Critical Release Notice

Publication number: 297-2621-370
Publication release: Standard 10.03

The content of this customer NTP supports the SN07 (DMS) software release.

Bookmarks used in this NTP highlight the changes between the baseline NTP and the current release. The bookmarks provided are color-coded to identify release-specific content changes. NTP volumes that do not contain bookmarks indicate that the baseline NTP remains unchanged and is valid for the current release.

Bookmark Color Legend

Black: Applies to new or modified content for the baseline NTP that is valid through the current release.

Red: Applies to new or modified content for NA017 that is valid through the current release.

Blue: Applies to new or modified content for NA018 (SN05 DMS) that is valid through the current release.

Green: Applies to new or modified content for SN06 (DMS) that is valid through the current release.

Purple: Applies to new or modified content for SN07 (DMS) that is valid through the current release.

Attention!
Adobe Acrobat Reader 5.0 or higher is required to view bookmarks in color.

Publication History

September 2004

For the SN07 (DMS) release, 10.03, the following changes were added:

Volume 1

Added additional NetworkBuilder-related data schema information to the CAINPARM table to address CR Q00816405.

Volume 2

No changes

Volume 3

Added notes for CAIN parameter TRTMTCD_COMPCODE_ZAPPED_ZERO to address CR Q00816405.

Volume 4

No changes

Volume 5

No changes

September 2003

For the SN06 (DMS) release, 10.02, the following changes were added:

Volume 1

SN06 (DMS) Standard release 10.02. Added LNP_EVALUATE_AFTER_OTC_CIC information per CR Q00 509677-06.

Volume 2

No changes

Volume 3

No changes

Volume 4

No changes

Volume 5

No changes

297-2621–370

Digital Switching Systems

UCS DMS-250

NetworkBuilder Application Guide, Volume 1 of 5

UCS17 Standard 10.01 July 2002

NORTEL NETWORKS™

How the world shares ideas.

Digital Switching Systems

UCS DMS-250

NetworkBuilder Application Guide, Volume 1 of 5

Publication number: 297-2621–370
Product release: UCS17
Document release: Standard 10.01
Date: July 2002

Printed in the United States of America

Publication history

July 2002

Standard release 10.01 for UCS17 (CSP17) software release.

Document changes affecting functionality

A59033229

— Added text describing new ANI profile screening to the ***JurisdictionInformation*** parameter in Chapter 8, "Outgoing CAIN Message Parameters" of Volume 3.

A59033997

— Added UCS17 – AIN ***ExtensionParameter*** parameter to Table 2–16, "Protocol version control based on stream designation" of Volume 1.

— Added text explaining how the AIN ***ExtensionParameter*** is now included in the ***Info_Analyzed*** message as well as the ***Info_Collected*** message to section "Info_Analyzed" in Chapter 5, "Analyze_Information PIC" of Volume 1.

— Added text explaining how the AIN ***ExtensionParameter*** is now included in the ***Info_Analyzed*** message as well as the ***Info_Collected*** message to section "ExtensionParameter" in Chapter 8, "Outgoing CAIN message parameters" of Volume 3.

A59034551

— Added new data schema table INWATNPA to Table 2–6 "NetworkBuilder–related data schema" in Chapter 2, "Provisioning NetworkBuilder" of Volume 1.

A59034561

— Added an additional PRI origination definition for the ***LATA***, ***Carrier*** and ***CollectedAddressInfo*** parameters in the "Info_Collected TDP–Request" section in "Offhook_Delay trigger" in Chapter 4, "Collect_Information PIC" of Volume 1.

— Added a description of enhancements to the Off_Hook_Delay trigger to the ***Carrier*** parameter section of Chapter 8, "Outgoing CAIN message parameters" of Volume 3.

— Added a description of enhancements to the Off_Hook_Delay trigger to the ***CollectedAddressInfo*** parameter section of Chapter 8, "Outgoing CAIN message parameters" of Volume 3.

— Added a description of enhancements to the Off_Hook_Delay trigger to the ***LATA*** parameter section of Chapter 8, "Outgoing CAIN message parameters" of Volume 3.

— Added a description of enhancements to the IN/1 Tollfree_Service trigger in the "Options" section of the "Trigger Evaluation" section of the "Tollfree_Service_trigger section in Chapter 4, "Collect_Information PIC" of Volume 1.

— Added an additional PRI origination definition for the ***Digits (calling party number)*** and ***Digits (LATA)*** parameters in the "Start" section in Chapter 7, "Outgoing IN/1 messages" of Volume 3.

December 2000

Standard release 09.02 for UCS14 (CSP14) software release.

Document changes affecting functionality

A60008663

— Added steps to check for the *Furnish_AMA_Information* component for the *Analyze_Route* and *Send_To_Resource* message processing in Chaper 1, "NetworkBuilder call processing," of Volume 2.

— Added data for the *Furnish_AMA_Information* component to Chapter 10, "Incoming CAIN messages," of Volume 3.

A60008667

— Added *Furnish_AMA_Information* procedure and other *Furnish_AMA_Information* data to Chapter 4, "SCP simulator," of Volume 5.

— Added the *CUSTOMER_SELECTOR* field to Step 2 of Procedure 4–4 in Volume 5.

A60008691

— Added procedures for the *PRIVATE_FACILITY_GROUP_USERID* and *RESTRICT_NETBUSY_BUSYCAUSE* parameters to Step 11 in Chapter 2, "Provisioning NetworkBuilder," of Volume 1.

A60008693

— Added subsection, “CAIN parameter OFFHKDEL_TRIG_AFTER_TREAT,” to the “Offhook_Delay trigger,” section in Chapter 4, “Collect_Information PIC,” of Volume 1.

AX1377

— Added text to the “Trigger evaluation” subsection in the “Tollfree_Service trigger” section in Chapter 4, “Collect_Information PIC,” of Volume 1. This text contains information about the 10–digit address requirement for the trigger action to occur.

September 2000

Preliminary release 09.01 for UCS14 (CSP14) software release.

Document changes affecting functionality

A60008441

— Added CDR fields AMABAFMD, AMASIZE, and HEXID to Table 2-8, “CDR field descriptions,” in Volume 1.

— Modified the definition for the ***AMAslpID*** parameter in Chapter 12, “Incoming CAIN message parameters,” of Volume 3.

A60008663

— Added the ***Furnish_AMA_Information*** message to the “CAIN non-call related messages” section in Chapter 1, “TCAP messaging,” of Volume 3. Also added CAINMGR2 and CAINOM OM groups to the “Associated OMs” section in Chapter 1 of Volume 3.

— Added the ***AMABAFModules*** and the ***AMASetHexABIndicator*** parameters to Table 12-1 of Volume 3.

— Added the ***OverflowBillingIndicator*** and the ***PrimaryBillingIndicator*** parameters to Table 12-1 of Volume 3. Removed UCS13-related information from the definitions of these two parameters.

A60008668

— Added LAMAFAIL, LAMAREXH, AINT1TO, AINT1ATO, AINRSVD1, AINRSVD2, AINRSVD3, and AINRSVD4 registers to the CAINOM OM group in Table 2-4, “NetworkBuilder CAIN-related OM,” of Volume 1.

— Added the FAMAINFO register to the CAINMSGR OM group in Table 2-4. Added the SPAREBLK and SPAROVFL registers to the TFREE533 OM group in Table 2-5, “NetworkBuilder TR–NWT–000533 IN/1-related OMs,” of Volume 1.

A60008690

— Added a subsection, “TR–533 ACG Cause to Treatment Mapping,” to Chaper 2, “Automatic Code Gapping,” in Volume 3.

A60008691

— Added an entry for table CAINPARM parameters, ***PRIVATE_FACILITY_GROUP_USERID*** and ***RESTRICT_NETBUSY_BUSYCAUSE*** to Table 2-6, “NetworkBuilder-related data schema,” of Volume 1.

— Added text to describe enhancements for the ***BusyCause*** and ***UserID*** parameters in Chapter 8, “Outgoing CAIN message parameters,” of Volume 3.

— Added text to describe the enhancements for the ***BusyCause*** parameter by the ***RESTRICT_NETBUSY_BUSYCAUSE*** CAIN office parameter to the “Network_Busy EDP–Request message parameters table in Chapter 6 of Volume 1.

A60008693

— Added an entry for table CAINPARM parameter, ***OFFHKDEL_TRIG_AFTER_TREAT***, to Table 2-6 “NetworkBuilder-related data schema,” in Volume 1.

— Added the DIGITS option for Step 6 in the “Provisioning the Offhook_Delay trigger” section in Chapter 4, “Collect_Information PIC,” of Volume 1.

SR60105565

— Added a note to Step 11, “Define the message-related parameters CAIN_PROTOCOL_STREAM,” in Chapter 2 of Volume 1.

May 2000

Standard release 08.02 for software release UCS13 (CSP13).

Document changes affecting functionality

Non–feature updates

— Re-write of outgoing ***ChargeNumber*** parameter population section in Volume 3, Chapter 8.

— Modified PRI terminations, PTS terminations and Restrictions under incoming ***ChargeNumber*** parameter section in Volume 3, Chapter 12.

March 2000

Preliminary release 08.01 for software release UCS13 (CSP13).

Document changes affecting functionality

A60098242

— Added AMP1 parameter in Volume 3, Chapter 8 snd 12 and updated UserID parameter encoding format in Volume 3, Chapter 8.

AT60007655

— Updated *Switch_Hook_Flash* tone detection range from .6 seconds to 40 milliseconds in Volume 3, Chapter 3.

SR60089576

— Added new log CAIN303 and updated CAINPARM parameters in Volume 1, Chapter 2.

A60008135

— Updated the LNP_FOR_RX_SELECTOR parameter in Volume 1, Chapter 2.

Non–feature updates

— Included size restrictions for ***FlexParmBlock*** and ***IPReturnBlock*** applicable to **Send_To_Resource** or **Connect_To_Resource** connections in Volume 4, Chapter 1.

November 1999

Standard release 07.02 for software release UCS12 (CSP12).

Document changes affecting functionality

Non–feature updates

— Updated Appendix–B engineering calculations in Volume 1. to include NCCBS parameter and updated NUM_CAIN_ECCBS .

August 1999

Preliminary release 07.01 for software release UCS12 (CSP12).

New functionality

No new functionality has been added.

Document changes affecting functionality

S60089224

— Updated IN1/ Connect message under Volume 2 Call processing.

PSD07023

— Updated the **Analyze_Route** message ***OutpulseNumber*** parameter in Chapter 10, Volume 3 .

— Updated route advancing for Network_Busy, O_Called_Party_Busy, O_No_Answer triggers in Chapters 6, 7 and 8, Volume 1.

Non–feature updates

— Updated the ***CollectedAddressInfo*** parameter population rules in Chapter 8, Volume 3.

— Updated the **O_Abandon**, **O_Disconnect**, and **Timeout** EDP messages to include the CAIN100 log in Chapter 3, Volume 3 .

— Updated Appendix–B engineering calculations in Volume 1.

Contents

Volume 1 of 5

Volume 2 of 5

Figures

Volume 3 of 5

Volume 4 of 5

Volume 5 of 5

About this document

This document describes NetworkBuilder's support for Carrier Advanced Intelligent Network (AIN) and Bellcore *TR-NWT-000533* Intelligent Network (IN/1) functionality. Information is provided for understanding, planning, datafilling, and testing.

Data tables and commands that support NetworkBuilder functionality are discussed in this document. The details provided in this document are limited to how these items apply to NetworkBuilder's support for Carrier AIN and *TR-NWT-000533* IN/1.

The intent of this document is to describe NetworkBuilder functions. These functions include setting up the switch to perform AIN and *TR-NWT-000533* IN/1, delivering messages, populating parameters, and processing messages received from an intelligent service control point (SCP). Refer to SCP provider documentation for more information on your SCP.

Intended audience

This publication assists telecommunications engineers, technicians, switching system developers, operating company personnel, and anyone else who requires technical information on NetworkBuilder functionality.

This document assumes the user's switch is installed, commissioned, and active.

Personnel implementing this feature require the following:

- Table Editor training
- Nortel Networks approved datafill, translations, and maintenance training

How this document is organized

The volumes and chapters are organized as follows:

Volume 1

Chapter 1, NetworkBuilder overview

Chapter 1 provides information on the NetworkBuilder's support for Carrier AIN and Bellcore *TR-NWT-000533* IN/1, the concept of a call model, and the AIN network model

Chapter 2, Provisioning NetworkBuilder

Chapter 2 provides instructions for provisioning the switch for NetworkBuilder.

Chapter 3, O_Null PIC

Chapter 3 provides general information on understanding and provisioning the **O_Null** PIC, ***Origination_Attempt*** TDP, and *Off_Hook_Immediate* trigger.

Chapter 4, Collect_Information PIC

Chapter 4 provides general information on understanding and provisioning the **Collect_Information** PIC, ***O_Feature_Requested*** and ***Info_Collected*** TDPs, and *O_Feature_Requested*, *Offhook_Delay*, *Shared_Interoffice_Trunk*, and *PRI_B-Channel* triggers.

Chapter 5, Analyze_Information PIC

Chapter 5 provides general information on understanding and provisioning the **Analyze_Information** PIC, ***Info_Analyzed*** TDP, and *Specific_Feature_Code*, *Customized_Dialing_Plan*, *Specific_Digit_String*, and *Office_Code* triggers.

Chapter 6, Select_Route PIC

Chapter 6 provides general information on understanding and provisioning the **Select_Route** PIC, ***Network_Busy*** DP, and *Network_Busy* trigger/event.

Chapter 7, Send_Call PIC

Chapter 7 provides general information on understanding and provisioning the **Send_Call** PIC, ***O_Term_Seized*** EDP, ***O_Called_Party_Busy*** DP, ***O_Mid_Call*** TDP, *O_Term_Seized* event, *O_Called_Party_Busy* trigger/event, and *O_IEC_Reorigination* trigger.

Chapter 8, O_Alerting PIC

Chapter 8 provides general information on understanding and provisioning the **O_Alerting** PIC, ***O_Answer*** EDP, ***O_No_Answer*** DP, ***O_Mid_Call*** TDP,

O_Answer event, *O_No_Answer* trigger/event, and *O_IEC_Reorigination* trigger.

Chapter 9, O_Active and O_Suspended PICs

Chapter 9 provides general information on understanding and provisioning the **O_Active** and **O_Suspended** PICs, ***O_Mid_Call*** DP, ***O_Disconnect*** EDP, *O_IEC_Reorigination* trigger, *Timeout* event, and *O_Disconnect* event.

Chapter 10, T_Null and O_Suspended PICs

Chapter 10 provides general information on understanding and provisioning the terminating call model, the **T_Null** PIC, ***Termination_Attempt*** TDP, *Termination_Attempt* trigger.

Appendix A, Service Migration

Appendix A provides an example of migrating N00 services from IN/1 (not Bellcore's *TR-NWT-000533* IN/1 specification) to CAIN.

Appendix B, Engineering guidelines

Appendix B provides engineering guidelines you should consider when provisioning your switch.

