Type 16** is the Information Reply message.

The TCP/IP subsystem (functioning as a host or gateway) sends this message with the source and destination addresses fully specified in reply to an Information Request message. This message is returned by the IP module of the TCP/IP subsystem. Note that the Information Request/Reply facility, although supported, is rarely used.

This field is reserved for future use.

**ZSTATS-ICMP-BADCODE**

is the number of packets received by the ICMP Layer of the TCP/IP subsystem during the sample period that contain an invalid ICMP packet type code in the header. For a description of the packet type codes currently implemented, see the description of the ZSTATS-ICMP-IN-HIST field.

**ZSTATS-ICMP-TOO-SHORT**

is the number of packets, received by the ICMP Layer of the TCP/IP subsystem during the sample period, that were shorter than the minimum length allowed for an ICMP packet. Short packets are usually caused by a noisy link.

**ZSTATS-ICMP-BAD-XSUM**

is the number of packets with an invalid checksum received by the ICMP Layer of the TCP/IP subsystem during the sample period. An invalid checksum is usually caused by a noisy link.

**ZSTATS-ICMP-BAD-LEN**

is the number of packets received by the ICMP Layer of the TCP/IP subsystem during the sample period with a length that is shorter than the length field.
specified in the header. This error, usually caused by a noisy link, is rarely reported because the checksum routine also detects this problem.

**ZSTATS-ICMP-REFLECT**

is the number of ICMP packets received by the ICMP Layer of the TCP/IP subsystem during the sample period that have been sent a response. Note that not all ICMP packets require a response.

**ZSTATS-ICMP-IN-HIST**

is the input histogram of ICMP messages recorded during the sample period. This histogram indicates the number of each type of ICMP message that the TCP/IP subsystem received from the network. Statistics are maintained in this histogram for these ICMP messages:

Type 0  is the Echo Reply message.

This ICMP message is the reply to the Echo (8) message.

Essentially, an Echo Reply message is just the original Echo message with the type changed from eight to zero and the destination and source addresses reversed. The data returned in the Echo Reply message is the same as that sent in the Echo message.

The receipt of an Echo Reply message informs the local host that the remote host is still alive. The data returned also gives the local host a means of testing the integrity of the link.

Type 3  is the Destination Unreachable message.

A Destination Unreachable message is sent to the TCP/IP subsystem when another host or gateway determines that a destination host or port is unreachable. This message can be caused by these reasons: either there is no route to the destination or the route to the destination has gone down, a nonexistent address has been specified, the process listening on the port has gone down, the destination host has crashed, or fragmentation is needed but the Don’t Fragment flag is set.

Type 4  is the Source Quench message.
A gateway sends this message to the TCP/IP subsystem to indicate that the gateway is receiving datagrams more quickly than it can process them. When the TCP/IP subsystem receives this message, it reduces the rate at which it is sending the datagrams by implementing a slow start. To implement a slow start, the TCP/IP subsystem first stops sending any datagrams at all and then gradually increases the number of datagrams sent. Note that if the TCP/IP subsystem is doing a lot of retransmissions, you should check to see if Source Quench packets are being received. If they are, you should reduce the number of packets being transmitted by your applications.

Type 5 is the Redirect message.

A gateway sends this message to the TCP/IP subsystem (functioning as a source host) to indicate that there is a shorter path to the destination through another gateway. When the TCP/IP subsystem receives a Redirect message, it corrects its routing table to reflect the new route. If a host receives many Redirect messages in a short period of time, it is usually an indication that the host is not correcting its routing table.

Type 8 is the Echo message.

This ICMP message is sent from the source address to the destination address. An Echo Reply message containing the same data is expected from the destination address.

Type 11 is the Time Exceeded message.

A gateway sends this message to notify the TCP/IP subsystem (functioning as a source host) that the time-to-live field is zero and that the gateway discarded the datagram. A destination host sends this message if the host cannot reassemble a fragmented datagram within the time limit because fragments are missing. The destination host discards the datagram. When a Time Exceeded message is received, you should check for routing loops.

Type 12 is the Parameter Problem message.

A host or gateway sends this message to notify the TCP/IP subsystem (functioning as a source host) that one of its datagrams has been discarded because the header parameters are incorrect.

Type 13 is the Timestamp message.

A host or gateway sends this message to indicate the time it last handled the message before sending it.

Type 14 is the Timestamp Reply message.

A host or a gateway sends this message in reply to a Timestamp message.
This message indicates the time in the original Timestamp message and the time at which the Timestamp message was received by the destination. The Timestamp facility is used to obtain the network time. Special applications can be written to use this facility.

**Type 15** is the Information Request message.

A host or gateway can send this message with the network portion of the source address and the destination address set to zero to determine the number of the network on which it is running. Any host on the local network can respond to this request with an Information Reply message.

**Type 16** is the Information Reply message.

A host or gateway sends this message with the source and destination addresses fully specified, in reply to an Information Request message.

Note that the Information Request/Reply facility, although supported, is rarely used.

**ZSTATS-ICMP-CURR-TIME**

is the time at which the TCP/IP process read the ICMP Layer statistics counters. This time indicates the end of the sample period.

**ZSTATS-ICMP-RESET-TIME**

is the time at which the ICMP Layer statistics were last reset. This time indicates the beginning of the sample period.

**ZTCI-MAP-STATS-SUBNET**

is the extensible structured token that lists the statistics counters maintained for the subnet interface during the sample period. These statistics are returned only for the SUBNET object. This token has the token number ZTCI-TNM-STATS-SUBNET and the DDL definition ZTCI-DDL-STATS-SUBNET.

Each counter is defined in a separate field. The fields in the ZTCI-MAP-STATS-SUBNET token are:

**ZSTATS-SUBNET-OUT-PKTS**

is the number of packets sent by the subnet interface during the sample period.

**ZSTATS-SUBNET-IN-PKTS**

is the number of packets received by the subnet interface during the sample period.
ZSTATS-SUBNET-OUT-ERRS

is the number of errors that occurred while the subnet interface was sending packets during the sample period. Each output error also generates one of these event messages: ZTCI-EVT-IOP-READ-ERR, ZTCI-EVT-IOP-WRITE-ERR, or ZTCI-EVT-IOP-AWAITIO-ERR.

ZSTATS-SUBNET-IN-ERRS

is the number of errors that occurred while the subnet interface was receiving packets during the sample period. Each input error also generates one of these event messages: ZTCI-EVT-IOP-READ-ERR, ZTCI-EVT-IOP-WRITE-ERR, or ZTCI-EVT-IOP-AWAITIO-ERR.

ZSTATS-SUBNET-CURR-TIME

is the time at which the TCP/IP process read the subnet statistics counters. This time indicates the end of the sample period.

ZSTATS-SUBNET-RESET-TIME

is the time at which the subnet statistics counters were last reset. This time indicates the beginning of the sample period.

ZTCI-MAP-STATS-ROUTE

is the extensible structured token that lists the statistics counters maintained for the route during the sample period. These statistics are returned only for the ROUTE object. This token has the token number ZTCI-TNM-STATS-ROUTE and the DDL definition ZTCI-DDL-STATS-ROUTE.

The fields in the ZTCI-MAP-STATS-ROUTE token are:

ZUSAGE

is the number of times the specified route has been used to send IP datagrams.

ZSTATS-ROUTE-CURR-TIME

is the time at which the TCP/IP process read the route statistics counters. This time indicates the end of the sample period.

ZSTATS-ROUTE-RESET-TIME

is the time at which the route statistics counters were last reset. This time indicates the beginning of the sample period.

ZTCI-MAP-SMEM-STAT

is the extensible structured token that lists the statistics counters relating to the different layers of the TCP/IP process as well as QIO statistics. These statistics are
returned only for the PROCESS object. This token has the token number ZTCI-TNM-SMEM-STAT and the DDL definition ZTCI-DDL-SMEM-STAT.

ZSTATS-SMEM-DATA-MDS
is an integer that specifies the current number of data message descriptors (MDs) in use by the process.

ZSTATS-SMEM-MAX-DATA-MDS
is an integer that specifies the maximum number of data MDs that have been in use.

ZSTATS-SMEM-DUP-MDS
is an integer that specifies the number of duplicate MDs in use by the process that are not assigned to inbound driver MDs.

ZSTATS-SMEM-MAX-DUP-MDS
is an integer that specifies the maximum number of duplicate MDs that have been in use that are not assigned to inbound driver MDs.

ZSTATS-SMEM-NO-DATA-MDS
is an integer that specifies the number of times the process failed to get a data MD.

ZSTATS-SMEM-NO-DUP-MDS
is an integer that specifies the number of times the process failed to get a duplicate MD.

ZSTATS-SMEM-RES-RECOV-RUN
is an integer that specifies the number of times that the send or receive queue on a TCP session exceeded a predefined limit of MDs queued. The process attempts to decrease the number queued by collapsing the data into a smaller number of MDs.

ZSTATS-SMEM-QIO-LIMIT-WARNS
is an integer that specifies the number of times that the process received from the QIO monitor process an event signifying a pool or an MD shortage.

ZSTATS-SMEM-QIO-DRIVER-ERRS
is an integer that specifies the number of times QIO driver returned an error.

ZSTATS-SMEM-CURR-POOL-SPACE
is an integer that specifies the number of bytes of pool space in use.
ZSTATS-SMEM-MAX-POOL-SPACE
  is an integer that specifies the maximum pool space used.

ZSTATS-SMEM-POOL-FAILS
  is an integer that specifies the number of times a pool space request failed.

ZSTATS-SMEM-TOTAL-MBUFS
  is an integer that specifies the total number of memory buffers available.

ZSTATS-SMEM-CURR-MBUF-INUSE
  is an integer that specifies the current number of memory buffers in use.

ZSTATS-SMEM-MAX-MBUFS-INUSE
  is an integer that specifies the maximum number of memory buffers in use.

ZSTATS-SMEM-MBUF-FAILS
  is an integer that specifies the number of times a pool space request failed.

ZSTATS-SMEM-DUP-DRIV-MDS
  is an integer that specifies the number of duplicate driver MDs.

ZSTATS-SMEM-MAX-DUP-DRIV-MDS
  is an integer that specifies the maximum number of duplicate driver MDs.

ZSTATS-SMEM-NO-DUP-DRIV-MDS
  is an integer that specifies the number of times the process failed to obtain a
duplicate driver MD.

ZSTATS-SMEM-SIZE-128
  is an integer that specifies the count of socket sent between 1 and 128 bytes.

ZSTATS-SMEM-SIZE-256
  is an integer that specifies the count of socket sends between 129 and
256 bytes.

ZSTATS-SMEM-SIZE-512
  is an integer that specifies the count of socket sends between 257 and
512 bytes.

ZSTATS-SMEM-SIZE-1024
  is an integer that specifies the count of socket sends between 513 and
1024 bytes.
ZSTATS-SMEM-SIZE-2048
  is an integer that specifies the count of socket sends between 1025 and 2048 bytes.

ZSTATS-SMEM-SIZE-4096
  is an integer that specifies the count of socket sends between 2049 and 4096 bytes.

ZSTATS-SMEM-SIZE-8192
  is an integer that specifies the count of socket sends between 4097 and 8192 bytes.

ZSTATS-SMEM-SIZE-12288
  is an integer that specifies the count of socket sends between 8193 and 12288 bytes.

ZSTATS-SMEM-SIZE-16384
  is an integer that specifies the count of socket sends between 12289 and 16384 bytes.

ZSTATS-SMEM-SIZE-32768
  is an integer that specifies the count of socket sends between 16385 and 32768 bytes.

ZSTATS-SMEM-CURR-TIME
  is the time at which the TCP/IP process read the QIO statistics counters.

ZSTATS-SMEM-RESET-TIME
  is the time at which the QIO statistics counters were last reset.

ZCOM-TKN-OBJNAME
  specifies the name of a TCP/IP object for which statistics information was requested. Unless there is an error that prevents the STATISTICS command from being attempted at all, this token is present in every response record. ZCOM-TKN-OBJNAME is described in Data Communications Standard Definitions on page 5-9.

ZCOM-TKN-OBJTYPE
  specifies the type of object for which information was requested. For more details, see Data Communications Standard Definitions on page 5-9.
**ZSPI-TKN-RETCODE**

specifies whether the command has been completed successfully. The value of this token remains zero (ZCOM-ERR-OK) when the command is completed successfully with or without a warning. Note that if the empty response is returned, the value of this token is nonzero, although the command is completed successfully.

If the command is not completed successfully, the value of this token indicates the reason the command failed. Always check the contents of the error list to determine the token or value in error. For details on the ZCOM return codes, see the *SPI Common Extensions Manual*.

The possible nonzero values of RETCODE in response to the STATISTICS command are:

- ZCOM-ERR-TKN-REQ (-29)
- ZCOM-ERR-TKN-DUP (-27)
- ZCOM-ERR-TKN-CODE-INV (-26)
- ZCOM-ERR-SSID-INV (-24)
- ZCOM-ERR-SPI-ERR (-23)
- ZCOM-ERR-OBJ-NOT-SUPP (-18)
- ZCOM-ERR-OBJ-NOT-FOUND (-17)
- ZCOM-ERR-CMD-TOO-LARGE (-8)
- ZCOM-ERR-CMD-NOT-SUPP (-5)

**ZSPI-TKN-ERRLIST**

indicates the beginning of an error list. For more details, see *SPI Standard Definitions* on page 5-1.

**ZSPI-TKN-ENDLIST**

indicates the end of a data list or an error list. For more details, see *SPI Standard Definitions* on page 5-1.

**ZSPI-TKN-CONTEXT**

specifies whether the response continues with more response messages. If this token is present, the response continues. If this token is not present, this response message is the last of the response. For more details, see *SPI Standard Definitions* on page 5-1.
Operational Notes

- STATISTICS is a nonsensitive command; any valid user on the system can issue this command.
- The summary state of the object does not prevent the successful completion of the STATISTICS command.
- To initialize the statistics counters after the response message is built, use the RESETSTATS command.

STATUS Command

The STATUS command returns information about the current summary state and other status information about the specified object(s). When the TCP/IP subsystem finishes processing this command, the specified object remains in the same summary state. You can inquire about the status of a process, subnets, and routes.

When your management application issues the STATUS command, the TCP/IP subsystem checks for object-name templates. Object-name templates allow your management application to obtain status information about several SUBNET or ROUTE objects.

Any user on the system can issue the STATUS command. The summary state of the object does not prevent the command from being completed successfully.

<table>
<thead>
<tr>
<th>Command</th>
<th>ZCOM–CMD–STATUS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object Type</td>
<td>ZCOM–OBJ–PROC or ZCOM–OBJ–SUBNET or ZCOM–OBJ–ROUTE or ZCOM–OBJ–ADDRMAP</td>
</tr>
<tr>
<td>Tokens in Command Buffer</td>
<td>ZSPI–TKN–SSID token-type ZSPI–TYP–SSID. !{}</td>
</tr>
<tr>
<td></td>
<td>ZCOM–TKN–OBJNAME token-type ZSPI–TYP–STRING. !{}</td>
</tr>
<tr>
<td></td>
<td>ZSPI–TKN–MAXRESP token-type ZSPI–TYP–INT.</td>
</tr>
<tr>
<td></td>
<td>ZSPI–TKN–CONTEXT token-type ZSPI–TYP–BYTESTRING.</td>
</tr>
<tr>
<td></td>
<td>ZSPI–TKN–ALLOW–TYPE token-type ZSPI–TYP–ENUM.</td>
</tr>
<tr>
<td></td>
<td>ZSPI–TKN–COMMENT token-type ZSPI–TYP–STRING.</td>
</tr>
</tbody>
</table>
Tokens in Response Buffer

ZSPI-TKN-SSID  token-type ZSPI-TYP-SSID.  \!{}
ZSPI-TKN-SERVER-VERSION  token-type ZSPI-TYP-UINT.  \!{}
ZSPI-TKN-DATALIST  token-type ZSPI-TYP-LIST.  \!...

ZTCI-MAP-STATUS-PROC  \!{}
\!OBJ-PROC

\quad def ZTCI-DDL-STATUS-PROC.  \!All fields are present.
\quad 02 ZINET-PROC-STATE  type ZSP-DDL-ENUM.
\quad end.

ZTCI-MAP-STATUS-PROTO  \!{}
\!OBJ-PROC

\quad def ZTCI-DDL-STATUS-PROTO  \!All fields are present.
\quad 02 ZNET-STATUS-LOC-ADDR  type ZSPI-DDL-INT2.
\quad 02 ZNET-STATUS-FORGN-ADDR  type ZSPI-DDL-INT2.
\quad 02 ZNET-STATUS-LOC-PORT  type ZSPI-DDL-INT.
\quad 02 ZNET-STATUS-FORGN-PORT  type ZSPI-DDL-INT.
\quad 02 ZNET-STATUS-PROTO  type ZSPI-DDL-ENUM.
\quad 02 ZNET-STATUS-STATE  type ZSPI-DDL-ENUM.
\quad 02 ZNET-STATUS-SEND  type ZSPI-DDL-INT.
\quad 02 ZNET-STATUS-RECV  type ZSPI-DDL-INT.
\quad end.

ZTCI-MAP-STATUS-SUBNET  \!{}
\!OBJ-SUBNET

\quad def ZTCI-DDL-STATUS-SUBNET.  \!All fields are present.
\quad 02 ZSUBNET-STATE  type ZSPI-DDL-ENUM.
\quad end.

ZTCI-MAP-STATUS-ROUTE  \!{}
\!OBJ-ROUTE

\quad def ZTCI-DDL-STATUS-ROUTE.  \!All fields are present.
\quad 02 ZSTATE  type ZSPI-DDL-ENUM.
\quad 02 ZREF-CNT  type ZSPI-DDL-INT.
\quad end.
Tokens in Command Buffer

ZSPI-TKN-SSID

specifies the subsystem ID, which your management application must pass to SSINIT. This token is described in SPI Standard Definitions on page 5-1, and in the SPI Programming Manual.

ZCOM-TKN-OBJNAME

specifies the object for which information is requested. Your application must provide this token in the command. The object name can include wild-card characters. Using wild-card characters allows the command to affect multiple objects. This token is described in Data Communications Standard Definitions on page 5-9.

(ZSPI-TKN-) MAXRESP, CONTEXT, ALLOW-TYPE, and COMMENT

are standard SPI tokens that your application can provide. These tokens are described in SPI Standard Definitions on page 5-1.

Tokens in Response Buffer

ZSPI-TKN-SSID

is an echo of the subsystem ID value provided in the command by your application. For more details, see SPI Standard Definitions on page 5-1.

ZSPI-TKN-SERVER-VERSION

indicates the software release version of the TCP/IP subsystem. For more details, see SPI Standard Definitions on page 5-1.
ZSPI-TKN-DATALIST

indicates the beginning of a data list. The ZSPI-TKN-MAXRESP token must have a value other than 0 for data lists to be included in the response. For more details, see SPI Standard Definitions on page 5-1.

ZTCI-MAP-STATUS-PROC

is the extensible structured token that lists the summary state of the PROC object. This token has the token number ZTCI-TNM-STATUS-PROC and the DDL definition ZTCI-DDL-STATUS-PROC. The only field in this token is:

ZINET-PROC-STATE

is the current summary state of the PROC object. Its value is always ZCOM-VAL-SUMSTATE-STARTED.

ZTCI-MAP-STATUS-PROTO

is the extensible structured token that lists status information about each socket associated with the PROC object. This token has the token number ZTCI-TNM-STATUS-PROTO and the DDL definition ZTCI-DDL-STATUS-PROTO. There can be multiple ZTCI-MAP-STATUS-PROTO structures for a process. They are returned in separate data lists or response messages, depending on the value of ZSPI-TKN-MAXRESP. The fields in the ZTCI-MAP-STATUS-PROTO token are:

ZNET-STATUS-LOC-ADDR

is the local Internet address associated with the socket IP address.

ZNET-STATUS-FORGN-ADDR

is the foreign Internet address associated with the socket IP address.

ZNET-STATUS-LOC-PORT

is the local port number for the protocol used.

ZNET-STATUS-FORGN-PORT

is the foreign port number for the protocol used.

ZNET-STATUS-PROTO

is the protocol associated with the socket. The valid types are TCP or UDP.

ZNET-STATUS-STATE

is the state of the connection. This is related only to TCP. The possible state values (ZTCI-VAL-TCP-STATE-) are:
LISTEN
  indicates that the TCP Layer is waiting for a connection request from any remote TCP Layer.

SYNC-SENT
  indicates that the TCP Layer has sent a SYN packet and is waiting for a SYN-ACK response.

SYNC-RECV
  indicates that the TCP Layer has received a SYN packet, responded with a SYN-ACK, and is now waiting for an ACK.

STATE-ESTAB
  indicates that there is an open connection. The TCP Layer can deliver data to the user that it has received from the remote. This is the normal operational state for data to be transferred over the connection.

CLOSE-WAIT
  indicates that the TCP Layer is waiting for a connection termination request from the local user.

FIN-WAIT-1
  indicates that the TCP Layer is waiting for a connection termination request from the remote or is waiting for an acknowledgment of the connection termination request it has previously sent.

CLOSING
  indicates that the TCP Layer is waiting for an acknowledgment, from the remote, of a connection termination request.

LAST-ACK
  indicates that the TCP Layer is waiting for an acknowledgment of the connection termination request previously sent to the remote. The previously sent request included an acknowledgment of a connection termination request sent by the remote.

FIN-WAIT-2
  indicates that the TCP Layer is waiting for a connection termination request from the remote.
TIME-WAIT

    indicates that the connection has been closed on both sides, but that the
    TCP Layer continues to maintain the protocol control block for a short time
    to trap stray packets.

ZNET-STATUS-SEND

    is the number of bytes of data currently in the send queue for the socket.

ZNET-STATUS-RECV

    is the number of bytes of data currently in the receive queue for the socket.

ZTCI-MAP-STATUS-SUBNET

    is the extensible structured token that lists the summary state of the SUBNET
    object. This token has the token number ZTCI-TNM-STATUS-SUBNET and the
    DDL definition ZTCI-DDL-STATUS-SUBNET.

    The only field in this token is:

ZSUBNET-STATE

    is the current summary state of the SUBNET object. Its value can be either
    ZCOM-VAL-SUMSTATE-STARTED or ZCOM-VAL-SUMSTATE-STOPPED.

ZTCI-MAP-STATUS-ROUTE

    is the extensible structured token that lists the summary state of the ROUTE
    object. This token has the token number ZTCI-TNM-STATUS-ROUTE and the DDL
    definition ZTCI-DDL-STATUS-ROUTE.

    The fields in this token are:

ZSTATE

    is the current summary state of the ROUTE object. Its value can be either
    ZCOM-VAL-SUMSTATE-STARTED or ZCOM-VAL-SUMSTATE-STOPPED.

ZREF-CNT

    is the number of users of the route.

ZCOM-TKN-OBJNAME

    specifies the name of the TCP/IP object for which information was requested.
    Unless there is an error that prevents the STATUS command from being attempted
    at all, this token is present in every response record. If you have used an object-
    name template, a separate ZCOM-TKN-OBJNAME token is returned for each
    subnet or route that meets the template requirements. ZCOM-TKN-OBJNAME is
    described in Data Communications Standard Definitions on page 5-9.
**ZCOM-TKN-OBJTYPE**

specifies the type of object for which information was requested. The possible values are ZCOM-OBJ-PROC, ZCOM-OBJ-SUBNET, and ZCOM-OBJ-ROUTE. This token is present in every response record. The value of the ZCOM-TKN-OBJTYPE token is the same as the value of the ZSPI-TKN-OBJECT-TYPE token in the header. For more details, see *Data Communications Standard Definitions* on page 5-9.

**ZSPI-TKN-RETCODE**

specifies whether the command has been completed successfully. The value of this token remains zero (ZCOM-ERR-OK) when the command is completed successfully with or without a warning. Note that if the empty response (ZCOM-ERR-EMPT-RSP) is returned, the value of this token is nonzero, although the command is completed successfully.

If the command is not completed successfully, the value of this token indicates the reason the command failed. Always check the contents of the error list to determine the token or value in error. For details on the ZCOM return codes and associated error lists, refer to the *SPI Common Extensions Manual*.

The possible nonzero values of RETCODE in response to the STATUS command are:

- ZCOM-ERR-TKN-REQ (-29)
- ZCOM-ERR-TKN-DUP (-27)
- ZCOM-ERR-TKN-CODE-INV (-26)
- ZCOM-ERR-SSID-INV (-24)
- ZCOM-ERR-SPI-ERR (-23)
- ZCOM-ERR-OBJ-NOT-SUPP (-18)
- ZCOM-ERR-OBJ-NOT-FOUND (-17)
- ZCOM-ERR-CMD-TOO-LARGE (-8)
- ZCOM-ERR-CMD-NOT-SUPP (-5)

**ZSPI-TKN-ERRLIST**

indicates the beginning of an error list. For more details, see *SPI Standard Definitions* on page 5-1.

**ZSPI-TKN-ENDLIST**

indicates the end of a data list or an error list. For more details, see *SPI Standard Definitions* on page 5-1.
ZSPI-TKN-CONTEXT

specifies whether the response continues with more response messages. If this token is present, the response continues. If this token is not present, this response message is the last of the response. For more details, see SPI Standard Definitions on page 5-1.

**Operational Notes**

- STATUS is a nonsensitive command; any user on the system can issue this command.
- When the STATUS command is completed, the object remains in the same summary state.
- The summary state of the object does not prevent successful completion of the STATUS command.

**STOP Command**

The STOP command terminates the operation of the specified object. You can stop a subnet and a route, but not a process. When the subsystem finishes processing the command, the specified object is placed in the STOPPED summary state.

When you are using the STOP command, the object must be in the STARTED summary state, with no existing connections to it. If it is either in the STOPPED summary state or in the midst of an active connection, when you issue the STOP command, a warning is returned.

You can use an object-name template to stop multiple subnets or routes. For more details, see Object-Name Templates on page 3-6.

Once the object is stopped, processing for that object cannot continue until it is started again.

### Command

ZCOM-CMD-STOP

### Object Type

ZCOM-OBJ-SUBNET or ZCOM-OBJ-ROUTE

### Tokens in Command Buffer

<table>
<thead>
<tr>
<th>Token Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZSPI-TKN-SSID</td>
<td>token-type ZSPI-TYP-SSID. !{}</td>
</tr>
<tr>
<td>ZCOM-TKN-OBJNAME</td>
<td>token-type ZSPI-TYP-STRING. !{}</td>
</tr>
<tr>
<td>ZSPI-TKN-MAXRESP</td>
<td>token-type ZSPI-TYP-INT.</td>
</tr>
<tr>
<td>ZSPI-TKN-CONTEXT</td>
<td>token-type ZSPI-TYP-BYTESTRING.</td>
</tr>
<tr>
<td>ZSPI-TKN-ALLOW-TYPE</td>
<td>token-type ZSPI-TYP-ENUM.</td>
</tr>
<tr>
<td>ZSPI-TKN-COMMENT</td>
<td>token-type ZSPI-TYP-STRING.</td>
</tr>
</tbody>
</table>
**Tokens in Command Buffer**

**ZSPI-TKN-SSID**

specifies the subsystem ID, which your management application must pass to SSINIT. This token is described in [SPI Standard Definitions](#) on page 5-1, and in the [SPI Programming Manual](#).

**ZCOM-TKN-OBJNAME**

specifies the subnet or route to be stopped. Your application must provide this token in the command. The object name can include wild-card characters. Using wild-card characters allows the command to affect multiple objects. This token is described more fully in [Data Communications Standard Definitions](#) on page 5-9.

**(ZSPI-TKN- ) MAXRESP, CONTEXT, ALLOW-TYPE, and COMMENT**

are standard SPI tokens that your application can provide. These tokens are described in [SPI Standard Definitions](#) on page 5-1.

**Tokens in Response Buffer**

**ZSPI-TKN-SSID**

is an echo of the subsystem ID value provided in the command by your application. For more details, see [SPI Standard Definitions](#) on page 5-1.

**ZSPI-TKN-SERVER-VERSION**

indicates the software release version of the TCP/IP subsystem. For more details, see [SPI Standard Definitions](#) on page 5-1.

**ZSPI-TKN-DATALIST**

indicates the beginning of a data list. The ZSPI-TKN-MAXRESP token must have a value other than 0 for data lists to be included in the response. For more details, see [SPI Standard Definitions](#) on page 5-1.
ZCOM–TKN–OBJNAME

specifies the name of a TCP/IP subnet or route stopped by the command. Unless there is an error that prevents the STOP command from being attempted at all, this token is present in every response record. If you have used an object-name template, a separate ZCOM–TKN–OBJNAME token is returned for each subnet or route that meets the template requirements. ZCOM–TKN–OBJNAME is described in Data Communications Standard Definitions on page 5-9.

ZCOM–TKN–OBJTYPE

specifies the type of object stopped. The possible values are ZCOM-OBJ-SUBNET and ZCOM-OBJ-ROUTE. This token is present in every response record. The value of the ZCOM–TKN–OBJTYPE token is the same as the value of the ZSPI–TKN–OBJECT–TYPE token in the header. For more details, see Data Communications Standard Definitions on page 5-9.

ZSPI–TKN–RETCODE

specifies whether the command has been completed successfully. The value of this token remains zero (ZCOM-ERR-OK) when the command is completed successfully with or without a warning. Note that if the empty response (ZCOM-ERR-EMPT-RSP) is returned, the value of this token is nonzero, although the command is completed successfully.

If the command is not completed successfully, the value of this token indicates the reason the command failed. Always check the contents of the error list to determine the token or value in error. For details on the ZCOM return codes and associated error lists, refer to the SPI Common Extensions Manual. The possible nonzero values of RETCODE in response to the STOP command are:

ZCOM-ERR-TKN-REQ (-29)
ZCOM-ERR-TKN-DUP (-27)
ZCOM-ERR-TKN-CODE-INV (-26)
ZCOM-ERR-SSID-INV (-24)
ZCOM-ERR-SPI-ERR (-23)
ZCOM-ERR-SECUVIOL (-22)
ZCOM-ERR-OBJ-NOT-SUPP (-18)
ZCOM-ERR-OBJ-NOT-FOUND (-17)
ZCOM-ERR-OBJ-ALRDY-IN-SUMSTATE (-16)
ZCOM-ERR-CMD-TOO-LARGE (-8)
ZCOM-ERR-CMD-NOT-SUPP (-5)
ZSPI-TKN-ERRLIST
indicates the beginning of an error list. For more details, see SPI Standard Definitions on page 5-1.

ZSPI-TKN-ENDLIST
indicates the end of a data list or an error list. For more details, see SPI Standard Definitions on page 5-1.

ZSPI-TKN-CONTEXT
specifies whether there are more response messages to be received for a response. If this token is present, the response will continue to the next response message. If this token is not present, this response message is the last to be received. For more details, see SPI Standard Definitions on page 5-1.

Operational Notes
- STOP is a sensitive command.
- When the STOP command is completed, the object is placed in the STOPPED summary state.
- To initiate the operation of subnets and routes, use the START command.
- To stop a subnet or route that is in use, use the ABORT command.
- You cannot stop a process.

TRACE Command
The TRACE command allows your management application to capture and store records that can be displayed using the PTRACE command. The TCP/IP subsystem does not support the TRACE command directly. Instead, your management application issues the command to SCP, which then invokes the SCP trace facility.

When your management application issues the TRACE command, the specified object enters trace mode; although the subsystem still functions normally, its operation might be slower. The trace parameters specified by your management application in ZCOM-MAP-TRACE-MODIF control the initialization of the trace file, the selection of the trace environment, and the termination of trace mode. This extensible structured token, along with one or more ZCOM-TKN-TRACE-OPT tokens, is sent to SCP.

The ZCOM-TKN-TRACE-OPT token(s) contains the trace option selected by the management application. There are several trace options available. From these trace options, SCP creates a bit mask that is sent to the TCP/IP subsystem. The TCP/IP
subsystem uses this bit mask to determine the records that have been selected by the
management application. The bit mask is 32 bits long.

**Note.** The trace operation can significantly increase CPU usage by the TCP/IP process. To
avoid problems with other processes in the processor, it is recommended that you lower the
priority of the TCP/IP process before you issue the TRACE command.

When the trace is to be terminated, the management application sends another
TRACE command where the ZSTOP field of the ZCOM-MAP-TRACE-MODIF
extensible structured token contains the value ZSPI-VAL-TRUE.

Object-name templates do not apply to this command.

