Dialogic®
Making Innovation Thrive™

Application Note

Enabling Dual-Chassis Fault Tolerance with the Dialogic® DSI SIGTRAN Stack

Enabling Dual-Chassis Fault Tolerance with the Dialogic® DSI SIGTRAN Stack

Application Note

Executive Summary

In seeking to achieve five-nines (99.999%) availability and a high degree of fault tolerance in a SIGTRAN signaling environment, the functionality of a SIGTRAN M3UA-based signaling node can be split over two chassis. With this type of dual configuration, the processes of a single chassis can continue if the other chassis fails, allowing the system as a whole to remain in service.

The Dialogic® DSI SIGTRAN Stack is well suited for this dual-chassis approach, and it provides the architecture for splitting one or more local point codes over two active protocol engines.

This application note describes how the DSI SIGTRAN Stack can be used to create a dual-resilient SIGTRAN M3UA-based platform that offers high availability and a high degree of fault tolerance.

Enabling Dual-Chassis Fault Tolerance with the Dialogic® DSI SIGTRAN Stack

Application Note

Table of Contents

Enabling Dual-Chassis Fault Tolerance with the Dialogic® DSI SIGTRAN Stack

Application Note

Introduction

Telephony systems are required to maintain high levels of availability, often characterized by a need for five-nines (99.999%) availability. However, the mean up-time between failures of a single system component is rarely long enough to achieve this goal. This means that system designers need to allow for individual component failures, while ensuring that the overall availability of the service is maintained.

One way to achieve five-nines availability is to build a fault-tolerant system with dual-resilient capabilities. These capabilities can make the system tolerant to a single failure, and they usually provide sufficient system resiliency to achieve the required up-time.

A fault in a telephony system is likely to involve one of the following types of failures:

- Failure of one or more system hardware components
- Failure of the physical connection between the active system and an adjacent system
- Complete failure of an adjacent system, such as a failure caused by a power loss
- Failure of a route to a remote system, caused by the failure of an unknown remote component

When configuring fault-tolerant systems, it must be clear which kind of faults the system can tolerate, the consequence of each fault, and how the system will maintain service levels in the event of each fault.

This application note discusses how dual-resilient SIGTRAN-based platforms can be configured using the Dialogic® DSI SIGTRAN Stack, resulting in telephony systems with high availability and a high degree of fault tolerance.

SIGTRAN Terminology

Table 1 describes the SIGTRAN terminology used in this application note, as defined by various IETF Request for Comments (RFC) documents:

Term	Acronym	Definition
Application Server	AS	"A logical entity serving a specific Routing Key. An example of an Application Server is a virtual switch element handling all call processing for a signaling relation, identified by an SS7 DPC/OPC. Another example is a virtual database element, handling all HLR transactions for a particular SS7 SIO/DPC/OPC combination." [RFC 4666]
Application Server Process	ASP	"A process instance of an Application Server. An Application Server Process serves as an active or backup process of an Application Server (e.g., part of a distributed virtual switch or database)." [RFC 4666]
IP Server Process	IPSP	"A process instance of an IP-based application. An IPSP is essentially the same as an ASP, except that it uses M3UA in a point-to-point fashion. Conceptually, an IPSP does not use the services of a Signaling Gateway node." [RFC 4666]
Multi-Homed SCTP Endpoints	N/A	"An SCTP endpoint is considered multi-homed if there are more than one transport address that can be used as a destination address to reach that endpoint." [RFC 2960]
Routing Context	RC	"A value that uniquely identifies a Routing Key." [RFC 4666]
Routing Key	RK	"A Routing Key describes a set of SS7 parameters and parameter values that uniquely define the range of signaling traffic to be handled by a particular Application Server." [RFC 4666]
SCTP Association	N/A	"The association provides the transport for the delivery of MTP3-User protocol data units and M3UA adaptation layer peer messages." [RFC 4666]
Signaling Gateway	SG	"…a signaling agent that receives/sends SCN [Switched Circuit Network] native signaling at the edge of the IP network. The SG function may relay, translate or terminate SS7 signaling in an SS7-Internet Gateway." [RFC 2719]

Table 1: SIGTRAN Terminology Used in This Application Note

SIGTRAN Resilience

SIGTRAN systems use IP to transport SS7 signaling, and much of the resilience of a SIGTRAN system is offered by features of the IP and SIGTRAN protocols.

The resilience of IP networks is enhanced by SCTP (Stream Control Transmission Protocol), which provides the ability to implement signaling connections to remote points using multiple Ethernet ports and multiple IP networks.

Above SCTP, the M3UA (MTP Level 3 User Adaption) Layer, as specified by RFC 4666, supports the distribution of services across one or more SCTP-enabled chassis and the routing of messages for a remote destination via multiple transfer points.

These features add to the existing resiliency features offered by the Dialogic® DSI Protocol Stacks, including SCCP synchronization of both remote point codes and remote subsystem states between resilient hosts, and the local distribution of ISUP and TCAP traffic between resilient hosts.

By combining the resilience measures described in this application note, fault-tolerant SIGTRAN systems can be built that protect against a single point of failure.

Concepts

This section describes how to build in resilience when implementing a SIGTRAN-based system. It highlights the resilient properties of such a system and discusses how the system handles potential points of failure.

Resilient SIGTRAN Architectures

A resilient SIGTRAN architecture can be implemented in two ways:

- With resilient M3UA ASPs that connect to the SS7 network via SGs or via the SG functions of two Signal Transfer Points (STPs)
- With resilient M3UA IPSPs that connect directly to a service provider's IPSPs via IP point-to-point signaling

The following notes apply to both types of implementations:

- SCTP associations support multi-homing across two IP subnets between the SGs and ASPs or between the local and remote IPSPs. For more information, see "SCTP Resilience," later in this application note.
- SS7 User Part protocols (such as ISUP and TCAP) synchronize state information and reroute traffic between hosts (ASPs or IPSPs) using a Dialogic® Remote Socket Interface (RSI) link. For more information, see "Service Distribution across Multiple Hosts," later in this application note.
- The Dialogic® DSI Resilient MTP Management (RMM) module allows outbound messages in a dual-resilient system to be forwarded to the partner M3UA module via an RSI link, providing resilience if the local route becomes unavailable. For more information, see "Resilient MTP Management," later in this application note.

Resilient M3UA Application Server Processes

Figure 1 depicts an example of a resilient SIGTRAN system in which resiliency is achieved by using two M3UA ASPs that connect to the SS7 network via two SGs:

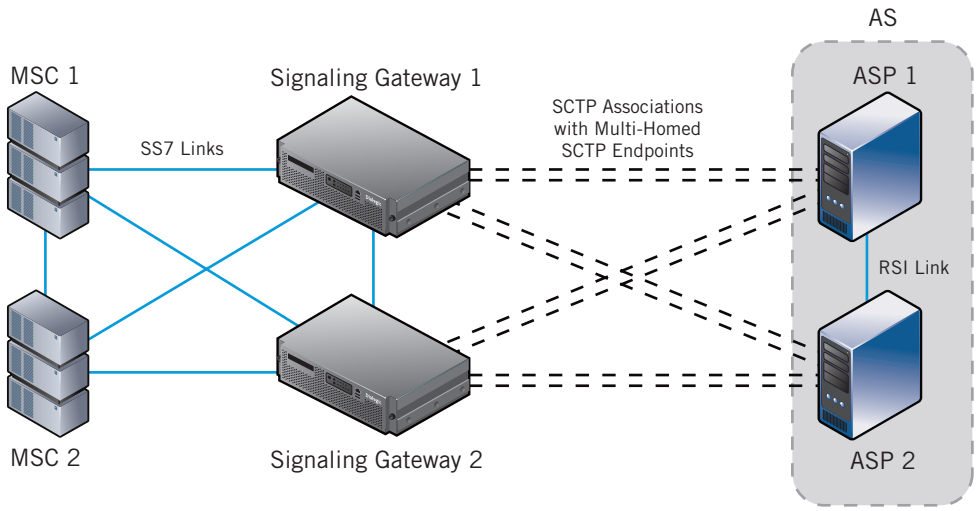

Figure 1. Example of Resilient M3UA Application Server Processes

In a configuration with resilient M3UA ASPs:

- An AS is distributed across two host servers.
- Each server hosts an instance of an ASP that connects to an SS7 network, either via two SGs (shown in this example) or via the SG functions of two Signal Transfer Points (STPs).
- The resilient hosts present themselves to the SGs as two distinct servers that can handle the same service.
- The SGs load-share traffic between the two hosts.
- An RSI link provides both an SS7 User Part connection between hosts and transport for the RMM.