Volume 2

Chapter 1, NetworkBuilder call processing

Chapter 1 provides a series of flowcharts illustrating NetworkBuilder call processing

Volume 3

Chapter 1, TCAP messaging

Chapter 1 provides an overview of TCAP messaging

Chapter 2, Automatic Code Gapping

Chapter 2 provides information for Automatic Code Gapping

Chapter 3, Event processing

Chapter 3 provides information for EDP call processing and EDP messages

Chapter 4, Call configuration model

Chapter 4 provides an overview of call configurations and diagrams of supported call configurations

Chapter 5, Termination_Notification processing

Chapter 5 provides information on **`Termination_Notification`** and **`Termination_Notification`** messaging scenarios

Chapter 6, Outgoing CAIN messages

Chapter 6 provides information on CAIN outgoing messages

Chapter 7, Outgoing IN/1 messages

Chapter 7 provides information on outgoing Bellcore *TR-NWT-000533* IN/1 messages

Chapter 8, Outgoing CAIN message parameters

Chapter 8 provides information on outgoing CAIN message parameters and extension parameters

Chapter 9, Outgoing IN/1 message parameters

Chapter 9 provides information on outgoing Bellcore *TR-NWT-000533* IN/1 message parameters

Chapter 10, Incoming CAIN messages

Chapter 10 provides information on incoming CAIN messages and incoming CAIN message processing

Chapter 11, Incoming IN/1 messages

Chapter 11 provides information on incoming Bellcore *TR-NWT-000533* IN/1 messages and incoming IN/1 message processing

Chapter 12, Incoming CAIN message parameters

Chapter 12 provides information on incoming CAIN message parameters and extension parameters

Chapter 13, Incoming IN/1 message parameters

Chapter 13 provides information on incoming Bellcore *TR-NWT-000533* IN/1 message parameters

Volume 4

Chapter 1, Conversational messages

Chapter 1 provides an overview of conversational TCAP messages

Chapter 2, STR-Connections

Chapter 2 provides information on conversational **`Send_To_Resource`** messages and STR-Connections to an IP

Chapter 3, CTR-Connections

Chapter 2 provides information on conversational **`Connect_To_Resource`** messages and CTR-Connections to an IP

Chapter 4, Virtual IP interaction

Chapter 4 provides information Virtual IP messaging scenarios and Virtual IP messaging

---

Volume 5

Chapter 1, NetworkBuilder tools

Chapter 1 provides an overview of NetworkBuilder tools

Chapter 2, CAINSCPT

Chapter 2 provides general information on the CAINSCPT tool

Chapter 3, CAINTEST

Chapter 3 provides general information on CAINTEST commands and messaging

Chapter 4, SCP simulator

Chapter 4 provides general information on the SCP simulator and SCP simulator CAINTEST interactions

Chapter 5, NetworkBuilder SOC functionality

Chapter 5 provides general information on the SOC tool and NetworkBuilder SOCs

Chapter 6, TRAVER

Chapter 6 provides general information on the TRAVER tool

Chapter 7, VPTRACE

Chapter 7 provides general information on the VPTRACE tool

How to check the version and issue of this document

The version and issue of the document are indicated by numbers, for example, 01.01.

The first two digits indicate the version. The version number increases each time the document is updated to support a new software release. For example, the first release of a document is 01.01. In the *next* software release cycle, the first release of the same document is 02.01.

The second two digits indicate the issue. The issue number increases each time the document is revised but rereleased in the *same* software release cycle. For example, the second release of a document in the same software release cycle is 01.02.

To determine which version of this document applies to the software in your office and how documentation for your product is organized, check the release information in the *UCS DMS-250 Master Index of Publications*, 297-2621-001.

This document is written for UCS DMS-250 offices. More than one version of this document may exist. To determine whether you have the latest version of this document and how documentation for your product is organized, check the release information in the *UCS DMS-250 Master Index of Publications*, 297-2621-001.

References in this document

The following documents are referred to in this document:

- *AIN 0.2 Switch-SCP/Adjunct Interface Generic Requirements*, Bellcore Specification GR-1299-CORE, Issue 3
- *AIN 0.2 Switching Systems Generic Requirements*, Bellcore Specification GR-1298-CORE, Issue 3
- *AIN 0.2 Switch Intelligent Peripheral Interface Generic Requirements*, Bellcore Specification GR-1129-CORE, Issue 2
- *Bell Communications Research Specification for Signalling System Number 7*, Bellcore Specification GR-246-CORE, Issue 1
- *Bellcore Automatic Message Accounting Format (BAF) Requirements*, Bellcore Specification GR-1100-CORE, Issue 2
- *Database Services Service Switching Points – Toll-Free Service*, Bellcore Specification TR-NWT-000533, Issue 3, Supplement 1, April 1995
- *Digital Recorded Announcement Machine DRAM and EDRAM Guide*, 297-1001-527
- *DMS-100 Family Software Optionality Control User's Manual*, 297-8991-901
- *DMS-100 Feature Description Manual*, PLN-1001-003

- *FCC Report and Order for Further Notice of Proposed Rulemaking in the Matter of Rules and Policies Regarding Calling Number Identification Service*
- *Switching Systems Generic Requirements for Interexchange Carrier Interconnection Using the Integrated Services Network User Part (ISUP)*, Bellcore Specification GR-394-CORE, Issue 1
- *UCS DMS-250 Software Release Manual for UCS05*, PLN-2621-004
- *UCS DMS-250 Software Release Manual for UCS06*, PLN-2621-004
- *UCS DMS-250 Software Release Manual for UCS07*, PLN-2621-004
- *UCS DMS-250 Software Release Manual for UCS08*, PLN-2621-004
- *UCS DMS-250 Software Release Manual for UCS09*, PLN-2621-004
- *UCS DMS-250 Software Release Manual for UCS11*, PLN-2621-004
- *UCS DMS-250 Software Release Manual for UCS12*, PLN-2621-004
- *UCS DMS-250 Local Number Portability Application Guide*, 297-2621-371
- *UCS DMS-250 CAIN/FlexDial Interactions*, 297-2621-372
- *UCS DMS-250 Commands Reference Manual*, 297-2621-819
- *UCS DMS-250 Conference Circuit Guide*, 297-1001-530
- *UCS DMS-250 Billing Records Application Guide*, 297-2621-395
- *UCS DMS-250 Data Schema Reference Manual*, 297-2621-851
- *UCS DMS-250 FlexDial Framework Application Guide*, 297-2621-390
- *UCS DMS-250 Global Application Guide*, 297-2621-327
- *UCS DMS-250 Logs Reference Manual*, 297-2621-840
- *UCS DMS-250 Master Index of Publications*, 297-2621-001
- *UCS DMS-250 NetworkBuilder AIN 0.2 TCAP Protocol Definition*, PLN-2621-010 (Contact your Nortel Networks Sales Representative for information on obtaining this document.)
- *UCS DMS-250 Office Parameters Reference Manual*, 297-2621-855
- *UCS DMS-250 Operational Measurements Reference Manual*, 297-2621-814
- *UCS DMS-250 PRI RLT Feature Application Guide*, 297-2621-347
- *UCS DMS-250 Programmable Service Node Application Guide*, 297-2621-380
- *UCS DMS-250 Software Optionality Control User's Manual*, 297-2621-301

- *UCS DMS-250 SS7 RLT Feature Application Guide*, 297-2621-345

Information about related documents can be found in either the *UCS DMS-250 Master Index of Publications*, 297-2621-001, or the *Product Documentation Directory*, 297-8991-001.

What precautionary messages mean

The types of precautionary messages used in Nortel Networks Technical Publications include attention boxes and danger, warning, and caution messages.

An attention box identifies information that is necessary for the proper performance of a procedure or task or the correct interpretation of information or data. Danger, warning, and caution messages indicate possible risks.

Examples of the precautionary messages follow.

ATTENTION Information needed to perform a task

ATTENTION

If the unused DS-3 ports are not deprovisioned before a DS-1/VT Mapper is installed, the DS-1 traffic will not be carried through the DS-1/VT Mapper, even though the DS-1/VT Mapper is properly provisioned.

DANGER Possibility of personal injury

DANGER

Risk of electrocution

Do not open the front panel of the inverter unless fuses F1, F2, and F3 have been removed. The inverter contains high-voltage lines. Until the fuses are removed, the high-voltage lines are active, and you risk being electrocuted.

WARNING Possibility of equipment damage

WARNING
Damage to the backplane connector pins
Align the card before seating it, to avoid bending the backplane connector pins. Use light thumb pressure to align the card with the connectors. Next, use the levers on the card to seat the card into the connectors.

CAUTION Possibility of service interruption or degradation

CAUTION
Possible loss of service
Before continuing, confirm that you are removing the card from the inactive unit of the peripheral module. Subscriber service will be lost if you remove a card from the active unit.

Document conventions

This document conforms to the following conventions.

PICs, TDPs, EDPs, triggers, and events

PICs are represented in Helvetica Bold font, for example:

O_Null

TDPs and EDPs are represented in Helvetica Bold-Italic font, for example:

Origination_Attempt

Triggers and events are represented in Helvetica Italic font, for example:

Off_Hook_Immediate

Messaging

Messages are represented in Courier Bold font, for example:

Origination_Attempt

Parameters are represented in Courier Bold-Italic font, for example:

CalledPartyID

ExtensionParameters and any sub-parameters are represented in Courier font, for example:

univeralAccess

Input prompt (>)

An input prompt (>) indicates that the information that follows is a command:

>CAINTEST

Commands and fixed parameters

Commands and fixed parameters that are entered at a MAP terminal are shown in uppercase letters:

>CAINTEST

Variables

Variables are shown in lowercase letters:

>SHOWFLDS fields

The letters or numbers that the variable represents must be entered. Each variable is explained in a list that follows the command string.

Responses

Responses correspond to the MAP display and are shown in a different type:

```
APPLICATION      : cain02
TIMEOUT          : 3     sec
MESSAGE          : info_analyzed
        Cldno    : N00   2145323773
        Clgno    : N00   2143562784
        Chgno    : N00   2143562784
```

The following excerpt from a procedure shows the command syntax used in this document:

1 Show the defined fields by typing the following:

>SHOWFLDS fields
and pressing the Enter key.

where

fields is the type of field to be shown.

Sample entry: **>SHOWFLDS all**

Example of a MAP response:

```
APPLICATION      : cain02
TRANSPORT        : tcap_sccp cain_clid_gt 214
TIMEOUT          : 3     sec
MESSAGE          : info_analyzed
        Cldno    : N00   2145323773
```

```
Clgno     : N00   2143562784
Chgno     : N00   2143562784
```

NetworkBuilder overview

Welcome to NetworkBuilder. NetworkBuilder provides optional intellegent networking (IN/1) and optional advanced intelligent networking (AIN) software to the UCS DMS-250 switching platform. The IN/1 system NetworkBuilder supports is based on Bellcore's *TR-NWT-000533* specifications. The AIN system is called Carrier AIN (CAIN). CAIN is based on Bellcore's AIN 0.2 specifications.

NetworkBuilder call processing changes the way the switch handles a call. Instead of controlling all aspects of a call, the switch off-loads some of the call processing functions to an intelligent service control point (SCP). This off-loading can relieve switch responsibilities and allow you greater control over the call services you offer to your subscribers.

History of IN

Before the concept of intelligent networking was developed, all network functions were performed in-switch. The development of an intelligent network (IN) off-loaded card validation, subscriber screening, and N00 services to a database located on an IN SCP.

In many cases, this database was a shared resource across a carrier's network of switches. The switches communicated with the SCP through CCS7/TCAP.

In most IN models, there was only one query/response transaction per call. The SCP was used as a remote database without intelligence and never took control of a call.

The next generation

The next generation of intelligent networking, AIN 0.2, allows call processing to be off-loaded from the SSP to a customer-defined SCP. The AIN SCP is able to take control of a call and direct call processing. This intelligent SCP may also contain feature logic and necessary databases.

AIN network model

You develop the software on your AIN peripherals to interact with AIN software on the SSP. The AIN network may consist of the following AIN peripherals:

- service switching point (SSP)
- service control point (SCP)
- signal transfer point (STP)
- service management system (SMS)
- service creation environment (SCE)
- intelligent peripheral (IP)

An SSP and an SCP are required for an AIN network. Peripheral software development is independent of SSP software development.

Note 1: CAIN requires the use of a UCS DMS-250 switch as the SSP.

Note 2: Nortel also offers an SCP product called ServiceBuilder.

What is AIN?

The concept of an advanced intelligent network (AIN) was developed by Bellcore to allow

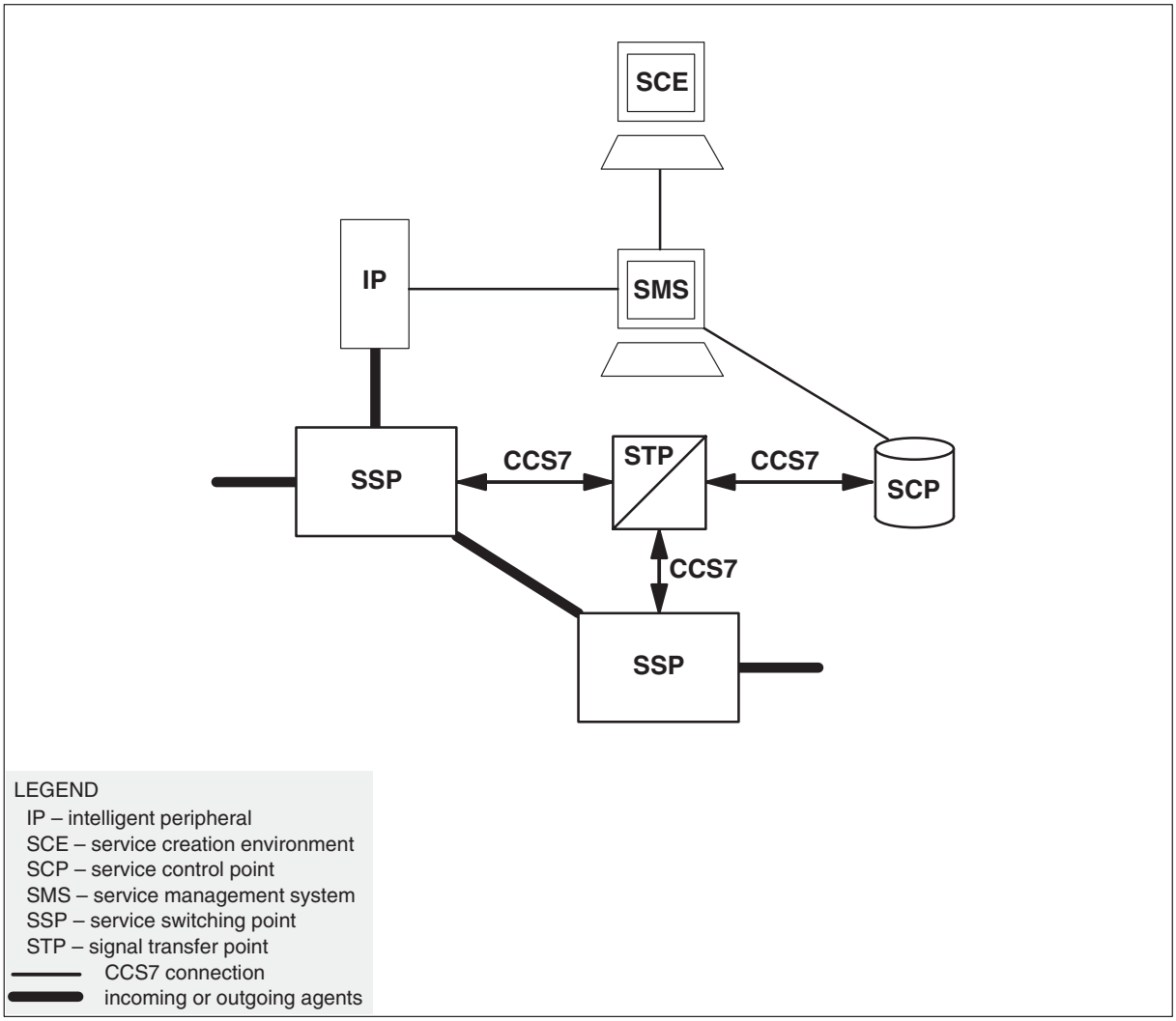
- multivendor hardware
- the AIN service control point (SCP) to take control of a call from the service switching point (SSP)
- transaction capabilities application part (TCAP) messaging using common channel signaling #7 (CCS7)
- rapid design process
- deployment of new customized services
- evolution capabilities

Bellcore specifications

Bellcore specification *TR-NWT-000533* defines the IN/1 functionality supported by NetworkBuilder. *GR-1298-CORE* and *GR-1299-CORE* define AIN 0.2 supported by NetworkBuilder. However, the Bellcore specifications define local exchange carrier (LEC) functions that don't necessarily apply to interexchange carriers (IEC). Also, the Bellcore specifications don't meet the needs of unique IEC applications.

Figure 1-1 shows the hardware components available for designing a NetworkBuilder system. Descriptions of each component follow Figure 1-1.

Figure 1-1
Example of an NetworkBuilder network model



Service switching point The SSP identifies calls requiring AIN service and initiates a dialog with the appropriate AIN service logic in the network.

Note: NetworkBuilder requires the use of a UCS DMS-250 switch as the SSP.

Service control point The SCP contains the service logic and data for AIN services. The SCP responds to requests for services from the SSP. The SCP may also provide a service creation environment (SCE) tool kit that you can use to create and customize services.

Signal transfer point The STP is an optional AIN component. STP functions include link management, routing of messages between SSPs and SCPs, and network recovery.

Service management system The SMS enables provisioning and administration of AIN services.

Service creation environment The SCE provides tools for creation and customization of services residing on the SCP.