---

### Command

ZCOM–CMD–TRACE

### Object Type

ZCOM–OBJ–PROC or ZCOM–OBJ–SUBNET

### Tokens in Command Buffer

- **ZSPI-TKN–SSID**: token-type ZSPI-TYP–SSID.
- **ZCOM–MAP–TRACE–MODIF**: token-type ZSPI-TYP–SSCTL.
  - def ZCOM–DDL–TRACE–MODIF !All fields are required.
    - 02 ZSTOP type ZSPI–DDL–BOOLEAN.
    - 02 ZPRIM type ZSPI–DDL–BOOLEAN.
    - 02 ZREC–SIZE type ZSPI–DDL–INT.
    - 02 ZCNT type ZSPI–DDL–INT2.
    - 02 ZXDS–PAGES type ZSPI–DDL–INT.
    - 02 ZCOLL type ZSPI–DDL–BOOLEAN.
    - 02 ZFNAME type ZSPI–DDL–FNAME32.
    - 02 ZWRAP type ZSPI–DDL–BOOLEAN.
  - end.

- **ZCOM–TKN–TRACE–OPT**: token-type ZSPI–TYP–ENUM.
- **ZCOM–TKN–OBJNAME**: token-type ZSPI–TYP–STRING.
- **ZSPI–TKN–MAXRESP**: token-type ZSPI–TYP–INT.
- **ZSPI–TKN–ALLOW–TYPE**: token-type ZSPI–TYP–ENUM.
- **ZSPI–TKN–COMMENT**: token-type ZSPI–TYP–STRING.

### Tokens in Response Buffer

- **ZSPI–TKN–SSID**: token-type ZSPI–TYP–SSID.
- **ZSPI–TKN–SERVER–VERSION**: token-type ZSPI–TYP–UINT.
- **ZSPI–TKN–DATALIST**: token-type ZSPI–TYP–LIST.
  - ZCOM–TKN–OBJNAME token-type ZSPI–TYP–STRING.
  - ZCOM–TKN–OBJTYPE token-type ZSPI–TYP–ENUM.
  - ZSPI–TKN–RETCODE token-type ZSPI–TYP–ENUM.
  - ZSPI–TKN–ERRLIST token-type ZSPI–TYP–LIST.
  ... ZSPI–TKN–ENDLIST token-type ZSPI–TYP–SSCTL.
- **ZSPI–TKN–ENDLIST**: token-type ZSPI–TYP–SSCTL.
Tokens in Command Buffer

ZSPI-TKN-SSID

specifies the subsystem ID, which your management application must pass to SSINIT. This token is described in SPI Standard Definitions on page 5-1 and in the SPI Programming Manual.

ZCOM-MAP-TRACE-MODIF

is a standard data communications token described in the SPI Common Extensions Manual. This token is required in the command, and duplicate occurrences are not allowed.

ZCOM-TKN-TRACE-OPT

specifies the trace option(s) that your management application has selected to control the trace file and records. Duplicate occurrences of ZCOM-TKN-TRACE-OPT are allowed. You can select more than one option. There are no options available for the subnet object. The possible trace values (ZTCI-VAL-TRACE-) for the process object are:

IP-IN

traces IP Layer input.

IP-OUT

traces IP Layer output.

IPC-CMDS

traces interprocess requests and replies.

MBUF-ALLOC

traces MBUF pool gets and puts.

ROUTE

traces routing routines.

SOCKET-CMDS

traces socket requests.

SOCKET-CREATE

traces socket creation.

TCP

traces TCP Layer.
UDP-DETL-IN
traces UDP Layer input detail.

UDP-DETL-OUT
traces UDP Layer output detail.

UDP-IN
traces UDP Layer input.

UDP-OUT
traces UDP Layer output.

UDP-USER-REQ
traces UDP user requests.

ZCOM-TKN-OBJNAME
specifies the process or subnet to be traced. Your application must provide this token in the command. The object name cannot be an object-name template. This token is described in Data Communications Standard Definitions on page 5-9.

(ZSPI-TKN-) MAXRESP, ALLOW-TYPE, and COMMENT
are standard SPI tokens that your application can provide. These tokens are described in SPI Standard Definitions on page 5-1.

Tokens in Response Buffer

ZSPI-TKN-SSID
is an echo of the subsystem ID value provided in the command by your application. For more details, see SPI Standard Definitions on page 5-1.

ZSPI-TKN-SERVER-VERSION
indicates the software release version of the TCP/IP subsystem. For more details, see SPI Standard Definitions on page 5-1.

ZSPI-TKN-DATALIST
indicates the beginning of a data list. The ZSPI-TKN-MAXRESP token must have a value other than zero for data lists to be included in the response. For more details, see SPI Standard Definitions on page 5-1.

ZCOM-TKN-OBJNAME
specifies the name of the TCP/IP process or subnet for which a trace was started or stopped. Unless there is an error that prevents the TRACE command from being
attempted at all, this token is present in every response record.
ZCOM-TKN-OBJNAME is described in Data Communications Standard Definitions
on page 5-9.

ZCOM-TKN-OBJTYPE
specifies the type of object for which a trace was started or stopped. This token
has the value ZCOM-OBJ-PROC or ZCOM-OBJ-SUBNET. For more details, see
Data Communications Standard Definitions on page 5-9.

ZSPI-TKN-RETCODE
specifies whether the command has been completed successfully. The value of this
token remains zero (ZCOM-ERR-OK) when the command is completed
successfully with or without a warning.

If the command is not completed successfully, the value of this token indicates the
reason the command failed. For details on the ZCOM return codes, see the SPI

The possible nonzero values of RETCODE in response to the TRACE command are:

ZCOM-ERR-TRACE-START-PEND (-849)
ZCOM-ERR-TRACE-COLL-INTL-ERR (-848)
ZCOM-ERR-TRACE-DEVICEINFO-INV (-847)
ZCOM-ERR-TRACE-SCP-INTL-ERR (-846)
ZCOM-ERR-TRACE-FCODE-INV (-845)
ZCOM-ERR-TRACE-DEV-NOT-FOUND (-841)
ZCOM-ERR-TRACE-DEVNAME-INV (-840)
ZCOM-ERR-TRACE-SEG-LOCK (-836)
ZCOM-ERR-TRACE-SEG-UNAVAIL (-835)
ZCOM-ERR-TRACE-SEG-ALLOC (-834)
ZCOM-ERR-TRACE-WORK-LOCK (-831)
ZCOM-ERR-TRACE-NEWPROCESS (-830)
ZCOM-ERR-TRACE-IN-PROGRESS (-822)
ZCOM-ERR-TRACE-SEG-OVERFLOW (-819)
ZCOM-ERR-TRACE-NOT-ACTV (-814)
ZCOM-ERR-TRACE-ATTR-INV (-812)
ZCOM-ERR-TRACE-LCB-INV (-810)
ZCOM-ERR-TRACE-ATTR-INV (-801)
ZCOM-ERR-TKN-REQ (-29)
ZCOM-ERR-TKN-DUP (-27)
ZCOM-ERR-TKN-CODE-INV (-26)
ZCOM-TKN-SSID-INV (-24)
ZCOM-ERR-SPI-ERR (-23)
ZCOM-ERR-SECUR-VIOL (-22)
ZCOM-ERR-OBJ-NOT-SUPP (-18)
ZCOM-ERR-OBJ-NOT-FOUND (-17)
ZCOM-ERR-CMD-TOO-LARGE (-8)
ZCOM-ERR-CMD-SEQ-INV (-6)
ZCOM-ERR-CMD-NOT-SUPP (-5)

ZSPI-TKN-ERRLIST

indicates the beginning of an error list. For more details, see SPI Standard Definitions on page 5-1.

ZSPI-TKN-ENDLIST

indicates the end of a data list or an error list. For more details, see SPI Standard Definitions on page 5-1.

Operational Notes

• TRACE is a sensitive command.

• When the TRACE command is completed, the process or subnet remains in the STARTED summary state.

• The TRACE command is processed by the SCP trace facility.

• For more detailed information on the specific trace records displayed by each of the trace options, including how these records are displayed with the PTrace program, see the TCP/IP Configuration and Management Manual.
Event Management

The TCP/IP subsystem reports information about events it detects by issuing event messages. A management application can obtain these event messages by opening an Event Management Service (EMS) distributor process and requesting the messages. The application can also specify that a filter be applied to select certain event messages. This section describes the event messages that the TCP/IP subsystem can issue and the specific programming considerations for dealing with these event messages in an application. For general information on how an application obtains event messages from a subsystem, see the *EMS Manual*.

In this section, event-message tokens and their values are represented in data definition language (DDL). For a brief explanation of DDL as it applies to SPI, see the *SPI Programming Manual*.

### Subsystem Processes That Report Events

Event messages are generated by the TCP/IP process. All event messages issued by the TCP/IP subsystem contain the subsystem ID, ZTCI-VAL-SSID, which is described in *SPI Standard Definitions* on page 5-1. For more information about subsystem IDs, see the *SPI Programming Manual*.

Table 7-1 lists the event messages that the TCP/IP subsystem can issue.

<table>
<thead>
<tr>
<th>Event Number</th>
<th>Symbolic Name (ZTCI-EVT- )</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>NO-MEM</td>
<td>Memory allocation failed</td>
</tr>
<tr>
<td>2</td>
<td>OP-READ-ERR</td>
<td>READ operation failed</td>
</tr>
<tr>
<td>3</td>
<td>IOP-WRITE-ERR</td>
<td>WRITE operation failed</td>
</tr>
<tr>
<td>4</td>
<td>IOP-AWAITIO-ERR</td>
<td>An error while waiting for I/O operation completion</td>
</tr>
<tr>
<td>5</td>
<td>SUMSTATE-CHG</td>
<td>Summary state change</td>
</tr>
<tr>
<td>6</td>
<td>INTL</td>
<td>Internal error (fatal)</td>
</tr>
<tr>
<td>7</td>
<td>DUP-IP-ADDR</td>
<td>Duplicate IP address</td>
</tr>
<tr>
<td>8</td>
<td>BCAST-ENET-ADDR</td>
<td>Broadcast address</td>
</tr>
<tr>
<td>9</td>
<td>MAX-DEV</td>
<td>Maximum number of X.25 devices opened</td>
</tr>
<tr>
<td>10</td>
<td>CONN-ESTAB</td>
<td>X.25 connection established</td>
</tr>
<tr>
<td>11</td>
<td>CONN-TIMEOUT</td>
<td>X.25 connection timed out</td>
</tr>
<tr>
<td>12</td>
<td>DEV-NOT-AVAIL</td>
<td>No X.25 device available</td>
</tr>
<tr>
<td>13</td>
<td>SYN-ERR</td>
<td>Syntax error in IP-to-X.25 address file</td>
</tr>
<tr>
<td>16</td>
<td>CONF-FILE-ACC-ERR</td>
<td>Configuration file access error</td>
</tr>
<tr>
<td>17</td>
<td>CONF-FILE-SYN-ERR</td>
<td>Configuration file syntax error</td>
</tr>
</tbody>
</table>
Critical Events

Events reported by the TCP/IP subsystem are divided into two classes: critical events and noncritical events. Critical events are errors and warnings for which the consequences could be serious, such as a hardware failure or a software error from which there is no automatic recovery. Noncritical events are often expected events that are informational, such as a summary state change.

To determine whether or not an event message reports a critical event, examine the value of the event-message token ZEMS-TKN-EMPHASIS. If the value is ZSPI-VAL-TRUE, the event message reports a critical event. If the value is ZSPI-VAL-FALSE, the event is not critical. In addition, check the value of the token
ZTCI-TKN-SEV. If the value is ZTCI-VAL-SEV-FATAL, the TCP/IP process is no longer operational. If the value is ZTCI-VAL-SEV-WARN, it is only a warning message. If the value is ZTCI-VAL-SEV-INFO, it is an informational message only. The ZTCI-VAL-SEV-INFO value is present if the ZEMS-TKN-EMPHASIS token has a value of ZSPI-VAL-FALSE.

Event-Management Programming

TCP/IP event messages are made up of individual tokens, each containing one piece of information about the event. There are no extensible structured tokens or lists in these event messages.

All TCP/IP event messages are reported to the EMS collector ($0).

The TCP/IP subsystem does not report action events. Action events prompt the operator for intervention, such as the need to mount a tape. For more information about action events, see the EMS Manual.

EMS provides you with the capability to create programs called filters, which allow applications to select particular event messages from among all the event messages issued. Filters select event messages to be returned to an application by examining the values of tokens in the event message.

For example, to select only event messages issued by the TCP/IP subsystem, a filter would examine the token that contains the subsystem ID of the issuing subsystem and pass through only those messages containing the TCP/IP subsystem ID.

Any of the tokens in an event message can be used for selecting the event messages to be returned to an application. You can create filters that return only critical event messages, all event messages with a certain event number, and so on.

For more information about filters, see the EMS Manual.

Event-Message Descriptions

This section describes each event message that can be issued by the subsystem. The descriptions are given in ascending order by event number; that is, in ascending order by the ZTCI-EVT- values.

Within each event-message description, there is a box that lists all the tokens that can appear in the event message. Header tokens are listed here because an application can filter event messages by header tokens, such as ZSPI-TKN-SSID, as well as by other tokens. The order in which tokens are presented in the token sections of the syntax box is arbitrary, except that the token ZEMS-TKN-SUBJECT-MARK always immediately precedes the subject token of an event message. For more information on the subject token, see the EMS Manual.

The syntax box also contains a portion of the text version of the event message. The text version of an event message is available through the EMSTEXT procedure, and it contains a standard header that includes the date, time, system name, and other
When you use the EMSTEXT procedure to get the text version of an event message, you request either the display-format version or the console-compatible-format version, as described in the *EMS Manual*. In the case of TCP/IP event messages, there is no difference between the two versions (except possibly the standard header).

The notation used in the syntax box for simple tokens is a shorthand version of the essential information given in the DDL TOKEN-CODE statement.

Following the box, the event-message tokens are described in the same order in which they appear in the box. However, because HP internal tokens are intended for HP use only, they are not described.

### 1: ZTCI-EVT-NO-MEM

The internal memory allocation of memory buffers (MBUFs) failed. An attempt was made to allocate memory to MBUFs, but the size specified was larger than the amount available. This could cause data transfer to be slower or to fail when memory cannot be allocated for data structures.

This event is critical. It is reported to the EMS collector ($0)$.

#### Unconditional Tokens

<table>
<thead>
<tr>
<th>Token</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZSPI-TKN-SSID</td>
<td>ZSPI-TYP-SSID</td>
</tr>
<tr>
<td>ZEMS-TKN-EVENTNUMBER</td>
<td>ZSPI-TYP-ENUM</td>
</tr>
<tr>
<td>ZEMS-TKN-SUBJECT-MARK</td>
<td>ZSPI-TYP-MARK</td>
</tr>
<tr>
<td>ZCOM-TKN-SUBJ-PROC</td>
<td>ZSPI-TYP-STRING</td>
</tr>
<tr>
<td>ZEMS-TKN-EMPHASIS</td>
<td>ZSPI-TYP-BOOLEAN</td>
</tr>
<tr>
<td>ZEMS-TKN-CONSOLE-PRINT</td>
<td>ZSPI-TYP-BOOLEAN</td>
</tr>
<tr>
<td>ZEMS-TKN-CPU</td>
<td>ZSPI-TYP-UINT</td>
</tr>
<tr>
<td>ZEMS-TKN-CRTPID</td>
<td>ZSPI-TYP-CRTPID</td>
</tr>
<tr>
<td>ZEMS-TKN-GENTIME</td>
<td>ZSPI-TYP-TIMESTAMP</td>
</tr>
<tr>
<td>ZEMS-TKN-LOGTIME</td>
<td>ZSPI-TYP-TIMESTAMP</td>
</tr>
<tr>
<td>ZEMS-TKN-PIN</td>
<td>ZSPI-TYP-UINT</td>
</tr>
<tr>
<td>ZEMS-TKN-SYSTEM</td>
<td>ZSPI-TYP-UINT</td>
</tr>
<tr>
<td>ZEMS-TKN-USERID</td>
<td>ZSPI-TYP-BYTE-PAIR</td>
</tr>
<tr>
<td>ZTCI-TKN-SEV</td>
<td>ZSPI-TYP-ENUM</td>
</tr>
<tr>
<td>ZTCI-TKN-SIZE</td>
<td>ZSPI-TYP-INT</td>
</tr>
</tbody>
</table>

#### Event-Message Text

```
CANT'T MALLOC size BYTES FOR MBUFS
```

### Unconditional Tokens

**ZSPI-TKN-SSID**

is a header token that specifies the subsystem ID. The value used for the TCP/IP subsystem is ZTCI-VAL-SSID.
ZEMS-TKN-EVENTNUMBER

is a header token that contains the event number. Its value is ZTCI-EVT-NO-MEM (1).

ZEMS-TKN-SUBJECT-MARK

is the standard EMS token that always immediately precedes the subject token here, ZCOM-TKN-SUBJ-PROC.

ZCOM-TKN-SUBJ-PROC

contains the name of the TCP/IP process that is the subject of this event message.

ZEMS-TKN-EMPHASIS

is a header token that contains the value ZSPI-VAL-TRUE, indicating that this is a critical event.

(ZEMS-TKN-) CONSOLE-PRINT, CPU, CRTPID, GENTIME, LOGTIME, PIN, SYSTEM, and USERID

are automatically placed in the header when the EMSINIT procedure is called and the event message buffer is initialized. For more information on these tokens, see Section 5, Common Definitions. For details on any of the EMS tokens, see the EMS Manual.

ZTCI-TKN-SEV

contains the value ZTCI-VAL-SEV-WARN. This value indicates that ZTCI-EVT-NO-MEM is only a warning message. The process continues to function, but some functionality has been lost; for example, the network may be going down, or a device accessed by the TCP/IP subsystem may be experiencing errors.

ZTCI-TKN-SIZE

contains the number of bytes of memory that the TCP/IP subsystem has tried to allocate to MBUFs.

**Event-Message Text**

In the text message, size corresponds to the value of ZTCI-TKN-SIZE.

**Cause.** The internal allocation of memory to MBUFs has failed.

**Recovery.** If this is a recurring problem, use the BINDER program and rebind the TCP/IP subsystem with a larger heap space. Otherwise, limit the data traffic and reduce the data request size.
2: ZTCI-EVT-IOP-READ-ERR

A READ operation to the X.25 (X25AM) I/O process failed. This can cause the I/O process to be closed if too many or certain critical failures occur. If this event results in the I/O process being closed, the TCP/IP process attempts to reopen the I/O process at one-minute intervals.

Note. This event is only valid for the X25AM subsystem. It does not apply to the SLSA subsystem.

This event is critical. It is reported to the EMS collector ($0).

Unconditional Tokens

<table>
<thead>
<tr>
<th>Token</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZSPI-TKN-SSID</td>
<td>ZSPI-TYP-SSID</td>
</tr>
<tr>
<td>ZEMS-TKN-EVENTNUMBER</td>
<td>ZSPI-TYP-ENUM</td>
</tr>
<tr>
<td>ZEMS-TKN-SUBJECT-MARK</td>
<td>ZSPI-TYP-MARK</td>
</tr>
<tr>
<td>ZCOM-TKN-SUBJ-SUBNET</td>
<td>ZSPI-TYP-STRING</td>
</tr>
<tr>
<td>ZEMS-TKN-EMPHASIS</td>
<td>ZSPI-TYP-BOOLEAN</td>
</tr>
<tr>
<td>ZEMS-TKN-CONSOLE-PRINT</td>
<td>ZSPI-TYP-BOOLEAN</td>
</tr>
<tr>
<td>ZEMS-TKN-CPU</td>
<td>ZSPI-TYP-UINT</td>
</tr>
<tr>
<td>ZEMS-TKN-CRTPID</td>
<td>ZSPI-TYP-CRTPID</td>
</tr>
<tr>
<td>ZEMS-TKN-GENTIME</td>
<td>ZSPI-TYP-TIMESTAMP</td>
</tr>
<tr>
<td>ZEMS-TKN-LOGTIME</td>
<td>ZSPI-TYP-TIMESTAMP</td>
</tr>
<tr>
<td>ZEMS-TKN-PIN</td>
<td>ZSPI-TYP-UINT</td>
</tr>
<tr>
<td>ZEMS-TKN-SYSTEM</td>
<td>ZSPI-TYP-UINT</td>
</tr>
<tr>
<td>ZEMS-TKN-USERID</td>
<td>ZSPI-TYP-BYTE-PAIR</td>
</tr>
<tr>
<td>ZTCI-TKN-SEV</td>
<td>ZSPI-TYP-ENUM</td>
</tr>
<tr>
<td>ZTCI-TKN-ERR-NUM</td>
<td>ZSPI-TYP-INT</td>
</tr>
<tr>
<td>ZTCI-TKN-IOPNAME</td>
<td>ZSPI-TYP-STRING</td>
</tr>
</tbody>
</table>

Event-Message Text

DEVICE READ ERROR error ON IOP iopname

Unconditional Tokens

ZSPI-TKN-SSID

is a header token that specifies the subsystem ID. The value used for the TCP/IP subsystem is ZTCI-VAL-SSID.

ZEMS-TKN-EVENTNUMBER

is a header token that contains the event number. Its value is ZTCI-EVT-IOP-READ-ERR (2).

ZEMS-TKN-SUBJECT-MARK

is the standard EMS token that always immediately precedes the subject token here, ZCOM-TKN-SUBJ-SUBNET.
ZCOM-TKN-SUBJ-SUBNET

contains the name of the TCP/IP subnet that is the subject of this event message.

ZEMS-TKN-EMPHASIS

is a header token that contains the value ZSPI-VAL-TRUE, indicating that this is a critical event.

(ZEMS-TKN-) CONSOLE-PRINT, CPU, CRTPID, GENTIME, LOGTIME, PIN, SYSTEM, and USERID

are automatically placed in the header when the EMSINIT procedure is called and the event message buffer is initialized. For a description of these tokens, see Section 5, Common Definitions. For details on any of the EMS tokens, see the EMS Manual.

ZTCI-TKN-SEV

contains the value ZTCI-VAL-SEV-WARN. This value indicates that ZTCI-EVT-IOP-READ-ERR is only a warning message. The process continues to function, but some functionality has been lost; for example, the network may be going down, or a device accessed by the TCP/IP subsystem may be experiencing errors.

ZTCI-TKN-ERR-NUM

contains the number of the file-system error that has occurred. For more details, see the Guardian Procedure Errors and Messages Manual.

ZTCI-TKN-IOPNAME

contains the name of the X25AM line and subdevice being accessed by the TCP/IP process for transmitting and receiving data when the READ operation failed.

Event-Message Text

In the text message, error corresponds to the value in ZTCI-TKN-ERR-NUM and iopname corresponds to the value in ZTCI-TKN-IOPNAME.

Cause. A READ operation to the I/O process has failed.

Recovery. The TCP/IP process automatically attempts to reopen the I/O process when it is closed as a result of an error. If successful, operator intervention is not required. However, if the attempt to reopen fails, the operator must check the X25AM line (for X.25 subnets) to verify that the problem is not at the subdevice or line level.

The TCP/IP process continues to try to open the port or subdevice at one-minute intervals.
3: ZTCI-EVT-IOP-WRITE-ERR

A WRITE operation to the X.25 (X25AM) I/O process failed. This can cause the I/O process to be closed if too many or certain critical failures occur. If this event results in the I/O process being closed, the TCP/IP process attempts to reopen the I/O process at one-minute intervals.

**Note.** This event is only valid for the X25AM subsystem. It does not apply to the SLSA subsystem.

This event is critical. It is reported to the EMS collector ($0).

### Unconditional Tokens

<table>
<thead>
<tr>
<th>Token Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZSPI-TKN-SSID</td>
<td>is a header token that specifies the subsystem ID. The value used for the TCP/IP subsystem is ZTCI-VAL-SSID.</td>
</tr>
<tr>
<td>ZEMS-TKN-EVENTNUMBER</td>
<td>is a header token that contains the event number. Its value is ZTCI-EVT-IOP-WRITE-ERR (3).</td>
</tr>
<tr>
<td>ZEMS-TKN-SUBJECT-MARK</td>
<td>is the standard EMS token that always immediately precedes the subject token here, ZCOM-TKN-SUBJ-SUBNET.</td>
</tr>
<tr>
<td>ZCOM-TKN-SUBJ-SUBNET</td>
<td></td>
</tr>
</tbody>
</table>

### Event-Message Text

DEVICE WRITE ERROR error ON IOP iopname
ZCOM-TKN-SUBJ-SUBNET

contains the name of the TCP/IP subnet that is the subject of this event message.

ZEMS-TKN-EMPHASIS

is a header token that contains the value ZSPI-VAL-TRUE, indicating that this is a critical event.

(ZEMS-TKN-) CONSOLE-PRINT, CPU, CRTPID, GENTIME, LOGTIME, PIN, SYSTEM, and USERID

are automatically placed in the header when the EMSINIT procedure is called and the event message buffer is initialized. For a description of these tokens, see Section 5, Common Definitions. For details on any of the EMS tokens, see the EMS Manual.

ZTCI-TKN-SEV

contains the value ZTCI-VAL-SEV-WARN. This value indicates that ZTCI-EVT-IOP-WRITE-ERR is only a warning message. The process continues to function, but some functionality has been lost; for example, the network may be going down, or a device accessed by the TCP/IP subsystem may be experiencing errors.

ZTCI-TKN-ERR-NUM

contains the number of the file-system error that has occurred. For more details, see the Guardian Procedure Errors and Messages Manual.

ZTCI-TKN-IOPNAME

contains the name of the X25AM line and subdevice being accessed by the TCP/IP process for transmitting and receiving data when the WRITE operation failed.

**Event-Message Text**

In the text message, *error* corresponds to the value in ZTCI-TKN-ERR-NUM and *iopname* corresponds to the value in ZTCI-TKN-IOPNAME.

**Cause.** A WRITE operation to the I/O process has failed.

**Recovery.** The TCP/IP process automatically attempts to reopen the I/O process when the I/O process is closed as a result of an error. If successful, operator intervention is not required. However, if the attempt to reopen fails, the operator must check the X25AM line (for X.25 subnets) to verify that the problem is not at the subdevice or line level.

The TCP/IP process continues to try to reopen the port or subdevice at one-minute intervals.
4: ZTCI-EVT-IOP-AWAITIO-ERR

The TCP/IP process called the AWAITIO procedure during communication with the X.25 (X25AM) I/O process but an error occurred before the operation completed. This can cause the I/O process to be closed if too many or certain critical failures occur. If this event results in the I/O process being closed, the TCP/IP process attempts to reopen the I/O process at one-minute intervals.

**Note.** This event is only valid for the X25AM subsystem. It does not apply to the SLSA subsystem.

This event is critical. It is reported to the EMS collector ($0).

For more information on nowait I/O, see the *Guardian Programmer’s Guide*.

### Unconditional Tokens

<table>
<thead>
<tr>
<th>Token ID</th>
<th>Token Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZSPI-TKN-SSID</td>
<td>ZSPI-TYP-SSID.</td>
</tr>
<tr>
<td>ZEMS-TKN-EVENTNUMBER</td>
<td>ZSPI-TYP-ENUM.</td>
</tr>
<tr>
<td>ZEMS-TKN-SUBJECT-MARK</td>
<td>ZSPI-TYP-MARK.</td>
</tr>
<tr>
<td>ZCOM-TKN-SUBJ-SUBNET</td>
<td>ZSPI-TYP-STRING.</td>
</tr>
<tr>
<td>ZEMS-TKN-EMPHASIS</td>
<td>ZSPI-TYP-BOOLEAN.</td>
</tr>
<tr>
<td>ZEMS-TKN-CONSOLE-PRINT</td>
<td>ZSPI-TYP-BOOLEAN.</td>
</tr>
<tr>
<td>ZEMS-TKN-CPU</td>
<td>ZSPI-TYP-UINT.</td>
</tr>
<tr>
<td>ZEMS-TKN-CRTPID</td>
<td>ZSPI-TYP-CRTPID.</td>
</tr>
<tr>
<td>ZEMS-TKN-GENTIME</td>
<td>ZSPI-TYP-TIMESTAMP.</td>
</tr>
<tr>
<td>ZEMS-TKN-LOGTIME</td>
<td>ZSPI-TYP-TIMESTAMP.</td>
</tr>
<tr>
<td>ZEMS-TKN-PIN</td>
<td>ZSPI-TYP-UINT.</td>
</tr>
<tr>
<td>ZEMS-TKN-SYSTEM</td>
<td>ZSPI-TYP-UINT.</td>
</tr>
<tr>
<td>ZEMS-TKN-USERID</td>
<td>ZSPI-TYP-BYTE-PAIR.</td>
</tr>
<tr>
<td>ZTCI-TKN-SEV</td>
<td>ZSPI-TYP-ENUM.</td>
</tr>
<tr>
<td>ZTCI-TKN-ERR-NUM</td>
<td>ZSPI-TYP-INT.</td>
</tr>
<tr>
<td>ZTCI-TKN-IOPNAME</td>
<td>ZSPI-TYP-STRING.</td>
</tr>
</tbody>
</table>

**Event-Message Text**

```
ERROR error ON IOP iopname
```

### Unconditional Tokens

- **ZSPI-TKN-SSID**
  - is a header token that specifies the subsystem ID. The value used for the TCP/IP subsystem is ZTCI-VAL-SSID.

- **ZEMS-TKN-EVENTNUMBER**
  - is a header token that contains the event number. Its value is ZTCI-EVT-IOP-AWAITIO-ERR (4).
ZEMS-TKN-SUBJECT-MARK

is the standard EMS token that always immediately precedes the subject token here, ZCOM-TKN-SUBJ-SUBNET.

ZCOM-TKN-SUBJ-SUBNET

contains the name of the TCP/IP subnet that is the subject of this event message.

ZEMS-TKN-EMPHASIS

is a header token that contains the value ZSPI-VAL-TRUE, indicating that this is a critical event.

(ZEMS-TKN-) CONSOLE-PRINT, CPU, CRTPID, GENTIME, LOGTIME, PIN, SYSTEM, and USERID

are automatically placed in the header when the EMSINIT procedure is called and the event message buffer is initialized. For a description of these tokens, see Section 5, Common Definitions. For details on any of the EMS tokens, see the EMS Manual.

ZTCI-TKN-SEV

contains the value ZTCI-VAL-SEV-WARN. This value indicates that ZTCI-EVT-IOP-AWAITIO-ERR is only a warning message. The process continues to function, but some functionality has been lost; for example, the network may be going down, or a device accessed by the TCP/IP subsystem may be experiencing errors.

ZTCI-TKN-ERR-NUM

contains the number of the file-system error that has occurred. For more details, see the Guardian Procedure Errors and Messages Manual.

ZTCI-TKN-IOPNAME

contains the name of the X25AM line and subdevice being accessed by the TCP/IP process for transmitting and receiving data when the error occurred.

**Event-Message Text**

In the text message, *error* corresponds to the value in ZTCI-TKN-ERR-NUM and *iopname* corresponds to the value in ZTCI-TKN-IOPNAME.

**Cause.** An error occurred while the TCP/IP process was waiting for an I/O operation to be completed. This error is returned only if the TCP/IP process called the AWAITIO procedure and the operator has aborted an X25AM line or subdevice while the TCP/IP process has the port or subdevice opened.

**Recovery.** The TCP/IP process automatically attempts to reopen the I/O process when it is closed as a result of an error. If successful, operator intervention is not
required. However, if the attempt to reopen fails, the operator must check the X25AM line (for X.25 subnets) to verify that the problem is not at the port or line level.

5: ZTCI-EVT-SUMSTATE-CHG

The summary state of a TCP/IP subnet has changed as a result of a command or an error on the subnet interface. This event is a noncritical event. The event message is reported to the EMS collector ($0).