Resilient M3UA IP Server Processes

Figure 2 depicts an example of a resilient SIGTRAN system in which resiliency is achieved by using two M3UA IPSPs that connect to a service provider's network nodes (IPSPs) via IP point-to-point signaling:

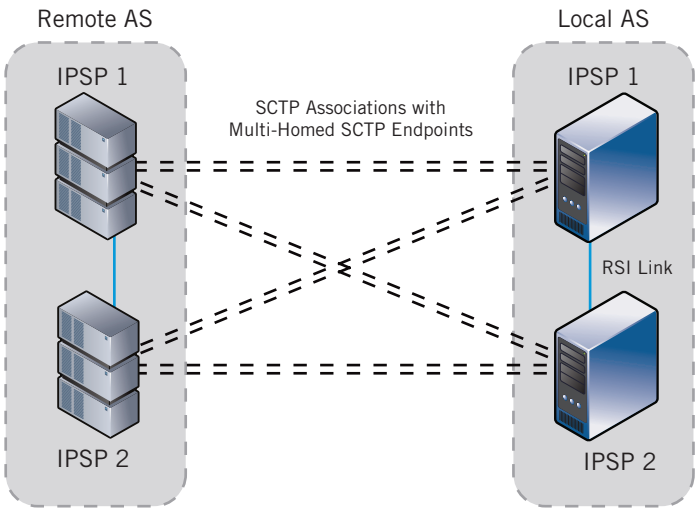

Figure 2. Example of Resilient IP Server Processes

In a configuration with resilient IPSPs:

- A local AS is distributed across two host servers.
- Each server hosts an instance of an IPSP that connects directly to a service provider's IPSP (a remote IPSP).
- Each remote IPSP acts as a node for handling traffic for a remote AS.
- An RSI link provides both an SS7 User Part connection between hosts and transport for the RMM.

Resilience Mechanisms

The DSI SIGTRAN Stack offers the following resilience mechanisms:

- SCTP resilience
- M3UA route resilience
- Resilient MTP management

SCTP Resilience

All SIGTRAN protocols use SCTP associations to provide reliable, in-sequence delivery of messages over IP. An SCTP association is similar to a TCP connection in that it offers both a logical connection across an IP network and reliable in-sequence delivery.

SCTP was designed to offer a reliable service for carrying SS7 traffic over IP. This reliability is achieved via enhanced flow control features, fast retransmission of dropped packets, and a multi-homing capability.

Multi-homing allows an SCTP association end point to support multiple IP addresses. Any IP address used by the association can receive traffic for the SCTP association, and each IP address can use a different Ethernet port. Having two IP addresses and two Ethernet ports at each end of an SCTP association means no single point of failure can cause the SCTP association to fail.

When configuring a multi-homed association, the underlying IP stack should be configured such that local IP addresses used by an association are bound to distinct Ethernet ports, and messages originating from a local Ethernet port use an IP address bound to that Ethernet port.

For additional resilience, the Ethernet ports used can be on different physical Ethernet cards, and the individual IP addresses used can be configured within different IP subnets.

M3UA Route Resilience

M3UA hosts use a table of pre-configured routes for forwarding messages via SGs to remote SS7 point codes. When M3UA has a signaling message to forward to a certain Destination Point Code (DPC), it searches the table for a route to this DPC.

M3UA routes can provide resiliency by identifying more than one SG. This protects against loss of a route in the event of an SG failure or if a single SG loses accessibility to the DPC. In this situation, if one SG fails, a second SG can be used, and the route will remain available.

For information about configuring M3UA route resilience, see "Configuring Routes," later in this application note.

Resilient MTP Management

The RMM module can be used in dual-resilient systems to improve the resilience of the whole system by passing traffic and status messages between different system units. The RMM module sits in the protocol stack between the SS7 User Part modules (ISUP/SCCP) and the Message Transfer modules (MTP3/M3UA), and it can be used to redirect outbound MTP traffic via the partner unit, if the local direct route for outbound traffic is unavailable.

The RMM module monitors route status indications from the M3UA module and provides an aggregate route status to the User Part modules (such as ISUP or SCCP) based on the accessibility via either unit of a dual resilient system.

For information about configuring the RMM module, see "Appendix C: Inter-Chassis Communication," later in this application note, and the *Dialogic® DSI Signaling Protocols RMM Programmer's Manual*.

Potential Points of Failure and Solutions

Potential points of failure in a SIGTRAN network include:

- Chassis failure
- Ethernet/IP failure
- Adjacent SG failure
- Adjacent IPSP failure

Chassis Failure

A chassis failure occurs when a component such as the chassis power supply causes the entire chassis to cease operating for a period of time. This type of failure also encompasses operating system failure, memory failure, certain software failures, and hard disk failure.

Solution: In the event of a chassis failure, it is a good practice to have a second chassis available to continue hosting the service. This requires synchronization of state information and distribution of messages across both chassis.

For information about configuring a dual-chassis SIGTRAN platform, see "Configuration," later in this application note. For information about message distribution, see "Service Distribution across Multiple Hosts," later in this application note.

Ethernet/IP Failure

An Ethernet/IP failure occurs due to a loss of Ethernet connectivity or the failure of Ethernet equipment, such as an Ethernet card or Ethernet port, within the network. This type of failure results in the inability of the IP network to route messages to certain IP addresses and, conversely, the inability of the remote equipment to route messages to the local system.

Solution: SIGTRAN systems communicate with each other via connections known as SCTP associations, enabling reliable, in-sequence delivery of messages. SIGTRAN systems also offer excellent resilience to Ethernet and IP failures through the use of SCTP multi-homing features, as described in "SCTP Resilience," earlier in this application note.

Additionally, the RMM module allows outbound messages in a dual-chassis SIGTRAN system to be sent via the remote M3UA, if the local route is unavailable. For more information, see "Appendix C: Inter-Chassis Communication," later in this application note, and the *Dialogic® DSI Signaling Protocols RMM Programmer's Manual*.

Figure 3 shows an example of a scenario in which all of the network-facing SCTP associations to one ASP have failed. In this scenario, the RMM module uses an RSI link to transport outbound messages to the partner ASP for delivery to the network.

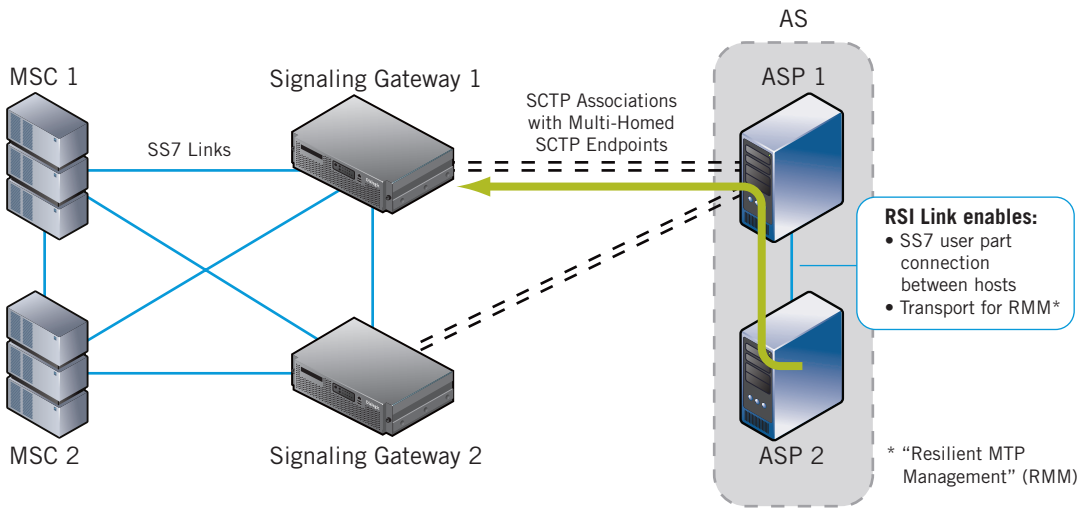

Figure 3. Using the RSI Link When All Local Network-Facing SCTP Associations Have Failed

Adjacent Signaling Gateway Failure

In configurations where M3UA ASPs connect to the SS7 network through SGs or STPs, an SG failure occurs when an SG is unavailable or when an SG cannot route messages to one or more destinations. A loss of connection to an SG can result in the loss of an SS7 route; consequentially, a locally hosted service may be unable to reach specific network destinations.

Solution: M3UA supports connections via two SGs, allowing continued SS7 accessibility despite the loss of a single SG. For more information, see "Configuring SCTP Associations," later in this application note.

Adjacent IP Server Process Failure

In configurations where M3UA services communicate directly with each other through IPSPs instead of through SGs, an adjacent IPSP failure occurs when a remote IPSP host is lost.

Solution: M3UA supports the distribution of a remote service over a number of hosts. In the event of the failure of one remote IPSP host, messages for the service continue to be sent to other hosts that support the service.

Configuration

Hosts using the DSI SIGTRAN Stack, including the M3UA layer, are configured using two files: a *config.txt* file and a *system.txt* file, as described in the *Dialogic® SS7 Protocols Programmer's Manual for SIGTRAN Host Software*. This section describes how to use a *config.txt* file to configure the following SS7 layers for dual-chassis fault tolerance:

- M3UA
- SCCP
- TCAP
- ISUP

The "Resilient SIGTRAN Configuration Examples" section, later in this application note, provides examples of resilient SIGTRAN configurations (*config.txt files* and *system.txt files*) for M3UA ASP systems and IPSP systems handling GSM MAP traffic.