Intelligent peripheral The IP is an optional AIN component. The IP contains functions and resources for exchanging information, such as voice announcements and dual-tone multifrequency (DTMF) digit collection with a subscriber.

TCAP messaging

TCAP messaging on CCS7 is the medium for communication between the SSP and SCP. Refer to Volume 3, Chapter 1, "TCAP messaging," for more information.

NetworkBuilder call model

AIN 0.2 introduces the concept of a call model. A call model represents the progression of a call and identifies different points in a call that are eligible to receive processing by the SCP.

Carrier AIN 0.2 utilizes the AIN 0.2 call model. The Carrier AIN 0.2 call model represents the progression of a call and identifies different points in a call that are eligible to receive NetworkBuilder services through interaction with the SCP. Only CAIN-capable originations and terminations are able to access NetworkBuilder.

Why NetworkBuilder?

NetworkBuilder enhances the intelligent network (IN) call processing functions by

- supporting calls originating from an interexchange carrier (IEC) on SS7 Inter-IMT and SS7 Global-IMT trunks
- supporting calls originating from a local exchange carrier (LEC) on FGD trunks
- supporting calls originating from a private branch exchange (PBX) directly linked to the switch by DAL or PRI agencies
- supporting calls originating on AXXESS agents

  Note: For more information on AXXESS agents interacting with NetworkBuilder refer to *UCS DMS-250 CAIN/FlexDial Interactions*.

- supporting calls terminating to an interexchange carrier (IEC) on IMT trunks

  Note: The ISUP92/Q.767 trunk is supported as a terminating agent for NetworkBuilder, however, SS7 parameters not supported by the ISUP92/Q.767 agent are not included in outgoing messages.

  Note: The NetworkBuilder terminating call model is not supported for the ISUP92/Q.767 agent.

- supporting calls terminating to a local exchange carrier (LEC) on FGB, FGC, and FGD trunks
- supporting calls terminating to a private branch exchange (PBX) directly linked to the switch by DAL or PRI agencies
- supporting calls terminating on AXXESS agents

  Note: For more information on AXXESS agents interacting with NetworkBuilder refer to *UCS DMS-250 CAIN/FlexDial Interactions*.

- supplying message encoders and decoders (Refer to Volume 3, Chapter 1, “TCAP messaging,” for more information.)
- allowing the switch to send queries to the SCP
- allowing conversation between the switch and the SCP
- supporting connections to an intelligent peripheral through GR-1129 style interactions
- creating a flexible, expandable table control environment (Refer to Volume 2, “NetworkBuilder call processing,” for more information.)
- supporting the following services as defined by the SCP software:
  - — Virtual Private Networking (VPN)
    - – OFFNET Overflow
    - – Forced ONNET
    - – Alternate Billing Numbers
    - – Origination/Termination Screening
    - – Customized Announcements
    - – Black Box Screening
    - – Universal Access Authorization
    - – Global Virtual Network Services (GVNS)
  - — Subscriber Screening
    - – automatic numbering identification (ANI) Screening

- Authorization Code Screening
- Account Code Screening
- Enhanced Travel Card Services

— N00 services

- Follow Me, Find Me, Do Not Disturb Me
- Call Screening
- Customized Call Branding
- Prepaid services
- Universal Access Authorization
- Toll-free services
- Take-back and Transfer

— Dial 1+ Services

- Speed Dial
- Hotline
- Intra-LATA Presubscription Screening (ANI Screening)
- Account Code Screening
- Bill to Office
- Prepaid services
- CIC Routing and Branding
- Chat Lines

— Customized Dialing Services

- Hotline calls
- Automatic Queries
- Automatic Call Blocking

— Network Select Enhanced Services

- Network Forwarding
- Call Forwarding
- Rerouting on busy signal or no answer
- Network Queuing

— Local Number Portability (LNP)

— Enhanced reorigination services

— Terminating Services

– Caller ID delivery with minimal delivery services tied up

Note: For more information on LNP and NetworkBuilder interactions, refer to the *UCS DMS-250 Local Number Portability Application Guide*.

NetworkBuilder network model

NetworkBuilder requires a service switching point (SSP) and a service control point (SCP). The SSP must be a UCS DMS-250 switch.

When the switch receives a call originating or terminating on a supported agency, NetworkBuilder software checks the user-defined datafill to determine if the call meets the criteria set forth for NetworkBuilder call processing.

The UCS DMS-250 receives a call from an IEC, LEC, or PBX through the following supported agencies:

- SS7 Inter-IMT and SS7 Global-IMT originating agents are supported from IECs.
- FGD originating agents are supported from LECs.
- DAL or PRI agents are supported from PBXs directly linked to a UCS DMS-250.
- AXXESS agents are supported from LECs and PBXs directly linked to a UCS DMS-250.

Note: For more information on AXXESS agent and NetworkBuilder interactions refer to *UCS DMS-250 CAIN/FlexDial Interactions*.

The UCS DMS-250 sends a call to an IEC, LEC, or PBX through the following supported agencies:

- IMT terminating agents are supported to IECs.
- FGB, FGC, FGD terminating agents are supported to LECs.
- DAL or PRI agents are supported to PBXs directly linked to a UCS DMS-250.
- AXXESS agents are supported to LECs and PBXs directly linked to a UCS DMS-250.

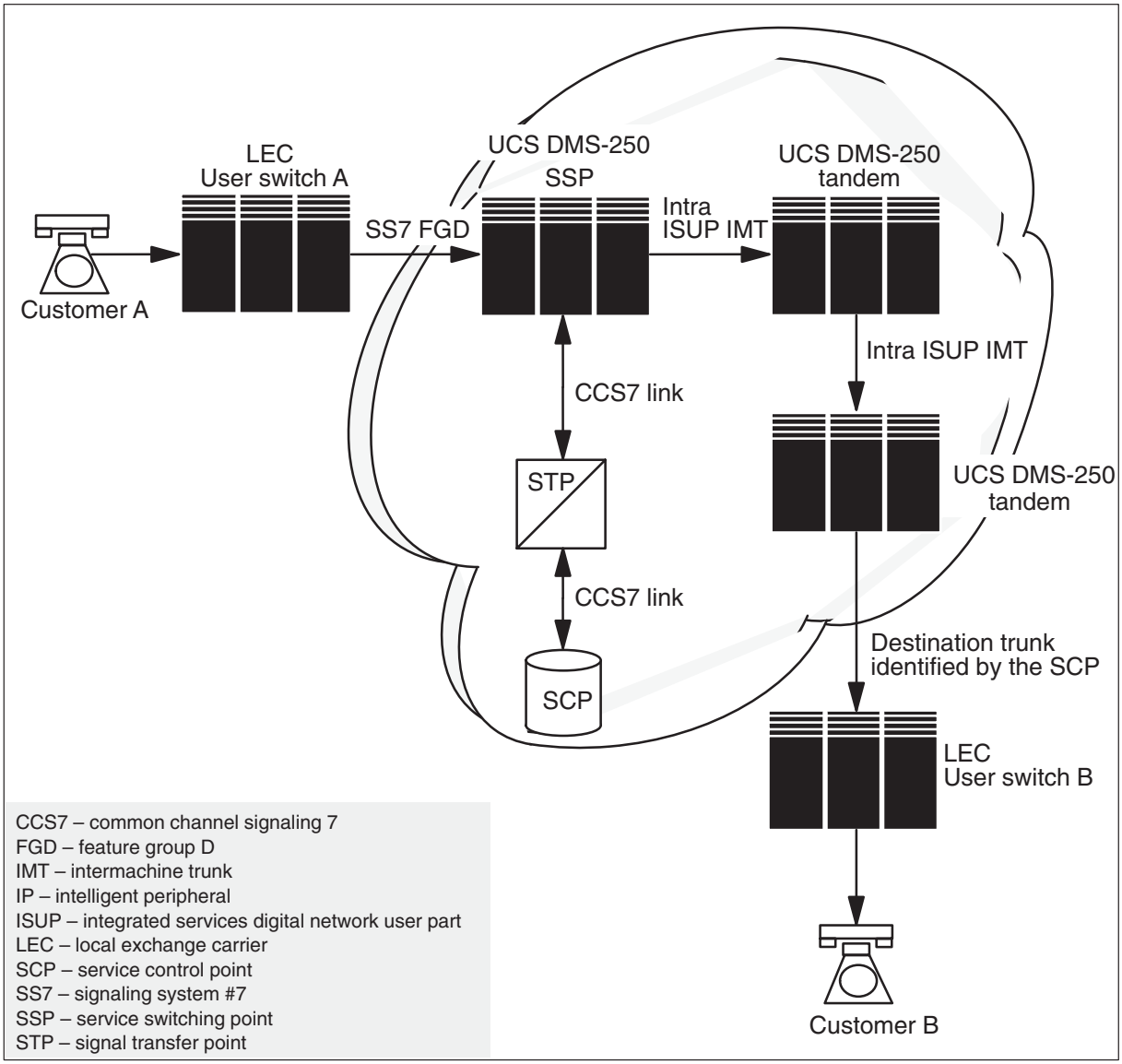
Note: For more information on AXXESS agent and NetworkBuilder interactions refer to *UCS DMS-250 CAIN/FlexDial Interactions*.

When the criteria is met and the action is QUERY or FEAT, a request is sent to the SCP for call processing information. The SCP responds and processing continues on the switch by

- directly routing to a trunk in the network (trunk determined by SCP)
- translating the digits returned by the SCP
- disconnecting the call
- playing an announcement and then disconnecting the call
- continuing in-switch processing as a non-CAIN call
- connecting to an in-switch resource and allowing conversational digit collection
- connecting to a local or remote IP
- authorizing the termination

Figure 1-2 shows a high-level example of a NetworkBuilder call (from a LEC) that receives routing information from the SCP.

Figure 1-2
Example of a LEC NetworkBuilder call



NetworkBuilder call model

The call model divides the call processing logic into key states (known as a point in call[PIC]). The switch performs a defined set of functions within the PIC.

Once certain procedures have occurred within the PIC, the call encounters trigger detection points (TDP) and examines triggers.

Once call processing encounters a TDP, in-switch logic and datafill is consulted to determine if the trigger criteria is met. When met, the switch can off-load call processing to the SCP when directed by datafill. The call may continue in-switch processing or call processing may be suspended and direction requested from the SCP.

NetworkBuilder also supports event detection points (EDP) at several PICs. EDPs are similar to TDPs. When a call queries the SCP, the SCP may return a conversational package which includes a call-related component such as an **`Analyze_Route`**, **`Continue`**, or **`Collect_Information`** message and a non-call related **`Request_Report_BCM_Event`** component containing a list of one or more EDPs that occur later in the call model. The list is used to activate or “arm” EDPs. EDP “arming” indicates that the SCP has informed the switch to send an EDP-Request or EDP-Notification message back to the SCP when the EDP is encountered. The list of EDPs to arm is called the next event list (NEL). After the call-related component is processed the call follows its standard logic, ignoring all TDPs and triggers, until the armed EDP is encountered or EDPs are deactivated.

Figure 1-3 shows the NetworkBuilder originating call model.

Figure 1-4 shows the NetworkBuilder terminating call model.

Figure 1-3
NetworkBuilder originating call model

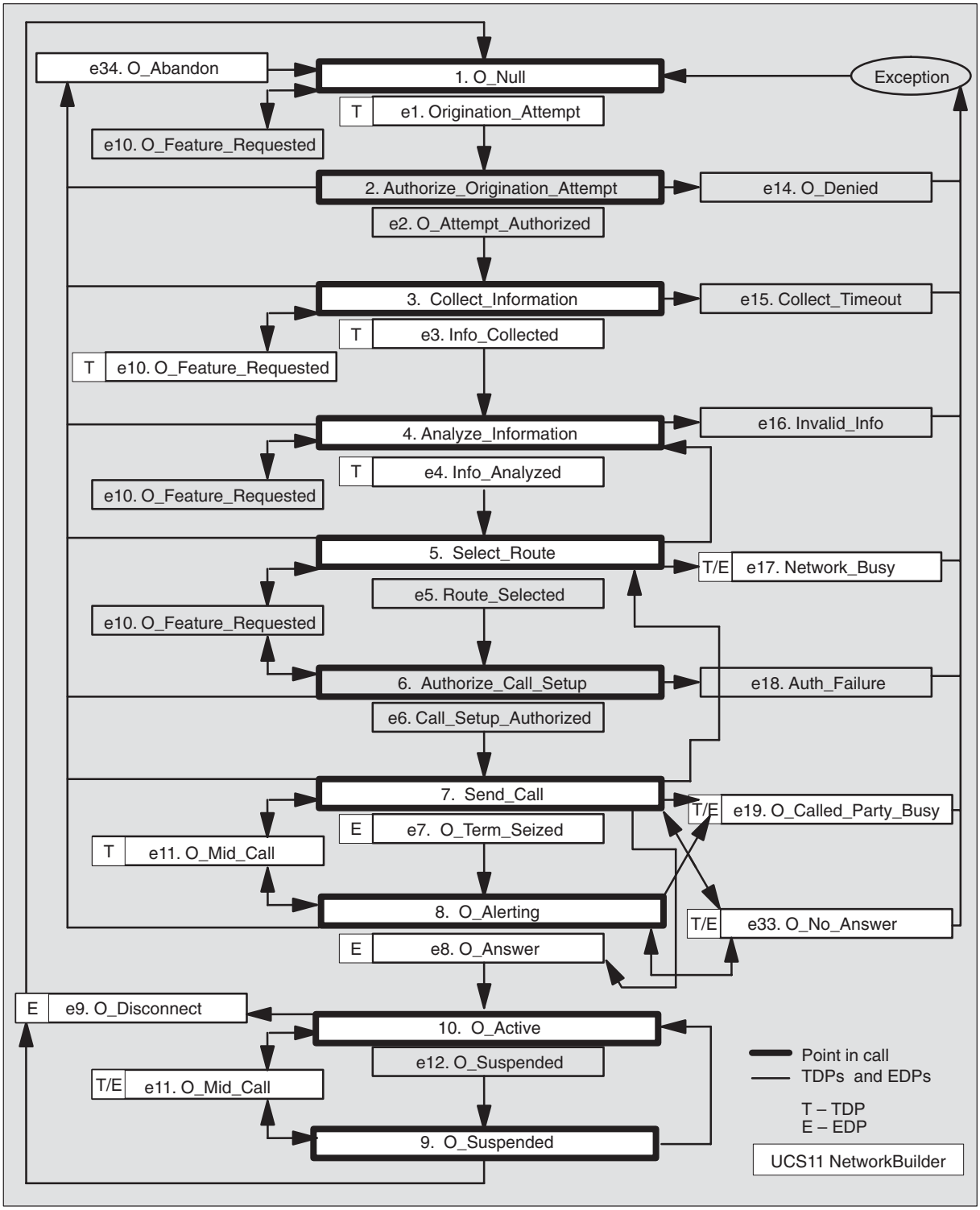
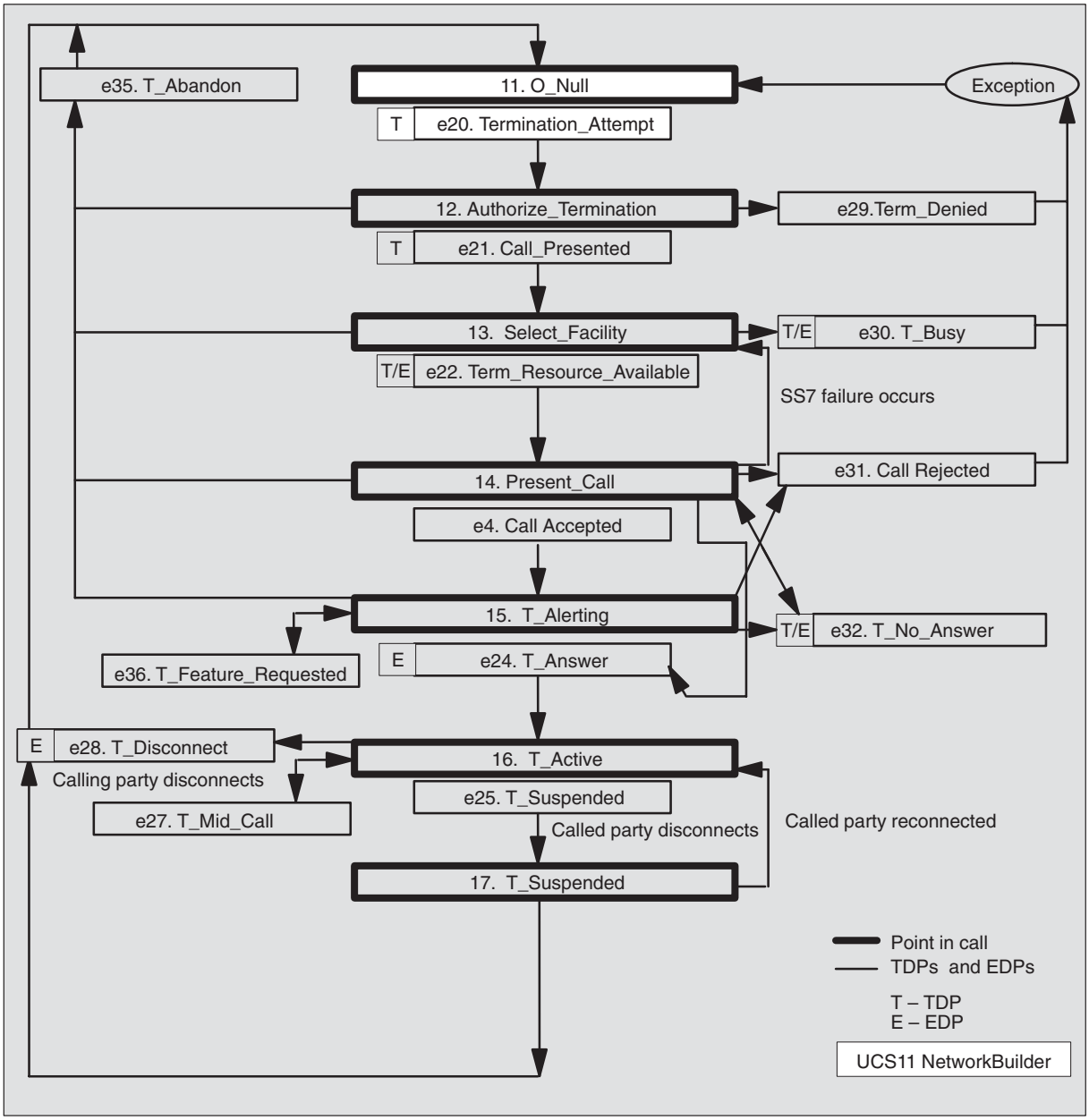