Unconditional Tokens

<table>
<thead>
<tr>
<th>Token Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZSPI-TKN-SSID</td>
<td>token-type ZSPI-TYP-SSID.</td>
</tr>
<tr>
<td>ZEMS-TKN-EVENTNUMBER</td>
<td>token-type ZSPI-TYP-ENUM.</td>
</tr>
<tr>
<td>ZEMS-TKN-SUBJECT-MARK</td>
<td>token-type ZSPI-TYP-MARK.</td>
</tr>
<tr>
<td>ZCOM-TKN-SUBJ-SUBNET</td>
<td>token-type ZSPI-TYP-STRING.</td>
</tr>
<tr>
<td>ZEMS-TKN-EMPHASIS</td>
<td>token-type ZSPI-TYP-BOOLEAN.</td>
</tr>
<tr>
<td>ZEMS-TKN-CONSOLE-PRINT</td>
<td>token-type ZSPI-TYP-BOOLEAN.</td>
</tr>
<tr>
<td>ZEMS-TKN-CPU</td>
<td>token-type ZSPI-TYP-UINT.</td>
</tr>
<tr>
<td>ZEMS-TKN-CRTPID</td>
<td>token-type ZSPI-TYP-CRTPID.</td>
</tr>
<tr>
<td>ZEMS-TKN-GENTIME</td>
<td>token-type ZSPI-TYP-TIMESTAMP.</td>
</tr>
<tr>
<td>ZEMS-TKN-LOGTIME</td>
<td>token-type ZSPI-TYP-TIMESTAMP.</td>
</tr>
<tr>
<td>ZEMS-TKN-PIN</td>
<td>token-type ZSPI-TYP-UINT.</td>
</tr>
<tr>
<td>ZEMS-TKN-SYSTEM</td>
<td>token-type ZSPI-TYP-UINT.</td>
</tr>
<tr>
<td>ZEMS-TKN-USERID</td>
<td>token-type ZSPI-TYP-BYTE-PAIR.</td>
</tr>
<tr>
<td>ZTCI-TKN-SEV</td>
<td>token-type ZSPI-TYP-ENUM.</td>
</tr>
<tr>
<td>ZTCI-TKN-IOPNAME</td>
<td>token-type ZSPI-TYP-STRING.</td>
</tr>
<tr>
<td>ZCOM-TKN-OBJSTATE</td>
<td>token-type ZSPI-TYP-ENUM.</td>
</tr>
<tr>
<td>ZCOM-TKN-OLD-OBJSTATE</td>
<td>token-type ZSPI-TYP-ENUM.</td>
</tr>
<tr>
<td>ZCOM-TKN-CAUSE-SUMSTATE-CHG</td>
<td>token-type ZSPI-TYP-ENUM.</td>
</tr>
</tbody>
</table>

Event-Message Text

SUBNET subname IOP iopname CHANGED FROM old-state TO new-state BECAUSE OF cause

Unconditional Tokens

ZSPI-TKN-SSID

is a header token that specifies the subsystem ID. The value used for the TCP/IP subsystem is ZTCI-VAL-SSID.

ZEMS-TKN-EVENTNUMBER

is a header token that contains the event number. Its value is ZTCI-EVT-SUMSTATE-CHG (5).

ZEMS-TKN-SUBJECT-MARK

is the standard EMS token that always immediately precedes the subject token here, ZCOM-TKN-SUBJ-SUBNET.
ZCOM-TKN-SUBJ-SUBNET
contains the name of the TCP/IP subnet that is the subject of this event message.

ZEMS-TKN-EMPHASIS
is a header token that contains the value ZSPI-VAL-FALSE, indicating that this is
not a critical event.

(ZEMS-TKN-) CONSOLE-PRINT, CPU, CRTPID, GENTIME, LOGTIME, PIN,
SYSTEM, and USERID
are automatically placed in the header when the EMSINIT procedure is called and
the event message buffer is initialized. For a description of these tokens, see
Section 5, Common Definitions. For details on any of the EMS tokens, see the
EMS Manual.

ZTCI-TKN-SEV
contains the value ZTCI-VAL-SEV-INFO. This value indicates that
ZTCI-EVT-SUMSTATE-CHG is an informational message only. No recovery action
is necessary.

ZTCI-TKN-IOPNAME
contains the name of the SLSA LIF or X25AM line and subdevice (for X.25
subnets) being accessed by the TCP/IP process for transmitting and receiving data
when the summary state change occurred.

ZCOM-TKN-OBJSTATE
is the summary state of the object after the change has taken place. It corresponds
to the value of new-state in the message text.

ZCOM-TKN-OLD-OBJSTATE
is the summary state of the object before the change took place. It corresponds to
the value of old-state in the message text.

ZCOM-TKN-CAUSE-SUMSTATE-CHG
is the reason the summary state has changed. The value can be one of these:

ZTCI-VAL-CAUSE-ERR
indicates the change occurred because of an error that occurred on the subnet
interface.

ZTCI-VAL-CAUSE-OP
indicates the change occurred because of a command.
indicates the change occurred because of a recovery operation.

**Event-Message Text**

In the text message, *subname* corresponds to the value in ZCOM-TKN-SUBJ-SUBNET, *iopname* corresponds to the value in ZTCI-TKN-IOPNAME, *old-state* corresponds to the value in ZCOM-TKN-OLD-OBJSTATE, *new-state* corresponds to the value in ZCOM-TKN-OBJSTATE, and *cause* corresponds to the value in ZCOM-TKN-CAUSE-SUMSTATE-CHG.

**Cause.** The summary state of a TCP/IP subnet has changed as a result of a command or an error.

**Recovery.** If a command caused the summary-state change, no action is necessary. If the state changed for any other reason, note the other event message returned and contact your HP representative.

6: ZTCI-EVT-INTL

The TCP/IP process detected an internal error and has halted. The TCP/IP process no longer exists.

This event is critical. It is reported to the EMS collector ($0).

<table>
<thead>
<tr>
<th><strong>Unconditional Tokens</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>ZSPI-TKN-SSID</td>
</tr>
<tr>
<td>ZEMS-TKN-EVENTNUMBER</td>
</tr>
<tr>
<td>ZEMS-TKN-SUBJECT-MARK</td>
</tr>
<tr>
<td>ZCOM-TKN-SUBJ-PROC</td>
</tr>
<tr>
<td>ZEMS-TKN-EMPHASIS</td>
</tr>
<tr>
<td>ZEMS-TKN-CONSOLE-PRINT</td>
</tr>
<tr>
<td>ZEMS-TKN-CPU</td>
</tr>
<tr>
<td>ZEMS-TKN-CRTPID</td>
</tr>
<tr>
<td>ZEMS-TKN-GENTIME</td>
</tr>
<tr>
<td>ZEMS-TKN-LOGTIME</td>
</tr>
<tr>
<td>ZEMS-TKN-PIN</td>
</tr>
<tr>
<td>ZEMS-TKN-SYSTEM</td>
</tr>
<tr>
<td>ZEMS-TKN-USERID</td>
</tr>
<tr>
<td>ZTCI-TKN-SEV</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Tandem Internal Tokens</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>ZTCI-TKN-TEXT</td>
</tr>
</tbody>
</table>

**Event-Message Text**

TCP/IP NO LONGER OPERATIONAL *cause*
Unconditional Tokens

ZSPI-TKN-SSID

is a header token that specifies the subsystem ID. The value used for the TCP/IP subsystem is ZTCI-VAL-SSID.

ZEMS-TKN-EVENTNUMBER

is a header token that contains the event number. Its value is ZTCI-EVT-INTL (6).

ZEMS-TKN-SUBJECT-MARK

is the standard EMS token that always immediately precedes the subject token here, ZCOM-TKN-SUBJ-PROC.

ZCOM-TKN-SUBJ-PROC

contains the name of the TCP/IP subsystem that is the subject of this event message.

ZEMS-TKN-EMPHASIS

is a header token that contains the value ZSPI-VAL-TRUE, indicating that this is a critical event.

(ZEMS-TKN-) CONSOLE-PRINT, CPU, CRTPID, GENTIME, LOGTIME, PIN, SYSTEM, and USERID

are automatically placed in the header when the EMSINIT procedure is called and the event message buffer is initialized. For a description of these tokens, see Section 5, Common Definitions. For details on any of the EMS tokens, see the EMS Manual.

ZTCI-TKN-SEV

contains the value ZTCI-VAL-SEV-FATAL. This value indicates that ZTCI-EVT-INTL is fatal to the TCP/IP process. The TCP/IP process halts itself and must be restarted.

Event-Message Text

In the text message, cause corresponds to the value in ZTCI-TKN-TEXT.

Cause. The TCP/IP process detected an internal error from which it cannot recover. The TCP/IP process no longer exists.

Recovery. Restart the TCP/IP process using the RUN command. Send the SAVEABEND file to HP for analysis.
7: ZTCI-EVT-DUP-IP-ADDR

The Address Resolution Protocol (ARP) module of the TCP/IP process has detected that another machine on the network is broadcasting the same IP address. If this were to continue, each of the two machines might receive packets intended for the other and would function improperly.

This event is critical. It is reported to the EMS collector ($0).

Unconditional Tokens

<table>
<thead>
<tr>
<th>Token Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZSPI-TKN-SSID</td>
<td>token-type ZSPI-TYP-SSID.</td>
</tr>
<tr>
<td>ZEMS-TKN-EVENTNUMBER</td>
<td>token-type ZSPI-TYP-ENUM.</td>
</tr>
<tr>
<td>ZEMS-TKN-SUBJECT-MARK</td>
<td>token-type ZSPI-TYP-MARK.</td>
</tr>
<tr>
<td>ZCOM-TKN-SUBJ-PROC</td>
<td>token-type ZSPI-TYP-STRING.</td>
</tr>
<tr>
<td>ZEMS-TKN-EMPHASIS</td>
<td>token-type ZSPI-TYP-BOOLEAN.</td>
</tr>
<tr>
<td>ZEMS-TKN-CONSOLE-PRINT</td>
<td>token-type ZSPI-TYP-BOOLEAN.</td>
</tr>
<tr>
<td>ZEMS-TKN-CPU</td>
<td>token-type ZSPI-TYP-UINT.</td>
</tr>
<tr>
<td>ZEMS-TKN-CRTPID</td>
<td>token-type ZSPI-TYP-CRTPID.</td>
</tr>
<tr>
<td>ZEMS-TKN-GENTIME</td>
<td>token-type ZSPI-TYP-TIMESTAMP.</td>
</tr>
<tr>
<td>ZEMS-TKN-LOGTIME</td>
<td>token-type ZSPI-TYP-TIMESTAMP.</td>
</tr>
<tr>
<td>ZEMS-TKN-PIN</td>
<td>token-type ZSPI-TYP-UINT.</td>
</tr>
<tr>
<td>ZEMS-TKN-SYSTEM</td>
<td>token-type ZSPI-TYP-UINT.</td>
</tr>
<tr>
<td>ZEMS-TKN-USERID</td>
<td>token-type ZSPI-TYP-BYTE-PAIR.</td>
</tr>
<tr>
<td>ZTCI-TKN-SEV</td>
<td>token-type ZSPI-TYP-ENUM.</td>
</tr>
<tr>
<td>ZTCI-TKN-IP-ADDR</td>
<td>token-type ZSPI-TYP-INT2.</td>
</tr>
<tr>
<td>ZTCI-TKN-ENET-ADDR</td>
<td>token-type ZSPI-TYP-STRING.</td>
</tr>
</tbody>
</table>

Event-Message Text

ARP:  DUPLICATE IP ADDRESS \textit{ip-addr} FROM ETHERNET ADDRESS \textit{enet-addr}

Unconditional Tokens

ZSPI-TKN-SSID

is a header token that specifies the subsystem ID. The value used for the TCP/IP subsystem is ZTCI-VAL-SSID.

ZEMS-TKN-EVENTNUMBER

is a header token that contains the event number. Its value is ZTCI-EVT-DUP-IPADDR (7).

ZEMS-TKN-SUBJECT-MARK

is the standard EMS token that always immediately precedes the subject token here, ZCOM-TKN-SUBJ-PROC.
ZCOM-TKN-SUBJ-PROC
contains the name of the TCP/IP subsystem that is the subject of this event message.

ZEMS-TKN-EMPHASIS
is a header token that contains the value ZSPI-VAL-TRUE, indicating that this is a critical event.

(ZEMS-TKN-) CONSOLE-PRINT, CPU, CRTPID, GENTIME, LOGTIME, PIN, SYSTEM, and USERID
are automatically placed in the header when the EMSINIT procedure is called and the event message buffer is initialized. For a description of these tokens, see Section 5, Common Definitions. For details on any of the EMS tokens, see the EMS Manual.

ZTCI-TKN-SEV
contains the value ZTCI-VAL-SEV-WARN. This value indicates that ZTCI-EVT-DUP-IP-ADDR is only a warning message. The process continues to function, but some functionality has been lost; for example, the network may be going down, or a device accessed by the TCP/IP subsystem may be experiencing errors.

ZTCI-TKN-IP-ADDR
contains the duplicate IP address.

ZTCI-TKN-ENET-ADDR
contains the Ethernet address of the other machine that is broadcasting a duplicate IP address.

Event-Message Text
In the text message, IP-addr corresponds to the value in ZTCI-TKN-IP-ADDR and enet-addr corresponds to the value in ZTCI-TKN-ENET-ADDR.

Cause. Another machine on the network is broadcasting the same IP address as the machine on which the TCP/IP process reporting the event is running.

Recovery. The operator should determine which machine is broadcasting incorrectly and reassign a unique IP address to it.
8: ZTCI-EVT-BCAST-ENET-ADDR

The Address Resolution Protocol (ARP) module of the TCP/IP process has detected that another machine on the Ethernet LAN is using the broadcast address FF.FF.FF.FF.FF.FF as its Ethernet address. This causes all packets sent to the broadcast address to be received by all machines on the Ethernet LAN. Performance of all machines is adversely affected. In addition, some implementations of TCP/IP might try to forward the packet(s) back over the Ethernet LAN, flooding the LAN and crashing most machines on the LAN.

This event is critical. It is reported to the EMS collector ($0).

Unconditional Tokens

- **ZSPI-TKN-SSID**: token-type ZSPI-TYP-SSID.
- **ZEMS-TKN-EVENTNUMBER**: token-type ZSPI-TYP-ENUM.
- **ZEMS-TKN-SUBJECT-MARK**: token-type ZSPI-TYP-MARK.
- **ZCOM-TKN-SUBJ-PROC**: token-type ZSPI-TYP-STRING.
- **ZEMS-TKN-EMPHASIS**: token-type ZSPI-TYP-BOOLEAN.
- **ZEMS-TKN-CONSOLE-PRINT**: token-type ZSPI-TYP-BOOLEAN.
- **ZEMS-TKN-CRTPID**: token-type ZSPI-TYP-CRTPID.
- **ZEMS-TKN-GENTIME**: token-type ZSPI-TYP-TIMESTAMP.
- **ZEMS-TKN-LOGTIME**: token-type ZSPI-TYP-TIMESTAMP.
- **ZEMS-TKN-PIN**: token-type ZSPI-TYP-UINT.
- **ZEMS-TKN-SYSTEM**: token-type ZSPI-TYP-UINT.
- **ZEMS-TKN-USERID**: token-type ZSPI-TYP-BYTE-PAIR.
- **ZTCI-TKN-SEV**: token-type ZSPI-TYP-ENUM.
- **ZTCI-TKN-IP-ADDR**: token-type ZSPI-TYP-INT2.

Event-Message Text

ARP: ETHERNET BROADCAST ADDRESS (FF.FF.FF.FF.FF.FF) USED BY IP ADDRESS *ip-addr*

Unconditional Tokens

**ZSPI-TKN-SSID**

is a header token that specifies the subsystem ID. The value used for the TCP/IP subsystem is ZTCI-VAL-SSID.

**ZEMS-TKN-EVENTNUMBER**

is a header token that contains the event number. Its value is ZTCI-EVT-BCAST-ENET-ADDR (8).

**ZEMS-TKN-SUBJECT-MARK**

is the standard EMS token that always immediately precedes the subject token here, ZCOM-TKN-SUBJ-PROC.
ZCOM-TKN-SUBJ-PROC
contains the name of the TCP/IP subsystem that is the subject of this event message.

ZEMS-TKN-EMPHASIS
is a header token that contains the value ZSPI-VAL-TRUE, indicating that this is a critical event.

(ZEMS-TKN-) CONSOLE-PRINT, CPU, CRTPID, GENTIME, LOGTIME, PIN, SYSTEM, and USERID
are automatically placed in the header when the EMSINIT procedure is called and the event message buffer is initialized. For a description of these tokens, see Section 5, Common Definitions. For details on any of the EMS tokens, see the EMS Manual.

ZTCI-TKN-SEV
contains the value ZTCI-VAL-SEV-WARN. This value indicates that ZTCI-EVT-BCAST-ENET-ADDR is only a warning message. The process continues to function, but some functionality has been lost; for example, the network may be going down, or a device accessed by the TCP/IP subsystem may be experiencing errors.

ZTCI-TKN-IP-ADDR
contains the IP address of the machine using the broadcast address as its Ethernet address.

Event-Message Text
In the text message, \textit{ip-addr} corresponds to the value in ZTCI-TKN-IP-ADDR.

\textbf{Cause.} A machine on the Ethernet LAN is using a broadcast address as its Ethernet address. This can cause performance deterioration, flooding of the Ethernet LAN, and machine crashes.

\textbf{Recovery.} The operator should determine which machine is using the broadcast address as its Ethernet address and fix the machine. Note that you cannot reassign an Ethernet address.
The TCP/IP process cannot open any more X25AM subdevices. This event is noncritical. It is reported to the EMS collector ($0).

**Unconditional Tokens**

- **ZSPI-TKN-SSID**
  - token-type ZSPI-TYP-SSID.
  - is a header token that specifies the subsystem ID. The value used for the TCP/IP subsystem is ZTCI-VAL-SSID.

- **ZEMS-TKN-EVENTNUMBER**
  - token-type ZSPI-TYP-ENUM.
  - is a header token that contains the event number. Its value is ZTCI-EVT-MAX-DEV (9).

- **ZEMS-TKN-SUBJECT-MARK**
  - token-type ZSPI-TYP-MARK.
  - is the standard EMS token that always immediately precedes the subject token here, ZCOM-TKN-SUBJ-PROC.

- **ZCOM-TKN-SUBJ-PROC**
  - token-type ZSPI-TYP-STRING.
  - contains the name of the TCP/IP subsystem that is the subject of this event message.

- **ZEMS-TKN-EMPHASIS**
  - token-type ZSPI-TYP-BOOLEAN.
  - is a header token that contains the value ZSPI-VAL-FALSE, indicating that this is a noncritical event.

**Event-Message Text**

NUMBER OF X25AM DEVICES AVAILABLE num-devices
(ZEMS-TKN-) CONSOLE-PRINT, CPU, CRTPID, GENTIME, LOGTIME, PIN, SYSTEM, and USERID

are automatically placed in the header when the EMSINIT procedure is called and the event message buffer is initialized. For a description of these tokens, see Section 5, Common Definitions. For details on any of the EMS tokens, see the EMS Manual.

ZTCI-TKN-SEV

contains the value ZTCI-VAL-SEV-INFO. This value indicates that ZTCI-EVT-MAX-DEV is an informational message only. No recovery action is necessary.

ZTCI-TKN-NUM-DEV

contains the number of X25AM subdevices opened by the TCP/IP process.

Event-Message Text

In the text message, num-devices corresponds to the value in ZTCI-TKN-NUM-DEV.

Cause. Either the operator has opened all the subdevices added to X25AM, or X25AM will not let the operator access subdevices.

Recovery. If the number of subdevices opened is less than the number that have been added to X25AM, the operator should check X25AM to determine why these devices cannot be accessed. If the number of subdevices opened is equal to the number that have been added to X25AM, this message should be treated as informational only and no recovery action is necessary.
10: ZTCI-EVT-CONN-ESTAB

An X.25 connection has been established with a remote TCP/IP process. This message is for informational purposes only.

This event is noncritical. It is reported to the EMS collector ($0).

**Unconditional Tokens**

<table>
<thead>
<tr>
<th>Token Name</th>
<th>Token Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZSPI-TKN-SSID</td>
<td>ZSPI-TYP-SSID</td>
</tr>
<tr>
<td>ZEMS-TKN-EVENTNUMBER</td>
<td>ZSPI-TYP-ENUM</td>
</tr>
<tr>
<td>ZEMS-TKN-SUBJECT-MARK</td>
<td>ZSPI-TYP-MARK</td>
</tr>
<tr>
<td>ZCOM-TKN-SUBJ-PROC</td>
<td>ZSPI-TYP-STRING</td>
</tr>
<tr>
<td>ZEMS-TKN-EMPHASIS</td>
<td>ZSPI-TYP-BOOLEAN</td>
</tr>
<tr>
<td>ZEMS-TKN-CONSOLE-PRINT</td>
<td>ZSPI-TYP-BOOLEAN</td>
</tr>
<tr>
<td>ZEMS-TKN-CPU</td>
<td>ZSPI-TYP-UINT</td>
</tr>
<tr>
<td>ZEMS-TKN-CRTPID</td>
<td>ZSPI-TYP-CRTPID</td>
</tr>
<tr>
<td>ZEMS-TKN-GENTIME</td>
<td>ZSPI-TYP-TIMESTAMP</td>
</tr>
<tr>
<td>ZEMS-TKN-LOGTIME</td>
<td>ZSPI-TYP-TIMESTAMP</td>
</tr>
<tr>
<td>ZEMS-TKN-PIN</td>
<td>ZSPI-TYP-UINT</td>
</tr>
<tr>
<td>ZEMS-TKN-SYSTEM</td>
<td>ZSPI-TYP-UINT</td>
</tr>
<tr>
<td>ZEMS-TKN-USERID</td>
<td>ZSPI-TYP-BYTE-PAIR</td>
</tr>
<tr>
<td>ZTCI-TKN-SEV</td>
<td>ZSPI-TYP-ENUM</td>
</tr>
<tr>
<td>ZTCI-TKN-IP-ADDR</td>
<td>ZSPI-TYP-INT2</td>
</tr>
</tbody>
</table>

**Event-Message Text**

Connection established with *ip-addr*

**Unconditional Tokens**

**ZSPI-TKN-SSID**

is a header token that specifies the subsystem ID. The value used for the TCP/IP subsystem is ZTCI-VAL-SSID.

**ZEMS-TKN-EVENTNUMBER**

is a header token that contains the event number. Its value is ZTCI-EVT-CONN-ESTAB (10).

**ZEMS-TKN-SUBJECT-MARK**

is the standard EMS token that always immediately precedes the subject token here, ZCOM-TKN-SUBJ-PROC.

**ZCOM-TKN-SUBJ-PROC**

contains the name of the TCP/IP subsystem that is the subject of this event message.
ZEMS-TKN-EMPHASIS

is a header token that contains the value ZSPI-VAL-FALSE, indicating that this is a noncritical event.

(ZEMS-TKN-) CONSOLE-PRINT, CPU, CRTPID, GENTIME, LOGTIME, PIN, SYSTEM, and USERID

are automatically placed in the header when the EMSINIT procedure is called and the event message buffer is initialized. See Section 5, Common Definitions, for a description of these tokens. For details on any of the EMS tokens, see the EMS Manual.

ZTCI-TKN-SEV

contains the value ZTCI-VAL-SEV-INFO. This value indicates that ZTCI-EVT-CONN-ESTAB is an informational message only. No recovery action is necessary.

ZTCI-TKN-IP-ADDR

contains the IP address of the remote host with which a connection has been established.

Event-Message Text

In the text message, ip-addr corresponds to the value in ZTCI-TKN-IP-ADDR.

Cause. An X.25 connection was established to a remote TCP/IP host.

Recovery. No action is necessary. This message is for information only.

11: ZTCI-EVT-CONN-TIMEOUT

An X.25 connection to a remote host has been deleted due to a timeout or an overload.
This event is noncritical. It is reported to the EMS collector ($0).

### Unconditional Tokens

<table>
<thead>
<tr>
<th>Token Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZSPI-TKN-SSID</td>
<td>is a header token that specifies the subsystem ID. The value used for the TCP/IP subsystem is ZTCI-VAL-SSID.</td>
</tr>
<tr>
<td>ZEMS-TKN-EVENTNUMBER</td>
<td>is a header token that contains the event number. Its value is ZTCI-EVT-CONN-TIMEOUT (11).</td>
</tr>
<tr>
<td>ZEMS-TKN-SUBJECT-MARK</td>
<td>is the standard EMS token that always immediately precedes the subject token here, ZCOM-TKN-SUBJ-PROC.</td>
</tr>
<tr>
<td>ZCOM-TKN-SUBJ-PROC</td>
<td>contains the name of the TCP/IP subsystem that is the subject of this event message.</td>
</tr>
<tr>
<td>ZEMS-TKN-EMPHASIS</td>
<td>is a header token that contains the value ZSPI-VAL-FALSE, indicating that this is a noncritical event.</td>
</tr>
</tbody>
</table>

### Event-Message Text

CONNECTION WITH ip-addr TIMED OUT
(ZEMS-TKN-) CONSOLE-PRINT, CPU, CRTPID, GENTIME, LOGTIME, PIN, SYSTEM, and USERID

are automatically placed in the header when the EMSINIT procedure is called and the event message buffer is initialized. For a description of these tokens, see Section 5, Common Definitions. For details on any of the EMS tokens, see the EMS Manual.

ZTCI-TKN-SEV
contains the value ZTCI-VAL-SEV-INFO. This value indicates that ZTCI-EVT-CONN-TIMEOUT is an informational message only. No recovery action is necessary.

ZTCI-TKN-IP-ADDR
contains the IP address of the remote host with which the connection was deleted.

Event-Message Text
In the text message, ip-addr corresponds to the value in ZTCI-TKN-IP-ADDR.

Cause. An X.25 connection to a remote host has been deleted due to a timeout or an overload.

Recovery. This message is for information only. If no free devices were available, at least one device is now available for new connections.

12: ZTCI-EVT-DEV-NOT-AVAIL
An X.25 connection to a remote host cannot be established because no free devices are available. All the connections that have been established are active and cannot be timed out.

The TCP/IP process also returns the socket error EHOSTUNREACH for the attempted connection.
This event is critical. It is reported to the EMS collector ($0).

### Unconditional Tokens

<table>
<thead>
<tr>
<th>Token</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZSPI-TKN-SSID</td>
<td>ZSPI-TYP-SSID</td>
</tr>
<tr>
<td>ZEMS-TKN-EVENTNUMBER</td>
<td>ZSPI-TYP-ENUM</td>
</tr>
<tr>
<td>ZEMS-TKN-SUBJECT-MARK</td>
<td>ZSPI-TYP-MARK</td>
</tr>
<tr>
<td>ZCOM-TKN-SUBJ-PROC</td>
<td>ZSPI-TYP-STRING</td>
</tr>
<tr>
<td>ZEMS-TKN-EMPHASIS</td>
<td>ZSPI-TYP-BOOLEAN</td>
</tr>
<tr>
<td>ZEMS-TKN-CONSOLE-PRINT</td>
<td>ZSPI-TYP-BOOLEAN</td>
</tr>
<tr>
<td>ZEMS-TKN-CPU</td>
<td>ZSPI-TYP-UINT</td>
</tr>
<tr>
<td>ZEMS-TKN-CRTPID</td>
<td>ZSPI-TYP-CRTPID</td>
</tr>
<tr>
<td>ZEMS-TKN-GENTIME</td>
<td>ZSPI-TYP-TIMESTAMP</td>
</tr>
<tr>
<td>ZEMS-TKN-LOGTIME</td>
<td>ZSPI-TYP-TIMESTAMP</td>
</tr>
<tr>
<td>ZEMS-TKN-PIN</td>
<td>ZSPI-TYP-UINT</td>
</tr>
<tr>
<td>ZEMS-TKN-SYSTEM</td>
<td>ZSPI-TYP-UINT</td>
</tr>
<tr>
<td>ZEMS-TKN-USERID</td>
<td>ZSPI-TYP-BYTE-PAIR</td>
</tr>
<tr>
<td>ZTCI-TKN-SEV</td>
<td>ZSPI-TYP-ENUM</td>
</tr>
<tr>
<td>ZTCI-TKN-IP-ADDR</td>
<td>ZSPI-TYP-INT2</td>
</tr>
</tbody>
</table>

### Event-Message Text

NO DEVICES AVAILABLE TO MAKE CONNECTION TO HOST ip-addr

### Unconditional Tokens

**ZSPI-TKN-SSID**

is a header token that specifies the subsystem ID. The value used for the TCP/IP subsystem is ZTCI-VAL-SSID.

**ZEMS-TKN-EVENTNUMBER**

is a header token that contains the event number. Its value is ZTCI-EVT-DEV-NOT-AVAIL (12).

**ZEMS-TKN-SUBJECT-MARK**

is the standard EMS token that always immediately precedes the subject token here, ZCOM-TKN-SUBJ-PROC.

**ZCOM-TKN-SUBJ-PROC**

contains the name of the TCP/IP subsystem that is the subject of this event message.

**ZEMS-TKN-EMPHASIS**

is a header token that contains the value ZSPI-VAL-TRUE, indicating that this is a critical event.
(ZEMS-TKN-) CONSOLE-PRINT, CPU, CRTPID, GENTIME, LOGTIME, PIN, SYSTEM, and USERID

are automatically placed in the header when the EMSINIT procedure is called and the event message buffer is initialized. For a description of these tokens, see Section 5, Common Definitions. For details on any of the EMS tokens, see the EMS Manual.

ZTCI-TKN-SEV

contains the value ZTCI-VAL-SEV-WARN. This value indicates that ZTCI-EVT-DEV-NOT-AVAIL is only a warning message. The process continues to function, but some functionality has been lost; for example, the network may be going down, or a device accessed by the TCP/IP subsystem may be experiencing errors.

ZTCI-TKN-IP-ADDR

contains the IP address of the remote host to which the connection could not be established.

Event-Message Text

In the text message, ip-addr corresponds to the value in ZTCI-TKN-IP-ADDR.

Cause. The X.25 connection to the remote host could not be established because no free devices were available. All the connections that have been established are active and cannot be timed out.

Recovery. No new connections can be established until the ZTCI-EVT-CONN-TIMEOUT event message is generated.
13: ZTCI-EVT-SYN-ERR

A syntax error has been detected in the IP-to-X.25 address file. The number of the line where the error occurred is returned in the message.

This event is noncritical. It is reported to the EMS collector ($0).

Unconditional Tokens

<table>
<thead>
<tr>
<th>Token Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZSPI-TKN-SSID</td>
<td>is a header token that specifies the subsystem ID. The value used for the TCP/IP subsystem is ZTCI-VAL-SSID.</td>
</tr>
<tr>
<td>ZEMS-TKN-EVENTNUMBER</td>
<td>is a header token that contains the event number. Its value is ZTCI-EVT-SYN-ERR (13).</td>
</tr>
<tr>
<td>ZEMS-TKN-SUBJECT-MARK</td>
<td>is the standard EMS token that always immediately precedes the subject token here, ZCOM-TKN-SUBJ-PROC.</td>
</tr>
<tr>
<td>ZCOM-TKN-SUBJ-PROC</td>
<td>contains the name of the TCP/IP subsystem that is the subject of this event message.</td>
</tr>
</tbody>
</table>

Event-Message Text

SYNTAX ERROR ON LINE line-num FILE fname
is a header token that contains the value ZSPI-VAL-FALSE, indicating that this is a noncritical event.

(CONSOLE-PRINT, CPU, CRTPID, GENTIME, LOGTIME, PIN, SYSTEM, and USERID) are automatically placed in the header when the EMSINIT procedure is called and the event message buffer is initialized. For a description of these tokens, see Section 5, Common Definitions. For details on any of the EMS tokens, see the EMS Manual.

contains the value ZTCI-VAL-SEV-INFO. This value indicates that ZTCI-EVT-SYN-ERR is an informational message only. No recovery action is necessary.

contains the number of the line where the syntax error occurred.

contains the name of the file where the syntax error occurred.

In the text message, line-num corresponds to the value in ZTCI-TKN-LINE-NUM and fname corresponds to the value in ZTCI-TKN-FNAME.

The IP-to-X.25 address file has a syntax error on the specified line.

The operator should correct the error and restart the TCP/IP process.

A configuration file is secured incorrectly and an access error has occurred. This error is generated by the X.25 interface, SMTP, or the Domain Name Server. When generated by the X.25 interface, this message is only a warning because the X.25 configuration file, the IP-to-X.25 address file, is not required. The TCP/IP process continues to function and uses the default DDN IP-to-X.25 address mapping instead. When generated by SMTP or the Domain Name Server, this message is fatal (that is, the program that detected the error exits because it cannot access the configuration file). In either case, the name of the file where the access error occurred is returned in the message.
This event is critical. It is reported to the EMS collector ($0).

### Unconditional Tokens

<table>
<thead>
<tr>
<th>Token Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZSPI-TKN-SSID</td>
<td>header token that specifies the subsystem ID. The value used for the TCP/IP subsystem is ZTCI-VAL-SSID.</td>
</tr>
<tr>
<td>ZEMS-TKN-EVENTNUMBER</td>
<td>header token that contains the event number. Its value is ZTCI-EVT-CONF-FILE-ACC-ERR (16).</td>
</tr>
<tr>
<td>ZEMS-TKN-SUBJECT-MARK</td>
<td>the standard EMS token that always immediately precedes the subject token here, ZCOM-TKN-SUBJ-PROC.</td>
</tr>
<tr>
<td>ZCOM-TKN-SUBJ-PROC</td>
<td>contains the name of the TCP/IP subsystem that is the subject of this event message.</td>
</tr>
<tr>
<td>ZEMS-TKN-EMPHASIS</td>
<td>header token that contains the value ZSPI-VAL-TRUE, indicating that this is a critical event.</td>
</tr>
</tbody>
</table>

### Event-Message Text

CONFIGURATION FILE fname ACCESS ERROR
ZTCI-EVT-CONF-FILE-ACC-ERR

(ZEMS-TKN-) CONSOLE-PRINT, CPU, CRTPID, GENTIME, LOGTIME, PIN, SYSTEM, and USERID

are automatically placed in the header when the EMSINIT procedure is called and the event message buffer is initialized. For a description of these tokens, see Section 5, Common Definitions. For details on any of the EMS tokens, see the EMS Manual.