M3UA Configuration

The following sub-sections describe how to use a *config.txt* file to create a resilient M3UA layer:

- Configuring SCTP Associations
- Configuring a Local Application Server
- Configuring Routes

Configuring SCTP Associations

Configure an RSI link to give each host (ASP) access to the other host's SCTP associations, and optionally configure multi-homing for the SCTP associations. Figure 4 depicts an example of this type of configuration:

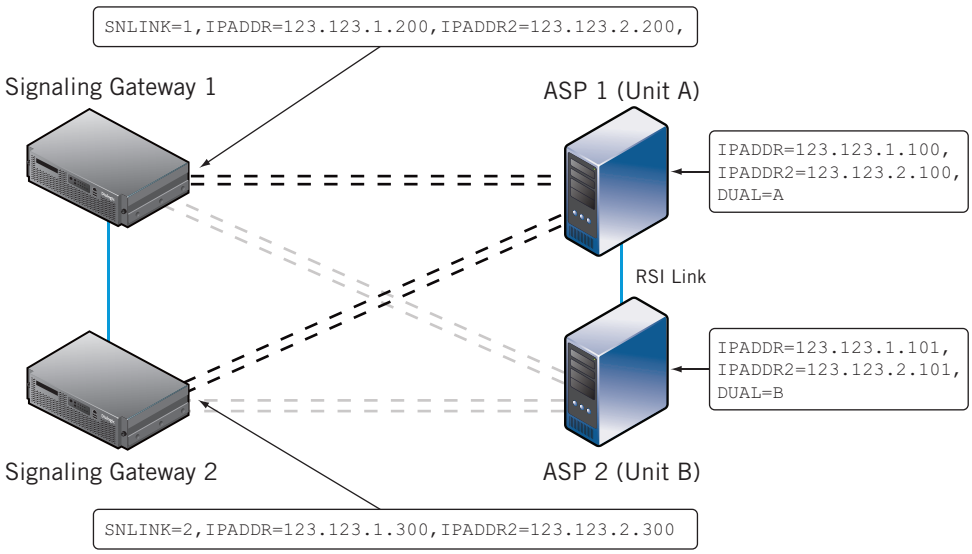

Figure 4. Example Configuration for a Host with Two Multi-Homed SCTP Associations and an RSI Link

Use the CNSYS command <DUAL> parameter in the *config.txt* file to enable RMM transport over the RSI link. Set the <DUAL> parameter value to 'A' or 'B' (depending upon whether Unit A or Unit B of the resilient system is being configured), as shown in Figure 4.

Set up multi-homing by configuring two local IP addresses and two remote IP addresses for each SCTP association. Set the <SNEND> parameter for one endpoint of the association to 'C' (client) and for the other to 'S' (server).

The following example shows how to enable an RSI link and configure SCTP multi-homing for ASP 1 as shown in Figure 4. In this example, the SCTP endpoints are configured as client endpoints:

```
CNSYS:IPADDR=123.123.1.100, IPADDR2=123.123.2.100, DUAL=A;
*
* Define SCTP Signaling Link IP Address/s, Signaling Gateway Number
SNSLI:SNLINK=1,IPADDR=123.123.1.200,IPADDR2=123.123.2.200,SG=1, SNEND=C,SNTYPE=M3UA;
*
SNSLI:SNLINK=2,IPADDR=123.123.1.300,IPADDR2=123.123.2.300,SG=2,SNEND=C,SNTYPE=M3UA;
*
```

For examples of *config.txt* files, see "Resilient SIGTRAN Configuration Examples," later in this application note. For more information about enabling an RSI link, see "Appendix C: Inter-Chassis Communication," later in this application note.

Configuring a Local Application Server

Use the SNAPI command in the *config.txt* file to configure a local AS:

- Use the <AS> parameter to declare a local AS.
- Use the Routing Context <RC> parameter to identify a Routing Key (RK) for the local AS. For definitions of the terms RC and RK, see "SIGTRAN Terminology," earlier in this application note.
- Set the Traffic Mode for Host System <TRMD> parameter to 'LS' (load-share). The host will request this mode of service from the SGs during activation.

The following example shows a local AS configuration:

```
* Define Application Server (local) and Routing Content (must be same as other end)
SNAPI:AS=1,OPC=333,RC=1,TRMD=LS;
*
```

For examples of *config.txt* files, see "Resilient SIGTRAN Configuration Examples," later in this application note.

Configuring Routes

To build resilient routes, configure connections to at least two SGs (or for an IPSP configuration, configure each remote service over two hosts). Then, configure each route to use two SGs. Figure 5 depicts an example of this type of configuration:

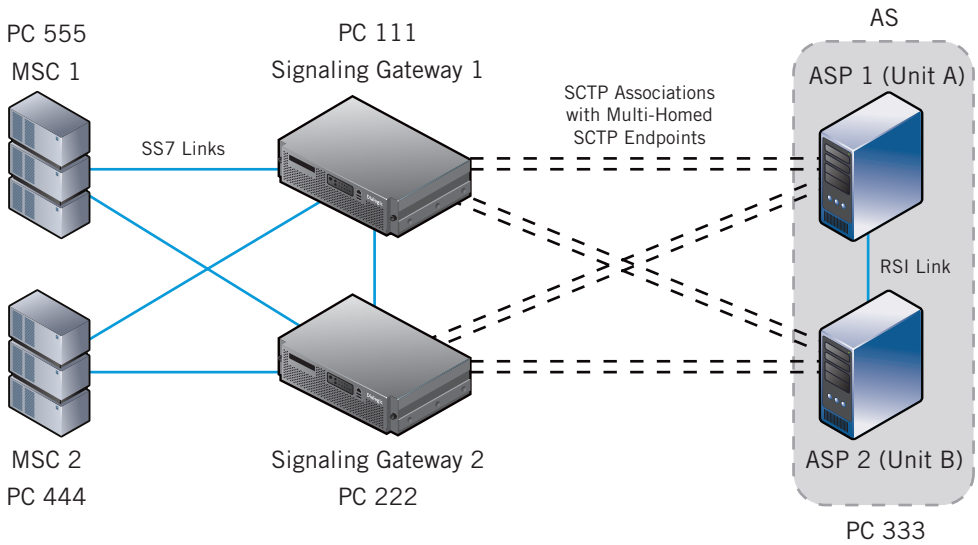

Figure 5. Example Route Configuration for Two ASPs Connecting via Two Signaling Gateways

Use the SNRTI command in the *config.txt* file to define a route, and the SNRLI command to associate a SG with a route.

The following example shows how to configure routes for a local ASP. The example assumes that the AS was already defined with local point code 333 as shown in Figure 5:

```
* Define Signaling Route number and the SG Destination Point Code
SNRTI:SNRT=1,DPC=111;   * to reach Signaling Gateway 1
SNRTI:SNRT=2,DPC=222;   * to reach Signaling Gateway 2
SNRTI:SNRT=3,DPC=444;   * to reach network MSC 1
SNRTI:SNRT=4,DPC=555;   * to reach network MSC 2
*
* Define Signaling Route to Signaling Gateway relationship
SNRLI:SNRL=1,SNRT=1,SG=1;
SNRLI:SNRL=2,SNRT=1,SG=2;
SNRLI:SNRL=3,SNRT=2,SG=1;
SNRLI:SNRL=4,SNRT=2,SG=2;
SNRLI:SNRL=5,SNRT=3,SG=1;
SNRLI:SNRL=6,SNRT=3,SG=2;
SNRLI:SNRL=7,SNRT=4,SG=1;
SNRLI:SNRL=8,SNRT=4,SG=2;
```

For examples of *config.txt* files, see "Resilient SIGTRAN Configuration Examples," later in this application note.

SCCP Configuration

At the SCCP layer, use the SCCP_CONFIG command in the *config.txt* file to configure SCCP for dual-resiliency:

- Configure the SCCP protocols with identical local subsystems, remote signaling points and remote subsystem data.
- Set the <sccp_instance> parameter to a unique number (between 0 and 15) for the SCCP instance running on each chassis. Assign an instance value of 0 to the first SCCP and 1 to the next SCCP.
- Set the SCCP_CONFIG options parameter <SCPF_SMB> to 1 to enable the SCCP Management Broadcast (SMB) mechanism. If SMB is enabled, SCCP will use SMB to communicate with other SCCP instances at the same signaling point; otherwise, the SCCP module is configured as a single instance.

For example, when the SMB mechanism is enabled, Local System A might see Remote SCCP Subsystem X as unavailable via its own signaling links. But if Local System B can still access Remote SCCP Subsystem X, both Systems A and B can reach Subsystem X.

Within the *system.txt* file, SCCP itself normally takes a module identifier (module ID) of 0x33, and the partner SCCP instance <partner id> should be set to 0x53 on Unit A and 0x43 on Unit B.

Use a REDIRECTION statement in the *system.txt* file to route messages to the RSI (or similar task), so they are delivered to the partner SCCP layer connected via Ethernet. For an example of how to configure module IDs for units A and B of a dual resilient system, see "Example M3UA system.txt Files (Windows®)," later in this application note.