Figure 1-4
NetworkBuilder terminating call model



NetworkBuilder subscription

Any call originating on a supported agency (DAL, FGD, SS7 Inter-IMT, SS7 Global-IMT, AXXESS, or PRI) can subscribe to NetworkBuilder originating services.

Addresses, authcodes, ANIs, originating agents, or the office can subscribe to an originating CAIN group provisioned in table CAINGRP. The

originating CAIN group, in turn, enables one or more triggers in the originating call model. Enabling a trigger provides an index into an appropriate trigger table, which is required to query the SCP.

Any call terminating to a supported agency (DAL, FGB, FGC, FGD, IMT, AXXESS, or PRI) can subscribe to NetworkBuilder terminating services.

Agents can subscribe to a terminating CAIN group provisioned in table CAINGRP. The terminating CAIN group, enables the supported trigger in the terminating call model. Enabling the trigger provides an index into the trigger table, which is required to query the SCP.

Note: In addition, the SCP can return a CAIN group, thereby controlling subscription for the originating portion of the call

Note: CAIN group subscription is handled differently for AXXESS agents, refer to *UCS DMS-250 CAIN/FlexDial Interactions* for more information.

Within the trigger tables, you define the call conditions required (trigger criteria) and the actions the switch takes when the conditions are met. If a call meets the trigger criteria, the switch performs one of the following:

- query the SCP for instructions
- route advance at *Network_Busy*, *O_Called_Party_Busy*, or *O_No_Answer* trigger
- ignore the criteria and continue processing
- block the call and apply AINF treatment
- exit the trigger detection point and continue processing as directed by datafill

NetworkBuilder call processing stores up to six subscription methods (groups) for use throughout the call. The six groups are determined by the following means:

1 SCP-returned CAIN group

2 Address subscription

3 Authorization code subscription

4 ANI subscription

5 Agent subscription

6 Office subscription

Supported PICs

NetworkBuilder software supports the following PICs, EDPs, TDPs, and triggers:

Table 1-1
Supported PICs, TDPs, EDPs, triggers, and events

PIC	TDP/EDP	Trigger/Event
PIC 1: **O_Null**	***Origination_Attempt*** TDP	*Off_Hook_Immediate* trigger
PIC 3: **Collect_Information**	***O_Feature_Requested*** TDP	*O_Feature_Requested* trigger
	Info_Collected TDP	*Tollfree_Service* (note)
		Offhook_Delay trigger
		Shared_Interoffice_Trunk trigger
		PRI_B-Channel trigger
	O_Abandon EDP	*O_Abandon* event
PIC 4: **Analyze_Information**	***Info_Analyzed*** TDP	*Specific_Feature_Code* trigger
		Customized_Dialing_Plan trigger
		Specific_Digit_String trigger
		Office_Code trigger
	O_Abandon EDP	*O_Abandon* event
PIC 5: **Select_Route**	***Network_Busy*** TDP	*Network_Busy* trigger
	Network_Busy EDP	*Network_Busy* event
	O_Abandon EDP	*O_Abandon* event
PIC 7: **Send_Call**	***O_Term_Seized*** EDP	*O_Term_Seized* event
	O_Called_Party_Busy TDP	*O_Called_Party_Busy* trigger
	O_Called_Party_Busy EDP	*O_Called_Party_Busy* event
	O_Mid_Call TDP	*O_IEC_Reorigination* trigger

Note: NetworkBuilder supports Bellcore's *TR-NWT-000533* toll-free service specifications. The ability to subscribe to multiple CAIN groups allows for a service integration of the CAIN triggers and the IN/1 *Tollfree_Service* trigger defined in *TR-NWT-000533*.

—continued—

Table 1-1
Supported PICs, TDPs, EDPs, triggers, and events (continued)

PIC	TDP/EDP	Trigger/Event
PIC 8: **O_Alerting**	***O_Mid_Call*** EDP	*Switch_Hook_Flash* event
	O_Abandon EDP	*O_Abandon* event
	O_Answer EDP	*O_Answer* event
	O_No_Answer TDP	*O_No_Answer* trigger
	O_No_Answer EDP	*O_No_Answer* event
PIC 9: **O_Active**	***O_Mid_Call*** TDP	*O_IEC_Reorigination* trigger
	O_Mid_Call EDP	*Switch_Hook_Flash* event
	O_Abandon EDP	*O_Abandon* event
	O_Disconnect EDP	*O_Disconnect* event
	O_Mid_Call TDP	*O_IEC_Reorigination* trigger
	O_Mid_Call EDP	*Timeout* event
PIC 10: **O_Suspended**		*Switch_Hook_Flash* event
	O_Disconnect EDP	*O_Disconnect* event
	O_Mid_Call TDP	*O_IEC_Reorigination* trigger
	O_Mid_Call EDP	*Timeout* event
PIC 11: **T_Null**	***Termination_Attempt*** TDP	*Termination_Attempt* trigger

Note: NetworkBuilder supports Bellcore's *TR-NWT-000533* toll-free service specifications. The ability to subscribe to multiple CAIN groups allows for a service integration of the CAIN triggers and the IN/1 *Tollfree_Service* trigger defined in *TR-NWT-000533*.

—end—

SCP interaction

NetworkBuilder call processing interacts with the SCP to determine how the call is handled. The SCP may direct the switch to collect digits, apply treatment and disconnect, play announcements, connect to an IP, or route the call according to SCP instructions.

To throttle the switch during periods of congestion, manual and automatic code gapping (ACG) can be implemented. Manual and automatic code gapping methods reduce the number of outgoing queries from the switch to a given SCP. The SCP provides the switch with information on which queries

to block by sending a message containing an ACG Control. The switch stores the controls in a control list. The control list is searched before sending each query message to determine if that query should be blocked.

Software optionality control

Software optionality control (SOC), part of the DMS Evolution product delivery process, controls the delivery of product computing module loads (PCL). All features in a PCL are categorized as either base or optional. Base applications are available for immediate use; optional applications are grouped into commercial units called SOC options. SOC options can be purchased by operating companies. NetworkBuilder uses the following SOC options:

- CAIN0100 (CAIN Messages)
- CAIN0200 (CAIN Extension Parms)
- CAIN0300 (CAIN SCP Simulator)
- CAIN0400 (CAIN Test Query Tool)
- CAIN0500 (CAIN CUSTDP Trigger)
- CAIN0501 (CAIN SPECDIG Trigger)
- CAIN0502 (CAIN OFFHKIM Trigger)
- CAIN0503 (CAIN SIOTRK Trigger)
- CAIN0504 (CAIN PRIBCHNL Trigger)
- CAIN0505 (CAIN ONOANSWER Trigger)
- CAIN0506 (CAIN NETBUSY Trigger)
- CAIN0507 (CAIN OCLDBUSY Trigger)
- CAIN0508 (CAIN OFTRREQ Trigger)
- CAIN0509 (CAIN OIECREO Trigger)
- CAIN0510 (CAIN TERMATT Trigger)
- CAIN0511 (CAIN SPECFEAT Trigger)
- CAIN0512 (CAIN OFFHKDEL Trigger)
- CAIN0513 (CAIN TOLLFREE Trigger)
- CAIN0600 (CAIN Con Digit Collect)
- CAIN0601 (CAIN SCP Trigger Sub)
- CAIN0602 (CAIN EDPs)
- CAIN0603 (CAIN STR Connection)
- CAIN0604 (CAIN Inter IMT Support)

- CAIN0605 (CAIN Global IMT Support)
- CAIN0606 (CAIN 1129-Style IP)
- CAIN0607 (CAIN Virtual IP)
- CAIN0609 (CAIN Term Notification)
- CAIN0610 (CAIN CainPrt Digit Coll)
- CAIN0700 (CAIN LNP QOO)
- CAIN0800 (CAIN Mid Call Services 1)
- CAIN0801 (CAIN Mid Call Services 2)
- CAIN0802 (CAIN Takeback & Transfer)
- CAIN0900 (CAIN Auto Code Gapping)
- CAIN0901 (CAIN Manual Code Gapping)

Refer to Volume 5, Chapter 5, "NetworkBuilder SOC functionality," for more information on NetworkBuilder SOCs and SOC dependencies.

The UCS11 software release adds the CAIN0802 (CAIN Takeback & Transfer) SOC.

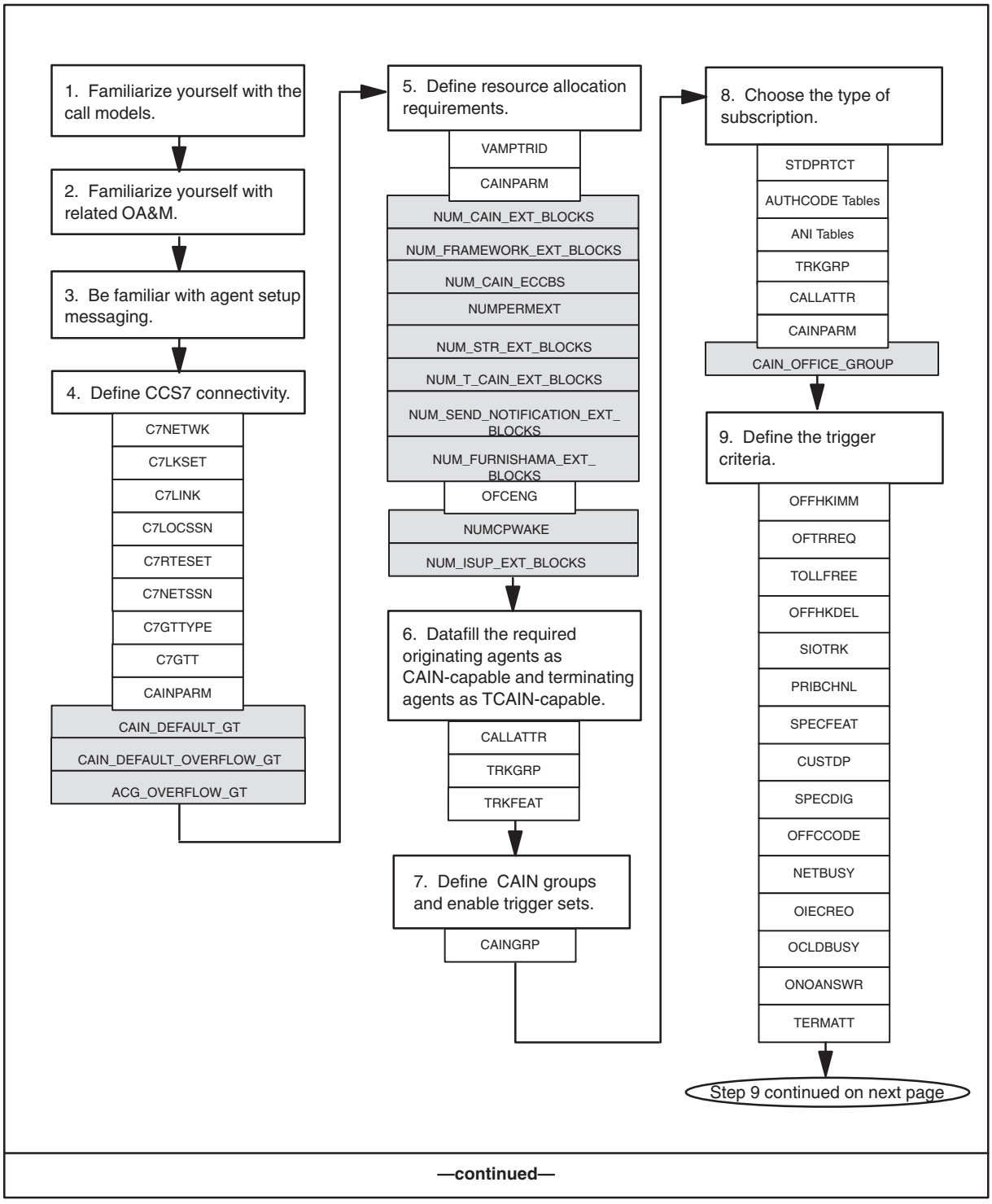
Provisioning NetworkBuilder

Implementing NetworkBuilder services requires a base knowledge of table control, the originating and terminating call model, CCS7 network connections, subscription, triggering, messaging, resource allocation requirements, and announcement and tone requirements.

This chapter provides a base level knowledge required to activate NetworkBuilder services into your network. Figure 2-1 shows a basic flow of the steps you should take to provision your switch.