ZTCI-TKN-SEV

contains the value ZTCI-VAL-SEV-WARN when this event is generated by the X.25 interface and the value ZTCI-VAL-SEV-FATAL when this event is generated by SMTP or the Domain Name Server. The WARN value indicates that ZTCI-EVT-CONF-FILE-ACC-ERR is only a warning message. The TCP/IP process continues to function, but some functionality has been lost; for example, the network may be going down, or a device accessed by the TCP/IP subsystem may be experiencing errors. The FATAL value indicates that ZTCI-EVT-CONF-FILE-ACC-ERR is fatal to the SMTP and Domain Name Server processes. These processes halt when this error occurs, and they must be restarted.

ZTCI-TKN-FNAME

contains the name of the file where the access error occurred.

Event-Message Text

In the text message, fname corresponds to the value in ZTCI-TKN-FNAME.

Cause. A configuration file has been secured incorrectly. This error is generated by either the X.25 interface, SMTP, or the Domain Name Server.

This error is generated by the X.25 interface when the IP-to-X.25 address file has been secured incorrectly. Because the IP-to-X.25 address file is not required, the TCP/IP process continues to function and uses the default DDN IP-to-X.25 address mapping instead.

This error is generated by SMTP or the Domain Name Server when one of these programs cannot access one of its configuration files. The program that detected the error exits because it cannot access the configuration file.

Recovery. If this error is generated by the X.25 interface, the IP-to-X.25 address file should be secured so that it is readable by the person starting the TCP/IP process, and the TCP/IP process should be restarted.

If this error is generated by SMTP or the Domain Name Server, the file should be secured so that it is readable by the person starting the TCP/IP process, and the process that detected the error should be restarted.
17: ZTCI-EVT-CONF-FILE-SYN-ERR

A configuration file contains a syntax error. This message is generated by SMTP or the Domain Name Server. The name of the file and the number of the line where the error occurred are returned in the message. The line containing the syntax error is ignored.

This event is critical. It is reported to the EMS collector ($0).

<table>
<thead>
<tr>
<th>Unconditional Tokens</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZSPI-TKN-SSID</td>
<td>Token-type ZSPI-TYP-SSID.</td>
</tr>
<tr>
<td>ZEMS-TKN-EVENTNUMBER</td>
<td>Token-type ZSPI-TYP-ENUM.</td>
</tr>
<tr>
<td>ZEMS-TKN-SUBJECT-MARK</td>
<td>Token-type ZSPI-TYP-MARK.</td>
</tr>
<tr>
<td>ZCOM-TKN-SUBJ-PROC</td>
<td>Token-type ZSPI-TYP-STRING.</td>
</tr>
<tr>
<td>ZEMS-TKN-EMPHASIS</td>
<td>Token-type ZSPI-TYP-BOOLEAN.</td>
</tr>
<tr>
<td>ZEMS-TKN-CONSOLE-PRINT</td>
<td>Token-type ZSPI-TYP-BOOLEAN.</td>
</tr>
<tr>
<td>ZEMS-TKN-CPU</td>
<td>Token-type ZSPI-TYP-UINT.</td>
</tr>
<tr>
<td>ZEMS-TKN-CRTPID</td>
<td>Token-type ZSPI-TYP-CRTPID.</td>
</tr>
<tr>
<td>ZEMS-TKN-GENTIME</td>
<td>Token-type ZSPI-TYP-TIMESTAMP.</td>
</tr>
<tr>
<td>ZEMS-TKN-LOGTIME</td>
<td>Token-type ZSPI-TYP-TIMESTAMP.</td>
</tr>
<tr>
<td>ZEMS-TKN-PIN</td>
<td>Token-type ZSPI-TYP-UINT.</td>
</tr>
<tr>
<td>ZEMS-TKN-SYSTEM</td>
<td>Token-type ZSPI-TYP-UINT.</td>
</tr>
<tr>
<td>ZEMS-TKN-USERID</td>
<td>Token-type ZSPI-TYP-BYTE-PAIR.</td>
</tr>
<tr>
<td>ZTCI-TKN-SEV</td>
<td>Token-type ZSPI-TYP-ENUM.</td>
</tr>
<tr>
<td>ZTCI-TKN-FNAME</td>
<td>Token-type ZSPI-TYP-FNAME32.</td>
</tr>
<tr>
<td>ZTCI-TKN-LINE-NUM</td>
<td>Token-type ZSPI-TYP-INT.</td>
</tr>
</tbody>
</table>

Event-Message Text

SYNTAX ERROR IN FILE fname LINE line-num

Unconditional Tokens

ZSPI-TKN-SSID

is a header token that specifies the subsystem ID. The value used for the TCP/IP subsystem is ZTCI-VAL-SSID.

ZEMS-TKN-EVENTNUMBER

is a header token that contains the event number. Its value is ZTCI-EVT-CONF-FILE-SYN-ERR (17).

ZEMS-TKN-SUBJECT-MARK

is the standard EMS token that always immediately precedes the subject token here, ZCOM-TKN-SUBJ-PROC.

ZCOM-TKN-SUBJ-PROC

contains the name of the TCP/IP subsystem that is the subject of this event message.
ZEMS-TKN-EMPHASIS

is a header token that contains the value ZSPI-VAL-TRUE, indicating that this is a critical event.

(ZEMS-TKN-) CONSOLE-PRINT, CPU, CRTPID, GENTIME, LOGTIME, PIN, SYSTEM, and USERID

are automatically placed in the header when the EMSINIT procedure is called and the event message buffer is initialized. For a description of these tokens, see Section 5, Common Definitions. For details on any of the EMS tokens, see the EMS Manual.

ZTCI-TKN-SEV

contains the value ZTCI-VAL-SEV-WARN. This value indicates that ZTCI-EVT-CONF-FILE-SYN-ERR is only a warning message. The process continues to function, but some functionality has been lost; for example, the network may be going down, or a device accessed by the TCP/IP subsystem may be experiencing errors.

ZTCI-TKN-FNAME

contains the name of the file where the syntax error occurred.

ZTCI-TKN-LINE-NUM

contains the number of the line where the syntax error occurred.

Event-Message Text

In the text message, fname corresponds to the value in ZTCI-TKN-FNAME and line-num corresponds to the value in ZTCI-TKN-LINE-NUM.

Cause. A configuration file contains a syntax error. This message is generated by SMTP, or the Domain Name Server. The line containing the error is ignored.

Recovery. The line containing the syntax error should be corrected and the process that detected the error restarted.
19: ZTCI-EVT-NEW-PROC-ERR

An attempt by the LISTNER process to start a server for a user request failed. A NEWPROCESS error that indicates the reason the server process could not be started is returned in the event message. The user is sent a connection refused message.

This event is critical. It is reported to the EMS collector ($0).

Unconditional Tokens

<table>
<thead>
<tr>
<th>Token Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZSPI-TKN-SSID</td>
<td>token-type ZSPI-TYP-SSID.</td>
</tr>
<tr>
<td>ZEMS-TKN-EVENTNUMBER</td>
<td>token-type ZSPI-TYP-ENUM.</td>
</tr>
<tr>
<td>ZEMS-TKN-SUBJECT-MARK</td>
<td>token-type ZSPI-TYP-MARK.</td>
</tr>
<tr>
<td>ZCOM-TKN-SUBJ-PROC</td>
<td>token-type ZSPI-TYP-STRING.</td>
</tr>
<tr>
<td>ZEMS-TKN-EMPHASIS</td>
<td>token-type ZSPI-TYP-BOOLEAN.</td>
</tr>
<tr>
<td>ZEMS-TKN-CONSOLE-PRINT</td>
<td>token-type ZSPI-TYP-BOOLEAN.</td>
</tr>
<tr>
<td>ZEMS-TKN-CPU</td>
<td>token-type ZSPI-TYP-UINT.</td>
</tr>
<tr>
<td>ZEMS-TKN-CRTPID</td>
<td>token-type ZSPI-TYP-CRTPID.</td>
</tr>
<tr>
<td>ZEMS-TKN-GENETIME</td>
<td>token-type ZSPI-TYP-TIMESTAMP.</td>
</tr>
<tr>
<td>ZEMS-TKN-LOGTIME</td>
<td>token-type ZSPI-TYP-TIMESTAMP.</td>
</tr>
<tr>
<td>ZEMS-TKN-PIN</td>
<td>token-type ZSPI-TYP-UINT.</td>
</tr>
<tr>
<td>ZEMS-TKN-SYSTEM</td>
<td>token-type ZSPI-TYP-UINT.</td>
</tr>
<tr>
<td>ZEMS-TKN-USERID</td>
<td>token-type ZSPI-TYP-BYTE-PAIR.</td>
</tr>
<tr>
<td>ZTCI-TKN-SEV</td>
<td>token-type ZSPI-TYP-ENUM.</td>
</tr>
<tr>
<td>ZTCI-TKN-NEW-PROC-ERR</td>
<td>token-type ZSPI-TYP-INT.</td>
</tr>
<tr>
<td>ZTCI-TKN-FNAME</td>
<td>token-type ZSPI-TYP-FNAME32.</td>
</tr>
</tbody>
</table>

Event-Message Text

LISTNER: ERROR new-proc-err STARTING
NEW-PROCESS FILE fname

Unconditional Tokens

ZSPI-TKN-SSID

is a header token that specifies the subsystem ID. The value used for the TCP/IP subsystem is ZTCI-VAL-SSID.

ZEMS-TKN-EVENTNUMBER

is a header token that contains the event number. Its value is ZTCI-EVT-NEW-PROC-ERR (19).

ZEMS-TKN-SUBJECT-MARK

is the standard EMS token that always immediately precedes the subject token here, ZCOM-TKN-SUBJ-PROC.

ZCOM-TKN-SUBJ-PROC

contains the name of the TCP/IP subsystem that is the subject of this event message.
ZEMS-TKN-EMPHASIS

is a header token that contains the value ZSPI-VAL-TRUE, indicating that this is a critical event.

(ZEMS-TKN-) CONSOLE-PRINT, CPU, CRTPID, GENTIME, LOGTIME, PIN, SYSTEM, and USERID

are automatically placed in the header when the EMSINIT procedure is called and the event message buffer is initialized. For a description of these tokens, see Section 5, Common Definitions. For details on any of the EMS tokens, see the EMS Manual.

ZTCI-TKN-SEV

contains the value ZTCI-VAL-SEV-WARN. This value indicates that ZTCI-EVT-NEW-PROC-ERR is only a warning message. The process continues to function, but some functionality has been lost; for example, the network may be going down, or a device accessed by the TCP/IP subsystem may be experiencing errors.

ZTCI-TKN-NEW-PROC-ERR

indicates the NEWPROCESS error that has occurred. For more details, see the Guardian Procedure Errors and Messages Manual.

ZTCI-TKN-FNAME

contains the name of server that could not be started.

Event-Message Text

In the text message, new-proc-err corresponds to the value in ZTCI-TKN-NEW-PROC-ERR, and fname corresponds to the value in ZTCI-TKN-FNAME.

Cause. This error is usually caused by an error in the PORTCONF file.

Recovery. If this error is caused by an error in the PORTCONF file, the error in the PORTCONF file should be corrected and the LISTNER process restarted.
**20: ZTCI-EVT-XFER-ERR**

An error has occurred during a TRANSFER request. This message is reported by SMTP.

This event is critical. It is reported to the EMS collector ($0).

### Unconditional Tokens

<table>
<thead>
<tr>
<th>Token Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZSPI-TKN-SSID</td>
<td>token-type ZSPI-TYP-SSIDID.</td>
</tr>
<tr>
<td>ZEMS-TKN-EVENTNUMBER</td>
<td>token-type ZSPI-TYP-ENUM.</td>
</tr>
<tr>
<td>ZEMS-TKN-SUBJECT-MARK</td>
<td>token-type ZSPI-TYP-MARK.</td>
</tr>
<tr>
<td>ZCOM-TKN-SUBJ-PROC</td>
<td>token-type ZSPI-TYP-STRING.</td>
</tr>
<tr>
<td>ZEMS-TKN-EMPHASIS</td>
<td>token-type ZSPI-TYP-BOOLEAN.</td>
</tr>
<tr>
<td>ZEMS-TKN-CONSOLE-PRINT</td>
<td>token-type ZSPI-TYP-BOOLEAN.</td>
</tr>
<tr>
<td>ZEMS-TKN-CPU</td>
<td>token-type ZSPI-TYP-UINT.</td>
</tr>
<tr>
<td>ZEMS-TKN-CRTPID</td>
<td>token-type ZSPI-TYP-CRTPID.</td>
</tr>
<tr>
<td>ZEMS-TKN-GENTIME</td>
<td>token-type ZSPI-TYP-TIMESTAMP.</td>
</tr>
<tr>
<td>ZEMS-TKN-LOGTIME</td>
<td>token-type ZSPI-TYP-TIMESTAMP.</td>
</tr>
<tr>
<td>ZEMS-TKN-PIN</td>
<td>token-type ZSPI-TYP-UINT.</td>
</tr>
<tr>
<td>ZEMS-TKN-SYSTEM</td>
<td>token-type ZSPI-TYP-UINT.</td>
</tr>
<tr>
<td>ZEMS-TKN-USERID</td>
<td>token-type ZSPI-TYP-BYTE-PAIR.</td>
</tr>
<tr>
<td>ZTCI-TKN-SEV</td>
<td>token-type ZSPI-TYP-ENUM.</td>
</tr>
<tr>
<td>ZTCI-TKN-XFER-UOW</td>
<td>token-type ZSPI-TYP-ENUM.</td>
</tr>
<tr>
<td>ZTCI-TKN-XFER-ERR</td>
<td>token-type ZSPI-TYP-ENUM.</td>
</tr>
<tr>
<td>ZTCI-TKN-XFER-ERR-DETL</td>
<td>token-type ZSPI-TYP-ENUM.</td>
</tr>
</tbody>
</table>

### Event-Message Text

SMTP: TRANSFER UOW REQUEST xfer-uow ERROR xfer-err, DETAIL xfer-err-detl

### Unconditional Tokens

**ZSPI-TKN-SSID**

is a header token that specifies the subsystem ID. The value used for the TCP/IP subsystem is ZTCI-VAL-SSID.

**ZEMS-TKN-EVENTNUMBER**

is a header token that contains the event number. Its value is ZTCI-EVT-XFER-ERR (20).

**ZEMS-TKN-SUBJECT-MARK**

is the standard EMS token that always immediately precedes the subject token here, ZCOM-TKN-SUBJ-PROC.

**ZCOM-TKN-SUBJ-PROC**

contains the name of the TCP/IP subsystem that is the subject of this event message.
ZEMS-TKN-EMPHASIS

is a header token that contains the value ZSPI-VAL-TRUE, indicating that this is a critical event.

(ZEMS-TKN-) CONSOLE-PRINT, CPU, CRTPID, GENTIME, LOGTIME, PIN, SYSTEM, and USERID

are automatically placed in the header when the EMSINIT procedure is called and the event message buffer is initialized. For a description of these tokens, see Section 5, Common Definitions. For details on any of the EMS tokens, see the EMS Manual.

ZTCI-TKN-SEV

contains the value ZTCI-VAL-SEV-WARN. This value indicates that ZTCI-EVT-XFER-ERR is only a warning message. The process continues to function, but some functionality has been lost; for example, the network may be going down, or a device accessed by the TCP/IP subsystem may be experiencing errors.

ZTCI-TKN-XFER-UOW

identifies the unit-of-work (UOW) that could not be processed because the error occurred. Each UOW code corresponds to a TRANSFER operation request. Processes communicate with TRANSFER by issuing requests that contain one or more UOW codes. The token value can be one of these:

ZTCI-VAL-XFER-START-SESS

indicates the error occurred during the processing of a request containing a START-SESSION UOW.

ZTCI-VAL-XFER-ADD-RECIP

indicates the error occurred during the processing of a request containing an ADD-RECIP UOW.

ZTCI-VAL-XFER-ADD-ITEM-REC

indicates the error occurred during the processing of a request containing an ADD-ITEM-REC UOW.

ZTCI-VAL-XFER-SUBMIT-PKG

indicates the error occurred during the processing of a request containing a SUBMIT-PKG UOW.

ZTCI-VAL-XFER-END-SESS

indicates the error occurred during the processing of a request containing an END-SESSION UOW.
ZTCI-VAL-XFER-CREATE-ITEM
indicates the error occurred during the processing of a request containing a CREATE-ITEM UOW.

ZTCI-VAL-XFER-ATTACH-COMPNT
indicates the error occurred during the processing of a request containing an ATTACH-COMPNT-A01 UOW.

ZTCI-VAL-XFER-GET-ITEM-REC
indicates the error occurred during the processing of a request containing a GET-ITEM-REC UOW.

ZTCI-VAL-XFER-CREATE-FOLDER
indicates the error occurred during the processing of a request containing a CREATE-FOLDER-B00 UOW.

ZTCI-VAL-XFER-SAVE-ITEM
indicates the error occurred during the processing of a request containing a SAVE-ITEM UOW.

ZTCI-VAL-XFER-DETACH-COMPNT
indicates the error occurred during the processing of a request containing a DETACH-COMPNT UOW.

ZTCI-VAL-XFER-SCAN-FOLDER
indicates the error occurred during the processing of a request containing a SCAN-FOLDER-B00 UOW.

ZTCI-VAL-XFER-GET-ITEM-DESCR
indicates the error occurred during the processing of a request containing a GET-ITEM-DESCR UOW.

ZTCI-VAL-XFER-GET-RECIP-REC
indicates the error occurred during the processing of a request containing a GET-RECIP-REC UOW.

ZTCI-VAL-XFER-GET-ITEM-COMPNT
indicates the error occurred during the processing of a request containing a GET-ITEM-COMPNT-A01 UOW.

ZTCI-VAL-XFER-UNSAVE-ITEM
indicates the error occurred during the processing of a request containing an UNSAVE-ITEM UOW.
For more detailed descriptions of these UOWs, including descriptions of the operations they perform, see the *TRANSFER Programming Manual*.

**ZTCI-TKN-XFER-ERR**

contains the error code value reported by TRANSFER. For descriptions of the values reported, see the *TRANSFER Programming Manual*.

**ZTCI-TKN-XFER-ERR-DETL**

contains any additional error detail values reported with the TRANSFER error. Certain TRANSFER errors return additional values containing details related to the main error code. For descriptions of the values reported, see the *TRANSFER Programming Manual*.

**Event-Message Text**

In the text message, `xfer-uow` corresponds to the value in ZTCI-TKN-XFER-ERR, `xfer-err` corresponds to the value in ZTCI-TKN-XFER-ERR, and `xfer-err-detl` corresponds to the value in ZTCI-TKN-XFER-ERR-DETL.

**Cause.** TRANSFER has returned an error to the SMTP process.

**Recovery.** For the recommended recovery action, see the *TRANSFER Programming Manual*.

**21: ZTCI-EVT-PARAM-ERR**

A configuration file contains a parameter error. This message is generated by SMTP or the Domain Name Server. The name of the file and the number of the line where the error occurred are returned in the message. The line containing the parameter error is ignored.
This event is critical. It is reported to the EMS collector ($0).

### Unconditional Tokens

<table>
<thead>
<tr>
<th>Token Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZSPI-TKN-SSID</td>
<td>token-type ZSPI-TYP-SSID.</td>
</tr>
<tr>
<td>ZEMS-TKN-EVENTNUMBER</td>
<td>token-type ZSPI-TYP-ENUM.</td>
</tr>
<tr>
<td>ZEMS-TKN-SUBJECT-MARK</td>
<td>token-type ZSPI-TYP-MARK.</td>
</tr>
<tr>
<td>ZCOM-TKN-SUBJ-PROC</td>
<td>token-type ZSPI-TYP-STRING.</td>
</tr>
<tr>
<td>ZEMS-TKN-EMPHASIS</td>
<td>token-type ZSPI-TYP-BOOLEAN.</td>
</tr>
<tr>
<td>ZEMS-TKN-CONSOLE-PRINT</td>
<td>token-type ZSPI-TYP-BOOLEAN.</td>
</tr>
<tr>
<td>ZEMS-TKN-CPU</td>
<td>token-type ZSPI-TYP-UINT.</td>
</tr>
<tr>
<td>ZEMS-TKN-CRTPID</td>
<td>token-type ZSPI-TYP-CRTPID.</td>
</tr>
<tr>
<td>ZEMS-TKN-GENTIME</td>
<td>token-type ZSPI-TYP-TIMESTAMP.</td>
</tr>
<tr>
<td>ZEMS-TKN-LOGTIME</td>
<td>token-type ZSPI-TYP-TIMESTAMP.</td>
</tr>
<tr>
<td>ZEMS-TKN-PIN</td>
<td>token-type ZSPI-TYP-UINT.</td>
</tr>
<tr>
<td>ZEMS-TKN-SYSTEM</td>
<td>token-type ZSPI-TYP-UINT.</td>
</tr>
<tr>
<td>ZEMS-TKN-USERID</td>
<td>token-type ZSPI-TYP-BYTE-PAIR.</td>
</tr>
<tr>
<td>ZTCI-TKN-SEV</td>
<td>token-type ZSPI-TYP-ENUM.</td>
</tr>
<tr>
<td>ZTCI-TKN-FNAME</td>
<td>token-type ZSPI-TYP-FNAME32.</td>
</tr>
<tr>
<td>ZTCI-TKN-LINE-NUM</td>
<td>token-type ZSPI-TYP-INT.</td>
</tr>
</tbody>
</table>

### Event-Message Text

```
ERROR IN PARAMETER SPECIFIED IN FILE fname
LINE line-num
```

### Unconditional Tokens

**ZSPI-TKN-SSID**

is a header token that specifies the subsystem ID. The value used for the TCP/IP subsystem is ZTCI-VAL-SSID.

**ZEMS-TKN-EVENTNUMBER**

is a header token that contains the event number. Its value is ZTCI-EVT-PARAM-ERR (21).

**ZEMS-TKN-SUBJECT-MARK**

is the standard EMS token that always immediately precedes the subject token here, ZCOM-TKN-SUBJ-PROC.

**ZCOM-TKN-SUBJ-PROC**

contains the name of the TCP/IP subsystem that is the subject of this event message.

**ZEMS-TKN-EMPHASIS**

is a header token that contains the value ZSPI-VAL-TRUE, indicating that this is a critical event.
(ZEMS-TKN-) CONSOLE-PRINT, CPU, CRTPID, GENTIME, LOGTIME, PIN, SYSTEM, and USERID

are automatically placed in the header when the EMSINIT procedure is called and the event message buffer is initialized. For a description of these tokens, see Section 5, Common Definitions. For details on any of the EMS tokens, see the EMS Manual.

ZTCI-TKN-SEV

contains the value ZTCI-VAL-SEV-WARN. This value indicates that ZTCI-EVT-PARAM-ERR is only a warning message. The process continues to function, but some functionality has been lost; for example, the network may be going down, or a device accessed by the TCP/IP subsystem may be experiencing errors.

ZTCI-TKN-FNAME

contains the name of the file where the parameter error occurred.

ZTCI-TKN-LINE-NUM

contains the number of the line where the parameter error occurred.

Event-Message Text

In the text message, fname corresponds to the value in ZTCI-TKN-FNAME and line-num corresponds to the value in ZTCI-TKN-LINE-NUM.

Cause. A configuration file contains a parameter error. This message is generated by SMTP or the Domain Name Server.

Recovery. The line containing the parameter error should be corrected and the process that detected the error restarted.
22: ZTCI-EVT-ADDR-ERR

A configuration file contains an address error.

This event is critical. It is reported to the EMS collector ($0).

Unconditional Tokens

<table>
<thead>
<tr>
<th>Token Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZSPI-TKN-SSID</td>
<td>token-type ZSPI-TYP-SSID.</td>
</tr>
<tr>
<td>ZEMS-TKN-EVENTNUMBER</td>
<td>token-type ZSPI-TYP-ENUM.</td>
</tr>
<tr>
<td>ZEMS-TKN-SUBJECT-MARK</td>
<td>token-type ZSPI-TYP-MARK.</td>
</tr>
<tr>
<td>ZCOM-TKN-SUBJ-PROC</td>
<td>token-type ZSPI-TYP-STRING.</td>
</tr>
<tr>
<td>ZEMS-TKN-EMPHASIS</td>
<td>token-type ZSPI-TYP-BOOLEAN.</td>
</tr>
<tr>
<td>ZEMS-TKN-CONSOLE-PRINT</td>
<td>token-type ZSPI-TYP-BOOLEAN.</td>
</tr>
<tr>
<td>ZEMS-TKN-CPU</td>
<td>token-type ZSPI-TYP-UINT.</td>
</tr>
<tr>
<td>ZEMS-TKN-CRTPID</td>
<td>token-type ZSPI-TYP-CRTPID.</td>
</tr>
<tr>
<td>ZEMS-TKN-GENTIME</td>
<td>token-type ZSPI-TYP-TIMESTAMP.</td>
</tr>
<tr>
<td>ZEMS-TKN-LOGTIME</td>
<td>token-type ZSPI-TYP-TIMESTAMP.</td>
</tr>
<tr>
<td>ZEMS-TKN-PIN</td>
<td>token-type ZSPI-TYP-UINT.</td>
</tr>
<tr>
<td>ZEMS-TKN-SYSTEM</td>
<td>token-type ZSPI-TYP-UINT.</td>
</tr>
<tr>
<td>ZEMS-TKN-USERID</td>
<td>token-type ZSPI-TYP-BYTE-PAIR.</td>
</tr>
<tr>
<td>ZTCI-TKN-SEV</td>
<td>token-type ZSPI-TYP-ENUM.</td>
</tr>
<tr>
<td>ZTCI-TKN-FNAME</td>
<td>token-type ZSPI-TYP-FNAME32.</td>
</tr>
<tr>
<td>ZTCI-TKN-LINE-NUM</td>
<td>token-type ZSPI-TYP-INT.</td>
</tr>
</tbody>
</table>

Event-Message Text

ERROR IN ADDRESS/NAME SPECIFIED IN FILE \textit{fname}
LINE \textit{line-num}

Unconditional Tokens

ZSPI-TKN-SSID

is a header token that specifies the subsystem ID. The value used for the TCP/IP subsystem is ZTCI-VAL-SSID.

ZEMS-TKN-EVENTNUMBER

is a header token that contains the event number. Its value is ZTCI-EVT-ADDR-ERR (22).

ZEMS-TKN-SUBJECT-MARK

is the standard EMS token that always immediately precedes the subject token here, ZCOM-TKN-SUBJ-PROC.

ZCOM-TKN-SUBJ-PROC

contains the name of the TCP/IP subsystem that is the subject of this event message.
ZEMS-TKN-EMPHASIS

is a header token that contains the value ZSPI-VAL-TRUE, indicating that this is a critical event.

(ZEMS-TKN-) CONSOLE-PRINT, CPU, CRTPID, GENTIME, LOGTIME, PIN, SYSTEM, and USERID

are automatically placed in the header when the EMSINIT procedure is called and the event message buffer is initialized. For a description of these tokens, see Section 5, Common Definitions. For details on any of the EMS tokens, see the EMS Manual.

ZTCI-TKN-SEV

contains the value ZTCI-VAL-SEV-WARN. This value indicates that ZTCI-EVT-ADDR-ERR is only a warning message. The process continues to function, but some functionality has been lost; for example, the network may be going down, or a device accessed by the TCP/IP subsystem may be experiencing errors.

ZTCI-TKN-FNAME

contains the name of the file where the parameter error occurred.

ZTCI-TKN-LINE-NUM

contains the number of the line where the parameter error occurred.

Event-Message Text

In the text message, $fname$ corresponds to the value in ZTCI-TKN-FNAME and $line-num$ corresponds to the value in ZTCI-TKN-LINE-NUM.

Cause. A configuration file contains a parameter error. This message is generated by SMTP or the Domain Name Server.

Recovery. The line containing the parameter error should be corrected and the process that detected the error restarted.
23: ZTCI-EVT-SOCK-ERR

A socket error was returned on a socket request. The error and the socket operation being performed when the error occurred are returned in the message.

This event is critical. It is reported to the EMS collector ($0).

Unconditional Tokens

<table>
<thead>
<tr>
<th>Token Name</th>
<th>Token Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZSPI-TKN-SSID</td>
<td>ZSPI-TYP-SSID</td>
</tr>
<tr>
<td>ZEMS-TKN-EVENTNUMBER</td>
<td>ZSPI-TYP-ENUM</td>
</tr>
<tr>
<td>ZEMS-TKN-SUBJECT-MARK</td>
<td>ZSPI-TYP-MARK</td>
</tr>
<tr>
<td>ZCOM-TKN-SUBJ-PROC</td>
<td>ZSPI-TYP-STRING</td>
</tr>
<tr>
<td>ZEMS-TKN-EMPHASIS</td>
<td>ZSPI-TYP-BOOLEAN</td>
</tr>
<tr>
<td>ZEMS-TKN-CONSOLE-PRINT</td>
<td>ZSPI-TYP-BOOLEAN</td>
</tr>
<tr>
<td>ZEMS-TKN-CPU</td>
<td>ZSPI-TYP-UINT</td>
</tr>
<tr>
<td>ZEMS-TKN-CRTPID</td>
<td>ZSPI-TYP-CRTPID</td>
</tr>
<tr>
<td>ZEMS-TKN-GENTIME</td>
<td>ZSPI-TYP-TIMESTAMP</td>
</tr>
<tr>
<td>ZEMS-TKN-LOGTIME</td>
<td>ZSPI-TYP-TIMESTAMP</td>
</tr>
<tr>
<td>ZEMS-TKN-PIN</td>
<td>ZSPI-TYP-UINT</td>
</tr>
<tr>
<td>ZEMS-TKN-SYSTEM</td>
<td>ZSPI-TYP-UINT</td>
</tr>
<tr>
<td>ZEMS-TKN-USERID</td>
<td>ZSPI-TYP-BYTE-PAIR</td>
</tr>
<tr>
<td>ZTCI-TKN-SEV</td>
<td>ZSPI-TYP-ENUM</td>
</tr>
<tr>
<td>ZTCI-TKN-SOCK-FUNC</td>
<td>ZSPI-TYP-INT</td>
</tr>
<tr>
<td>ZTCI-TKN-ERR-NUM</td>
<td>ZSPI-TYP-INT</td>
</tr>
</tbody>
</table>

Event-Message Text

ERROR err-num IN SOCKET OPERATION sock-func

Unconditional Tokens

ZSPI-TKN-SSID

is a header token that specifies the subsystem ID. The value used for the TCP/IP subsystem is ZTCI-VAL-SSID.

ZEMS-TKN-EVENTNUMBER

is a header token that contains the event number. Its value is ZTCI-EVT-SOCK-ERR (23).

ZEMS-TKN-SUBJECT-MARK

is the standard EMS token that always immediately precedes the subject token here, ZCOM-TKN-SUBJ-PROC.

ZCOM-TKN-SUBJ-PROC

contains the name of the TCP/IP subsystem that is the subject of this event message.
ZEMS-TKN-EMPHASIS

is a header token that contains the value ZSPI-VAL-TRUE, indicating that this is a critical event.

(ZEMS-TKN-) CONSOLE-PRINT, CPU, CRTPID, GENTIME, LOGTIME, PIN, SYSTEM, and USERID

are automatically placed in the header when the EMSINIT procedure is called and the event message buffer is initialized. For a description of these tokens, see Section 5, Common Definitions. For details on any of the EMS tokens, see the EMS Manual.

ZTCI-TKN-SEV

contains the value ZTCI-VAL-SEV-WARN. This value indicates that ZTCI-EVT-SOCK-ERR is only a warning message. The process continues to function, but some functionality has been lost; for example, the network may be going down, or a device accessed by the TCP/IP subsystem may be experiencing errors.

ZTCI-TKN-SOCK-FUNC

contains the socket operation that was being performed when the error occurred.

ZTCI-TKN-ERR-NUM

contains the error that has occurred.

Event-Message Text

In the text message, \textit{err-num} corresponds to the value in ZTCI-TKN-ERR-NUM and \textit{sock-func} corresponds to the value in ZTCI-TKN-SOCK-FUNC.

Cause. An error occurred in an HP application during a socket operation.

Recovery. Contact your HP support representative.
24: ZTCI-EVT-PROTO-ERR

A configuration file contains a prototype error. This message is generated by SMTP or the Domain Name Server. The name of the file and the number of the line where the error occurred are returned in the message. The line containing the parameter error is ignored.