The following example shows how to configure SCCP for resiliency:

```
* SCCP Configuration for Host A:
* SCCP Parameters:
* SCCP_CONFIG <local_spc> <ssf> <options> [<management_options> [<partner_id>
<instance>]]
SCCP_CONFIG 333 0x08 0x0102 0x00000000  0x53 0
*
* SCCP Configuration for Host B:
* SCCP_CONFIG <local_spc> <ssf> <options> [<management_options> [<partner_id>
<instance>]]
SCCP_CONFIG 333 0x08 0x0102 0x00000000  0x43 1
*
```

For examples of *config.txt* files, see "Resilient SIGTRAN Configuration Examples," later in this application note. For more information about RSI, see "Appendix C: Inter-Chassis Communication," later in this application note.

TCAP Configuration

In a multi-TCAP environment, assign a unique instance identifier to each TCAP instance. This is encoded in the transaction identifier at the SCCP boundary to allow quick resolution of the correct destination TCAP instance for received messages.

Use the TCAP_CONFIG command in the *config.txt* file to configure TCAP for resiliency:

- Set the <tcap_instance> parameter to 0 for the first TCAP instance and 1 for the next TCAP instance.
- Use the TCAP_CONFIG <partner_id> parameter to specify the module identifier of the partner TCAP protocol. For an example of how to configure module IDs for units A and B of a dual-resilient system, see "Example M3UA system.txt Files (Windows®)," later in this application note
- The logical dialog identifier ranges can be the same for both TCAP halves (so the two application processes running on the two hosts operate over the same dialog identifier range), or separate ranges can be used.

The following example shows how to configure TCAP for resiliency. In this example, the TCAP instances operate over the same dialog identifier range:

```
*
* TCAP Configuration for Host A:
* TCAP_CONFIG <base_ogdlg_id> <nog_dialogues><base_icdlg_id> <nic_dialogues>
<options>
<dlg_hunt> [ [<addr_format>] <partner_id><tcap_instance> ]
TCAP_CONFIG 0x0 8192 0x8000 8192 0x0000 0 0 0x34 0
*
* TCAP Configuration for Host B:
* TCAP_CONFIG <base_ogdlg_id> <nog_dialogues><base_icdlg_id> <nic_dialogues>
<options>
```

```
<dlg _ hunt> [ [<addr _ format>] <partner _ id><tcap _ inst> ]
TCAP _ CONFIG 0x0 8192 0x8000 8192 0x0000 0 0 0x24 1
```

For examples of *config.txt* files, see "Resilient SIGTRAN Configuration Examples," later in this application note.

ISUP Configuration

Configuring ISUP for dual-resiliency involves configuring the ISUP module and configuring ISUP circuit groups.

Configuring the ISUP Module

For resilient ISUP hosts, specify the module ID of the partner ISUP protocol (on the other host) by setting the optional <partner_id> parameter of the *config.txt* ISUP_CONFIG command. Set the ISUP_CONFIG options bit 'ISPF_DUAL' to indicate that ISUP should hand off to this software task any message received from MTP3 for an unrecognized circuit identity.

The module identifier normally assigned to ISUP is 0x23, and the task identifier to send ISUP messages for unrecognized circuits is 0x73 for unit A and 0x63 for unit B. Using the *system.txt* file, the user should arrange for a LOCAL task to convey messages sent to these tasks to the ISUP protocol layer running on the other platform. This can be achieved by use of the RSI task and the *system.txt* REDIRECTION command as described in "Appendix C: Inter-Chassis Communication," later in this application note.

The following example shows how to configure the ISUP module for dual-resilience:

Host A:

```
* Configure ISUP module:
* ISUP _ CONFIG <res1> <res2> <user _ id> <options> <num _ grps> <num _ ccts> [<partner _ id>]
 ISUP _ CONFIG     0      0       0x4d      0x0416        8           256          0x73
```

Host B:

```
* Configure ISUP module:
* ISUP _ CONFIG <res1> <res2> <user _ id> <options> <num _ grps> <num _ ccts> [<partner _ id>]
 ISUP _ CONFIG     0      0       0x4d      0x0416        8           256          0x63
```

Dual Chassis Configuration of ISUP Circuit Groups

ISUP circuit groups can be numbered on each host in two ways:

- By starting ISUP circuit group numbering at 0 for both hosts. With this method, the total system capacity is **double** that of a single ISUP protocol layer.
- By leaving gaps in the ISUP circuit group numbering on each host, where a circuit group exists on the partner host. With this method, the total system capacity is that of a **single** ISUP protocol layer.

In both types of numbering schemes, each host controls separate ISUP circuit group number ranges, as shown in Figures 6 and 7.

Figure 6 shows two hosts controling ISUP circuit groups starting from 0 (a double capacity circuit group distribution):

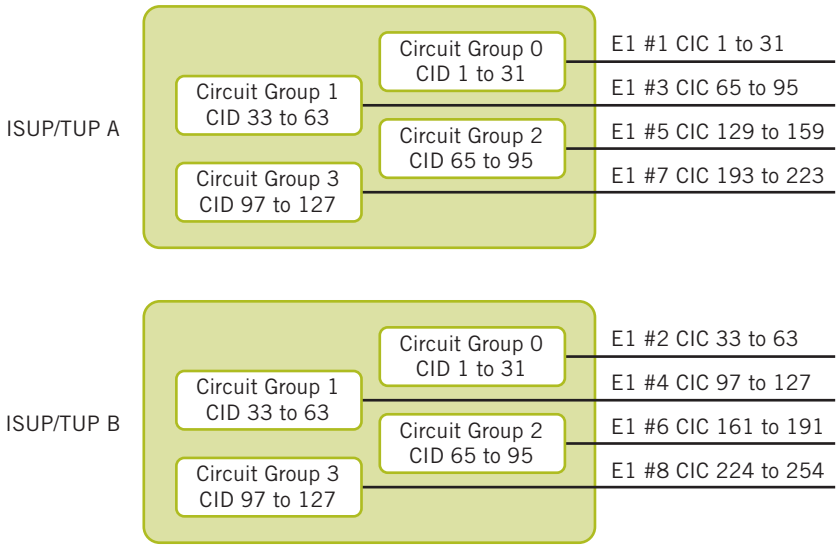

Figure 6. Double Capacity Circuit Group Distribution

Figure 7 shows two hosts controlling ISUP circuit groups with different values (a single capacity circuit group distribution):

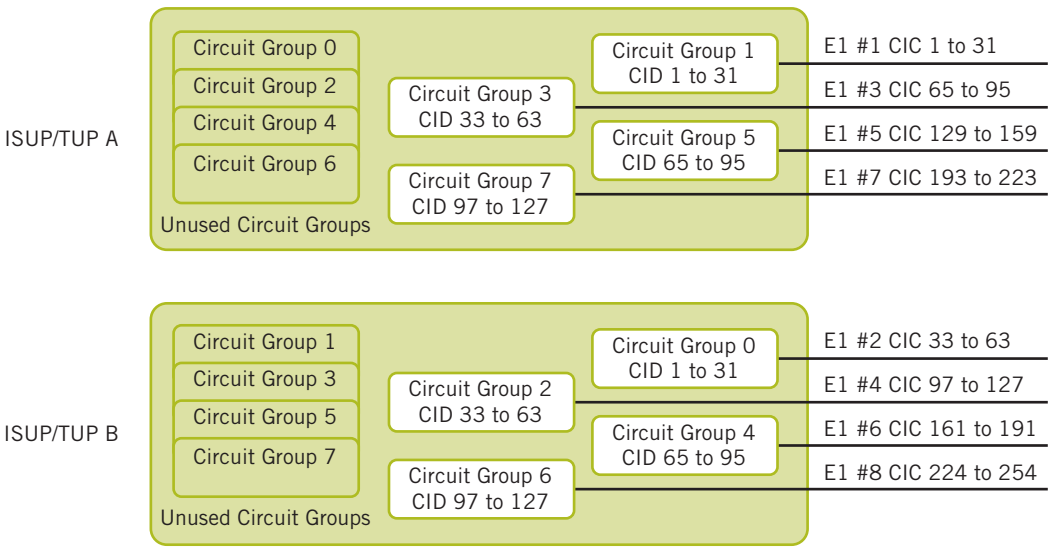

Figure 7. Single Capacity Circuit Group Distribution

The system that does not duplicate the circuit groups between the two halves (the single-capacity system shown in Figure 7) can activate (configure) unused circuit groups on one of the hosts if the other host fails.

This process is achieved using the ISP_MSG_CNF_GRP message to configure a circuit group. This will provide the SS7 signaling resources and ISUP state machines for circuits that have failed on the other host.

Note: The only process the signaling can achieve for circuit terminations that remain physically connected to a failed host is hardware blocking. For more information, see the Enabling Dual Chassis Fault Tolerance with Dialogic Signaling Boards application note.

Service Distribution across Multiple Hosts

Inherent in a dual-resilient service is the requirement for messages to be routed from the network to all hosts (ASPs or IPSPs) that make up the AS. M3UA supports the distribution of messages from an SG to multiple local hosts.

In a dual-resilient service, each host has its own multi-homed SCTP association to each SG, and each SG should be configured to send messages for the service across both hosts. The SG views each host as an ASP and views the service as an AS.