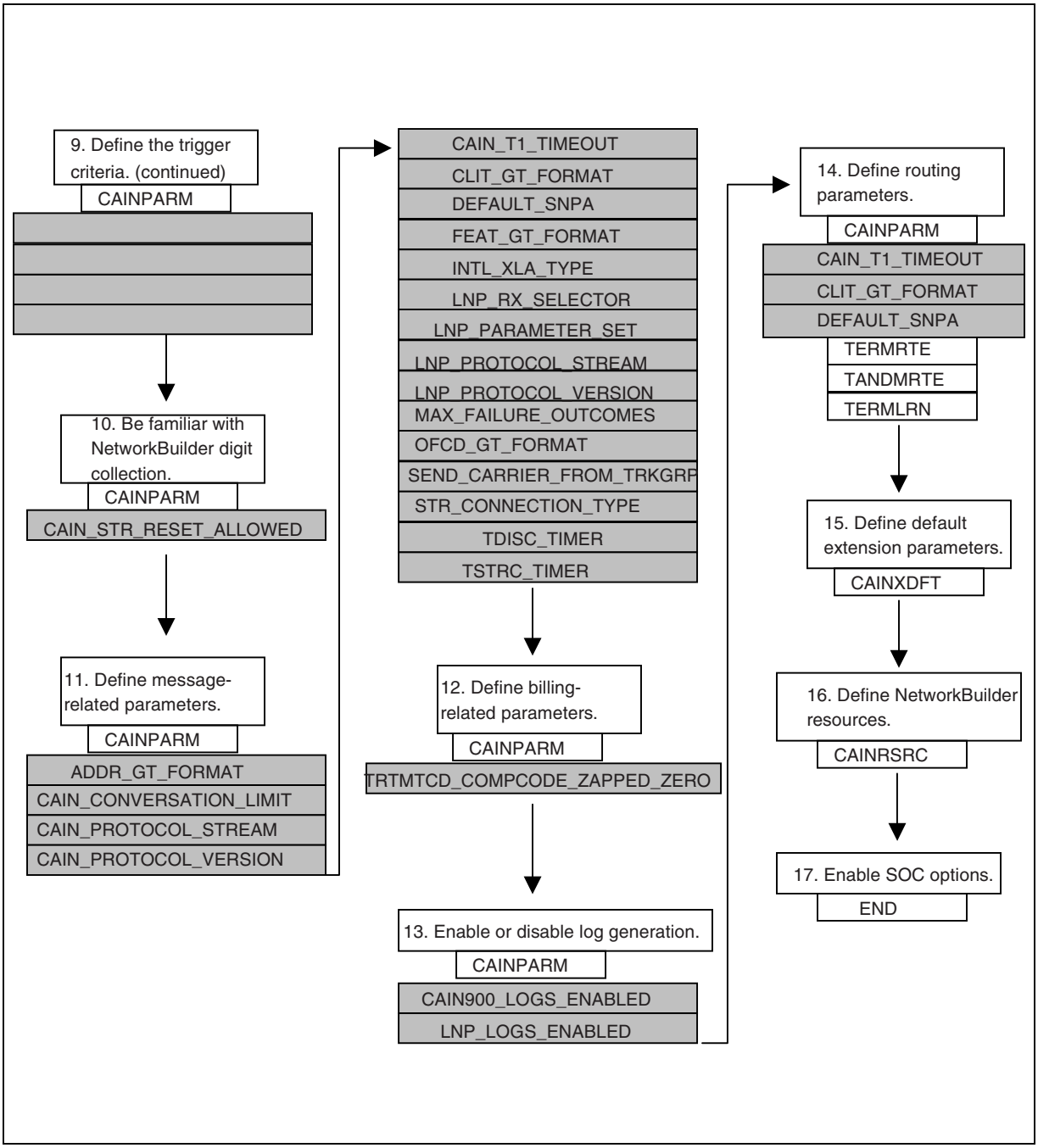
Note: Provisioning NetworkBuilder for AXXESS agents is explained in *UCS DMS-250 CAIN/FlexDial Interactions*.

Figure 2-1
Provisioning NetworkBuilder



—continued—

Figure 2-1
Provisioning NetworkBuilder (continued)



Provisioning

The remainder of this chapter shows you how to provision your switch for NetworkBuilder services. Figure 2-2 shows how sections are titled and divided.

Figure 2-2
Example of a provisioning section

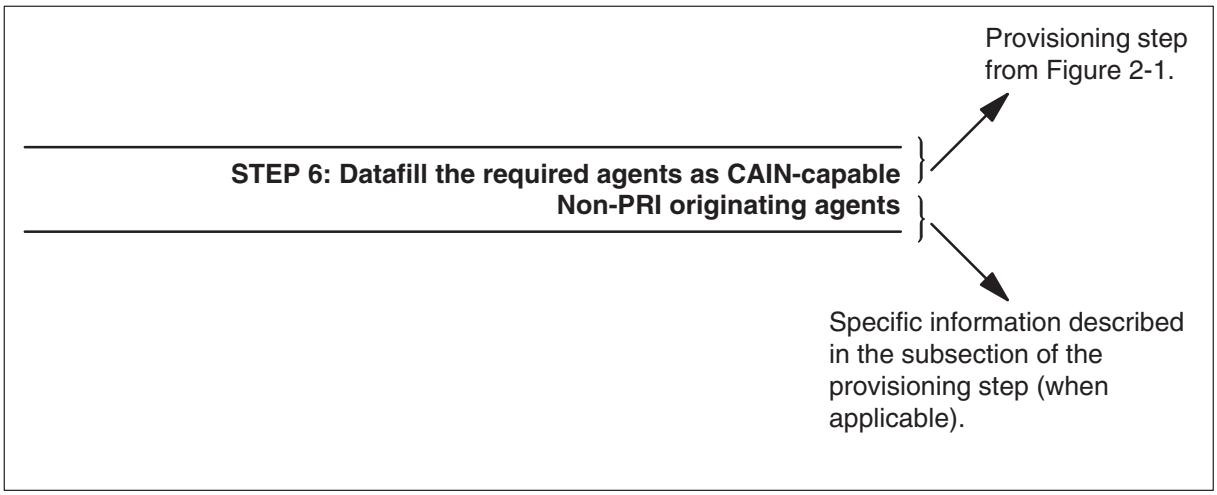


Table 2-1 shows the layout of the rest of this chapter and the information available.

Table 2-1
Chapter layout

Step	Title	Page
1	Familiarize yourself with the call models	2-9
2	Familiarize yourself with related OA&M	2-17
	Logs	2-18
	OMs	2-22
	DS	2-56
	Treatments	2-65
	Billing	2-68
	Commands	2-76
3	Be familiar with agent setup messaging	2-78

—continued—

Table 2-1
Chapter layout (continued)

Step	Title	Page
4	Define CCS7 connectivity	2-84
	CAIN_DEFAULT_GT	2-98
	CAIN_DEFAULT_OVERFLOW_GT	2-99
	ACG_OVERFLOW_GT	2-97
5	Define resource allocation requirements	2-100
	CAIN extension blocks	2-101
	T_CAIN extension blocks	2-103
	CAIN framework extension blocks	2-105
	VAMP transaction identifiers	2-107
	CAIN extended call condense blocks	2-109
	CAIN STR extension blocks	2-111
	ISUP extension blocks	2-113
	CAIN O_NO_Answer and Timeout timers	2-114
	CAIN send notification extension blocks	2-115
	FURNISHAMA extension blocks	2-118
	PORTPERM extension blocks	2-117
6	Datafill the required agencies as CAIN/T_CAIN-capable	2-119
	Non-PRI originating agents	2-121
	Non-PRI terminating agents	2-122
	AXXESS originating and terminating agents	2-123
	PRI originating agents	2-124
	PRI terminating agents	2-126
	Note: AXXESS agents are handled differently, refer to *UCS DMS-250 CAIN/FlexDial Interactions* for more information.	
7	Define CAIN groups and enable trigger sets	2-128

—continued—

Table 2-1
Chapter layout (continued)

—continued—

Table 2-1
Chapter layout (continued)

—continued—

Table 2-1
Chapter layout (continued)

Step 1: Familiarize yourself with the call models

Terminology

The following terms are related to the call model:

- originating call model – generic representation of a basic call in terms of the processing activities required to establish, maintain, and clear a call.
- terminating call model – generic representation of a basic call in terms of the processing activities required to terminate a call.
- point in call (PIC) – represents the call processing functionality required by a basic two-party call.
- detection point (DP) – points within the call model where events or triggers allow the switch to notify the SCP of events that occur or to temporarily suspend call processing to query the SCP before continuing to process the call.
- trigger detection point (TDP) – points within the call model where the switch can temporarily suspend call processing and send a query message to the SCP based on datafilled criteria evaluation.
- trigger criteria – identifies call conditions that must be met in order for the switch to perform a specified action.
- event detection point (EDP) – EDPs are similar to TDPs. When a call queries the SCP, the SCP may return a conversational package identifying one or more EDPs that occur later in the call model. At this point, the EDPs are said to be armed. An EDP is armed to send an EDP-Request or EDP-Notification message when encountered.
- next event list (NEL) – a list of event detection points (EDPs) that are armed to notify the SCP or send a request to the SCP when reached. EDPs become armed when a conversation package which includes an **`Analyze_Route`**, **`Continue`**, or **`Collect_Information`** message along with a **`Request_Report_BCM_Event`** non-call related component is returned by the SCP.
- call configuration (CC) – Call configurations identify key states in the progression of a call. Each call configuration provides a view of the connections between the call's parties. Refer to Volume 3, Chapter 4, "Call Configurations," for more information.
- call segment – Call segments are parts (segments) of a call. Call segments have one connection point and zero to three call legs. A call can have a maximum of two call segments:
  - — the first call segment is a call segment where a two-party call originated
  - — the second call segment is a call segment where a call to a third party originated

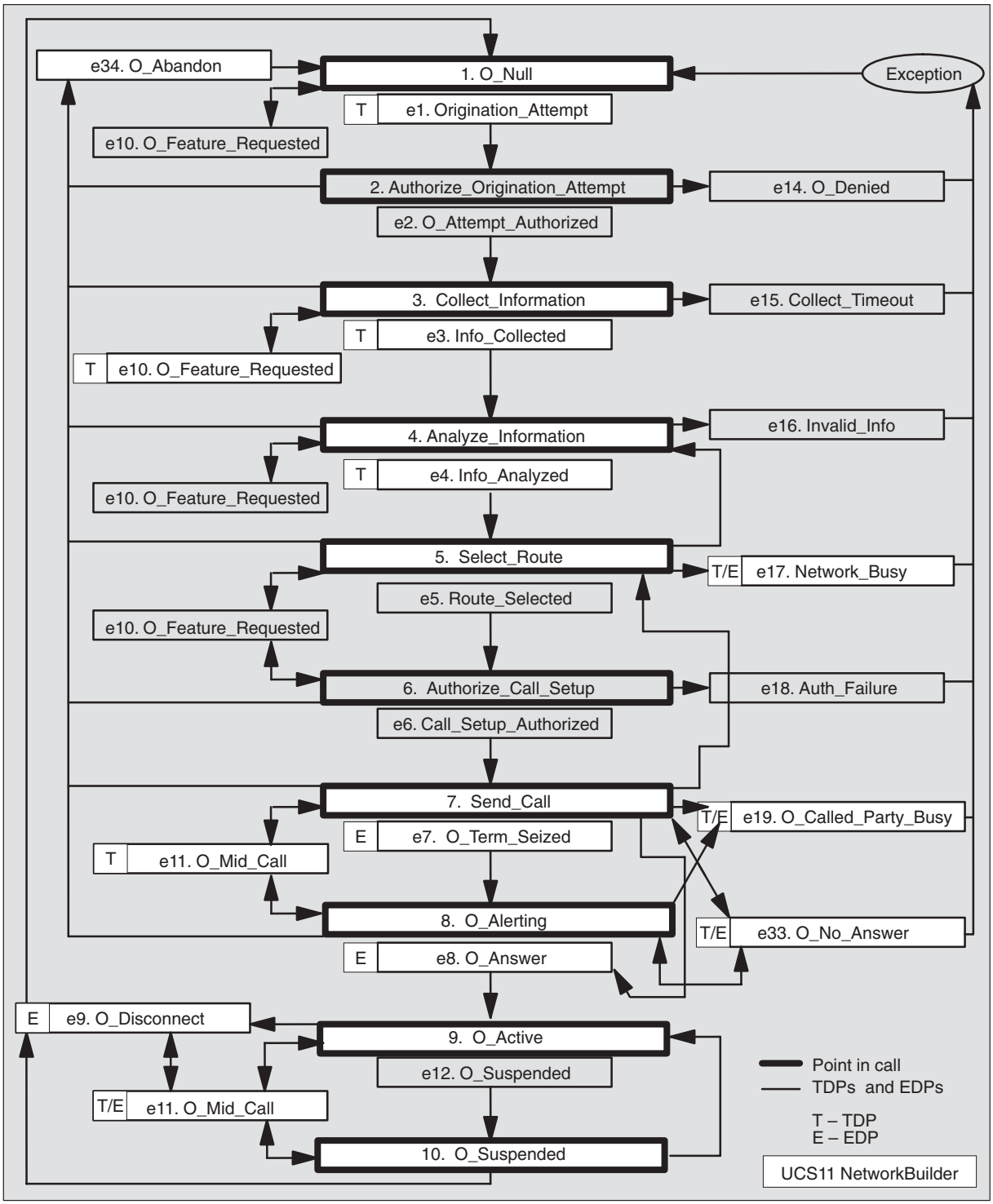
Step 1: Familiarize yourself with the call models (continued)

Note: Refer to Volume 3, Chapter 4, "Call Configurations," for more information on call segments.

Figures 2-3 and 2-4 show the NetworkBuilder originating and terminating call models.

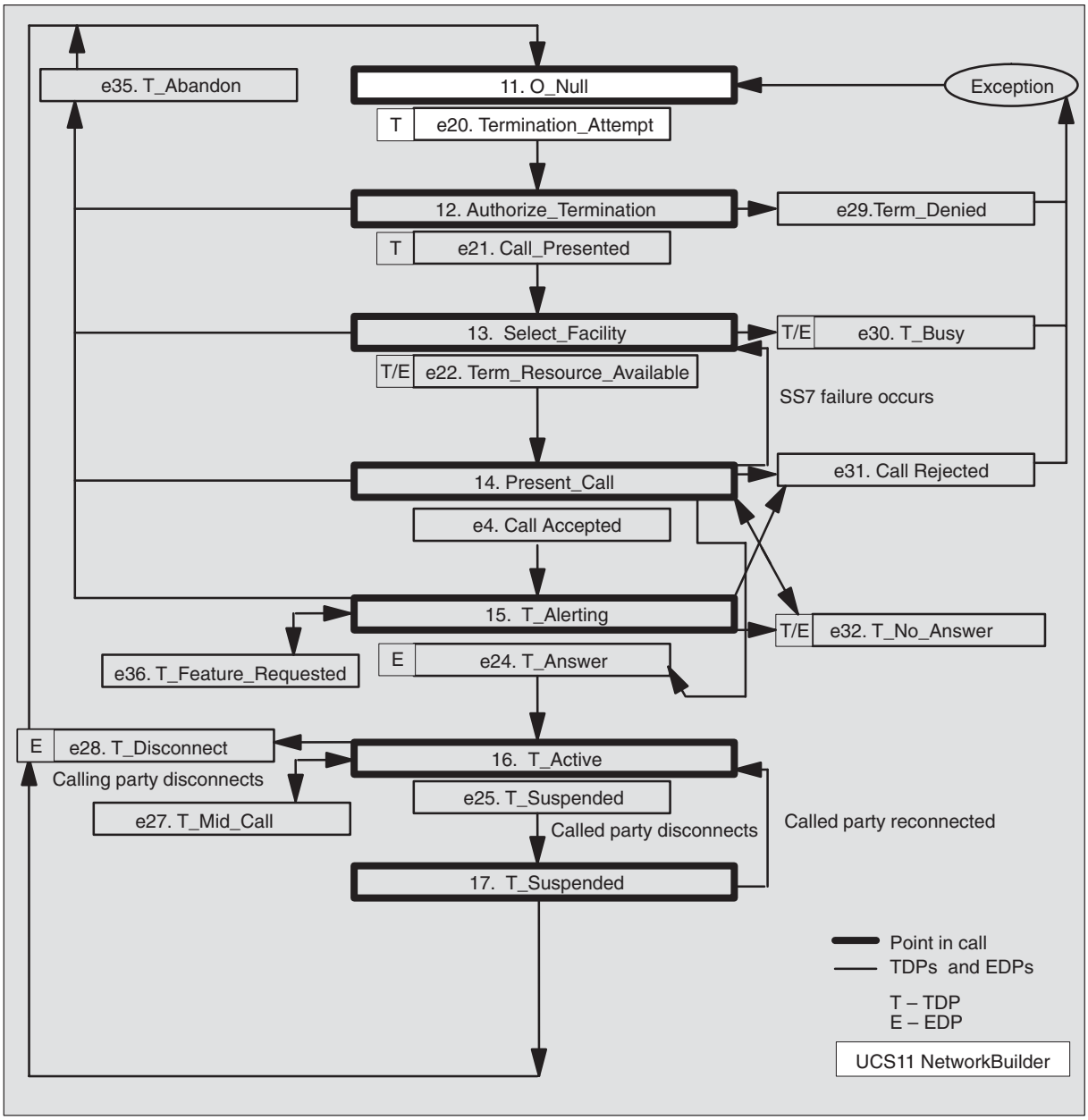
Step 1: Familiarize yourself with the call models (continued)

Figure 2-3
NetworkBuilder originating call model



Step 1: Familiarize yourself with the call models (continued)

Figure 2-4
NetworkBuilder terminating call model



Step 1: Familiarize yourself with the call models (continued)

Table 2-2
Call model defined

PIC	Detection Points	Definition
1. **O_Null**		Occurs upon call clearing, after exception handling, or system initialization. At this time, the originating agent is idle and no call exists.
	Origination_ Attempt TDP (Note 1)	Encountered when a call is initiated on the originating agent: • PTS agents – trunk is seized • SS7 agents – switch receives an IAM • PRI agents – switch receives a SETUP message
2. **Authorize_Origination _Attempt**		Corresponds to any screening performed based upon originating agent datafill.
3. **Collect_Information**		Corresponds to the switch collecting enough data (called party address, ANI, authcode) to process the call. The switch collects data based on the dialing plan.
	O_Feature_ Requested TDP	Encountered when the address digits are collected.
	Info_Collected TDP	Encountered when the switch completes the dialing plan.
	O_Abandon EDP	Encountered when the calling party disconnects before the called party answers.
4. **Analyze_Information**		Entered when the switch begins analyzing and translating the collected digits to determine a route index.
	Info_Analyzed TDP	Encountered once the switch has attempted to identify a route index.
	O_Abandon EDP	Encountered when the calling party disconnects before the called party answers.
Note 1: The switch only evaluates the ***Origination_Attempt*** TDP for calls originating from a DAL (loop or ground starts). ***Note 2:*** Refer to Chapter 9, “O_Active and O_Suspended PICs.” for more information.		