This event is critical. It is reported to the EMS collector ($0).

Unconditional Tokens

ZSPI-TKN-SSID            token-type ZSPI-TYP-SSID.
ZEMS-TKN-EVENTNUMBER     token-type ZSPI-TYP-ENUM.
ZEMS-TKN-SUBJECT-MARK    token-type ZSPI-TYP-MARK.
ZCOM-TKN-SUBJ-PROC       token-type ZSPI-TYP-STRING.
ZEMS-TKN-EMPHASIS        token-type ZSPI-TYP-BOOLEAN.
ZEMS-TKN-CONSOLE-PRINT   token-type ZSPI-TYP-BOOLEAN.
ZEMS-TKN-CPU             token-type ZSPI-TYP-UINT.
ZEMS-TKN-CRTPID          token-type ZSPI-TYP-CRTPID.
ZEMS-TKN-GENTIME         token-type ZSPI-TYP-TIMESTAMP.
ZEMS-TKN-LOGTIME         token-type ZSPI-TYP-TIMESTAMP.
ZEMS-TKN-PIN             token-type ZSPI-TYP-UINT.
ZEMS-TKN-SYSTEM          token-type ZSPI-TYP-UINT.
ZEMS-TKN-USERID          token-type ZSPI-TYP-BYTE-PAIR.
ZTCI-TKN-SEV             token-type ZSPI-TYP-ENUM.
ZTCI-TKN-FNAME           token-type ZSPI-TYP-FNAME32.
ZTCI-TKN-LINE-NUM        token-type ZSPI-TYP-INT.

Event-Message Text

ERROR IN PROTOCOL SPECIFIED IN FILE fname
LINE line-num

Unconditional Tokens

ZSPI-TKN-SSID

is a header token that specifies the subsystem ID. The value used for the TCP/IP subsystem is ZTCI-VAL-SSID.

ZEMS-TKN-EVENTNUMBER

is a header token that contains the event number. Its value is ZTCI-EVT-PROTO-ERR (24).

ZEMS-TKN-SUBJECT-MARK

is the standard EMS token that always immediately precedes the subject token here, ZCOM-TKN-SUBJ-PROC.
ZCOM-TKN-SUBJ-PROC

contains the name of the TCP/IP subsystem that is the subject of this event message.

ZEMS-TKN-EMPHASIS

is a header token that contains the value ZSPI-VAL-TRUE, indicating that this is a critical event.

(ZEMS-TKN-) CONSOLE-PRINT, CPU, CRTPID, GENTIME, LOGTIME, PIN, SYSTEM, and USERID

are automatically placed in the header when the EMSINIT procedure is called and the event message buffer is initialized. For a description of these tokens, see Section 5, Common Definitions. For details on any of the EMS tokens, see the EMS Manual.

ZTCI-TKN-SEV

contains the value ZTCI-VAL-SEV-WARN. This value indicates that ZTCI-EVT-PROTO-ERR is only a warning message. The process continues to function, but some functionality has been lost; for example, the network may be going down, or a device accessed by the TCP/IP subsystem may be experiencing errors.

ZTCI-TKN-FNAME

contains the name of the file where the parameter error occurred.

ZTCI-TKN-LINE-NUM

contains the number of the line where the parameter error occurred.

Event-Message Text

In the text message, $fname$ corresponds to the value in ZTCI-TKN-FNAME and $line-num$ corresponds to the value in ZTCI-TKN-LINE-NUM.

Cause. A configuration file contains a parameter error. This message is generated by SMTP or the Domain Name Server.

Recovery. The line containing the parameter error should be corrected and the process that detected the error restarted.
25: ZTCI-EVT-RECV-OPEN-FAIL

TCP/IP process could not open $RECEIVE.

This event is critical. It is reported to the EMS collector ($0).

Unconditional Tokens

<table>
<thead>
<tr>
<th>Token Type</th>
<th>Token Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZSPI-TKN-SSID</td>
<td></td>
<td>is a header token that specifies the subsystem ID. The value used for the TCP/IP subsystem is ZTCI-VAL-SSID.</td>
</tr>
<tr>
<td>ZEMS-TKN-EVENTNUMBER</td>
<td></td>
<td>is a header token that contains the event number. Its value is ZTCI-EVT-ADDR-ERR (22).</td>
</tr>
<tr>
<td>ZEMS-TKN-SUBJECT-MARK</td>
<td></td>
<td>is the standard EMS token that always immediately precedes the subject token here, ZCOM-TKN-SUBJ-PROC.</td>
</tr>
<tr>
<td>ZCOM-TKN-SUBJ-PROC</td>
<td></td>
<td>contains the name of the TCP/IP subsystem that is the subject of this event message.</td>
</tr>
</tbody>
</table>

Event-Message Text

OPEN OF $RECEIVE FAILED IN THE TCP/IP PROCESS,
ERR err-num
ZEMS-TKN-EMPHASIS

is a header token that contains the value ZSPI-VAL-TRUE, indicating that this is a critical event.

(ZEMS-TKN-) CONSOLE-PRINT, CPU, CRTPID, GENTIME, LOGTIME, PIN, SYSTEM, and USERID

are automatically placed in the header when the EMSINIT procedure is called and the event message buffer is initialized. For a description of these tokens, see Section 5, Common Definitions. For details on any of the EMS tokens, see the EMS Manual.

ZTCI-TKN-SEV

contains the value ZTCI-VAL-SEV-WARN. This value indicates that ZTCI-EVT-ADDR-ERR is only a warning message. The process continues to function, but some functionality has been lost; for example, the network may be going down, or a device accessed by the TCP/IP subsystem may be experiencing errors.

ZTCI-TKN-ERR-NUM

contains the error that has occurred.

Event-Message Text

In the text message, $err-num$ corresponds to the value in ZTCI-TKN-ERR-NUM.

Cause. The TCP/IP process could not open $RECEIVE$.

Effect. The TCP/IP process stops.

Recovery. Restart the TCP/IP process.
26: ZTCI-EVT-SSINT-FAILED

This is an SPI buffer initialization error.

This event is critical. It is reported to the EMS collector ($0).

Unconditional Tokens

<table>
<thead>
<tr>
<th>Token Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZSPI-TKN-SSID</td>
<td>is a header token that specifies the subsystem ID. The value used for the TCP/IP subsystem is ZTCI-VAL-SSID.</td>
</tr>
<tr>
<td>ZEMS-TKN-EVENTNUMBER</td>
<td>is a header token that contains the event number. Its value is ZTCI-EVT-ADDR-ERR (22).</td>
</tr>
<tr>
<td>ZEMS-TKN-SUBJECT-MARK</td>
<td>is the standard EMS token that always immediately precedes the subject token here, ZCOM-TKN-SUBJ-PROC.</td>
</tr>
<tr>
<td>ZCOM-TKN-SUBJ-PROC</td>
<td>contains the name of the TCP/IP subsystem that is the subject of this event message.</td>
</tr>
</tbody>
</table>

Event-Message Text

INIT_BUFF: SSINIT FAILED ERR err-num
ZEMS-TKN-EMPHASIS

is a header token that contains the value ZSPI-VAL-TRUE, indicating that this is a critical event.

(ZEMS-TKN-) CONSOLE-PRINT, CPU, CRTPID, GENTIME, LOGTIME, PIN, SYSTEM, and USERID

are automatically placed in the header when the EMSINIT procedure is called and the event message buffer is initialized. For a description of these tokens, see Section 5, Common Definitions. For details on any of the EMS tokens, see the EMS Manual.

ZTCI-TKN-SEV

contains the value ZTCI-VAL-SEV-WARN. This value indicates that ZTCI-EVT-ADDR-ERR is only a warning message. The process continues to function, but some functionality has been lost; for example, the network may be going down, or a device accessed by the TCP/IP subsystem may be experiencing errors.

ZTCI-TKN-ERR-NUM

contains the error that has occurred.

Event-Message Text

In the text message, err-num corresponds to the value in ZTCI-TKN-ERR-NUM.

Cause. SCF-interface related problem.

Effect. The error is returned to the user.

Recovery. Retry the SCF command.
## 27: ZTCI-EVT-CANT-HANDLE-AF

The protocol is not supported.

This event is critical. It is reported to the EMS collector ($0).

### Unconditional Tokens

<table>
<thead>
<tr>
<th>Token Type</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZSPI-TKN-SSID</td>
<td>token-type ZSPI-TYP-SSID.</td>
</tr>
<tr>
<td>ZEMS-TKN-EVENTNUMBER</td>
<td>token-type ZSPI-TYP-ENUM.</td>
</tr>
<tr>
<td>ZEMS-TKN-SUBJECT-MARK</td>
<td>token-type ZSPI-TYP-MARK.</td>
</tr>
<tr>
<td>ZCOM-TKN-SUBJ-PROC</td>
<td>token-type ZSPI-TYP-STRING.</td>
</tr>
<tr>
<td>ZEMS-TKN-CONSOLE-PRINT</td>
<td>token-type ZSPI-TYP-BOOLEAN.</td>
</tr>
<tr>
<td>ZEMS-TKN-CPU</td>
<td>token-type ZSPI-TYP-UINT.</td>
</tr>
<tr>
<td>ZEMS-TKN-CRTPID</td>
<td>token-type ZSPI-TYP-CRTPID.</td>
</tr>
<tr>
<td>ZEMS-TKN-GENTIME</td>
<td>token-type ZSPI-TYP-TIMESTAMP.</td>
</tr>
<tr>
<td>ZEMS-TKN-LOGTIME</td>
<td>token-type ZSPI-TYP-TIMESTAMP.</td>
</tr>
<tr>
<td>ZEMS-TKN-PIN</td>
<td>token-type ZSPI-TYP-UINT.</td>
</tr>
<tr>
<td>ZEMS-TKN-SYSTEM</td>
<td>token-type ZSPI-TYP-UINT.</td>
</tr>
<tr>
<td>ZEMS-TKN-USERID</td>
<td>token-type ZSPI-TYP-BYTE-PAIR.</td>
</tr>
<tr>
<td>ZTCI-TKN-SEV</td>
<td>token-type ZSPI-TYP-ENUM.</td>
</tr>
<tr>
<td>ZTCI-TKN-IF-UNIT</td>
<td>token-type ZSPI-TYP-INT.</td>
</tr>
<tr>
<td>ZTCI-TKN-PROTO</td>
<td>token-type ZSPI-TYP-INT.</td>
</tr>
</tbody>
</table>

### Event-Message Text

EN unit: CAN'T HANDLE AF proto

### Unconditional Tokens

**ZSPI-TKN-SSID**

is a header token that specifies the subsystem ID. The value used for the TCP/IP subsystem is ZTCI-VAL-SSID.

**ZEMS-TKN-EVENTNUMBER**

is a header token that contains the event number. Its value is ZTCI-EVT-ADDR-ERR (22).

**ZEMS-TKN-SUBJECT-MARK**

is the standard EMS token that always immediately precedes the subject token here, ZCOM-TKN-SUBJ-PROC.

**ZCOM-TKN-SUBJ-PROC**

contains the name of the TCP/IP subsystem that is the subject of this event message.
ZEMS-TKN-EMPHASIS

is a header token that contains the value ZSPI-VAL-TRUE, indicating that this is a
critical event.

(ZEMS-TKN-) CONSOLE-PRINT, CPU, CRTPID, GENTIME, LOGTIME, PIN,
SYSTEM, and USERID

are automatically placed in the header when the EMSINIT procedure is called and
the event message buffer is initialized. For a description of these tokens, see
Section 5, Common Definitions. For details on any of the EMS tokens, see the
EMS Manual.

ZTCI-TKN-SEV

contains the value ZTCI-VAL-SEV-WARN. This value indicates that
ZTCI-EVT-ADDR-ERR is only a warning message. The process continues to
function, but some functionality has been lost; for example, the network may be
going down, or a device accessed by the TCP/IP subsystem may be experiencing
effects.

ZTCI-TKN-IF-UNIT

contains the unit number of the interface being used.

ZTCI-TKN-PROTO

contains the protocol number.

Event-Message Text

In the text message, unit corresponds to the value in ZTCI-TKN-IF-UNIT and proto
corresponds to the value in ZTCI-TKN-PROTO.

Cause. Wrong protocol was used. The subsystem only supports AF_INET.

Effect. User receives the error EAFNOSUPPORT.

Recovery. Application must reissue the request using a valid protocol.
28: ZTCI-EVT-OUT-OF-MBUF

There is no buffer space for the request.

This event is critical. It is reported to the EMS collector ($0).

### Unconditional Tokens

<table>
<thead>
<tr>
<th>Token Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZSPI-TKN-SSID</td>
<td>is a header token that specifies the subsystem ID. The value used for the TCP/IP subsystem is ZTCI-VAL-SSID.</td>
</tr>
<tr>
<td>ZEMS-TKN-EVENTNUMBER</td>
<td>is a header token that contains the event number. Its value is ZTCI-EVT-ADDR-ERR (22).</td>
</tr>
<tr>
<td>ZEMS-TKN-SUBJECT-MARK</td>
<td>is the standard EMS token that always immediately precedes the subject token here, ZCOM-TKN-SUBJ-PROC.</td>
</tr>
<tr>
<td>ZCOM-TKN-SUBJ-PROC</td>
<td>contains the name of the TCP/IP subsystem that is the subject of this event message.</td>
</tr>
<tr>
<td>ZEMS-TKN-EMPHASIS</td>
<td>is a header token that contains the value ZSPI-VAL-TRUE, indicating that this is a critical event.</td>
</tr>
</tbody>
</table>

### Event-Message Text

IN_CKSUM: OUT OF MBUFS..
are automatically placed in the header when the EMSINIT procedure is called and the event message buffer is initialized. For a description of these tokens, see Section 5, Common Definitions. For details on any of the EMS tokens, see the EMS Manual.

ZTCI-TKN-SEV contains the value ZTCI-VAL-SEV-WARN. This value indicates that ZTCI-EVT-ADDR-ERR is only a warning message. The process continues to function, but some functionality has been lost; for example, the network may be going down, or a device accessed by the TCP/IP subsystem may be experiencing errors.

Cause. TCP/IP could not get enough buffer space for the request.

Effect. The TCP/IP statistics buffer for bad checksum is incremented.

Recovery. The TCP/IP protocol handles the recovery.

29: ZTCI-EVT-TRACE-STOP-ERR

The DSM trace stop failed.

This event is critical. It is reported to the EMS collector ($0).

### Unconditional Tokens

<table>
<thead>
<tr>
<th>Token</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZSPI-TKN-SSID</td>
<td>token-type ZSPI-TYP-SSID.</td>
</tr>
<tr>
<td>ZEMS-TKN-EVENTNUMBER</td>
<td>token-type ZSPI-TYP-ENUM.</td>
</tr>
<tr>
<td>ZEMS-TKN-SUBJECT-MARK</td>
<td>token-type ZSPI-TYP-MARK.</td>
</tr>
<tr>
<td>ZCOM-TKN-SUBJ-PROC</td>
<td>token-type ZSPI-TYP-STRING.</td>
</tr>
<tr>
<td>ZEMS-TKN-EMPHASIS</td>
<td>token-type ZSPI-TYP-BOOLEAN.</td>
</tr>
<tr>
<td>ZEMS-TKN-CONSOLE-PRINT</td>
<td>token-type ZSPI-TYP-BOOLEAN.</td>
</tr>
<tr>
<td>ZEMS-TKN-CPU</td>
<td>token-type ZSPI-TYP-UINT.</td>
</tr>
<tr>
<td>ZEMS-TKN-CRTPID</td>
<td>token-type ZSPI-TYP-CRTPID.</td>
</tr>
<tr>
<td>ZEMS-TKN-GENTIME</td>
<td>token-type ZSPI-TYP-TIMESTAMP.</td>
</tr>
<tr>
<td>ZEMS-TKN-LOGTIME</td>
<td>token-type ZSPI-TYP-TIMESTAMP.</td>
</tr>
<tr>
<td>ZEMS-TKN-PIN</td>
<td>token-type ZSPI-TYP-UINT.</td>
</tr>
<tr>
<td>ZEMS-TKN-SYSTEM</td>
<td>token-type ZSPI-TYP-UINT.</td>
</tr>
<tr>
<td>ZEMS-TKN-USERID</td>
<td>token-type ZSPI-TYP-BYTE-PAIR.</td>
</tr>
<tr>
<td>ZTCI-TKN-SEV</td>
<td>token-type ZSPI-TYP-ENUM.</td>
</tr>
<tr>
<td>ZTCI-TKN-ERR-NUM</td>
<td>token-type ZSPI-TYP-INT.</td>
</tr>
</tbody>
</table>

### Event-Message Text

SCPTRACE-STOP error err-num.
Unconditional Tokens

**ZSPI-TKN-SSID**

is a header token that specifies the subsystem ID. The value used for the TCP/IP subsystem is ZTCI-VAL-SSID.

**ZEMS-TKN-EVENTNUMBER**

is a header token that contains the event number. Its value is ZTCI-EVT-ADDR-ERR (22).

**ZEMS-TKN-SUBJECT-MARK**

is the standard EMS token that always immediately precedes the subject token here, ZCOM-TKN-SUBJ-PROC.

**ZCOM-TKN-SUBJ-PROC**

contains the name of the TCP/IP subsystem that is the subject of this event message.

**ZEMS-TKN-EMPHASIS**

is a header token that contains the value ZSPI-VAL-TRUE, indicating that this is a critical event.

**(ZEMS-TKN-) CONSOLE-PRINT, CPU, CRTPID, GENTIME, LOGTIME, PIN, SYSTEM, and USERID**

are automatically placed in the header when the EMSINIT procedure is called and the event message buffer is initialized. For a description of these tokens, see Section 5, Common Definitions. For details on any of the EMS tokens, see the EMS Manual.

**ZTCI-TKN-SEV**

contains the value ZTCI-VAL-SEV-WARN. This value indicates that ZTCI-EVT-ADDR-ERR is only a warning message. The process continues to function, but some functionality has been lost; for example, the network may be going down, or a device accessed by the TCP/IP subsystem may be experiencing errors.

**ZTCI-TKN-ERR-NUM**

contains the error that has occurred.
Event-Message Text

In the text message, \textit{err-num} corresponds to the value in ZTCI-TKN-ERR-NUM.

\textbf{Cause.} A TCP/IP call to DSM_TRACE_ failed.

\textbf{Effect.} There is a problem in stopping the trace.

\textbf{Recovery.} The operation may be retried. Recovery depends on the trace collector.

30: ZTCI-EVT-SSINIT-FAILED1

There was an SPI initialization failure.

This event is critical. It is reported to the EMS collector ($0$).

<table>
<thead>
<tr>
<th>Unconditional Tokens</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZSPI-TKN-SSID</td>
</tr>
<tr>
<td>ZEMS-TKN-EVENTNUMBER</td>
</tr>
<tr>
<td>ZEMS-TKN-SUBJECT-MARK</td>
</tr>
<tr>
<td>ZCOM-TKN-SUBJ-PROC</td>
</tr>
<tr>
<td>ZEMS-TKN-EMPHASIS</td>
</tr>
<tr>
<td>ZEMS-TKN-CONSOLE-PRINT</td>
</tr>
<tr>
<td>ZEMS-TKN-CPU</td>
</tr>
<tr>
<td>ZEMS-TKN-CRTPID</td>
</tr>
<tr>
<td>ZEMS-TKN-GENTIME</td>
</tr>
<tr>
<td>ZEMS-TKN-LOGTIME</td>
</tr>
<tr>
<td>ZEMS-TKN-PIN</td>
</tr>
<tr>
<td>ZEMS-TKN-SYSTEM</td>
</tr>
<tr>
<td>ZEMS-TKN-USERID</td>
</tr>
<tr>
<td>ZTCI-TKN-SEV</td>
</tr>
<tr>
<td>ZTCI-TKN-ERR-NUM</td>
</tr>
</tbody>
</table>

\textbf{Event-Message Text}

\textit{SN_INIT_BUF: SSINIT FAILED ERR err-num}.

\textbf{Unconditional Tokens}

\textbf{ZSPI-TKN-SSID}

is a header token that specifies the subsystem ID. The value used for the TCP/IP subsystem is ZTCI-VAL-SSID.

\textbf{ZEMS-TKN-EVENTNUMBER}

is a header token that contains the event number. Its value is ZTCI-EVT-ADDR-ERR (22).
ZEMS-TKN-SUBJECT-MARK  
is the standard EMS token that always immediately precedes the subject token here, ZCOM-TKN-SUBJ-PROC.

ZCOM-TKN-SUBJ-PROC  
contains the name of the TCP/IP subsystem that is the subject of this event message.

ZEMS-TKN-EMPHASIS  
is a header token that contains the value ZSPI-VAL-TRUE, indicating that this is a critical event.

(ZEMS-TKN-) CONSOLE-PRINT, CPU, CRTPID, GENTIME, LOGTIME, PIN, SYSTEM, and USERID  
are automatically placed in the header when the EMSINIT procedure is called and the event message buffer is initialized. For a description of these tokens, see Section 5, Common Definitions. For details on any of the EMS tokens, see the EMS Manual.

ZTCI-TKN-SEV  
contains the value ZTCI-VAL-SEV-WARN. This value indicates that ZTCI-EVT-ADDR-ERR is only a warning message. The process continues to function, but some functionality has been lost; for example, the network may be going down, or a device accessed by the TCP/IP subsystem may be experiencing errors.

ZTCI-TKN-ERR-NUM  
contains the error that has occurred.

Event-Message Text  
In the text message, \textit{err-num} corresponds to the value in ZTCI-TKN-ERR-NUM.

\textbf{Cause}. There is an SCF-interface related problem.

\textbf{Effect}. An error is returned to the user.

\textbf{Recovery}. Reissue the SCF command.
31: ZTCI-EVT-NEW-CL-MBUF

There is no buffer space for the request.

This event is critical. It is reported to the EMS collector ($0).

**Unconditional Tokens**

<table>
<thead>
<tr>
<th>Token Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZSPI-TKN-SSID</td>
<td>is a header token that specifies the subsystem ID. The value used for the TCP/IP subsystem is ZTCI-VAL-SSID.</td>
</tr>
<tr>
<td>ZEMS-TKN-EVENTNUMBER</td>
<td>is a header token that contains the event number. Its value is ZTCI-EVT-ADDR-ERR (22).</td>
</tr>
<tr>
<td>ZEMS-TKN-SUBJECT-MARK</td>
<td>is the standard EMS token that always immediately precedes the subject token here, ZCOM-TKN-SUBJ-PROC.</td>
</tr>
<tr>
<td>ZCOM-TKN-SUBJ-PROC</td>
<td>contains the name of the TCP/IP subsystem that is the subject of this event message.</td>
</tr>
<tr>
<td>ZEMS-TKN-EMPHASIS</td>
<td>is a header token that contains the value ZSPI-VAL-TRUE, indicating that this is a critical event.</td>
</tr>
</tbody>
</table>

**Event-Message Text**

ALLOCATING NEW CLUSTER MBUFS PANIC.
Unconditional Tokens

<table>
<thead>
<tr>
<th>Token</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZSPI-TKN-SSID</td>
<td>ZSPI-TYP-SSID</td>
</tr>
<tr>
<td>ZEMS-TKN-EVENTNUMBER</td>
<td>ZSPI-TYP-ENUM</td>
</tr>
<tr>
<td>ZEMS-TKN-SUBJECT-MARK</td>
<td>ZSPI-TYP-MARK</td>
</tr>
<tr>
<td>ZCOM-TKN-SUBJ-PROC</td>
<td>ZSPI-TYP-STRING</td>
</tr>
<tr>
<td>ZEMS-TKN-EMPHASIS</td>
<td>ZSPI-TYP-BOOLEAN</td>
</tr>
<tr>
<td>ZEMS-TKN-CONSOLE-PRINT</td>
<td>ZSPI-TYP-BOOLEAN</td>
</tr>
<tr>
<td>ZEMS-TKN-CPU</td>
<td>ZSPI-TYP-UINT</td>
</tr>
<tr>
<td>ZEMS-TKN-CRTPID</td>
<td>ZSPI-TYP-CRTPID</td>
</tr>
<tr>
<td>ZEMS-TKN-GENTIME</td>
<td>ZSPI-TYP-TIMESTAMP</td>
</tr>
<tr>
<td>ZEMS-TKN-LOGTIME</td>
<td>ZSPI-TYP-TIMESTAMP</td>
</tr>
<tr>
<td>ZEMS-TKN-PIN</td>
<td>ZSPI-TYP-UINT</td>
</tr>
<tr>
<td>ZEMS-TKN-SYSTEM</td>
<td>ZSPI-TYP-UINT</td>
</tr>
<tr>
<td>ZEMS-TKN-USERID</td>
<td>ZSPI-TYP-BYTE-PAIR</td>
</tr>
<tr>
<td>ZTCI-TKN-SEV</td>
<td>ZSPI-TYP-ENUM</td>
</tr>
<tr>
<td>ZTCI-TKN-SOCK</td>
<td>ZSPI-TYP-INT</td>
</tr>
</tbody>
</table>

Event-Message Text

SCHED_SOCKET: RUN_SOCK NON-ZERO socket
Unconditional Tokens

**ZSPI-TKN-SSID**

is a header token that specifies the subsystem ID. The value used for the TCP/IP subsystem is ZTCI-VAL-SSID.

**ZEMS-TKN-EVENTNUMBER**

is a header token that contains the event number. Its value is ZTCI-EVT-ADDR-ERR (22).

**ZEMS-TKN-SUBJECT-MARK**

is the standard EMS token that always immediately precedes the subject token here, ZCOM-TKN-SUBJ-PROC.

**ZCOM-TKN-SUBJ-PROC**

contains the name of the TCP/IP subsystem that is the subject of this event message.

**ZEMS-TKN-EMPHASIS**

is a header token that contains the value ZSPI-VAL-TRUE, indicating that this is a critical event.

**(ZEMS-TKN-) CONSOLE-PRINT, CPU, CRTPID, GENTIME, LOGTIME, PIN, SYSTEM, and USERID**

are automatically placed in the header when the EMSINIT procedure is called and the event message buffer is initialized. For a description of these tokens, see Section 5, Common Definitions. For details on any of the EMS tokens, see the EMS Manual.

**ZTCI-TKN-SEV**

contains the value ZTCI-VAL-SEV-WARN. This value indicates that ZTCI-EVT-ADDR-ERR is only a warning message. The process continues to function, but some functionality has been lost; for example, the network may be going down, or a device accessed by the TCP/IP subsystem may be experiencing errors.

**ZTCI-TKN-SOCK**

contains the current value of the run_sock variable.

Event-Message Text

In the text message, socket corresponds to the value in ZTCI-TKN-SOCK.

**Cause.** After scheduling all sockets, TCP/IP still had a socket left in a queue.
**Effect.** TCP/IP will abend.

**Recovery.** Restart the TCP/IP process.

### 33: ZTCI-EVT-NO-BACKUP

TCP/IP no longer has a backup process.

This event is critical. It is reported to the EMS collector ($0).

<table>
<thead>
<tr>
<th>Unconditional Tokens</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ZSPI-TKN-SSID</strong></td>
</tr>
<tr>
<td>token-type ZSPI-TYP-SSID.</td>
</tr>
<tr>
<td><strong>ZEMS-TKN-EVENTNUMBER</strong></td>
</tr>
<tr>
<td>token-type ZSPI-TYP-ENUM.</td>
</tr>
<tr>
<td><strong>ZEMS-TKN-SUBJECT-MARK</strong></td>
</tr>
<tr>
<td>token-type ZSPI-TYP-MARK.</td>
</tr>
<tr>
<td><strong>ZCOM-TKN-SUBJ-PROC</strong></td>
</tr>
<tr>
<td>token-type ZSPI-TYP-STRING.</td>
</tr>
<tr>
<td><strong>ZEMS-TKN-EMPHASIS</strong></td>
</tr>
<tr>
<td>token-type ZSPI-TYP-BOOLEAN.</td>
</tr>
<tr>
<td><strong>ZEMS-TKN-CONSOLE-PRINT</strong></td>
</tr>
<tr>
<td>token-type ZSPI-TYP-BOOLEAN.</td>
</tr>
<tr>
<td><strong>ZEMS-TKN-CPU</strong></td>
</tr>
<tr>
<td>token-type ZSPI-TYP-UINT.</td>
</tr>
<tr>
<td><strong>ZEMS-TKN-CRTPID</strong></td>
</tr>
<tr>
<td>token-type ZSPI-TYP-CRTPID.</td>
</tr>
<tr>
<td><strong>ZEMS-TKN-GENTIME</strong></td>
</tr>
<tr>
<td>token-type ZSPI-TYP-TIMESTAMP.</td>
</tr>
<tr>
<td><strong>ZEMS-TKN-LOGTIME</strong></td>
</tr>
<tr>
<td>token-type ZSPI-TYP-TIMESTAMP.</td>
</tr>
<tr>
<td><strong>ZEMS-TKN-PIN</strong></td>
</tr>
<tr>
<td>token-type ZSPI-TYP-UINT.</td>
</tr>
<tr>
<td><strong>ZEMS-TKN-SYSTEM</strong></td>
</tr>
<tr>
<td>token-type ZSPI-TYP-UINT.</td>
</tr>
<tr>
<td><strong>ZEMS-TKN-USERID</strong></td>
</tr>
<tr>
<td>token-type ZSPI-TYP-BYTE-PAIR.</td>
</tr>
<tr>
<td><strong>ZTCI-TKN-SEV</strong></td>
</tr>
<tr>
<td>token-type ZSPI-TYP-ENUM.</td>
</tr>
<tr>
<td><strong>ZTCI-TKN-ERR-NUM</strong></td>
</tr>
<tr>
<td>token-type ZSPI-TYP-INT.</td>
</tr>
</tbody>
</table>

**Event-Message Text**

TCP/IP PROCESS HAS NO BACKUP, ERR *err-num*

### Unconditional Tokens

**ZSPI-TKN-SSID**

is a header token that specifies the subsystem ID. The value used for the TCP/IP subsystem is ZTCI-VAL-SSID.

**ZEMS-TKN-EVENTNUMBER**

is a header token that contains the event number. Its value is ZTCI-EVT-ADDR-ERR (22).

**ZEMS-TKN-SUBJECT-MARK**

is the standard EMS token that always immediately precedes the subject token here, ZCOM-TKN-SUBJ-PROC.

**ZCOM-TKN-SUBJ-PROC**

contains the name of the TCP/IP subsystem that is the subject of this event message.
ZEMS-TKN-EMPHASIS

is a header token that contains the value ZSPI-VAL-TRUE, indicating that this is a critical event.

(ZEMS-TKN-) CONSOLE-PRINT, CPU, CRTPID, GENTIME, LOGTIME, PIN, SYSTEM, and USERID

are automatically placed in the header when the EMSINIT procedure is called and the event message buffer is initialized. For a description of these tokens, see Section 5, Common Definitions. For details on any of the EMS tokens, see the EMS Manual.

ZTCI-TKN-SEV

contains the value ZTCI-VAL-SEV-WARN. This value indicates that ZTCI-EVT-ADDR-ERR is only a warning message. The process continues to function, but some functionality has been lost; for example, the network may be going down, or a device accessed by the TCP/IP subsystem may be experiencing errors.

ZTCI-TKN-ERR-NUM

contains the error that has occurred.

**Event-Message Text**

In the text message, \textit{err-num} corresponds to the value in ZTCI-TKN-ERR-NUM.

**Cause.** The processor in which the backup process is running has gone down.

**Effect.** This is an informational message only.

**Recovery.** TCP/IP will restart the backup process when the processor is reloaded.
34: ZTCI-EVT-INVALID-CPU-NUM

An invalid backup CPU was specified.

This event is critical. It is reported to the EMS collector ($0).