When a host starts, it carries out the following sequence of tasks as detailed in RFC 4666:

1. Establishes its SCTP associations.
2. Brings up the local ASP, initiated by sending an "ASP Up" message to the SG. The SG should respond with an acknowledgement.
3. Sends an "ASP Active" message to the SG, indicating the ASP is ready to start processing traffic. The SG should respond with an acknowledgement.

Each host (AS) should request to be brought up in 'load-share' traffic mode, so the SG can distribute messages evenly across each host. Messages required to be processed by the partner host will automatically be sent across the dual-resilient RSI link by the SS7 User Parts.

M3UA Considerations

Due to the built in support for dual-resilient systems within the M3UA protocol, it is not necessary for partner M3UA instances to share state information such as DPC availability. The SGs take responsibility for forwarding network state information to each host (ASP) to provide ongoing synchronization of all routing information.

Additionally, the RMM modules monitor route status indications from the M3UA module via the RSI Link and provide an aggregate status to the SS7 User Part modules, such as ISUP and SCCP, based on the accessibility of either unit.

For information about configuring M3UA, see "M3UA Configuration," earlier in this application note. For information about configuring an RSI link, see "Appendix C: Inter-Chassis Communication," later in this application note.

ISUP and TCAP Considerations

Distributing the application across two hosts means that state information, such as the ISUP call state and TCAP dialog state, is also distributed across the two hosts.

User messages received from the network may refer to an existing circuit or TCAP dialog, which can be served by the partner host. Dialogic® SS7 User Part protocols can detect when a message should be forwarded to a partner protocol instance for processing. Partner hosts are configured with an RSI link between them for communication between partner protocols, enabling the simple forwarding of such messages.

For information about configuring ISUP and TCAP see "ISUP Configuration" and "TCAP Configuration," earlier in this application note. For information about configuring an RSI link, see "Appendix C: Inter-Chassis Communication," later in this application note.

SCCP Considerations

SCCP maintains a routing table that holds network state information. When a message is received from the network concerning the state of a remote signaling point or subsystem, this information is automatically synchronized across hosts. This communication uses the RSI link.

For information about configuring SCCP, see "SCCP Configuration," earlier in this application note. For information about configuring an RSI link, see "Appendix C: Inter-Chassis Communication," later in this application note.

Resilient SIGTRAN Configuration Examples

This section of the application note provides two examples that illustrate how to configure resilient SIGTRAN systems based on the Dialogic® DSI M3UA protocol:

- Dual-Resilient M3UA ASP Configuration Example
- Dual-Resilient M3UA IPSP Configuration Example

Finally, example *system.txt* files are given for dual-resilient M3UA systems. The *system.txt* file examples are for hosts running under Windows®, however small changes to the *system.txt* FORK_PROCESS commands will allow these examples to be used with a Linux- or Solaris-based host instead.

Dual-Resilient M3UA ASP Configuration Example

The following section provides example configuration files for a dual-resilient M3UA ASP system handling GSM MAP traffic with local and remote point codes and subsystem numbers, as shown in Figure 8. Example IP addresses are given as shown previously in Figure 4, earlier in this application note.

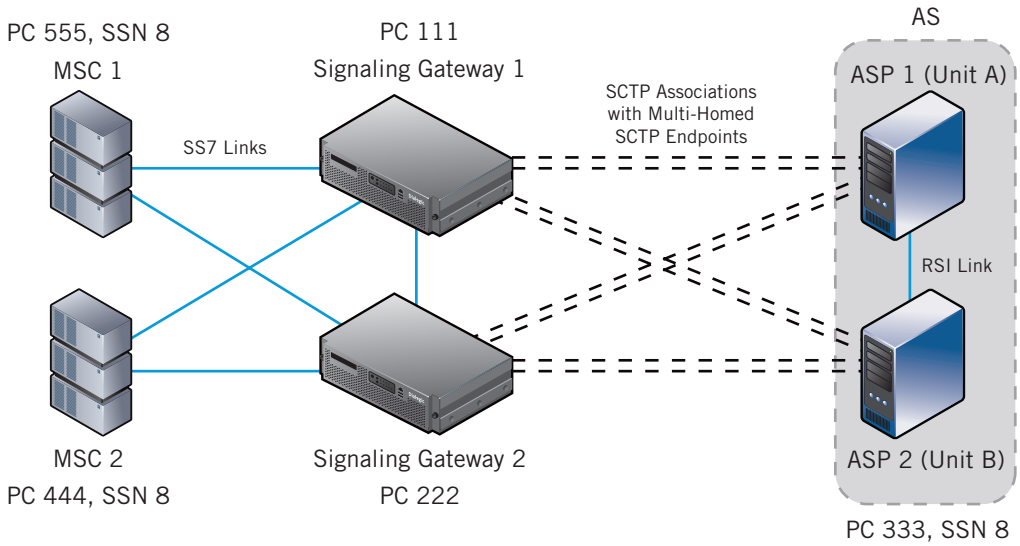

Figure 8. Dual-Resilient M3UA ASP Configuration Example

Host A config.txt File

The following example *config.txt* file defines Host A in a dual-resilient M3UA ASP system:

```
*
* Dual Resilient host A
*
****************************
* Local IP Address Configuration
*
CNSYS:IPADDR=123.123.1.100,IPADDR2=123.123.2.100, DUAL=A;
*
****************************
* Local Application Server Configuration for Communicating through signaling gateway
*
SNAPI:AS=1, OPC=333, RC=1,TRMD=LS;
*
****************************
* SCTP association configuration to Adjacent signaling gateways
*
SNSLI:SNLINK=1,IPADDR=123.123.1.200,IPADDR2=123.123.2.200,SS7MD=ITU14,SG=1;
SNSLI:SNLINK=2,IPADDR=123.123.1.300,IPADDR2=123.123.2.300,SS7MD=ITU14,SG=2;
*
****************************
* Route Configuration Via signaling gateways
*
* Define Signaling Route number and the SG Destination Point Code
*
SNRTI:SNRT=1,DPC=111;   * to reach Signaling Gateway 1
SNRTI:SNRT=2,DPC=222;   * to reach Signaling Gateway 2
*
SNRTI:SNRT=3,DPC=555;   * to reach network MSC 1
SNRTI:SNRT=4,DPC=444;   * to reach network MSC 2
*
* Define Signaling Route to Signaling Gateway relationship
*
SNRLI:SNRL=1,SNRT=1,SG=1;
SNRLI:SNRL=2,SNRT=1,SG=2;
*
```

```
SNRLI:SNRL=3,SNRT=2,SG=1;
SNRLI:SNRL=4,SNRT=2,SG=2;
*
SNRLI:SNRL=5,SNRT=3,SG=1;
SNRLI:SNRL=6,SNRT=3,SG=2;
*
SNRLI:SNRL=7,SNRT=4,SG=1;
SNRLI:SNRL=8,SNRT=4,SG=2;
*
**************************
*
* SCCP Parameters:
* SCCP _ CONFIG <local _ spc> <ssf> <options> [<management _ options> [<partner _ id>
<instance>]]
SCCP _ CONFIG 333 0xc 0x0102 0x00000001 0x53 0
*
* Enable SCCP traces:
* SCCP _ TRACE <op _ evt _ mask> <ip _ evt _ mask> <non _ prim _ mask>
SCCP _ TRACE 0x3 0x7 0x3
*
* Define Remote Signaling Points:
* SCCP _ SSR <ssr _ id> RSP <remote _ spc> <flags> <pc _ mask>
SCCP _ SSR 1 RSP 444 0 0x0000
SCCP _ SSR 2 RSP 555 0 0x0000
*
* Define Local Sub-Systems:
* SCCP _ SSR <ssr _ id> LSS <local _ ssn> <module _ id> <flags> <protocol>
SCCP _ SSR 3 LSS 0x08 0x2d 0 MAP
*
* Define Remote Sub-Systems:
* SCCP _ SSR <ssr _ id> RSS <remote _ spc> <remote _ ssn> <flags>
SCCP _ SSR 4 RSS 444 0x08 0
SCCP _ SSR 5 RSS 555 0x08 0
*
* Enable SCCP traces:
* SCCP _ TRACE <op _ evt _ mask> <ip _ evt _ mask> <non _ prim _ mask>
SCCP _ TRACE 0x3 0x7 0x3
```

```
*
* TCAP Parameters:
* TCAP _ CONFIG <base _ ogdlg _ id> <nog _ dialogs> <base _ icdlg _ id>
* <nic _ dialogs> <options> <dlg _ hunt> [<addr _ format><partner _ id><tcap _ inst> ]
TCAP _ CONFIG 0x0 8192 0x8000 8192 0x0000 0 0 0x34 0
*
* Enable TCAP traces:
* TCAP _ TRACE <op _ evt _ mask> <ip _ evt _ mask> <non _ prim _ mask>
TCAP _ TRACE 0x7 0xf 0x0
*
*
* MAP Parametes:
* MAP _ CONFIG <options>
MAP _ CONFIG 0x0
*
* Enable MAP traces:
* MAP _ TRACE <op _ evt _ mask> <ip _ evt _ mask> <non _ prim _ mask>
MAP _ TRACE 0xf 0xf 0x15
```