—continued—

Step 1: Familiarize yourself with the call models (continued)

Table 2-2
Call model defined (continued)

PIC	Detection Points	Definition
5. **Select_Route**		Entered when the switch selects a terminating agent from the route list that was determined in-switch or by the SCP.
	Network_Busy DP	Encountered when the triggering switch has attempted all routes and is unable to terminate the call or when an SS7 REL message or ISDN Release message is received indicating a network busy condition.
	O_Abandon EDP	Encountered when the calling party disconnects before the called party answers.
6. **Authorize_Call_Setup**		Entered as the switch verifies the authority of the calling party to place the call.
7. **Send_Call**		Entered as the switch attempts to terminate the call.
	O_Term_Seized EDP	Encountered when a terminating trunk is seized on the UCS DMS-250 switch.
	O_Called_Party_Busy DP	Encountered when the switch • attempts to route to a busy DAL or AXXESS agent (with ONNETTRK=Y) • receives an indication from the terminating agent that the end-user is busy Refer to Chapter 7, “Send_Call PIC,” for more information.
	O_Mid_Call TDP	Encountered when the calling party indicates reorigination.
Note 1: The switch only evaluates the ***Origination_Attempt*** TDP for calls originating from a DAL (loop or ground starts). ***Note 2:*** Refer to Chapter 9, “O_Active and O_Suspended PICs.” for more information.		
—continued—		

Step 1: Familiarize yourself with the call models (continued)

Table 2-2
Call model defined (continued)

PIC	Detection Points	Definition
	O_Abandon EDP	Encountered when the calling party disconnects before the called party answers.
8. **O_Alerting**		Entered as the terminating switch applies ringing and ring-back.
	O_Answer EDP	Encountered when the terminating party answers the call.
	O_No_Answer DP	Encountered when the called party does not answer within the specified amount of time.
	O_Mid_Call TDP	Encountered when the calling party indicates reorigination.
	O_Mid_Call EDP for the *Switch_Hook_Flash* event	Encountered during a valid call configuration when a valid call leg presses the asterisk "*" key for 40 milliseconds. (Note 2)
	O_Abandon EDP	Encountered when the calling party disconnects before the called party answers.
9. **O_Active**		Entered when answer indication is received from the called party and when the called party is reconnected after the switch receives an SS7 RESUME (RES) message.
	O_Disconnect EDP	Encountered when reorigination is detected or when a party disconnects.
	O_Mid_Call TDP	Encountered when the calling party indicates reorigination.
	O_Mid_Call EDP for the *Timeout* event	Encountered when the call has been active longer than the specified amount of time.

Note 1: The switch only evaluates the ***Origination_Attempt*** TDP for calls originating from a DAL (loop or ground starts).
Note 2: Refer to Chapter 9, "O_Active and O_Suspended PICs." for more information.

—continued—

Step 1: Familiarize yourself with the call models (end)

Table 2-2
Call model defined (continued)

PIC	Detection Points	Definition
	O_Mid_Call EDP for the *Switch_Hook_Flash* event	Encountered during a valid call configuration when a valid call leg presses the asterisk "*" key for 40 milliseconds. (Note 2)
10. **O_Suspended**		Entered when the switch receives an SS7 SUSPEND (SUS) message.
	O_Disconnect EDP	Encountered when reorigination is detected or when a party disconnects.
	O_Mid_Call TDP	Encountered when the calling party indicates reorigination.
	O_Mid_Call EDP *Timeout* event	Encountered when the call has been active longer than the specified amount of time.
11. **T_Null**		
	Termination_Attempt TDP	Encountered when indication of a desire to deliver the call is received from the originator.
Note 1: The switch only evaluates the ***Origination_Attempt*** TDP for calls originating from a DAL (loop or ground starts). ***Note 2:*** Refer to Chapter 9, "O_Active and O_Suspended PICs." for more information.		
—end—		

Step 2: Familiarize yourself with related OA&M

In order to fully understand, implement, and maintain NetworkBuilder, you need to be familiar with the related logs, operational measurements, data schema, treatments, billing, and commands used to process NetworkBuilder calls.

This section provides a brief overview of operating, administrative, and maintenance (OA&M) functions, which include the following:

- Logs – records of call activities that occur within the switch. Refer to *UCS DMS-250 Logs Reference Manual* for more logs information.
- OMs – operational measurements of data that are collected and displayed as the switch performs various operations. Refer to *UCS DMS-250 Operational Measurements Reference Manual* for more OMs information.
- Data schema – tables that direct how NetworkBuilder calls are processed. Refer to *UCS DMS-250 Data Schema Reference Manual* for more data schema information.
- Treatment – the method by which a call is disconnected or ended.
- Billing – billing records for calls on the switch. Refer to *UCS DMS-250 Billing Records Application Guide* for more billing information.
- Commands – commands used on the switch. Refer to *UCS DMS-250 Commands Reference Manual* for more commands information.

Step 2: Familiarize yourself with related OA&M Logs

The following table lists the switch-generated logs associated with NetworkBuilder:

Table 2-3
NetworkBuilder-related logs

Log	Description
AUD620	EXT Dump. Generated when there is a data dump for a CAIN framework extension block.
AUD621	EXT Dump. Generated when there is a data dump of a CAIN extension block.
AUD665	EXT block audit report. Generated when there is a data dump of a CAIN terminating call model extension block.
CAIN100	Non-Fatal Application Error. Generated when a nonfatal application error is encountered while transacting with the SCP. Specific error details are given in the log text. Call processing attempts to recover and proceeds with normal in-switch routing when nonfatal application errors occur.
CAIN101	Non-Fatal TCAP/SCCP Error. Generated when a nonfatal TCAP/SCCP error is encountered while transacting with the SCP. Specific error details are given in the log text. Call processing attempts to recover and proceeds.
CAIN102	CAIN SOC Access. Generated when the switch tries to access an idle, non-trigger CAIN SOC.
CAIN200	Fatal Application Error. Generated when a fatal application error is encountered while transacting with the SCP. Specific error details are given in the log text.
CAIN201	Fatal TCAP/SCCP Error. Generated when a fatal TCAP/SCCP error is encountered while transacting with the SCP. Specific error details are given in the log text.
CAIN300	SSP Routing Trouble Report. Generated when it determines the data returned by the SCP is in the correct format but does not correspond with data in the switch database.
CAIN301	ISUP Cause Indicator Received. Generated when the switch receives a REL message (cause 26) indicating that a call has been misrouted to a ported number or a REL message (cause 28) indicating that it received an improperly formatted GAP. Only generated when LNP is active on the call.
CAIN302	SSP Trouble Report. Generated when a call exceeds the number of times it can request information from the SCP.
Note: Refer to *UCS DMS-250 Logs Reference Manual* for more information on logs.	
—continued—	

Step 2: Familiarize yourself with related OA&M Logs (continued)

Table 2-3
NetworkBuilder-related logs (continued)

Log	Description
CAIN303	Networkbuilder Routing Log. The CAIN303 log is generated to identify and resolve network routing problems.
CAIN900	SCP Simulator Indices. Generated when the SCP simulator sends a response to the CAIN framework.
CAIN901	SCP Simulator Action. Generated when the SCP simulator receives an error or abort message from the CAIN framework. No response message from the SCP simulator is required; also produced when an SCP simulator error is detected.
CAIN902	CAIN Subscription Method. Generated to identify the current CAIN subscription method when evaluating trigger criteria.
CAIN903	CAINGRP To Trigger. Generated to identify the CAIN group associated with the call when the trigger criteria is met.
CAIN904	CAIN Collectible Overridden. Generated when a collectible on the switch is being overridden. The old information is discarded and the new information is retained.
CAIN905	CAIN Requested Event. Generated to identify the PIC, EDP, event, event action and originating trunk group for a call each time a requested event is reached.
CAIN906	OFFCCODE Trigger Blocked by STS. Generated when an *Office_Code* trigger is blocked by the NO_LNP option in table CAINSTS.
CAIN907	Unarmed EDP action. Generated when the switch detects an unarmed event necessary for multi-party call handling while the switch and the SCP are in conversation.
VAMP201	VAMPTRID High Resource Use. Generated when VAMP resources exceed the 75% or 90% usage threshold set in table VAMPTRID. If resource usage exceeds the 75% threshold, but not the 90% threshold, a minor alarm is indicated. If the 90% threshold is exceeded, a major alarm is indicated.
VAMP202	VAMPTRID Resource Overflow. Generated when attempted usage of a VAMP resource exceeds resource allocation set in table VAMPTRID. A critical alarm is indicated.
Note: Refer to *UCS DMS-250 Logs Reference Manual* for more information on logs.	

—continued—

Step 2: Familiarize yourself with related OA&M Logs (continued)

Table 2-3
NetworkBuilder-related logs (continued)

Log	Description
VAMP203	VAMPTRID Alarm Cleared. Generated after VAMP201 log reports are output indicating high resource usage and once usage of a VAMP resource drops below 75% of the allocation. Indicates the resource usage was lowered because changes occurred in the call volume or call mix, or the resource allocation was increased in table VAMPTRID.
VAMP301	VAMPTRID ACG Query Blocked. Generated when an outgoing query is blocked due to encountering an ACG control.
VAMP302	VAMPTRID ACG Control List Change. Generated when an ACG control is added, removed, expired, or updated on the ACG control lists.
VAMP303	VAMPTRID ACG Global Reset. Generated when an ACG Global Reset message is received by the switch.
VAMP304	VAMPTRID ACG Infinite Duration. Generated when an ACG control with an infinite duration is present in one of the control lists. Automatically occurs every ten minutes.
VAMP305	VAMPTRID ACG Overflow. Generated when an ACG control list has overflowed.
VAMP306	VAMPTRID ACG Global Outgoing Control. Generated when a Global Outgoing Control is present in the ACG control list. Automatically occurs every five minutes.
VAMP601	VAMPTRID Audit Summary. Generated when errors are found in the audit of a resource pool or free queue. If an audit completes successfully, this log report is not generated.
VAMP602	VAMPTRID Audit Block Error. Generated during a resource pool audit when an individual resource block is found damaged or in an inconsistent state and is recovered and placed in the free queue. Contains the error type that caused the block to be recovered, as well as a dump of the block contents in hexadecimal form.
VAMP603	VAMPTRID Free Queue Rebuilt. Generated when an application's free queue of a resource type must be rebuilt to recover from queue corruption. The rebuilding of a queue temporarily removes the queue from service, which may cause resource seizure attempts to fail.
Note: Refer to *UCS DMS-250 Logs Reference Manual* for more information on logs.	
—continued—	

Step 2: Familiarize yourself with related OA&M Logs (end)

Table 2-3
NetworkBuilder-related logs (continued)

Log	Description
VAMP901	VAMP Inbound Message. Generated when an inbound message is received from VAMP and the VPTRACE CI tool has enabled message monitoring. The outbound messages bounced back by the transport layer are not logged.
VAMP902	VAMP Outbound Message. Generated when an outbound message is sent through VAMP and the VPTRACE CI tool has enabled message monitoring.
Note: Refer to *UCS DMS-250 Logs Reference Manual* for more information on logs.	
—end—	

Step 2: Familiarize yourself with related OA&M Operational measurements

Operational measurements (OMs) related to CAIN are listed in table 2-4 and OMs related to Bellcore *TR-NWT-000533* IN/1 are listed in table 2-5.

The following table lists the OMs pegged by the switch in association with CAIN:

Table 2-4
NetworkBuilder CAIN-related OMs

Group	Register	Description
ANN		Provides information on traffic for recorded announcement machines.
	ANNATT	Counts calls that are routed to an announcement.
	ANNMBU	NetworkBuilder software does not peg this register.
	ANNOVFL	Counts calls that are routed to a recorded announcement, but fails to connect to the announcement because the maximum number of calls are connected or because the announcement is maintenance-busy.
	ANNSBU	NetworkBuilder software does not peg this register.
	ANNTRU	NetworkBuilder software does not peg this register.
	ANNFTRU	NetworkBuilder software does not peg this register.
CAINAGOM		Provides a tuple of OMs for each agency. The tuples are: • 0 TOTAL • 1 DAL • 2 EANT • 3 PRI • 4 IMT • 5 AXXESS • 6 ONAL
Note 1: Refer to *UCS DMS-250 Operational Measurements Reference Manual* for more information. ***Note 2:*** ITU protocol is not currently supported by NetworkBuilder.		
—continued—		

Step 2: Familiarize yourself with related OA&M Operational measurements (continued)

Table 2-4
NetworkBuilder CAIN-related OMs (continued)

Group	Register	Description
CAINAGOM (continued)		• 7 ONAT
	AGQUERY	Counts the number of queries sent to the SCP for calls originating on the agent type of the associated tuple.
	AGRESPR	Counts the number of responses the switch receives from the SCP for calls originating on the agent type of the associated tuple.
	AGERROR	Counts the number of times an error scenario occurs and the datafilled trigger action is performed for calls originating on the agent type of the associated tuple.
	AGEDPRCD	Counts the number of times a valid **`Request_Report_BCM_Event`** component is received.
	AGSTRCNV	Counts the number of **`Send_To_Resource`** messages (in a conversation package) received by the switch for calls originating on the agent type of the associated tuple.
	AGRSCLR	Counts the number of **`Resource_Clear`** messages the switch sends to the SCP during a conversational transaction for calls originating on the agent type of the associated tuple.
	AGEDPREQ	Counts the number of EDP requests sent for calls originating on the agent type of the associated tuple.
	AGEDPNOT	Counts the number of EDP notifications sent for calls originating on the agent type of the associated tuple.
	AGCTRCNV	Counts the number of conversational **`Connect_To_Resource`** messages, by agent type, received by the switch.
	AGCTRCLR	Counts the number of **`CTR_Clear`** messages, by agent type, sent to the SCP.
	AGVIPREQ	Counts the number of times the SCP requests VIP handling, for either message type (**`Send_To_Resource`** or **`Connect_To_Resource`**) on a per-agency basis.
Note 1: Refer to *UCS DMS-250 Operational Measurements Reference Manual* for more information. ***Note 2:*** ITU protocol is not currently supported by NetworkBuilder.		

—continued—

Step 2: Familiarize yourself with related OA&M Operational measurements (continued)

Table 2-4
NetworkBuilder CAIN-related OMs (continued)

Group	Register	Description
CAINAGOM (continued)	AGVIPRSP	Counts the number of times a response with a ***ClearCause*** value of `normal` is sent to the SCP after VIP handling, for either message type (**Resource_Clear** or **CTR_Clear**), on a per-agency basis.
	AGCITR	Counts the number of **Call_Info_To_Resource** messages received by the switch, on a per-agency basis.
	AGCIFR	Counts the number of **Call_Info_From_Resource** messages sent to the SCP, on a per-agency basis.
	TQUERY	Counts the number of terminating call model queries sent to the SCP for CAIN-related calls.
	TRESPR	Counts the number of response messages received by the switch in response to terminating call model queries sent to the SCP for CAIN-related calls.
	TERROR	Counts the number of times an error scenario is encountered during the processing of the switch to SCP communcations for terminating call model triggers, where the trigger table error action is invoked.
	TSTRCNV	Counts the number of conversational **Send_To_Resource** messages that are received by the switch for CAIN terminating call model queries.
	TRSCLR	Counts the number of **Resource_Clear** messages sent in conversation for CAIN terminating call model interactions.
	TVIPREQ	Counts the number Virtual IP requests that are sent in conversation for CAIN terminating call model interactions.
	TVIPRSP	Counts the number Virtual IP messages that are sent in conversation for CAIN terminating call model interactions.
CAINGOM2		Extension group of OM group CAINAGOM. CAINGOM2 provides threshold reporting of OMs on a per agency basis.