## Unconditional Tokens

<table>
<thead>
<tr>
<th>Token Name</th>
<th>Token Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZSPI-TKN-SSID</td>
<td>ZSPI-TYP-SSID</td>
</tr>
<tr>
<td>ZEMS-TKN-EVENTNUMBER</td>
<td>ZSPI-TYP-ENUM</td>
</tr>
<tr>
<td>ZEMS-TKN-SUBJECT-MARK</td>
<td>ZSPI-TYP-MARK</td>
</tr>
<tr>
<td>ZCOM-TKN-SUBJ-PROC</td>
<td>ZSPI-TYP-STRING</td>
</tr>
<tr>
<td>ZEMS-TKN-EMPHASIS</td>
<td>ZSPI-TYP-BOOLEAN</td>
</tr>
<tr>
<td>ZEMS-TKN-CONSOLE-PRINT</td>
<td>ZSPI-TYP-BOOLEAN</td>
</tr>
<tr>
<td>ZEMS-TKN-CPU</td>
<td>ZSPI-TYP-UINT</td>
</tr>
<tr>
<td>ZEMS-TKN-CRTPID</td>
<td>ZSPI-TYP-CRTPID</td>
</tr>
<tr>
<td>ZEMS-TKN-GENTIME</td>
<td>ZSPI-TYP-TIMESTAMP</td>
</tr>
<tr>
<td>ZEMS-TKN-LOGTIME</td>
<td>ZSPI-TYP-TIMESTAMP</td>
</tr>
<tr>
<td>ZEMS-TKN-PIN</td>
<td>ZSPI-TYP-UINT</td>
</tr>
<tr>
<td>ZEMS-TKN-SYSTEM</td>
<td>ZSPI-TYP-UINT</td>
</tr>
<tr>
<td>ZEMS-TKN-USERID</td>
<td>ZSPI-TYP-BYTE-PAIR</td>
</tr>
<tr>
<td>ZTCI-TKN-SEV</td>
<td>ZSPI-TYP-ENUM</td>
</tr>
<tr>
<td>ZTCI-TKN-ERR-NUM</td>
<td>ZSPI-TYP-INT</td>
</tr>
</tbody>
</table>

## Event-Message Text

INVALID BACKUP CPU NUMBER FOR TCP/IP PROCESS, ERR err-num

## Unconditional Tokens

**ZSPI-TKN-SSID**

is a header token that specifies the subsystem ID. The value used for the TCP/IP subsystem is ZTCI-VAL-SSID.

**ZEMS-TKN-EVENTNUMBER**

is a header token that contains the event number. Its value is ZTCI-EVT-ADDR-ERR (22).

**ZEMS-TKN-SUBJECT-MARK**

is the standard EMS token that always immediately precedes the subject token here, ZCOM-TKN-SUBJ-PROC.

**ZCOM-TKN-SUBJ-PROC**

contains the name of the TCP/IP subsystem that is the subject of this event message.
ZEMS-TKN-EMPHASIS

is a header token that contains the value ZSPI-VAL-TRUE, indicating that this is a critical event.

(ZEMS-TKN-) CONSOLE-PRINT, CPU, CRTPID, GENTIME, LOGTIME, PIN, SYSTEM, and USERID

are automatically placed in the header when the EMSINIT procedure is called and the event message buffer is initialized. For a description of these tokens, see Section 5, Common Definitions. For details on any of the EMS tokens, see the EMS Manual.

ZTCI-TKN-SEV

contains the value ZTCI-VAL-SEV-WARN. This value indicates that ZTCI-EVT-ADDR-ERR is only a warning message. The process continues to function, but some functionality has been lost; for example, the network may be going down, or a device accessed by the TCP/IP subsystem may be experiencing errors.

ZTCI-TKN-ERR-NUM

contains the error that has occurred.

**Event-Message Text**

In the text message, *err-num* corresponds to the value in ZTCI-TKN-ERR-NUM.

**Cause.** An invalid backup processor number was specified in the RUN command.

**Effect.** TCP/IP will abend.

**Recovery.** Restart the TCP/IP process using a valid backup processor number.
35: ZTCI-EVT-CANT-START-BACKUP

TCP/IP could not start a new backup process.

This event is critical. It is reported to the EMS collector ($0).

Unconditional Tokens

<table>
<thead>
<tr>
<th>Token Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZSPI-TKN-SSID</td>
<td>Token type ZSPI-TYP-SSID.</td>
</tr>
<tr>
<td>ZEMS-TKN-EVENTNUMBER</td>
<td>Token type ZSPI-TYP-ENUM.</td>
</tr>
<tr>
<td>ZEMS-TKN-SUBJECT-MARK</td>
<td>Token type ZSPI-TYP-MARK.</td>
</tr>
<tr>
<td>ZCOM-TKN-SUBJ-PROC</td>
<td>Token type ZSPI-TYP-STRING.</td>
</tr>
<tr>
<td>ZEMS-TKN-CONSOLE-PRINT</td>
<td>Token type ZSPI-TYP-BOOLEAN.</td>
</tr>
<tr>
<td>ZEMS-TKN-CPU</td>
<td>Token type ZSPI-TYP-UINT.</td>
</tr>
<tr>
<td>ZEMS-TKN-CRTPID</td>
<td>Token type ZSPI-TYP-CRTPID.</td>
</tr>
<tr>
<td>ZEMS-TKN-GENTIME</td>
<td>Token type ZSPI-TYP-TIMESTAMP.</td>
</tr>
<tr>
<td>ZEMS-TKN-LOGTIME</td>
<td>Token type ZSPI-TYP-TIMESTAMP.</td>
</tr>
<tr>
<td>ZEMS-TKN-PIN</td>
<td>Token type ZSPI-TYP-UINT.</td>
</tr>
<tr>
<td>ZEMS-TKN-SYSTEM</td>
<td>Token type ZSPI-TYP-UINT.</td>
</tr>
<tr>
<td>ZEMS-TKN-USERID</td>
<td>Token type ZSPI-TYP-BYTE-PAIR.</td>
</tr>
<tr>
<td>ZTCI-TKN-SEV</td>
<td>Token type ZSPI-TYP-ENUM.</td>
</tr>
<tr>
<td>ZTCI-TKN-ERR-NUM</td>
<td>Token type ZSPI-TYP-INT.</td>
</tr>
</tbody>
</table>

Event-Message Text

CAN NOT START TCP/IP BACKUP PROCESS ERR err-num

Unconditional Tokens

ZSPI-TKN-SSID

is a header token that specifies the subsystem ID. The value used for the TCP/IP subsystem is ZTCI-VAL-SSID.

ZEMS-TKN-EVENTNUMBER

is a header token that contains the event number. Its value is ZTCI-EVT-ADDR-ERR (22).

ZEMS-TKN-SUBJECT-MARK

is the standard EMS token that always immediately precedes the subject token here, ZCOM-TKN-SUBJ-PROC.

ZCOM-TKN-SUBJ-PROC

contains the name of the TCP/IP subsystem that is the subject of this event message.
**ZEMS-TKN-EMPHASIS**

is a header token that contains the value ZSPI-VAL-TRUE, indicating that this is a critical event.

**(ZEMS-TKN-) CONSOLE-PRINT, CPU, CRTPID, GENTIME, LOGTIME, PIN, SYSTEM, and USERID**

are automatically placed in the header when the EMSINIT procedure is called and the event message buffer is initialized. For a description of these tokens, see Section 5, Common Definitions. For details on any of the EMS tokens, see the EMS Manual.

**ZTCI-TKN-SEV**

contains the value ZTCI-VAL-SEV-WARN. This value indicates that ZTCI-EVT-ADDR-ERR is only a warning message. The process continues to function, but some functionality has been lost; for example, the network may be going down, or a device accessed by the TCP/IP subsystem may be experiencing errors.

**ZTCI-TKN-ERR-NUM**

contains the error that has occurred.

**Event-Message Text**

In the text message, \textit{err-num} corresponds to the value in ZTCI-TKN-ERR-NUM.

**Cause.** The TCP/IP process could not start its backup process.

**Effect.** This message is for informational purposes only.

**Recovery.** The TCP/IP process will try to restart its backup process, based on an ever-increasing delay with a maximum value of 10 minutes.
**36: ZTCI-EVT-BACKUP-IS DEAD**

The backup process has abended.

This event is critical. It is reported to the EMS collector ($0).

<table>
<thead>
<tr>
<th>Unconditional Tokens</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZSPI-TKN-SSID</td>
</tr>
<tr>
<td>ZEMS-TKN-EVENTNUMBER</td>
</tr>
<tr>
<td>ZEMS-TKN-SUBJECT-MARK</td>
</tr>
<tr>
<td>ZCOM-TKN-SUBJ-PROC</td>
</tr>
<tr>
<td>ZEMS-TKN-CONSOLE-PRINT</td>
</tr>
<tr>
<td>ZEMS-TKN-CPU</td>
</tr>
<tr>
<td>ZEMS-TKN-CRTPID</td>
</tr>
<tr>
<td>ZEMS-TKN-GENTIME</td>
</tr>
<tr>
<td>ZEMS-TKN-LOGTIME</td>
</tr>
<tr>
<td>ZEMS-TKN-PIN</td>
</tr>
<tr>
<td>ZEMS-TKN-SYSTEM</td>
</tr>
<tr>
<td>ZEMS-TKN-USERID</td>
</tr>
<tr>
<td>ZTCI-TKN-SEV</td>
</tr>
<tr>
<td>ZTCI-TKN-SIZE</td>
</tr>
</tbody>
</table>

**Event-Message Text**

TCP/IP BACKUP PROCESS ABENDED, ATTEMPTING RECOVERY IN *number* SECONDS

**Unconditional Tokens**

**ZSPI-TKN-SSID**

is a header token that specifies the subsystem ID. The value used for the TCP/IP subsystem is ZTCI-VAL-SSID.

**ZEMS-TKN-EVENTNUMBER**

is a header token that contains the event number. Its value is ZTCI-EVT-ADDR-ERR (22).

**ZEMS-TKN-SUBJECT-MARK**

is the standard EMS token that always immediately precedes the subject token here, ZCOM-TKN-SUBJ-PROC.

**ZCOM-TKN-SUBJ-PROC**

contains the name of the TCP/IP subsystem that is the subject of this event message.
ZEMS-TKN-EMPHASIS

is a header token that contains the value ZSPI-VAL-TRUE, indicating that this is a critical event.

(ZEMS-TKN-) CONSOLE-PRINT, CPU, CRTPID, GENTIME, LOGTIME, PIN, SYSTEM, and USERID

are automatically placed in the header when the EMSINIT procedure is called and the event message buffer is initialized. For a description of these tokens, see Section 5, Common Definitions. For details on any of the EMS tokens, see the EMS Manual.

ZTCI-TKN-SEV

contains the value ZTCI-VAL-SEV-WARN. This value indicates that ZTCI-EVT-ADDR-ERR is only a warning message. The process continues to function, but some functionality has been lost; for example, the network may be going down, or a device accessed by the TCP/IP subsystem may be experiencing errors.

ZTCI-TKN-SIZE

contains the time, in seconds, after which the TCP/IP process will attempt to restart its backup process.

Event-Message Text

In the text message, number corresponds to the value in ZTCI-TKN-SIZE.

Cause. The TCP/IP backup process abended due to internal inconsistency.

Effect. This message is for informational purposes only.

Recovery. The TCP/IP process will attempt recovery.
TCP/IP has successfully started its backup process.

This event is critical. It is reported to the EMS collector ($0).

### Unconditional Tokens

- **ZSPI-TKN-SSID**
  - token-type: ZSPI-TYP-SSID.
  - is a header token that specifies the subsystem ID. The value used for the TCP/IP subsystem is ZTCI-VAL-SSID.

- **ZEMS-TKN-EVENTNUMBER**
  - token-type: ZSPI-TYP-ENUM.
  - is a header token that contains the event number. Its value is ZTCI-EVT-ADDR-ERR (22).

- **ZEMS-TKN-SUBJECT-MARK**
  - token-type: ZSPI-TYP-MARK.
  - is the standard EMS token that always immediately precedes the subject token here, ZCOM-TKN-SUBJ-PROC.

- **ZCOM-TKN-SUBJ-PROC**
  - token-type: ZSPI-TYP-STRING.
  - contains the name of the TCP/IP subsystem that is the subject of this event message.

- **ZEMS-TKN-EMPHASIS**
  - token-type: ZSPI-TYP-BOOLEAN.
  - is a header token that contains the value ZSPI-VAL-TRUE, indicating that this is a critical event.

### Event-Message Text

TCP/IP BACKUP PROCESS STARTED
Event Management

38: ZTCI-EVT-CONN-EST AB-GATE

are automatically placed in the header when the EMSINIT procedure is called and the event message buffer is initialized. For a description of these tokens, see Section 5, Common Definitions. For details on any of the EMS tokens, see the EMS Manual.

ZTCI-TKN-SEV
contains the value ZTCI-VAL-SEV-WARN. This value indicates that ZTCI-EVT-ADDR-ERR is only a warning message. The process continues to function, but some functionality has been lost; for example, the network may be going down, or a device accessed by the TCP/IP subsystem may be experiencing errors.

Cause. The TCP/IP process successfully started its backup process.

Effect. This message is for informational purposes only.

38: ZTCI-EVT-CONN-ESTAB-GATE

An X.25 connection was established using a gateway.

This event is critical. It is reported to the EMS collector ($0).

Unconditional Tokens

<table>
<thead>
<tr>
<th>Token Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZSPI-TKN-SSID</td>
<td>token-type ZSPI-TYP-SSID.</td>
</tr>
<tr>
<td>ZEMS-TKN-EVENTNUMBER</td>
<td>token-type ZSPI-TYP-ENUM.</td>
</tr>
<tr>
<td>ZEMS-TKN-SUBJECT-MARK</td>
<td>token-type ZSPI-TYP-MARK.</td>
</tr>
<tr>
<td>ZCOM-TKN-SUBJ-PROC</td>
<td>token-type ZSPI-TYP-STRING.</td>
</tr>
<tr>
<td>ZEMS-TKN-EMPHASIS</td>
<td>token-type ZSPI-TYP-BOOLEAN.</td>
</tr>
<tr>
<td>ZEMS-TKN-CONSOLE-PRINT</td>
<td>token-type ZSPI-TYP-BOOLEAN.</td>
</tr>
<tr>
<td>ZEMS-TKN-CPU</td>
<td>token-type ZSPI-TYP-UINT.</td>
</tr>
<tr>
<td>ZEMS-TKN-CRTPID</td>
<td>token-type ZSPI-TYP-CRTPID.</td>
</tr>
<tr>
<td>ZEMS-TKN-GENTIME</td>
<td>token-type ZSPI-TYP-TIMESTAMP.</td>
</tr>
<tr>
<td>ZEMS-TKN-LOGTIME</td>
<td>token-type ZSPI-TYP-TIMESTAMP.</td>
</tr>
<tr>
<td>ZEMS-TKN-PIN</td>
<td>token-type ZSPI-TYP-UINT.</td>
</tr>
<tr>
<td>ZEMS-TKN-SYSTEM</td>
<td>token-type ZSPI-TYP-UINT.</td>
</tr>
<tr>
<td>ZEMS-TKN-USERID</td>
<td>token-type ZSPI-TYP-BYTE-PAIR.</td>
</tr>
<tr>
<td>ZTCI-TKN-SEV</td>
<td>token-type ZSPI-TYP-ENUM.</td>
</tr>
<tr>
<td>ZTCI-TKN-IP-ADDR-1</td>
<td>token-type ZSPI-TYP-INT.</td>
</tr>
<tr>
<td>ZTCI-TKN-IP-ADDR-2</td>
<td>token-type ZSPI-TYP-INT.</td>
</tr>
<tr>
<td>ZTCI-TKN-IP-ADDR-3</td>
<td>token-type ZSPI-TYP-INT.</td>
</tr>
<tr>
<td>ZTCI-TKN-IP-ADDR-4</td>
<td>token-type ZSPI-TYP-INT.</td>
</tr>
<tr>
<td>ZTCI-TKN-IP-ADDR-5</td>
<td>token-type ZSPI-TYP-INT.</td>
</tr>
<tr>
<td>ZTCI-TKN-IP-ADDR-6</td>
<td>token-type ZSPI-TYP-INT.</td>
</tr>
<tr>
<td>ZTCI-TKN-IP-ADDR-7</td>
<td>token-type ZSPI-TYP-INT.</td>
</tr>
<tr>
<td>ZTCI-TKN-IP-ADDR-8</td>
<td>token-type ZSPI-TYP-INT.</td>
</tr>
</tbody>
</table>
Unconditional Tokens

**ZSPI-TKN-SSID**

is a header token that specifies the subsystem ID. The value used for the TCP/IP subsystem is ZTCI-VAL-SSID.

**ZEMS-TKN-EVENTNUMBER**

is a header token that contains the event number. Its value is ZTCI-EVT-ADDR-ERR (22).

**ZEMS-TKN-SUBJECT-MARK**

is the standard EMS token that always immediately precedes the subject token here, ZCOM-TKN-SUBJ-PROC.

**ZCOM-TKN-SUBJ-PROC**

contains the name of the TCP/IP subsystem that is the subject of this event message.

**ZEMS-TKN-EMPHASIS**

is a header token that contains the value ZSPI-VAL-TRUE, indicating that this is a critical event.

**(ZEMS-TKN-) CONSOLE-PRINT, CPU, CRTPID, GENTIME, LOGTIME, PIN, SYSTEM, and USERID**

are automatically placed in the header when the EMSINIT procedure is called and the event message buffer is initialized. For a description of these tokens, see Section 5, Common Definitions. For details on any of the EMS tokens, see the EMS Manual.

**ZTCI-TKN-SEV**

contains the value ZTCI-VAL-SEV-WARN. This value indicates that ZTCI-EVT-ADDR-ERR is only a warning message. The process continues to function, but some functionality has been lost; for example, the network may be going down, or a device accessed by the TCP/IP subsystem may be experiencing errors.

**ZTCI-TKN-IP-ADDR-1 through ZTCI-TKN-IP-ADDR-4**

contains the IP address of the source.
ZTCI-TKN-IP-ADDR-5 through ZTCI-TKN-IP-ADDR-8 contains the IP address of the gateway being used.

**Event-Message Text**


**Cause.** An X.25 connection was established to a remote TCP/IP host.

**Effect.** This message is for informational purposes only.

### 39: ZTCI-EVT-CHKPT-FAILED

A checkpoint to the backup process failed.

This event is critical. It is reported to the EMS collector ($0).

<table>
<thead>
<tr>
<th><strong>Unconditional Tokens</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>ZSPI-TKN-SSID</td>
</tr>
<tr>
<td>ZEMS-TKN-EVENTNUMBER</td>
</tr>
<tr>
<td>ZEMS-TKN-SUBJECT-MARK</td>
</tr>
<tr>
<td>ZCOM-TKN-SUBJ-PROC</td>
</tr>
<tr>
<td>ZEMS-TKN-EMPHASIS</td>
</tr>
<tr>
<td>ZEMS-TKN-CONSOLE-PRINT</td>
</tr>
<tr>
<td>ZEMS-TKN-CPU</td>
</tr>
<tr>
<td>ZEMS-TKN-CRTPID</td>
</tr>
<tr>
<td>ZEMS-TKN-GENTIME</td>
</tr>
<tr>
<td>ZEMS-TKN-LOGTIME</td>
</tr>
<tr>
<td>ZEMS-TKN-PIN</td>
</tr>
<tr>
<td>ZEMS-TKN-SYSTEM</td>
</tr>
<tr>
<td>ZEMS-TKN-USERID</td>
</tr>
<tr>
<td>ZTCI-TKN-SEV</td>
</tr>
<tr>
<td>ZTCI-TKN-ERR-NUM</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Event-Message Text</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>CHECKPOINT FAILED ERROR <em>err-num</em>, ATTEMPTING TO RESTART BACKUP PROCESS</td>
</tr>
</tbody>
</table>

**Unconditional Tokens**

ZSPI-TKN-SSID

is a header token that specifies the subsystem ID. The value used for the TCP/IP subsystem is ZTCI-VAL-SSID.
ZEMS-TKN-EVENTNUMBER

is a header token that contains the event number. Its value is ZTCI-EVT-ADDR-ERR (22).

ZEMS-TKN-SUBJECT-MARK

is the standard EMS token that always immediately precedes the subject token here, ZCOM-TKN-SUBJ-PROC.

ZCOM-TKN-SUBJ-PROC

contains the name of the TCP/IP subsystem that is the subject of this event message.

ZEMS-TKN-EMPHASIS

is a header token that contains the value ZSPI-VAL-TRUE, indicating that this is a critical event.

(ZEMS-TKN-) CONSOLE-PRINT, CPU, CRTPID, GENTIME, LOGTIME, PIN, SYSTEM, and USERID

are automatically placed in the header when the EMSINIT procedure is called and the event message buffer is initialized. For a description of these tokens, see Section 5, Common Definitions. For details on any of the EMS tokens, see the EMS Manual.

ZTCI-TKN-SEV

contains the value ZTCI-VAL-SEV-WARN. This value indicates that ZTCI-EVT-ADDR-ERR is only a warning message. The process continues to function, but some functionality has been lost; for example, the network may be going down, or a device accessed by the TCP/IP subsystem may be experiencing errors.

ZTCI-TKN-ERR-NUM

contains the error that has occurred.

Event-Message Text

In the text message, err-num corresponds to the value in ZTCI-TKN-ERR-NUM.

Cause. A TCP/IP process checkpoint failed.

Effect. The backup process is no longer considered operational.

Recovery. The TCP/IP process will attempt to restart the backup process.
40: ZTCI-EVT-NO-BACKUP-CPU

No backup processor was specified.

This event is critical. It is reported to the EMS collector ($0).

<table>
<thead>
<tr>
<th>Unconditional Tokens</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZSPI-TKN-SSID</td>
</tr>
<tr>
<td>ZEMS-TKN-EVENTNUMBER</td>
</tr>
<tr>
<td>ZEMS-TKN-SUBJECT-MARK</td>
</tr>
<tr>
<td>ZCOM-TKN-SUBJ-PROC</td>
</tr>
<tr>
<td>ZEMS-TKN-EMPHASIS</td>
</tr>
<tr>
<td>ZEMS-TKN-CONSOLE-PRINT</td>
</tr>
<tr>
<td>ZEMS-TKN-CPU</td>
</tr>
<tr>
<td>ZEMS-TKN-CRTPID</td>
</tr>
<tr>
<td>ZEMS-TKN-GENTIME</td>
</tr>
<tr>
<td>ZEMS-TKN-LOGTIME</td>
</tr>
<tr>
<td>ZEMS-TKN-PIN</td>
</tr>
<tr>
<td>ZEMS-TKN-SYSTEM</td>
</tr>
<tr>
<td>ZEMS-TKN-USERID</td>
</tr>
<tr>
<td>ZTCI-TKN-SEV</td>
</tr>
</tbody>
</table>

Event-Message Text

NO BACKUP CPU SPECIFIED FOR TCP/IP PROCESS.

Unconditional Tokens

ZSPI-TKN-SSID

is a header token that specifies the subsystem ID. The value used for the TCP/IP subsystem is ZTCI-VAL-SSID.

ZEMS-TKN-EVENTNUMBER

is a header token that contains the event number. Its value is ZTCI-EVT-ADDR-ERR (22).

ZEMS-TKN-SUBJECT-MARK

is the standard EMS token that always immediately precedes the subject token here, ZCOM-TKN-SUBJ-PROC.

ZCOM-TKN-SUBJ-PROC

contains the name of the TCP/IP subsystem that is the subject of this event message.

ZEMS-TKN-EMPHASIS

is a header token that contains the value ZSPI-VAL-TRUE, indicating that this is a critical event.
ZTCI-SEV contains the value ZTCI-VAL-SEV-WARN. This value indicates that ZTCI-EVT-ADDR-ERR is only a warning message. The process continues to function, but some functionality has been lost; for example, the network may be going down, or a device accessed by the TCP/IP subsystem may be experiencing errors.

**Cause.** The TCP/IP process started without a backup CPU specified.

**Effect.** This message is for informational purposes only.

### 41: ZTCI-EVT-IOP-QIO-ERR

An error occurred on the QIO driver call.

This event is critical. It is reported to the EMS collector ($0).

#### Unconditional Tokens

<table>
<thead>
<tr>
<th>Token Type</th>
<th>Token Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZSPI-TKN-SSID</td>
<td>ZSPI-TYP-SSID</td>
</tr>
<tr>
<td>ZEMS-TKN-EVENTNUMBER</td>
<td>ZSPI-TYP-ENUM</td>
</tr>
<tr>
<td>ZEMS-TKN-SUBJECT-MARK</td>
<td>ZSPI-TYP-MARK</td>
</tr>
<tr>
<td>ZCOM-TKN-SUBJ-PROC</td>
<td>ZSPI-TYP-STRING</td>
</tr>
<tr>
<td>ZEMS-TKN-EMPHASIS</td>
<td>ZSPI-TYP-BOOLEAN</td>
</tr>
<tr>
<td>ZEMS-TKN-CONSOLE-PRINT</td>
<td>ZSPI-TYP-BOOLEAN</td>
</tr>
<tr>
<td>ZEMS-TKN-CPU</td>
<td>ZSPI-TYP-UINT</td>
</tr>
<tr>
<td>ZEMS-TKN-CRTPID</td>
<td>ZSPI-TYP-CRTPID</td>
</tr>
<tr>
<td>ZEMS-TKN-GENTIME</td>
<td>ZSPI-TYP-TIMESTAMP</td>
</tr>
<tr>
<td>ZEMS-TKN-LOGTIME</td>
<td>ZSPI-TYP-TIMESTAMP</td>
</tr>
<tr>
<td>ZEMS-TKN-PIN</td>
<td>ZSPI-TYP-UINT</td>
</tr>
<tr>
<td>ZEMS-TKN-SYSTEM</td>
<td>ZSPI-TYP-UINT</td>
</tr>
<tr>
<td>ZEMS-TKN-USERID</td>
<td>ZSPI-TYP-BYTE-PAIR</td>
</tr>
<tr>
<td>ZTCI-TKN-SEV</td>
<td>ZSPI-TYP-ENUM</td>
</tr>
<tr>
<td>ZTCI-TKN-ERR-NUM</td>
<td>ZSPI-TYP-INT</td>
</tr>
<tr>
<td>ZTCI-TKN-IOPNAME</td>
<td>ZSPI-TYP-STRING</td>
</tr>
</tbody>
</table>

#### Event-Message Text

QIO ERROR err-num on IOP iop-name
Unconditional Tokens

ZSPI-TKN-SSID

is a header token that specifies the subsystem ID. The value used for the TCP/IP subsystem is ZTCI-VAL-SSID.

ZEMS-TKN-EVENTNUMBER

is a header token that contains the event number. Its value is ZTCI-EVT-ADDR-ERR (22).

ZEMS-TKN-SUBJECT-MARK

is the standard EMS token that always immediately precedes the subject token here, ZCOM-TKN-SUBJ-PROC.

ZCOM-TKN-SUBJ-PROC

contains the name of the TCP/IP subsystem that is the subject of this event message.

ZEMS-TKN-EMPHASIS

is a header token that contains the value ZSPI-VAL-TRUE, indicating that this is a critical event.

(ZEMS-TKN-) CONSOLE-PRINT, CPU, CRTPID, GENTIME, LOGTIME, PIN, SYSTEM, and USERID

are automatically placed in the header when the EMSINIT procedure is called and the event message buffer is initialized. For a description of these tokens, see Section 5, Common Definitions. For details on any of the EMS tokens, see the EMS Manual.

ZTCI-TKN-SEV

contains the value ZTCI-VAL-SEV-WARN. This value indicates that ZTCI-EVT-ADDR-ERR is only a warning message. The process continues to function, but some functionality has been lost; for example, the network may be going down, or a device accessed by the TCP/IP subsystem may be experiencing errors.

ZTCI-TKN-ERR-NUM

contains the QIO error that has occurred.

ZTCI-TKN-IOPNAME

contains the QIO process name.
Event-Message Text

In the text message, `err-num` corresponds to the value in ZTCI-TKN-ERR-NUM and `iop-name` corresponds to the value in ZTCI-TKN-IOPNAME.

**Cause.** A QIO driver error has occurred.

**Effect.** This message is for informational purposes only.

**Recovery.** The TCP/IP process will close and attempt to reopen the I/O process.

### 42: ZTCI-EVT-QIOMODE-ON

The subnet is now using QIO direct mode.

This event is critical. It is reported to the EMS collector ($0).

#### Unconditional Tokens

- **ZSPI-TKN-SSID**
  - token-type: ZSPI-TYP-SSID.
- **ZEMS-TKN-EVENTNUMBER**
  - token-type: ZSPI-TYP-ENUM.
- **ZEMS-TKN-SUBJECT-MARK**
  - token-type: ZSPI-TYP-MARK.
- **ZCOM-TKN-SUBJ-PROC**
  - token-type: ZSPI-TYP-STRING.
- **ZEMS-TKN-EMPHASIS**
  - token-type: ZSPI-TYP-BOOLEAN.
- **ZEMS-TKN-CONSOLE-PRINT**
  - token-type: ZSPI-TYP-BOOLEAN.
- **ZEMS-TKN-CPU**
  - token-type: ZSPI-TYP-UINT.
- **ZEMS-TKN-CRTPID**
  - token-type: ZSPI-TYP-CRTPID.
- **ZEMS-TKN-GENTIME**
  - token-type: ZSPI-TYP-TIMESTAMP.
- **ZEMS-TKN-LOGTIME**
  - token-type: ZSPI-TYP-TIMESTAMP.
- **ZEMS-TKN-PIN**
  - token-type: ZSPI-TYP-UINT.
- **ZEMS-TKN-SYSTEM**
  - token-type: ZSPI-TYP-UINT.
- **ZEMS-TKN-USERID**
  - token-type: ZSPI-TYP-BYTE-PAIR.
- **ZTCI-TKN-SEV**
  - token-type: ZSPI-TYP-ENUM.
- **ZTCI-TKN-TEXT**
  - token-type: ZSPI-TYP-STRING.

#### Event-Message Text

```
SUBNET subnet-name QIO DRIVER MODE ON
```

#### Unconditional Tokens

- **ZSPI-TKN-SSID**
  - is a header token that specifies the subsystem ID. The value used for the TCP/IP subsystem is ZTCI-VAL-SSID.

- **ZEMS-TKN-EVENTNUMBER**
  - is a header token that contains the event number. Its value is ZTCI-EVT-ADDR-ERR (22).
ZEMS-TKN-SUBJECT-MARK

is the standard EMS token that always immediately precedes the subject token here, ZCOM-TKN-SUBJ-PROC.

ZCOM-TKN-SUBJ-PROC

contains the name of the TCP/IP subsystem that is the subject of this event message.

ZEMS-TKN-EMPHASIS

is a header token that contains the value ZSPI-VAL-TRUE, indicating that this is a critical event.

(ZEMS-TKN-) CONSOLE-PRINT, CPU, CRTPID, GENTIME, LOGTIME, PIN, SYSTEM, and USERID

are automatically placed in the header when the EMSINIT procedure is called and the event message buffer is initialized. For a description of these tokens, see Section 5, Common Definitions. For details on any of the EMS tokens, see the EMS Manual.

ZTCI-TKN-SEV

contains the value ZTCI-VAL-SEV-WARN. This value indicates that ZTCI-EVT-ADDR-ERR is only a warning message. The process continues to function, but some functionality has been lost; for example, the network may be going down, or a device accessed by the TCP/IP subsystem may be experiencing errors.

ZTCI-TKN-TEXT

contains the name of the subnet.

Event-Message Text

In the text message, subnet-name corresponds to the value in ZTCI-TKN-TEXT.

Cause. The TCP/IP subnet has entered QIO direct mode to the controller.

Effect. This message is for informational purposes only.
Error Numbers and Error Lists

This appendix describes the error numbers and associated error lists defined by the TCP/IP subsystem (that is, the error values whose symbolic names begin with ZTCI-ERR-). These error numbers occur as values of the ZSPI-TKN-RETCODE and ZSPI-TKN-ERROR tokens.

These errors cannot occur for all commands. To determine which commands return these TCP/IP-specific errors, see the following descriptions of each of the error numbers.

For descriptions of the common data communications errors returned by the TCP/IP subsystem (that is, the error values whose symbolic names begin with ZCOM-ERR-), see the SPI Common Extensions Manual. For descriptions of the common SPI errors returned by the TCP/IP subsystem (that is, the error values whose symbolic names begin with ZSPI-ERR-), see the SPI Programming Manual.

Notation Used

For each error-list description on the pages that follow, there is a box that contains a list of the simple and extensible structured tokens that can appear in the error list. Included are the tokens ZSPI-TKN-ERRLIST and ZSPI-TKN-ENDLIST, which start and end all error lists. With the exception of ZSPI-TKN-ERRLIST and ZSPI-TKN-ENDLIST, the order of the tokens in the box is not necessarily the order in which they actually occur.

The notation used in the box for simple tokens is a shorthand version of the essential information given in the DDL TOKEN-CODE statement.

Following the box, each token in the error list, except ZSPI-TKN-ERRLIST and ZSPI-TKN-ENDLIST, is described in the same order in which it appears in the box.

The tokens ZSPI-TKN-ERRLIST and ZSPI-TKN-ENDLIST serve the same function in all error lists. They are not described in this section. Information about these tokens can be found in the SPI Programming Manual.