Host B config.txt file

The following example *config.txt* file defines Host B in a dual-resilient M3UA ASP system:

```
*
* Dual Resilient host B
*
***************************
* Local IP Address Configuration
*
CNSYS:IPADDR=123.123.1.101,IPADDR2=123.123.2.101, DUAL=B;
*
***************************
* Local Application Server Configuration for Communicating through signaling gateway
*
SNAPI:AS=1,RC=1,OPC=333,TRMD=LS;
*
***************************
* SCTP association configuration to Adjacent signaling gateways
*
```

```
SNSLI:SNLINK=1,IPADDR=123.123.1.200,IPADDR2=123.123.2.200,SS7MD=ITU14,SG=1;
SNSLI:SNLINK=2,IPADDR=123.123.1.300,IPADDR2=123.123.2.300,SS7MD=ITU14,SG=2;
*
**************************
* Route Configuration Via signaling gateways
*
* Define Signaling Route number and the SG Destination Point Code
*
SNRTI:SNRT=1,DPC=111;   * to reach Signaling Gateway 1
SNRTI:SNRT=2,DPC=222;   * to reach Signaling Gateway 2
*
SNRTI:SNRT=3,DPC=555;   * to reach network MSC 1
SNRTI:SNRT=4,DPC=444;   * to reach network MSC 2
*
* Define Signaling Route to Signaling Gateway relationship
*
SNRLI:SNRL=1,SNRT=1,SG=1;
SNRLI:SNRL=2,SNRT=1,SG=2;
*
SNRLI:SNRL=3,SNRT=2,SG=1;
SNRLI:SNRL=4,SNRT=2,SG=2;
*
SNRLI:SNRL=5,SNRT=3,SG=1;
SNRLI:SNRL=6,SNRT=3,SG=2;
*
SNRLI:SNRL=7,SNRT=4,SG=1;
SNRLI:SNRL=8,SNRT=4,SG=2;
*
**************************
*
* SCCP Parameters:
* SCCP _ CONFIG <local _ spc> <ssf> <options> [<management _ options> [<partner _ id>
<instance>]]
SCCP _ CONFIG 333 0xc 0x0102 0x00000001 0x43 1
*
* Enable SCCP traces:
* SCCP _ TRACE <op _ evt _ mask> <ip _ evt _ mask> <non _ prim _ mask>
SCCP _ TRACE 0x3 0x7 0x3
```

```
*
* Define Remote Signaling Points:
* SCCP_SSR <ssr_id> RSP <remote_spc> <flags> <pc_mask>
SCCP_SSR 1 RSP 444 0 0x0000
SCCP_SSR 2 RSP 555 0 0x0000
*
* Define Local Sub-Systems:
* SCCP_SSR <ssr_id> LSS <local_ssn> <module_id> <flags> <protocol>
SCCP_SSR 3 LSS 0x08 0x2d 0 MAP
*
* Define Remote Sub-Systems:
* SCCP_SSR <ssr_id> RSS <remote_spc> <remote_ssn> <flags>
SCCP_SSR 4 RSS 444 0x08 0
SCCP_SSR 5 RSS 555 0x08 0
*
* Enable SCCP traces:
* SCCP_TRACE <op_evt_mask> <ip_evt_mask> <non_prim_mask>
SCCP_TRACE 0x3 0x7 0x3
*
* TCAP Parameters:
* TCAP_CONFIG <base_ogdlg_id> <nog_dialogs> <base_icdlg_id>
* <nic_dialogs> <options> <dlg_hunt> [<addr_format><partner_id><tcap_inst> ]
TCAP_CONFIG 0x0 8192 0x8000 8192 0x0000 0 0 0x24 1
*
* Enable TCAP traces:
* TCAP_TRACE <op_evt_mask> <ip_evt_mask> <non_prim_mask>
TCAP_TRACE 0x7 0xf 0x0
*
*
* MAP Parametes:
* MAP_CONFIG <options>
MAP_CONFIG 0x0
*
* Enable MAP traces:
* MAP_TRACE <op_evt_mask> <ip_evt_mask> <non_prim_mask>
MAP_TRACE 0xf 0xf 0x15
```

Dual-Resilient M3UA IPSP Configuration Example

The following section provides example configuration files for a dual-resilient M3UA IPSP system handling GSM MAP traffic with local and remote point codes, subsystem numbers, and IP addresses, as shown in Figure 9. In this example, the SCTP associations are single-homed; however, the configuration can be updated to add multi-homing.

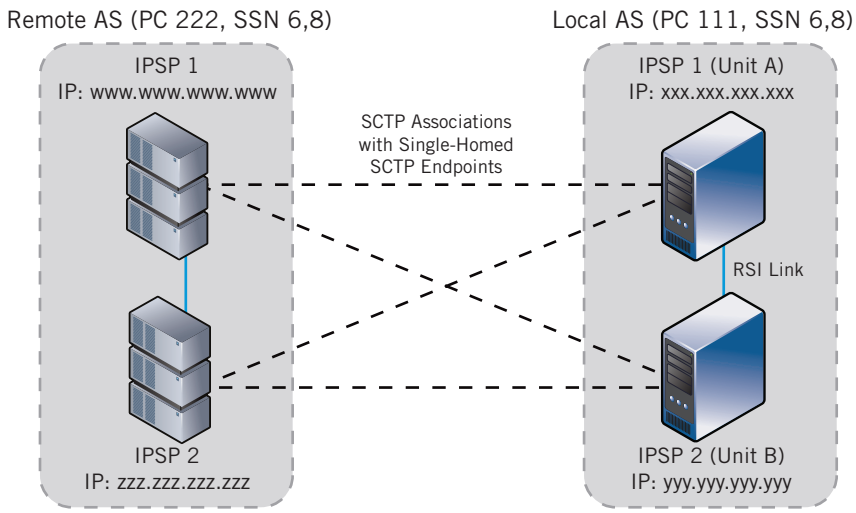

Figure 9. Dual-Resilient M3UA IPSP Configuration Example

Host A config.txt File

The following example *config.txt* file defines Host A in a dual-resilient M3UA IPSP system:

```
************************************************************************
* Example Protocol Configuration File (config.txt)
*
* This file needs to be modified to suit individual circumstances.
* Refer to the relevant Programmer's Manuals for further details.
*
************************************************************************
* Update IP addresses as shown in figure 9
*
CNSYS:IPADDR=xxx.xxx.xxx.xxx,PER=0,DUAL=A;
*
SNSLI:SNLINK=1,IPADDR=www.www.www.www,SNEND=S,SNTYPE=m3ua;
SNSLI:SNLINK=2,IPADDR=zzz.zzz.zzz.zzz,SNEND=S,SNTYPE=m3ua;
*
SNAPI:AS=1,OPC=111,RC=1,TRMD=ls;
*
```

```
SNRAI:RAS=1,RC=1,DPC=222;
*
SNALI:SNAL=1,RAS=1,SNLINK=1;
SNALI:SNAL=2,RAS=1,SNLINK=2;
*
* SCCP Parameters:
* SCCP _ CONFIG <local _ spc> <ssf> <options> [<management _ options> [<partner _ id>
<instance>]]
SCCP _ CONFIG 111 0xc 0x0102 0x00000001 0x53 0
*
*
* Enable SCCP traces:
* SCCP _ TRACE <op _ evt _ mask> <ip _ evt _ mask> <non _ prim _ mask>
SCCP _ TRACE 0x3 0x7 0x3
*
* Define Remote Signaling Points:
* SCCP _ SSR <ssr _ id> RSP <remote _ spc> <flags> <pc _ mask>
SCCP _ SSR 1 RSP 222 0 0x0000
*
* Define Local Sub-Systems:
* SCCP _ SSR <ssr _ id> LSS <local _ ssn> <module _ id> <flags> <protocol>
SCCP _ SSR 2 LSS 0x08 0x2d 0 MAP
SCCP _ SSR 3 LSS 0x06 0x2d 0 MAP
*
* Define Remote Sub-Systems:
* SCCP _ SSR <ssr _ id> RSS <remote _ spc> <remote _ ssn> <flags>
SCCP _ SSR 4 RSS 222 0x08 0
SCCP _ SSR 5 RSS 222 0x06 0
*
* Enable SCCP traces:
* SCCP _ TRACE <op _ evt _ mask> <ip _ evt _ mask> <non _ prim _ mask>
SCCP _ TRACE 0x3 0x7 0x3
*
* TCAP Parameters:
* TCAP _ CONFIG <base _ ogdlg _ id> <nog _ dialogs> <base _ icdlg _ id>
* <nic _ dialogs> <options> <dlg _ hunt> [<addr _ format><partner _ id><tcap _ inst> ]
TCAP _ CONFIG 0x0 8192 0x8000 8192 0x0000 0 0 0x34 0
*
```

```
* Enable TCAP traces:
* TCAP _ TRACE <op _ evt _ mask> <ip _ evt _ mask> <non _ prim _ mask>
TCAP _ TRACE 0x7 0xf 0x0
*
*
* MAP Parametes:
* MAP _ CONFIG <options>
MAP _ CONFIG 0x0
*
* Enable MAP traces:
* MAP _ TRACE <op _ evt _ mask> <ip _ evt _ mask> <non _ prim _ mask>
MAP _ TRACE 0xf 0xf 0x15
```