Note 1: Refer to *UCS DMS-250 Operational Measurements Reference Manual* for more information.

Note 2: ITU protocol is not currently supported by NetworkBuilder.

—continued—

Step 2: Familiarize yourself with related OA&M Operational measurements (continued)

Table 2-4
NetworkBuilder CAIN-related OMs (continued)

Group	Register	Description
CAINGOM2 (continued)	AGQUERY2	Extension register of register AGQUERY.
	AGRESPR2	Extension register of register AGRESPR.
	AGERROR2	Extension register of register AGERROR.
	AGEDPRC2	Extension register of register AGEDPRCD.
	AGSTRCN2	Extension register of register AGSTRCNV.
	AGRSCLR2	Extension register of register AGRSCLR.
	AGEDPRE2	Extension register of register AGEDPREQ.
	AGEDPNO2	Extension register of register AGEDPNOT.
	AGCTRCN2	Extension register of register AGCTRCNV.
	AGCTRCL2	Extension register of register AGCTRCLR.
	AGCITR2	Extension register of register AGCITR.
	AGCIFR2	Extension register of register AGCIFR.
CAINIP		Provides OMs to track the number of times each of the various ***ClearCause*** values is sent to the SCP in a **Resource_Clear** message during 1129-style IP interaction. Also tracked is the number of times the TSTRC and TDISC timers expire at the local switch.
	IPNORMAL	Counts the number of **Resource_Clear** messages sent to the SCP with a ***ClearCause*** parameter value of `normal`.
	IPTMO	Counts the number of **Resource_Clear** messages sent to the SCP with a ***ClearCause*** parameter value of `timeout`.
	IPRESCAN	Counts the number of **Resource_Clear** messages sent to the SCP with a ***ClearCause*** parameter value of `resourceCancelled`.

Note 1: Refer to *UCS DMS-250 Operational Measurements Reference Manual* for more information.

Note 2: ITU protocol is not currently supported by NetworkBuilder.

—continued—

Step 2: Familiarize yourself with related OA&M Operational measurements (continued)

Table 2-4
NetworkBuilder CAIN-related OMs (continued)

Group	Register	Description
CAINIP (continued)	IPUANLEG	Counts the number of **Resource_Clear** messages sent to the SCP with a ***ClearCause*** parameter value of unansweredLeg.
	IPINVLEG	Counts the number of **Resource_Clear** messages sent to the SCP with a ***ClearCause*** parameter value of invalidLeg.
	IPUABNDN	Counts the number of **Resource_Clear** messages sent to the SCP with a ***ClearCause*** parameter value of userAbandon.
	IPINVCOD	Counts the number of **Resource_Clear** messages sent to the SCP with a ***ClearCause*** parameter value of invalidCode.
	IPFAIL	Counts the number of **Resource_Clear** messages sent to the SCP with a ***ClearCause*** parameter value of failure.
	IFCHBSY	Counts the number of **Resource_Clear** messages sent to the SCP with a ***ClearCause*** parameter value of channelsBusy.
	IPRESNAV	Counts the number of **Resource_Clear** messages sent to the SCP with a ***ClearCause*** parameter value of resourceNotAvailable.
	IPISDNTO	Counts the number of **Resource_Clear** messages sent to the SCP with a ***ClearCause*** parameter value of isdnTimeout.
	IPRESTNS	Counts the number of **Resource_Clear** messages sent to the SCP with a ***ClearCause*** parameter value of resourceTypeNotSupported.
	IPTSKRFS	Counts the number of **Resource_Clear** messages sent to the SCP with a ***ClearCause*** parameter value of taskRefused.
	IPINVCRS	Counts the number of **Resource_Clear** messages sent to the SCP with a ***ClearCause*** parameter value of invalidCallerResponse.
	IPCAPFL	Counts the number of **Resource_Clear** messages sent to the SCP with a ***ClearCause*** parameter value of capabilityFailure.
Note 1: Refer to *UCS DMS-250 Operational Measurements Reference Manual* for more information. ***Note 2:*** ITU protocol is not currently supported by NetworkBuilder.		
—continued—		

Step 2: Familiarize yourself with related OA&M Operational measurements (continued)

Table 2-4
NetworkBuilder CAIN-related OMs (continued)

Group	Register	Description
CAINIP (continued)	IPPROTER	Counts the number of **Resource_Clear** messages sent to the SCP with a ***ClearCause*** parameter value of protocolError.
	IPABORT	Counts the number of **Resource_Clear** messages sent to the SCP with a ***ClearCause*** parameter value of abort.
	IPSUPINV	Counts the number of **Resource_Clear** messages sent to the SCP with a ***ClearCause*** parameter value of suppServiceInvoked.
	IPSTRCAN	Counts the number of **Resource_Clear** messages sent to the SCP with a ***ClearCause*** parameter value of strCancelled.
	IPTMPLF	Counts the number of **Resource_Clear** messages sent to the SCP with a ***ClearCause*** parameter value of temporaryFailure.
	IPIPTMO	Counts the number of **Resource_Clear** messages sent to the SCP with a ***ClearCause*** parameter value of ipTimeout.
	IPCTRCAN	Counts the number of **Resource_Clear** messages sent to the SCP with a ***ClearCause*** parameter value of ctrCancelled.
	IPTSTRC	Counts the number of times the TSTRC timer expires.
	IPTDISC	Counts the number of times the TDISC timer expires.
CAINLNP		Provides OMs to track Local Number Portability functionality on a Carrier Advanced Intelligent Network.
	OFCDLOOK	Counts the number of calls subscribed to the *Office_Code* trigger and number of times table OFFCCODE is referenced.
	LNPQUERY	Counts the number of times an LNP query is sent to the SCP.
	LRNONLNP	Counts the number of times an LNP query is made to the SCP and a response message is received containing the location routing number (LRN).

Note 1: Refer to *UCS DMS-250 Operational Measurements Reference Manual* for more information.

Note 2: ITU protocol is not currently supported by NetworkBuilder.

—continued—

Step 2: Familiarize yourself with related OA&M Operational measurements (continued)

Table 2-4
NetworkBuilder CAIN-related OMs (continued)

Group	Register	Description
CAINLNP (continued)	INLNPINF	Counts the number of initial address messages received that indicate LNP has been performed by a previous switch.
	LNPDISCD	Counts the number of times LNP has been performed, but the information obtained is discarded because the terminating agent is a non-ISUP agent, or the terminating agent has been datafilled with option SIGPTDNO in table TRKGRP or TRKFEAT.
	DESTFAIL	Counts the number of times the release cause 26, Ported_Dest_Failed, is received by the switch that performed the query. Release cause 26 is received when the terminating switch could not find the subscriber whose address was in the LNP Generic Address Parameter (GAP). Also counts the number of times release cause 28 is received. Release cause 28 is received when the switch receives an improperly formatted LNP GAP.
	BADGAP	Counts the number of times the switch receives a RELease message with a release cause of 28 indicating that an incorrectly encoded LNP GAP parameter was detected. LNP must be active on the switch.
	BLKBYSTS	Counts the number of times the *Office_Code* trigger is blocked because the NO_LNP option is datafilled against the STS when the ***Info_Analyzed*** TDP is encountered and table CAINSTS referenced.
	TERMLRN	Counts the number of times the switch processes a terminating LRN for the call that matches its own.
	BLKBYST2	Extension register of BLKBYSTS.
	DESTFAI2	Extension register of DESTFAIL.
	INLNPIN2	Extension register of INLNPINF.
	LNPDISC2	Extension register of LNPDISCD.

Note 1: Refer to *UCS DMS-250 Operational Measurements Reference Manual* for more information.

Note 2: ITU protocol is not currently supported by NetworkBuilder.

—continued—

Step 2: Familiarize yourself with related OA&M Operational measurements (continued)

Table 2-4
NetworkBuilder CAIN-related OMs (continued)

Group	Register	Description
CAINLNP (continued)	LNPQUER2	Extension register of LNPQUERY.
	LRNONLN2	Extension register of LRNONLNP.
	OFCDLOO2	Extension register of OFCDLOOK.
	TERMLRN2	Extension register of TERMLRN2.
CAINMGR2		An extension OM group of CAINMSGR. Reports the number of CAIN TCAP messages received from the SCP related to call processing.
	ANLZRTE2	Extension register of register ANLZRTE.
	CONTINU2	Extension register of register CONTINUE.
	SND2RSR2	Extension register of register SND2RSRC.
	CANCRSR2	Extension register of register CANCRSRC.
	RRBCMEV2	Extension register of register RRBCMEVT.
	COLLINF2	Extension register of register COLLINFO.
	CON2RSR2	Extension register of register CON2RSRC.
	CITRSC2	Extension register of register CITRSC.
	AUTHTER2	Extension register of register AUTHTERM.
	SENDNOT2	Extension register of register SENDNOT.
	ACG2	Extension register of register ACG.
	FAMAINF2	Extension register of register FAMAINFO.
CAINMGS2		Extension OM group of CAINMSGS. CAINMGS2 provides OMs for TCAP messages sent to SCP for CAIN processing.
	ORIGATT2	Extension register of register ORIGATT.

Note 1: Refer to *UCS DMS-250 Operational Measurements Reference Manual* for more information.

Note 2: ITU protocol is not currently supported by NetworkBuilder.

—continued—

Step 2: Familiarize yourself with related OA&M Operational measurements (continued)

Table 2-4
NetworkBuilder CAIN-related OMs (continued)

Group	Register	Description
CAINMGS2 (continued)	O_FTRRE2	Extension register of register O_FTRREQ.
	INFOCOL2	Extension register of register INFOCOLL.
	INFOANL2	Extension register of register INFOANLZ.
	NETWBUS2	Extension register of register NETWBUSY.
	OCLDBUS2	Extension register of register OCLDBUSY.
	O_NOANS2	Extension register of register O_NOANSW.
	RSRCCLR2	Extension register of register RSRCCLR.
	OTERMSZ2	Extension register of register OTERMSZ.
	OANSWR2	Extension register of register OANSWR.
	OMIDCAL2	Extension register of register OMIDCALL.
	CTRCLR2	Extension register of register CTRCLR.
	CIFRSC2	Extension register of register CIFRSC.
	TERMATT2	Extension register of register TERMATT.
	TIMEOUT2	Extension register of register TIMEOUTS.
	ODISC2	Extension register of register ODISC.
CAINMSGR		Provides OMs for messages received by the switch.
	ANLZRTE	Counts the number of **`Analyze_Route`** messages received by the switch.
	CONTINUE	Counts the number of **`Continue`** messages received by the switch.
	DISCON	Counts the number of **`Disconnect`** messages received by the switch.

Note 1: Refer to *UCS DMS-250 Operational Measurements Reference Manual* for more information.

Note 2: ITU protocol is not currently supported by NetworkBuilder.

—continued—

Step 2: Familiarize yourself with related OA&M Operational measurements (continued)

Table 2-4
NetworkBuilder CAIN-related OMs (continued)

Group	Register	Description
CAINMSGR (continued)	SND2RSRC	Counts the number of **Send_To_Resource** messages received by the switch.
	FAILREPR	Counts the number of **Failure_Report** messages received by the switch.
	REPERRR	Counts the number of **Report_Error** messages received by the switch.
	CLOSER	Counts the number of **Close** messages received by the switch.
	APPLERRR	Counts the number of **Application_Error** messages received by the switch.
	CANCRSRC	Counts the number of **Cancel_Resource_Event** messages received by the switch.
	RRBCMEVT	Counts the number of **Request_Report_BCM_Event** operations received by the switch.
	CITRSC	Counts the number of **Call_Info_To_Resource** messages received by the switch.
	AUTHTERM	Counts the number of **Authorize_Termination** messages received by the switch.
	COLLINFO	Counts the number of **Collect_Information** messages received by the switch.
	CON2RSRC	Counts the number of **Connect_To_Resource** messages received by the switch.
	SENDNOT	Counts the number of **Send_Notification** components received by the switch.
	ACG	Counts the number of **ACG** messages which are received by the switch.
	ACGRESTR	Counts the number of **ACG_Global_Ctrl_Restore** messages which are received by the switch.

Note 1: Refer to *UCS DMS-250 Operational Measurements Reference Manual* for more information.

Note 2: ITU protocol is not currently supported by NetworkBuilder.

—continued—

Step 2: Familiarize yourself with related OA&M Operational measurements (continued)

Table 2-4
NetworkBuilder CAIN-related OMs (continued)

Group	Register	Description
CAINMSGR (continued)	TR533R	This register is related to Bellcore *TR-NWT-000533* IN/1. See Table 2-5.
	ORIGCALL	Counts the number of **`Originate_Call`** messages the switch receives.
	MERGECL	Counts the number of **`Merge_Call`** messages the switch receives.
	DISCLEG	Counts the number of **`Disconnect_Leg`** messages the switch receives.
	ACKNOW	Counts the number of **`Acknowledge`** messages the switch receives.
	FAMAINFO	Counts the number of **`Furnish_AMA_Information`** TCAP messages received by the switch from the SCP.
CAINMSGS		Provides OMs for CAIN messages sent to the SCP from the switch.
	ORIGATT	Counts the number of **`Origination_Attempt`** messages sent to the SCP.
	O_FTRREQ	Counts the number of **`O_Feature_Requested`** messages sent to the SCP.
	INFOCOLL	Counts the number of **`Info_Collected`** messages sent to the SCP.
	INFOANLZ	Counts the number of **`Info_Analyzed`** messages sent to the SCP.
	NETWBUSY	Counts the number of **`Network_Busy`** messages sent to the SCP.
	OCLDBUSY	Counts the number of **`O_Called_Party_Busy`** messages sent to the SCP.
	O_NOANSW	Counts the number of **`O_No_Answer`** messages sent to the SCP.

Note 1: Refer to *UCS DMS-250 Operational Measurements Reference Manual* for more information.

Note 2: ITU protocol is not currently supported by NetworkBuilder.

—continued—

Step 2: Familiarize yourself with related OA&M Operational measurements (continued)

Table 2-4
NetworkBuilder CAIN-related OMs (continued)

Group	Register	Description
CAINMSGS (continued)	FAILREPS	Counts the number of **Failure_Report** messages sent to the SCP.
	REPERRS	Counts the number of **Report_Error** messages sent to the SCP.
	CLOSES	Counts the number of **Close** messages sent to the SCP.
	APPLERRS	Counts the number of **Application_Error** messages sent to the SCP.
	RSRCCLR	Counts the number of **Resource_Clear** messages sent to the SCP.
	OTERMSZ	Counts the number of **O_Term_Seized** messages sent to the SCP.
	OANSWR	Counts the number of **O_Answer** messages sent to the SCP.
	TIMEOUTS	Counts the number of **Timeout** messages sent to the SCP.
	ODISC	Counts the number of **O_Disconnect** messages sent to the SCP.
	CIFRSC	Counts the number of **Call_Info_From_Resource** messages sent to the SCP, on a per-switch basis.
	TERMATT	Counts the number of **Termination_Attempt** messages sent to the SCP.
	OMIDCALL	Counts the number of **O_Mid_Call** messages sent to the SCP.
	CTRCLR	Counts the number of **CTR_Clear** messages sent to the SCP.
	TR533S	This register is related to Bellcore *TR-NWT-000533* IN/1. See Table 2-5.
	OABANDON	Counts the number of **O_Abandon** EDP-Request messages the switch sends to the SCP.
	FAILUREO	Counts the number of **Failure_Outcome** messages the switch sends to the SCP.
Note 1: Refer to *UCS DMS-250 Operational Measurements Reference Manual* for more information. ***Note 2:*** ITU protocol is not currently supported by NetworkBuilder.		