Error Descriptions

The error descriptions that follow are in ascending order by error number (that is, ascending order by ZTCI-ERR- values).
1: **ZTCI-ERR-INV-FNAME**

The ADD SUBNET command has been issued but cannot be executed because an invalid I/O process name has been specified for the ZIOPNAME field.

| ZSPI-TKN-ERRLIST         | token-type ZSPI-TYP-LIST. |
| ZSPI-TKN-ERROR           | token-type ZSPI-TYP-ERROR. |
| ZCOM-TKN-OBJNAME         | token-type ZSPI-TYP-STRING. |
| ZCOM-TKN-OBJTYPE         | token-type ZSPI-TYP-ENUM.  |
| ZSPI-TKN-ENDLIST         | token-type ZSPI-TYP-SSCTL. |

**Tokens**

**ZSPI-TKN-ERROR**

is the standard SPI error token described in the *SPI Programming Manual*. Its value consists of the TCP/IP subsystem ID and the error number ZTCI-ERR-INV-FNAME. This token is always present in the error list.

**ZCOM-TKN-OBJNAME**

is the object name on which the error occurred in the command. This token is present in the error list if its presence is relevant to the cause of the error; that is, if the error was detected after the subsystem made choices based on the object name, this token is present in the error list. Otherwise, this token is omitted from the list.

**ZCOM-TKN-OBJTYPE**

is the object type provided with the command on which the error occurred. Its value defines the type of object against which the command was issued. Its only possible value is ZCOM-OBJ-SUBNET.

**Recommended Action**

Reissue the ADD SUBNET command with a valid I/O process name

5: **ZTCI-ERR-ATTR-OUT-OF-RANGE**

The ALTER PROCESS command has been issued but cannot be executed because a value specified for one of the attributes is outside the valid range.

| ZSPI-TKN-ERRLIST         | token-type ZSPI-TYP-LIST. |
| ZSPI-TKN-ERROR           | token-type ZSPI-TYP-ERROR. |
| ZCOM-TKN-OBJNAME         | token-type ZSPI-TYP-STRING. |
| ZCOM-TKN-OBJTYPE         | token-type ZSPI-TYP-ENUM.  |
| ZSPI-TKN-ENDLIST         | token-type ZSPI-TYP-SSCTL. |
Tokens

ZSPI-TKN-ERROR

is the standard SPI error token described in the SPI Programming Manual. Its value consists of the TCP/IP subsystem ID and the error number ZTCI-ERR-ATTR-OUT-OF-RANGE. This token is always present in the error list.

ZCOM-TKN-OBJNAME

is the object name on which the error occurred in the command. This token is present in the error list if its presence is relevant to the cause of the error; that is, if the error was detected after the subsystem made choices based on the object name, this token is present in the error list. Otherwise, this token is omitted from the list.

ZCOM-TKN-OBJTYPE

is the object type provided with the command on which the error occurred. Its value defines the type of object against which the command was issued. Its only possible value is ZCOM-OBJ-PROC.

Recommended Action

Reissue the ALTER PROCESS command with values within the valid range.

6: ZTCI-ERR-ENET-ADDR-NOT-AVAIL

The ADD SUBNET command has been issued but cannot be executed because the TCP/IP subsystem cannot determine the Ethernet address of the subnet.

<table>
<thead>
<tr>
<th>Token</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZSPI-TKN-ERRLIST</td>
<td>ZSPI-TYP-LIST</td>
</tr>
<tr>
<td>ZSPI-TKN-ERROR</td>
<td>ZSPI-TYP-ERROR</td>
</tr>
<tr>
<td>ZCOM-TKN-OBJNAME</td>
<td>ZSPI-TYP-STRING</td>
</tr>
<tr>
<td>ZCOM-TKN-OBJTYPE</td>
<td>ZSPI-TYP-ENUM</td>
</tr>
<tr>
<td>ZSPI-TKN-ENDLIST</td>
<td>ZSPI-TYP-SSCTL</td>
</tr>
</tbody>
</table>

Tokens

ZSPI-TKN-ERROR

is the standard SPI error token described in the SPI Programming Manual. Its value consists of the TCP/IP subsystem ID and the error number ZTCI-ERR-ENET-ADDR-NOT-AVAIL. This token is always present in the error list.

ZCOM-TKN-OBJNAME

is the object name on which the error occurred in the command. This token is present in the error list if its presence is relevant to the cause of the error; that is, if the error was detected after the subsystem made choices based on the object name.
name, this token is present in the error list. Otherwise, this token is omitted from the list.

**ZCOM-TKN-OBJTYPE**

is the object type provided with the command on which the error occurred. Its value defines the type of object against which the command was issued. Its only possible value is ZCOM-OBJ-SUBNET.

**Recommended Action**

Either the subsystem is not operational or the SCP process ($ZNET) is not running. Verify that both these processes are up before reissuing the command.

**7: ZTCI-ERR-DUP-ADDR**

The ADD SUBNET or ALTER SUBNET command has been issued but cannot be executed because the IP address specified for the ZIP-ADDR field is already being used by another interface.

<table>
<thead>
<tr>
<th>Token Type</th>
<th>Token Type</th>
<th>Token Type</th>
<th>Token Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZSPI-TKN-ERRLIST</td>
<td>token-type ZSPI-TYP-LIST.</td>
<td>ZSPI-TKN-ERROR</td>
<td>token-type ZSPI-TYP-ERROR.</td>
</tr>
<tr>
<td>ZCOM-TKN-OBJNAME</td>
<td>token-type ZSPI-TYP-STRING.</td>
<td>ZCOM-TKN-OBJTYPE</td>
<td>token-type ZSPI-TYP-ENUM.</td>
</tr>
<tr>
<td>ZSPI-TKN-ENDLIST</td>
<td>token-type ZSPI-TYP-SSCTL.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Tokens**

**ZSPI-TKN-ERROR**

is the standard SPI error token described in the *SPI Programming Manual*. Its value consists of the TCP/IP subsystem ID and the error number ZTCI-ERR-DUP-ADDR. This token is always present in the error list.

**ZCOM-TKN-OBJNAME**

is the object name on which the error occurred in the command. This token is present in the error list if its presence is relevant to the cause of the error; that is, if the error was detected after the subsystem made choices based on the object name, this token is present in the error list. Otherwise, this token is omitted from the list.

**ZCOM-TKN-OBJTYPE**

is the object type provided with the command on which the error occurred. Its value defines the type of object against which the command was issued. Its only possible value is ZCOM-OBJ-SUBNET.
Recommended Action

Reissue the ADD SUBNET or ALTER SUBNET command with another IP address.

8: ZTCI-ERR-NET-UNREACH

An ADD ROUTE command has been issued but cannot be executed because the gateway specified for the route is unavailable.

<table>
<thead>
<tr>
<th>Tokens</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZSPI-TKN-ERRLIST token-type ZSPI-TYP-LIST.</td>
</tr>
<tr>
<td>ZSPI-TKN-ERROR token-type ZSPI-TYP-ERROR.</td>
</tr>
<tr>
<td>ZCOM-TKN-OBJNAME token-type ZSPI-TYP-STRING.</td>
</tr>
<tr>
<td>ZCOM-TKN-OBJTYPE token-type ZSPI-TYP-ENUM.</td>
</tr>
<tr>
<td>ZSPI-TKN-ENDLIST token-type ZSPI-TYP-SSCTL.</td>
</tr>
</tbody>
</table>

Tokens

ZSPI-TKN-ERROR

is the standard SPI error token described in the SPI Programming Manual. Its value consists of the TCP/IP subsystem ID and the error number ZTCI-ERR-NET-UNREACH. This token is always present in the error list.

ZCOM-TKN-OBJNAME

is the object name on which the error occurred in the command. This token is present in the error list if its presence is relevant to the cause of the error; that is, if the error was detected after the subsystem made choices based on the object name, this token is present in the error list. Otherwise, this token is omitted from the list.

ZCOM-TKN-OBJTYPE

is the object type provided with the command on which the error occurred. Its value defines the type of object against which the command was issued. Its only possible value is ZCOM-OBJ-ROUTE.

Recommended Action

Reissue the ADD ROUTE command with an available gateway.
10: ZTCI-ERR-SNAP-MTU-NOT-AVAIL

The ADD SUBNET command has been issued for a SNAP subnet but could not be executed because the SLSA LAN interface did not report the frame size that could be sent over the line.

<table>
<thead>
<tr>
<th>Tokens</th>
<th>Token Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZSPI-TKN-ERRLIST</td>
<td>ZSPI-TYP-LIST</td>
</tr>
<tr>
<td>ZSPI-TKN-ERROR</td>
<td>ZSPI-TYP-ERROR</td>
</tr>
<tr>
<td>ZCOM-TKN-OBJNAME</td>
<td>ZSPI-TYP-STRING</td>
</tr>
<tr>
<td>ZCOM-TKN-OBJTYPE</td>
<td>ZSPI-TYP-ENUM</td>
</tr>
<tr>
<td>ZSPI-TKN-ENDLIST</td>
<td>ZSPI-TYP-SSCTL</td>
</tr>
</tbody>
</table>

**Tokens**

**ZSPI-TKN-ERROR**

is the standard SPI error token described in the *SPI Programming Manual*. Its value consists of the TCP/IP subsystem ID and the error number ZTCI-ERR-SNAP-MTU-NOT-AVAIL. This token is always present in the error list.

**ZCOM-TKN-OBJNAME**

is the object name on which the error occurred in the command. This token is present in the error list if its presence is relevant to the cause of the error; that is, if the error was detected after the subsystem made choices based on the object name, this token is present in the error list. Otherwise, this token is omitted from the list.

**ZCOM-TKN-OBJTYPE**

is the object type provided with the command on which the error occurred. Its value defines the type of object against which the command was issued. Its only possible value is ZCOM-OBJ-SUBNET.

**Recommended Action**

The SLSA LAN interface is not reporting this value because the SLSA LIF is not started. Start the SLSA LAN interface and then reissue the START SUBNET command.
Glossary

This glossary defines terms used both in this manual and in other NonStop TCP/IP manuals. Both industry-standard terms and HP terms are included. Because this glossary for NonStop TCP/IP as a whole, not all of the terms listed here appear in this manual.

**address mask.** A bit mask used to select bits from an Internet address for subnet addressing. The mask is 32 bits long and selects the network portion of the Internet address and one or more bits from the local portion.

**address resolution.** Conversion of an Internet address into a corresponding physical address. Depending on the underlying network, resolution may require broadcasting on a local network. See also Address Resolution Protocol (ARP).

**Address Resolution Protocol (ARP).** The Internet protocol used to dynamically bind a high-level Internet Address to a low-level physical hardware address. ARP applies only across a single physical network and is limited to networks that support hardware broadcast.

**Advanced Projects Research Agency (ARPA).** Former name of DARPA, the government agency that funded the ARPANET and DARPA Internet.

**ARP.** See Address Resolution Protocol (ARP).

**ARPA.** See Advanced Projects Research Agency (ARPA).

**ARPANET.** A pioneering long-haul network funded by ARPA (later DARPA) and built by Bolt, Baranek, and Newman (BBN). It served as the basis for early networking research as well as a central backbone during the development of the Internet.

**attribute.** In DSM, a characteristic of an entity. For example, two attributes of a communications line might be its baud rate and its retry count. In a token-oriented interface based on SPI, an attribute of an object is usually expressed as either a simple token or as a field within an extensible structured token. See also simple token or extensible structured token.

**autonomous system.** A collection of gateways and networks that fall under one administrative entity and cooperate closely to propagate network reachability (and routing) information among themselves using an interior gateway protocol of their choice. Gateways within an autonomous system have a high degree of trust. At least one gateway in an autonomous system must advertise networks in that system to a core gateway using EGP.

**baseband.** Characteristic of any network technology (like Ethernet) that uses a single carrier frequency and requires all stations attached to the network to participate in every transmission. See broadband.
**bridge.** A router that connects two or more networks and forwards packets among them. Usually, bridges operate at the physical network level. For example, an Ethernet bridge connects two physical Ethernet cables and forwards from one cable to the other exactly those packets that are not local. Bridges differ from repeaters; bridges store and forward complete packets, while repeaters forward electrical signals.

**broadband.** Characteristic of any network technology that multiplexes multiple, independent network carriers onto a single cable (usually using frequency division multiplexing). For example, a single 100 mbps broadband cable can be divided into ten 10 mbps carriers, with each treated as an independent Ethernet. The advantage of broadband is less cable; the disadvantage is higher cost for equipment. See broadband.

**broadcast.** A packet delivery system that delivers a copy of a given packet to all hosts that attach to it is said to broadcast the packet. Broadcast may be implemented with hardware or software.

**BSD.** Berkeley Software Distribution.

**Class A.** The network number is 1 through 127 (1 octet); that is, the first octet is in the range 1-127. The remaining three octets in the address are used for the subnet number and host number.

**Class B.** The network number is 128 through 191.255 (2 octets); that is, the first octet is in the range 128-191, the second octet is in the range 0-255. The remaining two octets are used for the subnet number and host number.

**Class C.** The network number is 192.0.0 through 255.255.255 (3 octets); that is, the first octet is in the range 192-255, the second octet is in the range 0-255, and the third octet is in the range 0-255. The remaining octet is used for the subnet number and host number. The subnet number varies in length. The subnet number's width is typically represented by a bit mask. The rest of the available bits uniquely identify the host connected to the subnetwork. LANs connected by way of a gateway to the INTERNET get their subnet class from the DCA's NIC (Network Information Center). The address classes of standalone, or entirely private, LANs are administered by the LAN administrator. Typical usage calls for all CLASS A addresses to have private LANs.

**collector.** An EMS process that accepts event messages from subsystems and logs them in the event log. See also Event Management Service (EMS). Compare distributor.

**command message.** A SPI message, containing a command, sent from an application program to a subsystem. See also SPI message. Compare response message or event message.

**common definition.** In DSM programmatic interfaces, a definition (data declaration) used in several commands, responses, or event messages in an SPI interface to a subsystem. See also definition.
compatibility distributor. An EMS distributor process that filters event messages according to fixed (rather than user-specified) criteria, obtains text for these messages that is compatible with the operator console of Guardian operating system versions earlier than C00, and writes the text to the standard Guardian console-message destinations. See also distributor.

conditional token. In DSM event management, a token that is sometimes, but not always, present in a particular event message.

connection. The path between two protocol modules that provides reliable stream delivery service. In the Internet, a connection extends from a TCP module on one machine to a TCP module on another machine.

connectionless service. Characteristic of the packet delivery service offered by most hardware and by the Internet Protocol (IP). The connectionless service treats each packet or datagram as a separate entity that contains the source and destination address. Usually, connectionless services can drop packets or deliver them out of sequence.

consumer distributor. An EMS distributor process that returns selected event messages to management applications upon request. See also distributor.

context token. In DSM programmatic interfaces, a token in an SPI response message that indicates (by its presence or absence) whether or not the response is continued in the following message. If this token is present, the response is continued. To obtain the next message, the application program reissues the original command with one modification: the context token is included in the new command message. When the subsystem sends a response message that does not contain a context token, the series of response messages is complete.

control and inquiry. In DSM, those aspects of object management that affect the state or configuration of an object, such as inquiries about the object and commands pertaining to the environment (for example, commands that set default values for the session). Compare event management.

critical event. A DSM event that is considered to be crucial to the operation of the system or network. Each subsystem determines which of its events are critical, designating them as such by setting the value of the emphasis token to TRUE. Compare noncritical event.

Carrier Sense Multiple Access (CSMA). A characteristic of network hardware that operates by allowing multiple stations to contend for access to a transmission medium by listening to see if it is idle.

Carrier Sense Multiple Access with Collision Detection (CSMA/CD). A characteristic of network hardware that uses CSMA access combined with a mechanism that allows the hardware to detect when two stations simultaneously attempt transmission. Ethernet is an example of a well-known network based on CSMA/CD technology.
Defense Advanced Projects Research Agency (DARPA). Formerly called ARPA. The government agency that funded research and experimentation with the ARPANET and DARPA Internet.

Data communications standard definitions. In DSM, the set of declarations provided by Tandem for use in all management programs that manage or retrieve event messages from Tandem data communications subsystems. The names of these definitions start with either ZCOM or ZCMK. See also definition or definition files. Compare SPI standard definitions or EMS standard definitions.

data list. In DSM programmatic interfaces, a group of tokens used to separate response records within an SPI message for a response, or used to enclose a single response record, if the program so requests. A data list consists of a list token that denotes a data list (different from the token that starts an error list or a generic list), followed by a response record and an end-list token. See also response record.

Defense Data Network (DDN). Used to loosely refer to the MILNET, ARPANET, and the TCP/IP protocols they use. More literally, it is the MILNET and associated parts of the Internet that connect military installations.

Definition. One of the declarations provided by Tandem for use in applications that call APS or SPI procedures. These definitions are provided in definition files. See also definition files.

Definition files. A set of files containing declarations for use in applications that call SPI procedures. SPI has a standard definition file for the Data Definition Language (DDL) and one for each of the programming languages supporting SPI; the latter files are derived from the DDL definition file. Likewise, each subsystem that has a token-oriented programmatic interface has one definition file for DDL and one for each programming language. Some subsystems for instance, data communications subsystems have additional, shared definition files. See also SPI standard definitions, data communications standard definitions, or EMS standard definitions.

Distributed Systems Management (DSM). A set of tools used to manage NonStop systems and EXPAND networks. These tools include the VIEWPOINT console application, the Subsystem Control Facility (SCF) for data communications subsystems, the Subsystem Programmatic Interface (SPI), the Event Management Service (EMS), the Distributed Name Service (DNS), and token-oriented programmatic interfaces to the management processes for various NonStop subsystems.

distributor. An EMS process that distributes event messages from event logs to requesting management applications, to Guardian console message destinations, or to a collector on another node. See also consumer distributor, and compatibility distributor. Contrast collector.

DNS. See Domain Name Server (DNS).
**Domain Name Server (DNS).** A method for naming resources. The basic function of the domain name server is to provide information about network objects by answering queries.

**Domain.** In the Internet, a part of the naming hierarchy. Syntactically, a domain name consists of a sequence of names (labels) separated by periods (dots).

**DSM.** See Distributed Systems Management (DSM).

**E4SA.** See Ethernet 4 ServerNet adapter (E4SA).

**ECHO.** The name of a program used in the Internet to test the reachability of destinations by sending them an ICMP echo request and waiting for a reply.

**Exterior Gateway Protocol (EGP).** The protocol used by a gateway in one autonomous system to advise the Internet addresses of networks in that autonomous system to a gateway in another autonomous system. Every autonomous system must use EGP to advertise network reachability to the core gateway system.

**empty response record.** In DSM programmatic interfaces, a response record containing only a return token with a value that means “no more response records.” See also return token.

**EMS.** See Event Management Service (EMS).

**EMS standard definitions.** The set of declarations provided by EMS for use in event management, regardless of the subsystem. Any application that retrieves tokens from event messages needs the EMS standard definitions. See also definition or definition files. Compare data communications standard definitions or SPI standard definitions.

**error.** In DSM interfaces, a condition that causes a command or other operation to fail. Contrast Warning.

**error list.** In DSM programmatic interfaces, a group of tokens used within a response record to provide error and warning information. An error list consists of a list token that denotes an error list (different from the token that starts a data list or a generic list), followed by an error token, other tokens explaining the error (optional), and an end-list token. Error lists can be nested within other error lists. The return token cannot be included in an error list. See also return token.

**error number.** In DSM programmatic interfaces, a value that can be assigned to a return token, or to the last field of an error token, to identify an error that occurred. Some error numbers are defined in the data communications (ZCOM) definitions; others are defined by individual subsystems.

**error token.** In DSM programmatic interfaces, a token in a response message that indicates the reason an error occurred during a programmatic command. NonStop subsystems enclose each error token in an error list, which can also contain additional information about the error. A response record must contain a return token and can also contain
error lists to explain the error further. The token code for the error token is ZSPI-TKN-ERROR. Its value is a structure consisting of the subsystem ID and an error number identifying the error. See also error list, error number, or return token.

**Ethernet.** A popular local area network technology invented at the Xerox Corporation Palo Alto Research Center. An Ethernet itself is a passive coaxial cable; the interconnections all contain active components. Ethernet is a best-effort delivery system that uses CSMA/CD technology. Xerox Corporation, Digital Equipment Corporation, and Intel Corporation developed and published the standard for 10 Mbps Ethernet.

**Ethernet 4 ServerNet adapter (E4SA).** A ServerNet adapter for Ethernet local area networks (LANs) that contains four Ethernet ports.

**Ethernet meltdown.** An event that causes saturation or near saturation on an Ethernet. It usually results from illegal or misrouted packets and typically lasts only a short time. As an example, consider an IP datagram directed to a nonexistent host and delivered by way of hardware broadcast to all machines on the network. Gateways receiving the broadcast will send out ARP packets in an attempt to find the host and deliver the datagram.

**event.** In DSM terms, a significant change in some condition in the system or network. Events can be operational errors, notifications of limits exceeded, requests for action needed, and so on.

**event log.** A file or set of files maintained by EMS to store event messages generated by subsystems.

**Event Management Service (EMS).** A part of DSM used to provide event collection, event logging, and event distribution facilities. It provides for different event descriptions for interactive and programmatic interfaces, lets an operator or application select specific event-message data, and allows for the flexible distribution of event messages within a system or network. EMS has an SPI-based programmatic interface for both reporting and retrieving events. See also DSM or event message.

**event management.** The reporting and logging of events, the distribution and retrieval of information concerning those events, and the actions taken by operations personnel or software in response to the events. Compare control and inquiry.

**event message.** A special kind of SPI message that describes an event occurring in the system or network. Compare command message.

**Expand.** The Expand subsystem enables you to link together as many as 255 geographically dispersed NonStop systems to create a network with the same reliability, capacity to preserve data integrity, and potential expansion as a single Tandem system.
**extensible structure.** In DSM programmatic interfaces, a structure declared for the value of an extensible structured token. See also extensible structured token. Compare fixed structure.

**extensible structured token.** In DSM programmatic interfaces, a token consisting of a token code and a value that is an extensible structure. Extensible structures can be extended by adding new fields at the end in later RVUs. Such structures are typically used to indicate the attributes of an object being operated on and to return status and statistics information in responses; they can also be used for other purposes. The token is referenced by a token map that describes the structure to SPI so that SPI can provide compatibility between different versions of the structure. Compare simple token.

**fabric.** A simplified way of representing a complex set of interconnections through which there can be multiple and (to the user) unknown paths from point to point. The term fabric is used to refer to the X or Y portion of the ServerNet system area network (ServerNet SAN), for example the X fabric.

**FDDI.** See Fiber Distribution Data Interface (FDDI).

**Fiber Distribution Data Interface (FDDI).** An emerging standard for a network technology based on fiber optics. FDDI specifies a 100-mbps data rate using 1300-nanometer light wavelength, and limits networks to approximately 200 km in length, with repeaters every 2 km or less. The access control mechanism uses token-ring technology.

**File Transfer Protocol (FTP).** The Internet standard, high-level protocol for transferring files from one machine to another. Usually implemented as application level programs, FTP uses the TELNET and TCP protocols. The server side requires a client to supply a login identifier and password before it will honor requests.

**filter.** In EMS, a file containing a list of criteria against which incoming event messages can be compared. The messages are allowed to pass (all criteria met) or not pass (one or more criteria failed). In the ServerNet LAN Systems Access (SLSA) subsystem (for NonStop S-series systems), filters are logical entities which allow frames received from the LAN to be sorted and delivered to a client. In the SLSA subsystem, filters replace the PORT objects used in K-series systems in the sense that filters are the final destination for data received from the LAN.

**FINGER.** A protocol providing a method for retrieving status information about one or all of the users on a particular system.

**fixed structure.** In DSM programmatic interfaces, a multifield structure declared for the value of a simple token. Fields cannot be added to fixed structures in later RVUs. Compare extensible structure.

**forwarding distributor.** An EMS distributor process that sends selected event messages to an EMS collector on another network node. See also distributor.

**FTP.** See File Transfer Protocol (FTP).
**full-duplex mode.** The communication mode in which data can be transferred in both directions simultaneously. In the Session Layer, no data token is needed.

**gateway.** A special-purpose, dedicated computer that attaches to two or more networks and routes packets from one to the other. In particular, an Internet gateway routes IP datagrams among the networks to which it is connected. Gateways route packets to other gateways until they can be delivered to the final destination directly across one physical network. The term is loosely applied to any machine that transfers information from one network to another, as in “mail gateway.”

**Gateway to Gateway Protocol (GGP).** The protocol core gateways used to exchange routing information, GGP implements a distributed shortest path routing computation. Under normal circumstances, all GGP participants reach a steady state in which the routing information at all gateways agrees.

**GGP.** See [Gateway to Gateway Protocol (GGP)](#).

**half-duplex mode.** The communications mode in which data can be transferred in both directions, but only in one direction at a time, and in which the direction of data flow alternates. In the Session Layer, the data token indicates which side can send data.

**header.** The initial part of an SPI message. The first word of this header always contains the value -28; the remainder of the header contains descriptive information about the SPI message, most of which is accessible as header tokens. The tokens in an SPI message header differ according to the type of message: the header of a message that contains a command or response differs somewhat from the header of an event message. An application can use SSGET or EMSGET calls to retrieve the values of header tokens, and can use SSPUT calls to change the values of some tokens. However, there are certain basic differences between header tokens and other tokens. See also [header token](#).

**header token.** In an SPI message, a token that provides information pertaining to the message as a whole. Header tokens differ from other tokens in several ways: they exist in the buffer at initialization and their values are usually set by SSINIT; they can occur only once in a buffer, they are never enclosed in a list, they cannot be moved to another buffer with SSMOVE, and programs cannot position to them or retrieve their values using the NEXTCODE or NEXTTOKEN operation. Programs retrieve the values of header tokens by passing appropriate token codes to SSGET and can change the values of some header tokens by passing their token codes to SSPUT.

Examples of header tokens for commands are the command number, the object type, the maximum-response token, the server-version token, the maximum-field-version token, and the checksum token. Command and response messages contain a specified set of header tokens; event messages, a different set with some overlap. See also [SPI message](#).

**hierarchical routing.** Routing based on a hierarchical addressing scheme. Most Internet routing is based on a two-level hierarchy in which an Internet address is divided into a network portion and a host portion. Gateways use only the network portion until the...
datagram reaches a gateway that can deliver it directly. Subnetting introduces additional levels of hierarchical routing.

**hop count.** A measure of distance between two points in the Internet. A hop count of $n$ means that $n$ gateways separate the source destination.

**ICMP.** See [Internet Control Message Protocol (ICMP)](#).

**IEEE.** See [Institute of Electrical and Electronics Engineers (IEEE)](#).

**IEN.** See [Internet Engineering Note (IEN)](#).

**IGP.** See [Interior Gateway Protocol (IGP)](#).

**Interior Gateway Protocol (IGP).** The generic term applied to any protocol used to propagate network reachability and routing information within an autonomous system. Although no standard Internet IGP exists, RIP is among the most popular.

**Institute of Electrical and Electronics Engineers (IEEE).** An international industry group that develops standards for many areas of electrical engineering and computers.

**interactive command.** In DSM, a command entered by a human operator rather than by a program. See also [programmatic command](#).

**International Organization for Standardization (ISO).** A United Nations organization, established to promote the development of standards to facilitate the international exchange of goods and services and to develop mutual cooperation in areas of intellectual, scientific, technological, and economic activity.

**International Telecommunications Union Telecommunications (ITU-T).** An international body of member countries whose task is to define recommendations and standards relating to the international telecommunications industry. The fundamental standards for ATM have been defined and published by the ITU-T (previously CCITT).

**Internet.** Physically, a collection of packet switching networks interconnected by gateways, along with protocols that allow them to function logically as a single, large, virtual network. When written in uppercase, INTERNET refers specifically to the DARPA Internet and the TCP/IP protocols it uses.

**Internet address.** The 32-bit address assigned to hosts that want to participate in the Internet using TCP/IP. Internet addresses are the abstraction of physical hardware addresses, just as the Internet is an abstraction of physical networks. Actually assigned to the interconnection of a host to a physical network, an Internet address consists of a network portion and a host portion. The partition makes routing efficient.

**Internet Control Message Protocol (ICMP).** An integral part of the Internet Protocol (IP) that handles error and control messages. Specifically, gateways and hosts use ICMP to send reports of problems about datagrams back to the original source that sent the
Internet Engineering Note (IEN). A series of notes developed in parallel to RFCs and available across the Internet from the INIC. IENs contain many of the early theories on the Internet.

Internet Protocol (IP). The Internet standard protocol that defines the Internet datagram as the unit of information passed across the Internet, and that provides the basis for the Internet, connectionless, best-effort, packet-delivery service.

Interoperability. The ability of software and hardware on multiple machines from multiple vendors to communicate meaningfully.

IOP. Input/output process. An input/output process (IOP) is a privileged process, residing in a NonStop system processor, which provides an application access to a communications line.

IP. See Internet Protocol (IP).

IP datagram. The basic unit of information passed across the Internet. An IP datagram is to the Internet as a hardware packet is to a physical network. It contains source and destination addresses, along with data.

ISO. See International Organization for Standardization (ISO).


LANMAN. See LAN manager (LANMAN) process.

LAN. See local area network (LAN).

LAN manager (LANMAN) process. The process provided as part of the ServerNet local area network (LAN) systems access (SLSA) subsystem that starts and manages the SLSA subsystem objects and the LAN monitor (LANMON) process and assigns ownership of Ethernet adapters to the LANMON processes in the system. Subsystem Control Facility (SCF) commands are directed to the LANMON processes for configuring and managing the SLSA subsystem and the Ethernet adapters.

LANMON. See LAN monitor (LANMON) process.

LAN monitor (LANMON) process. The process provided as part of the ServerNet local area network (LAN) systems access (SLSA) subsystem that has ownership of the Ethernet adapters controlled by the SLSA subsystem.

**Level 2.** A reference to LINK LEVEL communication (for example, frame formats) or link-level connections derived from the ISO 7-layer reference model. For long-haul networks, level 2 refers to the communication between a host computer and a network packet switch (for example, HDLC/LAPB). For local area networks, level 2 refers to physical packet transmission. Thus, a level 2 address is a physical hardware address.

**Level 3.** A reference to NETWORK-level communication derived from the ISO 7-layer reference model. For the Internet, level 3 refers to the IP and IP datagram formats. Thus, a level 3 address is an Internet address.

**LIF.** See [logical interface (LIF)].

**LLC.** See [Logical Link Control (LLC)].

**local area network (LAN).** A network that is located in a small geographical area and whose communications technology provides a high-bandwidth, low-cost medium to which low-cost nodes can be connected. One or more LANs can be connected to the system such that the LAN users can access the system as if their workstations were connected directly to it.

**logical interface (LIF).** The interface that allows an application or another process to communicate with data communications hardware.

**Logical Link Control (LLC).** An IEEE 802.2 standard for the Data Link Layer of the OSI Reference Model that defines both connection-oriented and connectionless standards over LAN networks.

**Media Access Control (MAC) address.** A MAC address is a value in the Medium Access Control sublayer of the IEEE/ISO/ANSI LAN architecture, that uniquely identifies an individual station that implements a single point of physical attachment to a LAN.

**management applications.** In DSM, an application process that opens a management or subsystem process to control a subsystem. This process can issue SPI commands to subsystems and retrieve EMS event messages to assist in the management of a computer system or a network of systems. A management application is a requester to the subsystems to which it sends commands; the subsystems are servers to the management application.

**management process.** In DSM, a Tandem process through which an application issues commands to a subsystem. A management process can be part of a subsystem, or it can be associated with more than one subsystem; in the latter case, the management process is logically part of each of the subsystems. SCP is the management process for all Tandem data communications subsystems that support DSM. See also [subsystem].

**manager process.** In DSM, a Tandem subsystem process with which the SCP management process communicates to control a particular data communications subsystem.
MFIOB. See multifunction I/O board (MFIOB).

MILNET (Military Network). Originally part of the ARPANET, MILNET was partitioned in 1984 to make it possible for military installations to have reliable network service, while the ARPANET continues to be used for research. MILNET uses exactly the same hardware and protocol technology as ARPANET, and there are several interconnection points between the two. Thus, under normal circumstances, MILNET sites are part of the Internet.

multicast. A technique that allows copies of a single packet to be passed to a selected subset of all possible destinations. Some hardware (for example, Ethernet) supports multicast by allowing a network interface to belong to one or more multicast groups. Broadcast is a special form of multicast in which the subset of machines selected to receive a copy of a packet consists of the entire set.

multifunction I/O board (MFIOB). A ServerNet adapter that contains ServerNet addressable controllers (SACs) for SCSI and Ethernet; a service processor; ServerNet links to the processor, to the two ServerNet adapter slots, and to one of the ServerNet expansion board (SEB) slots; and provides connections to the serial maintenance bus (SMB), which connects components within an enclosure to the service processor.