Host B config.txt File

The following example *config.txt* file defines Host B in a dual-resilient M3UA IPSP system:

```
*****************************************************************************
* Example Protocol Configuration File (config.txt)
*
* This file needs to be modified to suit individual circumstances.
* Refer to the relevant Programmer's Manuals for further details.
*
*****************************************************************************
*
* Update IP addresses as shown in figure 9
*
CNSYS:IPADDR=yyy.yyy.yyy.yyy,per=0,DUAL=B;
*
SNSLI:SNLINK=1,IPADDR=www.www.www.www,SNEND=S,SNTYPE=M3UA;
SNSLI:SNLINK=2,IPADDR=zzz.zzz.zzz.zzz,SNEND=S,SNTYPE=M3UA;
*
SNAPI:AS=1,OPC=111,RC=1,TRMD=ls;
*
SNRAI:RAS=1,RC=1,DPC=222;
*
SNALI:SNAL=1,RAS=1,SNLINK=1;
SNALI:SNAL=2,RAS=1,SNLINK=2;
*
```

```
* SCCP _ CONFIG <local _ spc> <ssf> <options> [<management _ options> [<partner _ id>
<instance>]]
SCCP _ CONFIG 111 0x0c 0x102 0x00000001 0x43 1
*
SCCP _ TRACE 0x3 0x7 0x3
*
*SCCP _ SSR [<nc _ id>] <ssr _ id> RSP <remote _ spc> <rsp _ flags> [<pc _ mask>]
*
SCCP _ SSR 1 RSP 222 0 0
*
*SCCP _ SSR [<nc _ id>] <ssr _ id> LSS <local _ ssn> <module _ id> <lss _ flags> <protocol>
*
SCCP _ SSR 2 LSS 0x8 0x2d 0 MAP
SCCP _ SSR 3 LSS 0x6 0x2d 0 MAP
*
*
*SCCP _ SSR [<nc _ id>] <ssr _ id> RSS <remote _ spc> <remote _ ssn> <rss _ flags>
*
SCCP _ SSR 4 RSS 222 0x8 0
SCCP _ SSR 5 RSS 222 0x6 0
*
* TCAP Parameters:
* TCAP _ CONFIG <base _ ogdlg _ id> <nog _ dialogs> <base _ icdlg _ id>
*        <nic _ dialogs> <options> <dlg _ hunt> [<addr _ format>]
TCAP _ CONFIG 0x0 8192 0x8000 8192 0x0000 0 0 0x24 1
*
* Enable TCAP traces:
* TCAP _ TRACE <op _ evt _ mask> <ip _ evt _ mask> <non _ prim _ mask>
TCAP _ TRACE 0x7 0xf 0x0
*
MAP _ CONFIG 0x0
*
MAP _ TRACE 0xf 0xf 0x15
*
*
```

Example M3UA system.txt Files

The *system.txt* file examples in this section are for hosts running under Windows®, however small changes to the FORK_PROCESS commands will allow these examples to be used with a Linux- or Solaris-based host instead. Table 2 provides examples of using the FORK_PROCESS command under Windows®, Solaris and Linux:

Operating System	FORK_PROCESS Command Example
Windows®	FORK_PROCESS m3ua_nt.exe
Solaris	./FORK_PROCESS m3ua_sol
Linux	./FORK_PROCESS m3ua_lnx6

Table 2. FORK_PROCESS Command Examples

Special considerations apply for using SCTP with Solaris 10 and Linux releases that have a built-in SCTP stack. For information, see "Appendix A: Using SCTP with Solaris 10 and Linux Releases That Have a Built-In SCTP Stack," later in this application note.

Host A system.txt File for Hosts Running under Windows®

The following example *system.txt* file for Host A contains REDIRECT commands that provide a mechanism for sending messages to Host B for dual resiliency:

```
* Example system.txt for the Windows Development Package.
*
* If necessary, edit this file to reflect your configuration.
*
* Modules running on the host:
*
LOCAL       0x00       * Timer Task
LOCAL       0xcf       * s7_mgt - Management/config task
LOCAL       0xef       * s7_log - Display and logging utility
*
*
LOCAL       0xd0       * SCTPD module
LOCAL       0xd1       * SCTP module
LOCAL       0xd2       * M3UA module
LOCAL       0x33       * SCCP module
LOCAL       0x14       * TCAP module
LOCAL       0x15       * MAP module
LOCAL       0x2d       * MTR or MTU module
LOCAL       0x1d       * User App
*
LOCAL       0x32       * RMM module
LOCAL       0xb0       * RSI module
```

```
LOCAL       0xfd       * rsicmd module
*
REDIRECT     0x22 0xd2     * MTP3 -> M3UA
REDIRECT     0xc2 0xd2     * MBM -> M3UA
*
* Definitions for Unit A:
*
REDIRECT     0x52 0xb0     * RMM to unit B
REDIRECT     0x12 0xb0     * M3UA to unit B
REDIRECT     0x53 0xb0     * SCCP to unit B
REDIRECT     0x34 0xb0     * TCAP to unit B
*
REDIRECT     0x42 0x32     * RMM from unit B
REDIRECT     0x02 0xd2     * M3UA from unit B
REDIRECT     0x43 0x33     * SCCP from unit B
REDIRECT     0x24 0x14     * TCAP from unit B
*
NUM_MSGS 10000
*
* Now start-up all local tasks:
*
FORK_PROCESS tim_nt.exe
FORK_PROCESS tick_nt.exe
FORK_PROCESS sctpd.exe
FORK_PROCESS sctp.exe
FORK_PROCESS m3ua_nt.exe -t
FORK_PROCESS sccp_nt.exe -t
FORK_PROCESS tcp_nt.exe -t
FORK_PROCESS map_nt.exe -t
FORK_PROCESS rmm.exe -m0x32 -d
FORK_PROCESS s7_mgt.exe -d
FORK_PROCESS s7_log.exe -fss7.log

FORK_PROCESS    rsi.exe -m0xb0 -r./rsi_lnk.exe -l1
FORK_PROCESS    rsicmd.exe 0 0x32 0 yyy.yyy.yyy.yyy 9005 0xb0
*
* Insert the IP address of unit B in place of yyy.yyy.yyy.yyy
```

Host B system.txt File

The following example *system.txt* file for Host B contains REDIRECT commands that provide a mechanism for sending messages to Host A for dual resiliency:

```
* Example system.txt for the Windows Development Package.
*
* If necessary, edit this file to reflect your configuration.
*
* Essential modules running on the host:
*
LOCAL       0x00       * Timer Task
*
* Optional modules running on the host:
*
LOCAL       0xcf       * s7_mgt - Management/config task
LOCAL       0xd0       * SCTPD module
LOCAL       0xd1       * SCTP module
LOCAL       0xd2       * M3UA module
LOCAL       0x33       * SCCP module
LOCAL       0x14       * TCAP module
LOCAL       0x15       * MAP module*
LOCAL       0xef       * S7_LOG
LOCAL       0x2d       * mtu
LOCAL       0x32       * RMM
LOCAL       0xb0       * RSI
LOCAL       0xfd       * rsicmd module
*
REDIRECT     0x22 0xd2     * MTP3 -> M3UA
REDIRECT     0xc2 0xd2     * MBM -> M3UA
*
* definitions for unit B:
*
REDIRECT     0x42 0xb0     * RMM to unit A
REDIRECT     0x02 0xb0     * M3UA to unit A
REDIRECT     0x43 0xb0     * SCCP to unit A
REDIRECT     0x24 0xb0     * TCAP to unit A
*
```

```
REDIRECT     0x52 0x32     * RMM from unit A
REDIRECT     0x12 0xd2     * M3UA from unit A
REDIRECT     0x53 0x33     * SCCP from unit A
REDIRECT     0x34 0x14     * TCAP from unit A
*
NUM_MSGS 10000
*
* Now start-up all local tasks:
*
FORK_PROCESS tim_nt.exe
FORK_PROCESS tick_nt.exe
FORK_PROCESS sctpd.exe
FORK_PROCESS sctp.exe
FORK_PROCESS m3ua_nt.exe -t
FORK_PROCESS sccp_nt.exe -t
FORK_PROCESS tcp_nt.exe -t
FORK_PROCESS map_nt.exe -t
FORK_PROCESS rmm.exe -m0x32 -d
FORK_PROCESS s7_mgt.exe -d
FORK_PROCESS s7_log.exe -fss7.log
*

FORK_PROCESS    rsi.exe -m0xb0 -r./rsi_lnk.exe -l1
FORK_PROCESS    rsicmd.exe 0 0x32 1 0 9005 0xb0
*
* Start RSI link in "server" mode as detailed in Appendix C
```

Appendix A: Using SCTP with Solaris 10 and Linux Releases That Have a Built-In SCTP Stack

Solaris 10 and some later Linux releases include a built-in SCTP stack that is sometimes referred to as "native SCTP." The SCTPN module provides an interface to these native SCTP stacks. This module works as follows:

- Uses the same API as the Dialogic® SCTP implementation, but uses the operating system kernel's SCTP stack instead of the Dialogic DSI SCTP stack.
- Imposes no hard limits on the number of SCTP associations and streams that can be used. Instead, limits are determined by the limitations of the operating system's SCTP stack.
- The SCTPD module **must not be running** when the SCTPN module is used.