Network File System (NFS). A protocol developed by SUN Microsystems that uses IP to allow a set of cooperating computers to access each other’s file systems as if they were local. The key advantage of NFS over conventional file transfer protocols is that NFS hides the differences between local and remote files by placing them in the same name space. NFS is used primarily on UNIX systems, but has been implemented for many systems, including personal computers like an IBM PC and Apple Macintosh.

noncritical event. A DSM event not too crucial to system or network operations. Each subsystem determines which of its events are noncritical by setting the value of the emphasis token to FALSE. Compare critical event.

nonsensitive command. A DSM command that can be issued by any user or program allowed access to the target subsystem—that is, a command on which the subsystem imposes no further security restrictions. For Tandem data communications subsystems, the nonsensitive commands are all those that cannot change the state or configuration of objects (usually information commands). Compare sensitive command.

nowait mode. In Guardian file-system operations and in some APS operations, the mode in which the called procedure initiates an I/O operation but does not wait for it to complete before returning control to the caller. In order to make the called procedure wait for the completion of the operation, the application calls a separate procedure. Compare wait mode.

object. (1) In general, use one or more of the devices, lines, processes, and files in a NonStop subsystem; any entity subject to independent reference or control by one or more subsystems. (2) In DSM use, an entity subject to independent reference and control by a subsystem: for example, the disk volume $DATA or the data
communications line $X2502. An object typically has a name and a type known to the controlling subsystem.

**object-name template.** In DSM, a name that stands for more than one object. Such a name includes one or more wild-card characters, such as * and ?. See also wild-card character.

**object type.** In DSM, the category of objects to which a specific object belongs: for example, a specific disk file might have the object type FILE, and a specific terminal might have the object type SU (subdevice). A subsystem identifies a set of object types by the objects it manages. The SCF interfaces to Tandem data communications subsystems use standard keywords to identify the types. The corresponding programmatic interfaces have object-type numbers (represented by symbolic names such as ZCOM-OBJ-SU) suitable for passing to the SPI SSINIT procedure.

**open system.** Any computer system that adheres to the OSI standards.

**Open Systems Interconnection (OSI).** A set of standards used for the interconnection of heterogeneous computer systems, thus providing universal connectivity.

**OSI.** See Open Systems Interconnection (OSI).

**OSI Reference Model.** A communications architecture, adopted by the ISO in 1984, that includes seven layers that define the functions involved in communications between two systems, the services required to perform these functions, and the protocols associated with these services.

**packet.** The unit of data sent across a packet switching network. While some Internet literature uses it to refer specifically to data sent across a physical network, other literature views the Internet as a packet switching network and describes IP datagrams as packets.

**Packet Internet Groper (PING).** The name of a program used in the Internet to test the reachability of destinations by sending them an ICMP echo request and waiting for a reply. The term has survived the original program and is now used as a verb, as in “please ping host A to see if it is alive.”

**packet switching.** A technique in which messages are broken into smaller units, called packets, that can be individually addressed and routed through the network. The receiving-end node ascertains whether all the packets are received and in the proper sequence before forwarding the complete message to the addressee.

**PDN.** See Public Data Network (PDN).

**physical interface (PIF).** The hardware components that connect a system node to a network.

**physical layer.** Layer 1 in the OSI Reference Model. This layer establishes the actual physical connection between the network and the computer equipment. Protocols at
the Physical Layer include rules for the transmission of bits across the physical medium and rules for connectors and wiring.

**PIF.** See physical interface (PIF).

**PING.** See PING.

**predefined value.** A commonly used value, for instance, a value for a token or a field in a token that is given a name in a set of definition files.

**process.** A running entity that is managed by the operating system, as opposed to a program, which is a collection of code and data. When a program is taken from a file on a disk and run in a processor, the running entity is called a process.

**programmatic command.** In DSM, a command issued by a program rather than by a human operator. Compare interactive command.

**protocol.** A formal description of message formats and the rules two or more machines must follow to exchange those messages. Protocols can describe low level details of machine-to-machine interfaces (for example, the order in which the bits from a byte are sent across a wire), or high-level exchanges between application programs (for example, the way in which two programs transfer a file across the Internet). Most protocols include both intuitive descriptions of the expected interactions as well as more formal specifications using finite state-machine models.

**Public Data Network (PDN).** A network with data communications services available to any subscriber.

**Request for Comments (RFC).** The name of a series of notes that contain surveys, measurements, ideas, techniques, and observations, as well as proposed and accepted Internet protocol standards. RFCs are edited but not referenced. They are available across the Internet.

**response.** In DSM use, the information or confirmation supplied (as part of a response message) to an application by a subsystem in response to a DSM command.

**response message.** An SPI message sent from a subsystem to an application program in reaction to a command message. Compare command message or event message.

**response record.** In DSM programmatic interfaces, a set of response tokens usually describing the result when a command is performed on one object. Every response record in a response from a NonStop subsystem contains a return token; a response record can also contain error lists that include error tokens. A response can consist of multiple response records, spread across one or more response messages. A response record cannot be split between two response messages. If multiple response records are in a response message, each response record is enclosed in a data list. See also data list. Each response record is required to contain a return token. See also return token.
**return token.** In DSM programmatic interfaces, the token that indicates whether a command was successful and, if not, why it failed. The token code for the return token is ZSPI-TKN-RETCODE. Its value consists of a single integer field. Compare **error token**.

**RFC.** See Request for Comments (RFC).

**SAC.** See ServerNet addressable controller (SAC).

**SCF.** See Subsystem Control Facility (SCF).

**SCP.** See Subsystem Control Point (SCP).

**sensitive command.** In DSM, a command that can be issued only by a restricted set of Guardian users, such as the owner of a subsystem process. For Nonsotp data communications subsystems, the sensitive commands are those that can change the state or configuration of objects, start or stop tracing, or change the values of statistics counters. Compare nonsensitive command.

**ServerNet adapter.** A customer-replaceable unit (CRU) that connects peripheral devices to the rest of the system through a ServerNet bus interface (SBI). A ServerNet adapter is similar in function to an I/O controller logic board (LB) and backplane interconnect card (BIC) in NonStop K-series servers.

**ServerNet addressable controller (SAC).** A controller that is uniquely addressable within one or more ServerNet address domains (SADs) through the node ID and address fields in a request packet. A SAC typically is implemented on some portion of a processor multifunction (PMF) customer-replaceable unit (CRU), an I/O multifunction (IOMF) CRU, or a ServerNet adapter.

**ServerNet LAN Systems Access (SLSA) subsystem.** A subsystem of the NonStop operating system. The SLSA subsystem enables the protocol I/O processes (IOPs) and drivers to access the ServerNet adapters.

**ServerNet wide area network (SWAN) concentrator.** A Tandem data communications peripheral that provides connectivity to a NonStop S-series server. The SWAN concentrator supports both synchronous and asynchronous data over RS-232, RS-449, X.21, and V.35 electrical and physical interfaces.

**service.** A set of primitives (operations) that a layer provides to the layer above it. The service defines what operations the layer can perform on behalf of its users, but not how these operations are implemented. A service relates to an interface between two layers: the lower layer is the service provider, and the upper layer is the service user. Compare protocol.

**session.** For a management application, the period during which an application can issue commands to a subsystem.
Simple Mail Transfer Protocol (SMTP). The Internet standard protocol for transferring electronic mail messages from one machine to another. SMTP specifies how two mail systems interact, and specifies the format of control messages the two mail systems exchange to transfer mail.

simple token. In DSM programmatic interfaces, a token consisting of a token code and a value that is either a single elementary field, such as an integer or a character string, or a fixed (nonextensible) structure. Compare extensible structured token.

SLSA Subsystem. See ServerNet wide area network (SWAN) concentrator on page -15

SMTP. See Simple Mail Transfer Protocol (SMTP).


SPI. See Subsystem Programmatic Interface (SPI).

SPI buffer. The buffer that contains an SPI message. See also SPI message.

SPI message. In DSM programmatic interfaces, a message specially formatted by the SPI procedures for communication between a management application and a subsystem or between one subsystem and another. An SPI message consists of a collection of tokens. Note that an SPI message is a single block of information sent at one time, as one interprocess message. There are two types of SPI messages, distinguished by different sets of tokens in the header: command and response messages, and event messages.

SPI procedures. In DSM, the set of Guardian procedures used to build and decode buffers for use in system and network management and in certain other applications.

SPI standard definitions. In DSM programmatic interfaces, the set of declarations available for use with the SPI procedures, regardless of the subsystem. There is also a set of subsystem-specific declarations for each subsystem, and some sets of declarations that apply to multiple subsystems. An application using SPI needs the SPI standard definitions and also the subsystem definitions for all subsystems with which it communicates. See also definition. Compare data communications standard definitions or EMS standard definitions.

subject token. In event management, a device, process, or other named entity about which a given event message has information.

subnet address. An extension of the Internet addressing scheme that allows a site to use a single Internet address for multiple physical networks. Outside of the site using subnet addressing, routing continues as usual by dividing the destination address into an Internet portion and local portion. Gateways and hosts inside a site using subnet addressing interpret the local portion of the address by dividing it into a physical network portion and host portion.
subnetwork. One or more intermediate systems that provide relaying and through which end open systems may establish network connections.

Subnetwork Access Protocol (SNAP). In order to run the TCP/IP protocol suite over IEEE networks, the Subnetwork Access Protocol (SNAP) defines the interface between the IP layer and the LLC layer. The interface is accomplished through the use of an extension of the LLC header that contains a predefined Service Access Point (SAP) for use in the Source SAP (SSAP) and the Destination SAP (DSAP) fields of the LLC header.

subsystem. (1) The software and/or hardware facilities that provide users with access to a set of communications service. (2) For DSM, a program or set of processes that manages a cohesive set of objects. Each subsystem has a process through which applications can request services by issuing commands defined by that subsystem; in some cases, this process is the entire subsystem. Many subsystems also have interactive interfaces.

Subsystem Control Facility (SCF). A part of DSM, used to provide a common, interactive management interface for configuring, controlling, and collecting information from Tandem data communications products.

Subsystem Control Point (SCP). In DSM, the management process for all Tandem data communications subsystems. There can be several instances of this process. Applications using SPI send all commands for data communications subsystems to an instance of this process, which in turn sends the commands on to the manager processes of the target subsystems. SCP also processes a few commands itself. It provides security features, version compatibility, support for tracing, and support for applications implemented as NonStop process pairs. See also management process or manager process.

Subsystem ID (SSID). In DSM programmatic interfaces, a data structure that uniquely identifies a subsystem to SPI. It consists of the name of the owner of the subsystem (such as Tandem), a subsystem number that identifies that particular subsystem, and a subsystem version number. The subsystem ID is an argument to most of the SPI procedures.

Subsystem Programmatic Interface (SPI). In DSM, a set of procedures and associated definition files used to define common message-based programmatic interfaces for communication between requesters and servers—for instance, in a management application. SPI includes procedures to build and decode specially formatted messages; definition files in Pascal, TAL, C, COBOL, and TACL for inclusion in programs, macros, and routines using the SPI procedures; and definition files in DDL for programmers writing their own subsystems.

summary state. In DSM interfaces to NonStop data communications subsystems, one of the generally defined possible conditions of an object, with respect to the management of that object. A summary state differs from a state in two ways. First, a summary state pertains to the management of an object, whereas a state may convey other kinds of
information about the object. Second, summary states are defined the same way for all NonStop data communications subsystems, whereas the set of possible states differs from subsystem to subsystem. The management programming interfaces to NonStop data communications subsystems refer to summary states rather than to states. Examples of summary states are STARTED, STOPPED, SUSPENDED, and ABORTING.

**SWAN concentrator.** See [ServerNet wide area network (SWAN) concentrator](#).

**symbolic name.** In DSM programmatic interfaces, a name used in programs to refer to commonly used values, token codes, token maps, extensible structures, and other related variables for use in management programs.

**TCP (Transmission Control Protocol).** The Internet standard transport-level protocol that provides the reliable, full-duplex stream service on which many application protocols depend. TCP allows a process on one machine to send a stream of data to a process on another. It is connection-oriented, in the sense that before transmitting data participants must establish a connection. Software implementing TCP usually resides on the operating system and uses the IP protocol to transmit information across the Internet. It is possible to terminate (shut down) one direction of flow across a TCP connection, leaving a one-way (simplex) connection. The Internet protocol suite is often referred to as TCP/IP because TCP is one of the two most fundamental protocols.

**TELNET.** The Internet standard protocol for remote terminal connection service. TELNET allows a user at one site to interact with remote timesharing systems at another site just as if the user's terminal is connected directly to the remote machine. That is, the user invokes a TELNET application program that connects to a remote machine, prompts for a login ID and password, then passes keystrokes from the user's terminal to the remote machine and displays output from the remote machine on the user's terminal.

**TFTP.** See [Trivial File Transfer Protocol (TFTP)](#).

**token.** In DSM use, a distinguishable unit in a SPI message. Programs place tokens in an SPI buffer using the SSPUT or SSINIT procedures and retrieve them from the buffer with the SSGET procedure. A token has two parts: an identifying code, or token code, and a token value. In command and response messages, a token normally represents a parameter to a command, an item of information in a response, or control information for the subsystem. In event messages, a token normally represents an item of information about an event or about the event message itself. See also [header token](#).

**token number.** In DSM programmatic interfaces, the number used by a subsystem to identify each DSM token that it defines. The token type and the token number together form the token code.
token ring LAN. A ring network that allows unidirectional data transmission between data stations, by a token passing procedure, such that the transmitted data return to the transmitting station.

token ring ServerNet adapter. A ServerNet adapter that provides access to token ring LANs.

token type. In DSM programmatic interfaces, the part of a DSM token code that identifies the data type and length of the token value. The token type and the token number together form the token code.

token value. In DSM programmatic interfaces, the value assigned to a DSM token.

Trivial File Transfer Protocol (TFTP). The Internet standard protocol for file transfer with minimal capability and minimal overhead. TFTP depends only on the unreliable, connectionless datagram delivery service (UDP), so it can be used on machines like diskless workstations that keep such software in ROM and use it to bootstrap themselves.

TRSA. See token ring ServerNet adapter.

UDP. See User Datagram Protocol (UDP).

User Datagram Protocol (UDP). The Internet standard protocol that allows an application program on one machine to send a datagram to an application program on another machine. UDP uses the Internet Protocol to deliver datagrams. Conceptually, the important difference between UDP and IP is that UDP messages include a protocol port number, allowing the sender to distinguish among multiple destinations (application programs) on the remote machine. In practice, UDP also includes a checksum over the data being sent.

wait mode. In the Guardian operating system, the mode in which the called procedure waits for the completion of an I/O operation before returning a condition code to the caller. Compare nowait mode.

WAN. See wide area network (WAN).

WAN manager process. The WAN manager process starts and manages the WAN subsystem objects including the ConMgr and WANBoot processes.

WAN subsystem. See wide area network (WAN) subsystem.

Warning. In DSM interfaces, a condition encountered in performing a command or other operation, that can be significant but does not cause the command or operation to fail. A warning is less serious than an error. Compare error.

well-known port. Any of a set of protocol ports preassigned for specific uses by transport level protocols (that is, TCP and UDP). Servers follow the well-known port assignments so clients can locate them. Examples of well-known port numbers include ports
assigned to echo servers, time servers, remote login (TELNET) servers, and file transfer (FTP) servers.

**wide area network (WAN).** A network that operates over a larger geographical area than a local area network (LAN)—typically, an area with a radius greater than one kilometer. The elements of a WAN may be separated by distances great enough to require telephone communications. Contrast with local area network (LAN).

**wide area network (WAN) subsystem.** The Subsystem Control Facility (SCF) subsystem for configuration and management of WAN objects in G-series RVUs.

**wild-card character.** A character that stands for any possible character(s) in a search string or in a name applying to multiple objects. In DSM object-name templates, two wild-card characters can appear: ? for a single character and * for zero, one, or more consecutive characters. See also object-name template.

**X.25.** The CCITT standard protocol for transport-level network service. Originally designed to connect terminals to computers, X.25 provides a reliable stream transmission service that can support remote login.

**X.25 Access Method.** See X25AM.

**X25AM (X.25 Access Method).** An HP product that implements, for WANs, the services of the Network Layer and layers below.

**X.25 network.** Any network or subnetwork linked using X.25 standards. X.25 standards are CCITT standards that define packet switching carrier communication in the Network Layer over wide area networks (WANs). See also International Telecommunications Union Telecommunications (ITU-T) and packet switching.

**$ZZLAN.** See LAN manager (LANMAN) process.

**$ZZWAN.** See WAN manager process.
Index

A
ABORT command 6-5
Action events 7-3
ADD command 6-8
Allow-Type token 5-3
ALTER command 6-16
Applications
  naming rules and guidelines for 3-3
  obtaining event messages 7-1
  using letter Z 3-4

B
Buffer declarations,TCP/IP 5-15
Buffer size
  recommended for commands 4-1
  recommended for responses 4-2
Building the command buffer 4-1

C
Checksum token 5-3
Command buffer, building 4-1
Command numbers 3-4
Commands
  ABORT 6-5
  ADD 6-8
  ALTER 6-16
    building the buffer for 4-1
    by object type 6-1
  compared to SCF commands 2-4
  components 3-10
  continuation of 4-1
  DELETE 6-25
  description of 6-1
  discontinuing 4-2
  duplicate occurrences of tokens in 4-1
  GETVERSION 6-28

Commands (continued)
  INFO 6-31
  list of 2-4
  LISTOBJECTS 6-42
  LISTOPENS 6-45
    multiple, in single request message 4-1
    nonsensitive 6-2
    notation used to describe 6-4
    numbers 3-4
    order of tokens in 6-4
    recommended buffer size for 4-1
  RESETSTATS 6-50
    sensitive 6-2
  START 6-70
  STATISTICS 6-73
  STATUS 6-100
  STOP 6-107
    symbolic names 6-1
  TRACE 6-110
Commands and responses 6-1
Command-number token 5-3
Comment token 5-3
Common definitions 5-1
Common syntax elements 3-4
Constructs involving multiple tokens 3-9
Context token 5-3
Continuation
  of commands 4-1
  of responses 4-2
Creating a process 1-2
Critical events 7-2
CRUs
Customer replaceable units
  See CRUs

D
Data communications standard definitions 5-9
Data list token 5-4
Data lists
  description of 3-9
  representation in responses 6-5
Decoding the response buffer 4-2
Definition files 3-1
Definitions
  common 5-1
  data communications standard 5-9
  EMS standard 5-7
  files 3-1
  SPI standard 5-1
  TCP/IP subsystem 5-14
DELETE command 6-25
Discontinuing a command 4-2
Dynamic route switching 4-3

E

Elements of SPI messages 3-1
Empty response record 4-3
EMS standard definitions 5-7
End list token 5-4
Error handling 4-3
Error list token 5-4
Error lists, description of 3-9
Error lists, representation in responses 6-5
Error numbers with no error lists 4-3
Event management 7-1
Event messages
  components 3-10
  console-compatible-format text 7-4
  description of 2-6
  display-format text 7-4
  event numbers of 3-7
  filters 7-3
  notation 7-3
  obtaining 7-1
  order of tokens 7-3
  reported to EMS 7-3
  standard header in text version 7-4

Event messages (continued)
  subjects 3-7
  TCP/IP subsystem 7-1
  TCP/IP subsystem ID 7-1
  text version 7-4
Event numbers 3-7
Events
  action 7-3
  critical 7-2
  description of 2-6
  management of 7-1
  noncritical 7-2
  processes that report 7-1
Event-management programming 7-3
Event-number token 5-8
Extensible structured tokens
  description of 3-8
  using SSNULL to initialize 4-1

F

Field types 3-9
Fields
  initializing 4-1
  null values of 4-1
  types of 3-9
Files, definition 3-1
Filters, definition of 1-5, 7-3

G

GETVERSION command 6-28

H

Header type token 5-5

I

INFO command 6-31
Initializing structured tokens 4-1
Index

L
Letter Z, using in applications 3-4
LIF, definition of 1-5
LISTOBJECTS command 6-42
LISTOPENS command 6-45
Lists
  data lists 3-9
  error lists 3-9
Logical device name 3-7
Logical device number 3-7

M
Maximum field version token 5-5
Maximum response token 5-5
Maximum values for ZTCI token values 5-17
MAXRESP token 5-5
Minimum values for ZTCI token values 5-17
Monitoring statistics 6-73
Multiple objects, in commands 3-6
Multiple tokens, constructs involving 3-9

N
Names
  of objects
    See Object names
  of values 3-9
Naming rules and guidelines 3-3
Noncritical events 7-2
Nonsensitive commands
description of 6-2
GETVERSION 6-28
INFO 6-31
LISTOBJECTS 6-42
LISTOPENS 6-45
STATISTICS 6-73
STATUS 6-100
NULL object
  commands affecting 6-2
  object type of 5-11
NULL, getting release version 6-28

O
Object names
  format 3-5
  processes 5-13
  routes 5-12
  subnets 5-13
  when required in a command 5-13
Object types
  commands listed by 6-1
  description 3-5
  of NULL object 5-11
  process 5-11
  route 5-11
  subnet 5-12
Objects
  description 1-1
  format of names 3-5
  number affected by one command 4-1
  number described in one response record 4-2
Object-name templates
  commands accepted in 3-7
  description of 3-6
Object-name token 5-12
Object-type header token 5-5
Object-type number 3-5
Object-type token 5-13

P
PIF, definition of 1-6
Predefined value names 3-9
Private field types 5-15
Private token types 5-15
Process
  altering  6-16
  creating  1-2
  current status  6-100
  event messages reported  7-1
  format of object name  3-5
  monitoring statistics  6-73
  object names  5-13
  obtaining statistics  6-50
  resetting statistical counters  6-50
  tracing  6-110

Process object  5-11
Processes that report events  7-1
Programming considerations  4-1
Programming for event management  7-3

R
Receiving the response buffer  4-2
Release version  6-28
RESETSTATS command  6-50
Response buffer, receiving and decoding  4-2
Response records
  duplicate occurrences of tokens in  4-2
  empty  4-3
  multiple  4-2
  number of in one reply message  4-2
  number of objects described in  4-2
  tokens in  6-4

Responses
  components  3-10
  continuation of  4-2
  decoding the buffer for  4-2
  description of  6-1
  multiple response records in  4-2
  notation used to describe  6-4
  order of tokens in  6-4
  receiving the buffer for  4-2
  recommended buffer sizes for  4-2
  that describe multiple objects  4-2

Return token  5-6
Route object  5-11
Routes
  aborting  6-5
  adding  6-8
  commands affecting  6-2
  current status  6-100
  deleting  6-25
  description of  4-3
  format of object names  3-6
  monitoring statistics  6-73
  object names  5-12
  obtaining statistics  6-50
  resetting statistics counters  6-50
  starting  6-70
  stopping  6-107
  terminating operation  6-5

S
SAC, definition of  1-6
Security  4-3
Sensitive commands
  ABORT  6-5
  ADD  6-8
  ALTER  6-16
  DELETE  6-25
  description of  6-2
  RESETSTATS  6-50
  START  6-70
  STOP  6-107
Server banner token  5-6
Server version token  5-6
Simple tokens  3-8
SPI error token  5-4
SPI messages, elements  3-1
SPI programming considerations  4-1
SPI standard definitions  5-1
SSNULL procedure  4-1
START command  6-70
STATISTICS command  6-73
STATUS command 6-100
STOP command 6-107
Structured tokens
  See Extensible structured tokens
Subject token 3-7
Subjects of event messages 3-7
Subnet object 5-12
Subnets
  aborting 6-5
  adding 6-8
  altering 6-16
  commands affecting 6-2
  current status 6-100
  deleting 6-25
  format of object names 3-6
  monitoring statistics 6-73
  object names 5-13
  obtaining statistics 6-50
  resetting statistics counters 6-50
  starting 6-70
  stopping 6-107
  terminating operation 6-5
  tracing 6-110
Subsystem identifier 5-17
Syntax elements, command numbers 3-4
Syntax elements, common 3-4
TCP/IP process, commands affecting 6-2
TCP/IP subsystem
  command numbers supported 3-4
  commands accepted by 6-1
  common syntax elements 3-4
  definitions provided by 5-14
  elements of SPI messages 3-1
  EMS definitions 5-7
  object names 3-5
  object types supported 3-5
  responses issued by 6-1
  SPI definitions 5-1
  SPI programming considerations 4-1
  subsystem number 5-2
TCP/IP subsystem ID in event messages 7-1
TCP/IP, data communications definitions 5-10
Token codes 3-8
Token maps 3-8
Token types 3-9
Tokens
  constructs involving multiple 3-9
  description of 3-1, 3-8
  duplicate occurrences of in commands 4-1
  duplicate occurrences of in responses 4-2
  extensible structured 3-8
  in one response record 6-4
  order in event messages 7-3
  order of in commands 6-4
  order of in responses 6-4
  predefined value names 3-9
  simple 3-8
  token codes 3-8
  token maps 3-8
  token types 3-9
TRACE command 6-110
Trace mode 6-110

Tandem token value 5-7
TCI subsystem, subsystem ID 5-17
TCP/IP definitions
  buffer declarations 5-15
  description of 5-14
  extensible structured tokens 5-18
  field types 5-15
  field values 5-16
  token types 5-15
  tokens in error lists 5-19
  tokens in event messages 5-19
Index

Types
  of fields 3-9
  of objects 3-5
  of tokens 3-9

V
  Value names, predefined 3-9

Z
  ZCOM-CMD-
    ABORT 6-5
    ADD 6-9
    ALTER 6-16
    DELETE 6-25
    GETVERSION 6-28
    INFO 6-31
    LISTOBJECTS 6-42
    LISTOPENS 6-45
    RESETSTATS 6-50
    START 6-70
    STATISTICS 6-74
    STATUS 6-100
    STOP 6-107
    TRACE 6-111
  ZCOM-MAP-
    REQID 5-11
    TRACE-MODIF 5-11
  ZCOM-OBJ-
    NULL 5-11
    PROC 5-11
    ROUTE 5-11
    SUBNET 5-12
  ZCOM-TKN-
    CAUSE-SUMSTATE-CHG 5-12
    OBJNAME 5-12
    OBJSTATE 5-13
    OBJTYPE 5-13
    OLD-OBJSTATE 5-13
    SUBJ-PROC 5-13
    SUBJ-SUBNET 5-14
    TRACE-OPT 5-14
  ZCOM-VAL-
    BUFLEN 5-14
    MAXBUFLEN 5-14
    SUMSTATE-STARTED 5-14
    SUMSTATE-STOOPED 5-14
  ZEMS-TKN-
    CONSOLE-PRINT 5-8
    CPU 5-8
    CRTPID 5-8
    EMPHASIS 5-8, 7-2
    EVENTNUMBER 5-8
    GENTIME 5-8
    LOGTIME 5-9
    PIN 5-9
    SUBJECT-MARK 5-9, 7-3
    SYSTEM 5-9
    USERID 5-9
  ZSPI-ERR-OK 4-3
  ZSPI-SSN-ZTCI 5-2
  ZSPI-TKN-
    ALLOW-TYPE 4-2, 5-3
    CHECKSUM 5-3
    COMMAND 5-3
    COMMENT 5-3
    CONTEXT 5-3
    DATALIST 5-4, 6-4
    ENDLIST 5-4, 6-4
    ERRLIST 5-4, 6-4
    ERROR 5-4
    HDRTYPE 5-5
    MAXRESP 4-2, 5-5
    MAX-FIELD-VERSION 5-5
    OBJECT-TYPE 5-5
    RETCODE 5-6
    SERVER-BANNER 5-6
    SERVER-VERSION 5-6
ZSPI-VAL-
  CMDHDR 5-7
  EVTHDR 5-7
  NORM-ONLY 5-7
  TANDEM 5-7
ZTCI subsystem ID 5-17
ZTCI-DDL-MSG-BUFFER 5-15
ZTCI-ERR-
  ATTR-OUT-OF-RANGE A-2
  DUP-ADDR A-4
  ENET-ADDR-NOT-AVAIL A-3
  INV-FNAME A-2
  NET-UNREACH A-5
  SNAP-MTU-NOT-AVAIL A-6
ZTCI-EVT-
  ADDR-ERR 7-42
  BACKUP-IS-DEAD 7-68
  BACKUP-IS-STARTED 7-70
  BCAST-ENET-ADDR 7-18
  CANT-HANDLE-AF 7-52
  CANT-START-BACKUP 7-66
  CHKPT-FAILED 7-73
  CONF-FILE-ACC-ERR 7-30
  CONF-FILE-SYN-ERR 7-32
  CONN-ESTAB 7-22
  CONN-ESTAB-GATE 7-71
  CONN-TIMEOUT 7-24
  DEV-NOT-AVAIL 7-25
  DUP-IP-ADDR 7-16
  INTL 7-14
  INVALID-CPU-NUM 7-64
  IOP-AWAITIO-ERR 7-10
  IOP-QIO-ERR 7-76
  IOP-READ-ERR 7-6
  IOP-WRITE-ERR 7-8
  MAX-DEV 7-20
  NEW-CL-MBUF 7-59
  NEW-PROC-ERR 7-34
  NON-ZERO-RUN-SOCK 7-60
ZTCI-EVT- (continued)
  NO-BACKUP 7-62
  NO-BACKUP-CPU 7-75
  NO-MEM 7-4
  OUT-OF-MBUF 7-54
  PARAM-ERR 7-40
  PROTO-ERR 7-46
  QIOMODE-ON 7-78
  RECV-OPEN-FAIL 7-48
  SOCK-ERR 7-44
  SSINIT-FAILED1 7-57
  SSINT-FAILED 7-50
  SUMSTATE-CHG 7-12
  SYN-ERR 7-28
  TRACE-STOP-ERR 7-55
  XFER-ERR 7-36
ZTCI-TKN-SEV 7-3
ZTCI-TYP-
  CAUSE 5-15
  SEV 5-15
  SUBNET-TYPE 5-15
  TCP-STATE 5-15
  TRACE 5-16
  XFER 5-16
ZTCI-VAL-
  BUFLEN 5-16
  CAUSE-ERR 5-17
  CAUSE-OP 5-12, 5-17
  CAUSE-RECOV 5-12, 5-17
  MAX-BUFLEN 5-17
  MAX-CAUSE 5-17
  MAX-SEV 5-17
  MAX-SUBNET-TYPE 5-17
  MAX-TCP-STATE 5-17
  MAX-TRACE 5-17
  MAX-XFER 5-17
  MIN-CAUSE 5-17
  MIN-SEV 5-17
  MIN-SUBNET-TYPE 5-17
ZTCI-VAL- (continued)
    MIN-TCP-STATE 5-17
    MIN-TRACE 5-17
    MIN-XFER 5-17
    SEV-FATAL 5-17
    SEV-INFO 5-17
    SEV-WARN 5-17
    SSID
        description 5-17
        structure definition 5-6
    SUBNET-TYPE-ENET 5-17
    SUBNET-TYPE-LOOP 5-17
    SUBNET-TYPE-SNAP 5-17
    SUBNET-TYPE-X25 5-17
    TCP-STATE-CLOSE-WAIT 5-18
    TCP-STATE-CLOSING 5-18
    TCP-STATE-FIN-WAIT-1 5-18
    TCP-STATE-FIN-WAIT-2 5-18
    TCP-STATE-LAST-ACK 5-18
    TCP-STATE-LISTEN 5-18
    TCP-STATE-SYNC-RECV 5-18
    TCP-STATE-SYNC-SENT 5-18
    TCP-STATE-TIME-WAIT 5-18
    TRACE-IPC-CMDS 5-18
    TRACE-IP-IN 5-18
    TRACE-IP-OUT 5-18
    TRACE-MBUF-ALLOC 5-18
    TRACE ROUTE 5-18
    TRACE-SOCKET-CMDS 5-18
    TRACE-SOCKET-CREATE 5-18
    TRACE-TCP 5-18
    TRACE-UDP-DETL-IN 5-18
    TRACE-UDP-DETL-OUT 5-18
    TRACE-UDP-IN 5-18
    TRACE-UDP-OUT 5-18
    TRACE-UDP-USER-REQ 5-18
    VERSION 5-18
    XFER-ADD-ITEM-REC 5-18

ZTCI-VAL- (continued)
    XFER-ADD-RECIPIENT 5-18
    XFER-ATTACH-COMPNT 5-18
    XFER-CREATE-FOLDER 5-18
    XFER-CREATE-ITEM 5-18
    XFER-DETACH-COMPNT 5-18
    XFER-END-SESS 5-18
    XFER-GET-ITEM-COMPNT 5-18
    XFER-GET-ITEM-DESCR 5-18
    XFER-GET-ITEM-REC 5-18
    XFER-GET-RECIPIENT-REC 5-18
    XFER-SAVE-ITEM 5-18
    XFER-SCAN-FOLDER 5-18
    XFER-START-SESS 5-18
    XFER-SUBMIT-PKG 5-18
    XFER-UNSAVE-ITEM 5-18
Z, using in applications 3-4