To use the SCTPN module to interface with the native SCTP stack, modify the *system.txt* file as follows:

1. Remove the following lines from the *system.txt* file:

```
FORK _ PROCESS ./sctp
FORK _ PROCESS ./sctpd
```

2. Add this new line to start the SCTPN module:

```
FORK _ PROCESS ./sctpn
```

3. Replace the LOCAL declaration for SCTPD with a REDIRECT command to ensure that messages previously routed to SCTPD are now routed to SCTPN:

   Remove:

```
  LOCAL 0xd0 * SCTPD module
```

   Replace with:

```
  REDIRECT 0xd0 0xd1 * SCTPD messages sent to SCTPN
```

4. Use the gctload -Ci parameter to inform the SCTPN module of congestion events:

```
./gctload –Ci0xd1
```

For more see the Dialogic® SS7 Protocols SCTP Programmer's Manual.

Appendix B: Signaling Gateway Configuration

Detailed configuration of SGs is beyond the scope of this application note. SGs in the signaling network are usually provided by a third party, and configuration requirements should be available in the applicable product documentation.

The high-level configuration should be similar for most SGs, including the configuration of SCTP associations to each host, a remote ASP for each host, a single remote AS that runs on the remote ASPs, and an RK identifying SS7 parameters such as the point code of the service and referencing the AS.

Table 3 lists the high-level configuration information required for an SG by component:

Components	Required Configuration Information
Each association and remote ASP	• Remote IP addresses and port numbers • Local IP addresses and port numbers
Each remote AS	• Remote ASPs on which the AS is supported • RK identifying SS7 parameters such as the point code of the service • RC used by both the SG and hosts to identify the service

Table 3. High-Level Configuration Information Required for a Signaling Gateway

Appendix C: Inter-Chassis Communication

The Remote Socket Interface (RSI) software takes messages that have a destination other than that of the RSI itself and sends them to a peer RSI task on the remote end of a TCP/IP Ethernet. This communication uses TCP/IP sockets, with one side acting as a server, and the other acting as a client.

At the receiving end, the RSI takes messages received from the Ethernet and delivers them through the local message-passing system to the task identified by the original message destination (header destination field). If the communication between two RSI tasks over an Ethernet fails, messages passed to the RSI for transmission over the Ethernet are discarded.

For dual-chassis fault tolerance, use the FORK_PROCESS command in the *system.txt* file to configure two RSI tasks (one for each half of the system). Assign the same module identifier to each task, and declare this identifier on both systems with a LOCAL definition. The module identifier is usually 0xb0.

The RSI program takes the task module identifier as an optional command line parameter prefixed by –m, for example:

```
./rsi -m0xb0
```

Messages sent to the module identifier assigned to the RSI are processed by the local RSI task and not passed over the Ethernet.

The rsicmd utility can be used to activate the RSI links between the master and each slave host. This utility configures one side of the system as a client and the other as a server. The syntax for rsicmd is:

rsicmd <link_id> <conc_id> <link_type> <rem_addr> <rem_port> [<rsi_id>]

Where:

<link_id> Logical identifier for this particular communication channel. RSI selects an outgoing channel by matching the message instance value (set by the GCT_set_instance() library function to the **<link_id>** value. For most dual-resilient systems, a single RSI link between the two halves is sufficient, and therefore, **<link_id>** should be set to 0.

<conc_id> Module identifier for the task that will receive a message whenever the RSI link fails. This module should exist within the system, such that when these status messages are issued by RSI, they are received and then released by this module.

<link_type> and **<rem_addr>** Set these parameters according to Table 4:

Connection Type	Value of <link_type>	Value of <rem_addr>
Client	0 (client)	IP address of remote end
Server	1 (server)	0

Table 4. Values for the link_type and rem_addr Parameters

<rem_port> Specifies the TCP/IP socket port used for the RSI link. Each RSI link has a unique **<link_id>** value, and each must have a unique port value, starting from 9000.

<rsi_id> Is optional and identifies the RSI module used for message passing.

RSI Link Configuration and Activation Example

The following example shows rsicmd commands for inter-chassis communication:

Chassis A (client)

```
*rsicmd   <link _ id>   <conc _ id>   <link _ type>   <rem _ addr>    <rem _ port> [<rsi _ id>]
 rsicmd        0            0xef            0         193.195.185.8       9000
```

Chassis B (server)

```
*rsicmd   <link _ id>   <conc _ id>   <link _ type>   <rem _ addr>    <rem _ port> [<rsi _ id>]
 rsicmd        0            0xef            1               0             9000
```

Accessing Partner Modules Using RSI

To allow access to partner modules using RSI, insert a REDIRECT statement in *system.txt* for all destinations in the other half of the system that should be accessible through the RSI link. The destinations are the module ID values given to the remote ISUP, TUP, SCCP, and/or TCAP protocol.

For example, a dual ISUP system will consist of two ISUP halves, each with a default module ID of 0x23. The configuration data assigned to ISUP on Unit A indicates the other ISUP half (on unit B) is addressed as module ID 0x73, hence there will be a REDIRECT statement on Unit A to route any messages for module ID 0x73 via RSI.

On Unit B, there should be a REDIRECT statement to route all messages received by RSI on Unit B for destination 0x73 to the local ISUP module identifier, identified as 0x23.

Figure 10 depicts how RSI messages are rerouted between Units A and B:

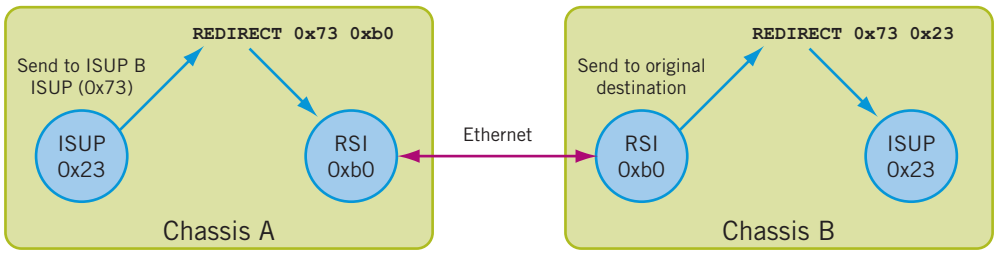

Figure 10. Use of REDIRECT and RSI for ISUP-to-ISUP Communication

Table 5 lists the module ID values that should be used on the two units of a dual-resilient system:

Module	Unit A setting	Unit B setting
Local RMM	0x32	0x32
Partner RMM	0x52	0x42
Local M3UA	0xd2	0xd2
Partner M3UA	0x12	0x02
Local ISUP	0x23	0x23
Partner ISUP	0x73	0x63
Local SCCP	0x33	0x33
Partner SCCP	0x53	0x43
Local TCAP	0x14	0x14
Partner TCAP	0x34	0x24
Local MAP	0x15	0x15
Local IS41	0x25	0x25
Local INAP	0x35	0x35

Table 5: Module ID Values in a Dual-Resilient System

References

[RFC 2719] *Framework Architecture for Signaling Transport*
http://www.ietf.org/rfc/rfc2719

[RFC 4666] *Signaling System 7 (SS7) Message Transfer Part 3 (MTP3) - User Adaptation Layer (M3UA)*
http://www.ietf.org/rfc/rfc4666

Acronyms

AS	Application Server
ASP	Application Server Process
CIC	Circuit Identification Code
DSI	Distributed Signaling Interface
DPC	Destination Point Code
GSM	Global System for Mobile Communications
IETF	Internet Engineering Task Force
INAP	Intelligent Network Application Part
IPSP	IP Server Process
ISUP	ISDN User Part
M3UA	MTP Level 3 (MTP3) User Adaption Layer
MAP	Mobile Application Part
MSC	Mobile Switching Center
MTP	Message Transfer Part Layer
OPC	Originating Point Code
RC	Routing Context
RFC	Request for Comments
RK	Routing Key
RMM	Resilient MTP Management
RSI	Remote Socket Interface
SCCP	Signaling Connection Control Part
SCTP	Stream Control Transmission Protocol
SG	Signaling Gateway
SS7	Signaling System 7
STP	Signal Transfer Point
TCAP	Transaction Capabilities Application Part
TUP	Telephone User Part

For More Information

Dialogic® Distributed Signaling Interface Protocol Stacks user documentation

Dialogic® Distributed Signaling Interface (DSI) – Signaling and SS7 Products on the Dialogic Service Center website

Dialogic®
Making Innovation Thrive™

www.dialogic.com

Dialogic Corporation
9800 Cavendish Blvd., 5th floor
Montreal, Quebec
CANADA H4M 2V9

05/10 11770-01