—continued—

Step 2: Familiarize yourself with related OA&M Operational measurements (continued)

Table 2-4
NetworkBuilder CAIN-related OMs (continued)

Group	Register	Description
CAINNCRS		Provides OMs for non-call-related CAIN messages sent to the SCP from the switch.
	TERMNOT	Counts the number of **`Termination_Notification`** components sent by the switch.
	ACGOVFLW	Counts the number of **`ACG_Overflow`** messages sent by the switch.
	ACGRSCSS	Counts the number of **`ACG_Global_Ctrl_Restore_Success`** messages sent by the switch.
CAINOM		Provides threshold reporting for NetworkBuilder.
	AINTIMO	Counts the number of T1 timeouts that occur when the switch queries the SCP.
	AINTRDNA	Counts the number of times the switch attempts to query the SCP when TCAP transaction identifiers are not available.
	AINSBOUT	Counts the number of queries attempted while the CAIN subsystem is out-of-service.
	AINABNDN	Counts the total number of call abandons on all CAIN calls.
	AINTOVFL	Counts the number of times a query is attempted after a T1 timeout occurs.
	TERMGNCT	Counts the number of termination overflows that occur when the switch attempts direct termination routing methods.
	AINACGBK	Counts number of CAIN queries blocked by ACG controls.
	AINACGRQ	Counts the number of requeries performed due to a query being blocked by an ACG control.
	LAMAFAIL	Counts the number of lost AMA call data due to failure.
	LAMAREXH	Counts the number of lost AMA call data due to resource exhaustion.

Note 1: Refer to *UCS DMS-250 Operational Measurements Reference Manual* for more information.

Note 2: ITU protocol is not currently supported by NetworkBuilder.

—continued—

Step 2: Familiarize yourself with related OA&M Operational measurements (continued)

Table 2-4
NetworkBuilder CAIN-related OMs (continued)

Group	Register	Description
CAINOM (continued)	AINT1TO	Counts the number of T1 timeouts that occur at the *Offhook_Delay* trigger.
	AINT1ATO	Counts the number of abandoned calls while the T1 timer is running at the *Offhook_Delay* trigger.
	AINRSVD1	Reserved register
	AINRSVD2	Reserved register
	AINRSVD3	Reserved register
	AINRSVD4	Reserved register
CAINTRI2		Extension OM group of CAINTRIG. CAINTRI2 OM group provides OMs for each CAIN trigger and EDP.
	QUERY2	Extension register of register QUERY.
	RESPRCV2	Extension register of register RESPRCVD.
	BLOCKED2	Extension register of register BLOCKED.
	IGNORE2	Extension register of register IGNORE.
	NOTRIG2	Extension register of register NOTRIG.
	LEAVETD2	Extension register of register LEAVETDP.
	NO_MATC2	Extension register of register NO_MATCH.
	STRCONV2	Extension register of register STRCONV.
	RCLRCON2	Extension register of register RCLRCONV.
	EDPSRCV2	Extension register of register EDPSRCVD.
	EDPREQ2	Extension register of register EDPREQ.
	EDPNOTI2	Extension register of register EDPNOTIF.
	FEATADD2	Extension register of register FEATADDR.

Note 1: Refer to *UCS DMS-250 Operational Measurements Reference Manual* for more information.

Note 2: ITU protocol is not currently supported by NetworkBuilder.

—continued—

Step 2: Familiarize yourself with related OA&M Operational measurements (continued)

Table 2-4
NetworkBuilder CAIN-related OMs (continued)

Group	Register	Description
CAINTRI2 (continued)	FEATAUT2	Extension register of register FEATAUTH.
	NEXTRTE2	Extension register of register NEXTRTE.
	NXTCNRT2	Extension register of register NXTCNRTE.
	CITR2	Extension register of register CITR.
	CIFR2	Extension register of register CIFR.
	CTRCONV2	Extension register of register CTRCONV.
	CCLRCON2	Extension register of register CCLRCONV.
	FEATCAR2	Extension register of register FEATCARD.
CAINTRIG		Provides a tuple of OMs for each trigger/event. The tuples are: • 0 TOTAL • 1 OFFHKIMM • 3 OFFHKDEL • 4 PRIBCHNL • 5 SIOTRK • 8 SPECFEAT • 10 CUSTDP • 11 SPECDIG • 13 NETBUSY • 14 OCLDBSY • 15 ONOANSW • 16 TERMATT

Note 1: Refer to *UCS DMS-250 Operational Measurements Reference Manual* for more information.

Note 2: ITU protocol is not currently supported by NetworkBuilder.

—continued—

Step 2: Familiarize yourself with related OA&M Operational measurements (continued)

Table 2-4
NetworkBuilder CAIN-related OMs (continued)

Group	Register	Description
CAINTRIG (continued)		• 20 OFTRREQ • 21 OFFCCODE • 22 OTERMSZE • 23 OANSWRE • 24 NETBUSYE • 25 OCLDBSYE • 26 ONOANSRE • 27 TIMEOUT • 28 ODISC • 29 OIECREO • 30 OABANDON • 31 SHF
	QUERY	Counts the number of times a call made on a CAIN-capable agency queries the SCP.
	RESPRCVD	Counts the number of responses the switch receives from the SCP.
	BLOCKED	Counts the number of times a call made on a CAIN-capable agency has BLOCK as the trigger action.
	IGNORE	Counts the number of times a call made on a CAIN-capable agency has IGNORE as the trigger/event action.
	NOTRIG	Counts the number of times a call made on a CAIN-capable agency has CONT_NOTRIG as the trigger action.
	LEAVETDP	Counts the number of times a call made on a CAIN-capable agency has LEAVE_TDP as the trigger action.
Note 1: Refer to *UCS DMS-250 Operational Measurements Reference Manual* for more information. ***Note 2:*** ITU protocol is not currently supported by NetworkBuilder.		

—continued—

Step 2: Familiarize yourself with related OA&M Operational measurements (continued)

Table 2-4
NetworkBuilder CAIN-related OMs (continued)

Group	Register	Description
CAINTRIG (continued)	ERROR	Counts the number of times a call made on a CAIN-capable agency queries the SCP and receives an error message.
	NO_MATCH	Counts the number of times a call made on a CAIN-capable trunk is evaluated for trigger criteria and datafill does not match.
	QUERYSCU	Counts the number of times a call made on a CAIN-capapble trunk has QUERY_SCU datafilled as the trigger action in the trigger table.
	STRCONV	Counts the number of times the switch receives a **`Send_To_Resource`** message from the SCP in a conversation package.
	RCLRCONV	Counts the number of times the switch sends a **`Resource_Clear`** message to the SCP in a conversation package.
	EDPSRCVD	Counts the number of times the switch receives a valid **`Request_Report_BCM_Event`** component.
	EDPREQ	Counts the number of EDP requests sent to the SCP.
	EDPNOTIF	Counts the number of EDP notifications sent to the SCP.
	FEATADDR	Counts the number of times a call made on a CAIN-capable agency has FEAT datafilled as the trigger action in the table OFTRREQ and provisions the ADDR feature processor.
	FEATAUTH	Counts the number of times a call made on a CAIN-capable agency has FEAT datafilled as the trigger action in the table OFTRREQ and provisions the AUTH feature processor.
	FEATCARD	Counts the number of times a call made on a CAIN-capable agency has FEAT datafilled as the trigger action in the table OFTRREQ and provisions the CARD feature processor.
	NEXTRTE	Counts the number of times a call made on a CAIN-capable agency has NEXTRTE as the trigger/event action.
Note 1: Refer to *UCS DMS-250 Operational Measurements Reference Manual* for more information. ***Note 2:*** ITU protocol is not currently supported by NetworkBuilder.		
—continued—		

Step 2: Familiarize yourself with related OA&M Operational measurements (continued)

Table 2-4
NetworkBuilder CAIN-related OMs (continued)

Group	Register	Description
CAINTRIG (continued)	NXTCNRTE	Counts the number of times a call made on a CAIN-capable agency has NEXTCNRTE as the trigger/event action.
	VIPREQ	Counts the number of times the SCP requests VIP handling, for either message type (**Send_To_Resource** or **Connect_To_Resource**) on a per-trigger basis.
	VIPRESP	Counts the number of times a response with a ***ClearCause*** value of `normal` is sent to the SCP after VIP handling, for either message type (**Resource_Clear** or **CTR_Clear**), on a per-trigger basis.
	CITR	Counts the number of **Call_Info_To_Resource** messages received by the switch, on a per-trigger basis.
	CIFR	Counts the number of **Call_Info_From_Resource** messages sent to the SCP, on a per-trigger basis.
	CTRCONV	Counts the number of conversational **Connect_To_Resource** messages the switch received from the SCP.
	CCLRCONV	Counts the number of **CTR_Clear** conversational messages sent to the SCP.
CAINUIF		Provides OMs for TCAP **Send_To_Resource** and **Connect_To_Resource** messages.
	AINTOTDC	Counts the total number of digit collection attempts by the UIF.
	AINTOTAN	Counts the total number of attempts to play an announcement by the UIF.
	AINUSRAB	Counts the total number of user abandons during use of the UIF.
	AINRSCNA	Counts the number of times resources were determined to be unavailable by the UIF.
	AINRSCNI	Counts the number of times resources were determined to be not implemented or installed by the UIF.

Note 1: Refer to *UCS DMS-250 Operational Measurements Reference Manual* for more information.

Note 2: ITU protocol is not currently supported by NetworkBuilder.

—continued—

Step 2: Familiarize yourself with related OA&M Operational measurements (continued)

Table 2-4
NetworkBuilder CAIN-related OMs (continued)

Group	Register	Description
CAINUIF (continued)	AINTOTTN	Counts the total number of attempts to play a tone by the UIF.
	AINBUFFR	Counts the number of buffering attempts by the UIF.
	AINBUFOV	counts the number of buffer overflow events by the UIF.
	AINIPSIGN	Counts the number of times permanent signal was received by the UIF.
	AINPDIAL	Counts the number of times partial dial was received by the UIF.
EXT		Monitors use of extension blocks. The tuples related to CAIN are: • 152 CAIN_FRAMEWORK_EXT_BLOCK • 153 CAIN_EXT_BLOCK • 176 CAIN_ECCB_BLOCK • 177 CAIN_STR_EXT_BLOCK • 197 T_CAIN_EXT_BLOCK • 199 SN_CAIN_EXT_BLOCK
	EXTSEIZ	Counts the times a request for an extension block is successful.
	EXTOVFL	Counts number of times an extension block is unavailable.
	EXTHI	Records the maximum number of extensions that are in simultaneous use during the preceding OM transfer period.
	EXTSEIZ2	Extension register for EXTSEIZ.
	EXTHI2	Extension register for EXTHI.
VAMPACG		Tuple 1 CAIN02 measures the use of the VAMP ACG control system for applications in table VAMPTRID for CAIN.

Note 1: Refer to *UCS DMS-250 Operational Measurements Reference Manual* for more information.

Note 2: ITU protocol is not currently supported by NetworkBuilder.

—continued—

Step 2: Familiarize yourself with related OA&M Operational measurements (continued)

Table 2-4
NetworkBuilder CAIN-related OMs (continued)

Group	Register	Description
VAMPACG (continued)	VACGADD	Counts ACG controls that have been added to the control list for the application.
	VACGEXP	Counts ACG controls that have expired on the duration timer.
	VACGREM	Counts ACG controls that have been removed from the duration timer.
	VACGUPD	Counts ACG controls that have been updated. A control is updated when VAMP receives an **ACG** component that contains a control that already exists in the control list for that application.
	VACGQCAC	Counts the queries for an application that have been checked against the ACG controls in the control list.
	VACGBLK	Counts an applications queries that have been blocked by an ACG control in the control list.
	VACGRSET	Counts **ACG_Global_Ctrl_Restore** messages received by VAMP for an application.
	VACGBLK2	Extension register of VACGBLK.
	VACGQCA2	Extension register of VACGQCAC.
VPTRUSAG		Tuple 1 CAIN02 measures the use of the transaction identifier blocks (TRID) for CAIN.
	TRIDSEIZ	Measures seizures of TRID.
	TRIDUSED	Tracks the highest number of TRID blocks in use at one time.
	TRIDOVFL	Measures failed attempts to seize TRID blocks.
	TRIDCMPS	Tracks the highest number of component identifier (COMP) blocks allocated against a single TRID block.
	TRIDMSGS	Tracks the highest number of message buffers (MESG) allocated against a single TRID block.
	COMPSEIZ	Measures seizures of COMP blocks.

Note 1: Refer to *UCS DMS-250 Operational Measurements Reference Manual* for more information.

Note 2: ITU protocol is not currently supported by NetworkBuilder.

—continued—

Step 2: Familiarize yourself with related OA&M Operational measurements (continued)

Table 2-4
NetworkBuilder CAIN-related OMs (continued)

Group	Register	Description
VPTRUSAG (continued)	COMPUSED	Tracks highest number of COMP blocks in use at any one time.
	COMPOVFL	Measures failed attempts to seize COMP blocks.
	COMPT1TM	Measures number of COMP blocks for which T1 timing was requested.
	COMPT1TO	Measures number of COMP blocks for which T1 timer timed out without receiving a response.
	MESGSEIZ	Measures seizures of MESG blocks.
	MESGUSED	Tracks highest number of MESG blocks in use at any one time.
	MESGOVFL	Measures failed attempts to seize MESG blocks.
VTCAPERR (note 2)		Tuple 1 CAIN02 counts the number of protocol errors detected in TCAP messages, packages, and components received using VAMP TCAP for CAIN. This OM group decodes the peg count of VAMP TCAP detected errors.
	VTETPBAD	Counts the number of bad or incorrect transaction portion errors.
	VTETPPKG	Counts the number of unrecognized package type errors.
	VTETPPRM	Counts the number of "permission to release" errors when transactions are terminated. This OM groups' peg count also includes query or last conversation package errors sent "without permission."
	VTETPUCS	Counts the number of unexpected component sequence errors.
	VTETPUID	Counts the number of unrecognized transaction ID errors.
	VTETPNID	Counts the number of missing respond ID errors.
	VTEDPBAD	Counts the number of bad or incorrect dialog portion errors.
	VTEDPDLG	Counts the number of unrecognized dialog ID errors.

Note 1: Refer to *UCS DMS-250 Operational Measurements Reference Manual* for more information.

Note 2: ITU protocol is not currently supported by NetworkBuilder.

—continued—

Step 2: Familiarize yourself with related OA&M Operational measurements (continued)

Table 2-4
NetworkBuilder CAIN-related OMs (continued)

Group	Register	Description
VTCAPERR (continued)	VTEDPINC	Counts the number of inconsistent or missing dialog portion errors.
	VTECPBAD	Counts the number of bad or incorrect component portion errors.
	VTECPCMP	Counts the number of unrecognized component errors.
	VTECPDID	Counts the number of duplicate invoke ID errors.
	VTECPCID	Counts the number of unrecognized correlation ID errors.
	VTECPOPC	Counts the number of unrecognized operation or code errors.
	VTECPUPS	Counts the number of unexpected parameter set/sequence errors.
	VTEPMBAD	Counts the number of bad or incorrect parameter errors.
	VTEPMPID	Counts the number of unrecognized parameter ID errors.
	VTEPMMND	Counts the number of missing mandatory parameter errors.
	VTEPMCND	Counts the number of missing conditional parameter errors.
	VTEPMERR	Counts the number of incorrect parameter data errors.
	VTEUCOMM	Counts the number of unexpected communication errors.
	VTENETWK	Counts the number of network errors, when an undeliverable VAMP message is returned.
	VTENRSRC	Counts the number of lack of resources or decoding software errors using VAMP TCAP.
VTCAPRCV (note 2)	VTEOTHER	Counts the number of other errors including internal or miscellaneous errors.
		Tuple 1 CAIN02 is pegged when TCAP messages, packages, and components are received using VAMP TCAP for CAIN.
	VTRMESGS	Counts all TCAP messages received using VAMP TCAP regardless of transport mechanism.

Note 1: Refer to *UCS DMS-250 Operational Measurements Reference Manual* for more information.

Note 2: ITU protocol is not currently supported by NetworkBuilder.

—continued—

Step 2: Familiarize yourself with related OA&M Operational measurements (continued)

Table 2-4
NetworkBuilder CAIN-related OMs (continued)

Group	Register	Description
VTCAPRCV (continued)	VTRQWPRM	Counts the number of TCAP packages received with ANSI Query with Permission or ITU-T Begin.
	VTRQWOPM	Counts the number of TCAP packages received using ANSI Query without permission.
	VTRCWPRM	Counts the number of TCAP packages received that contain package type ANSI Conversation with Permission or ITU-T Continue.
	VTRCWOPM	Counts the number of TCAP packages received using ANSI Conversation without permission.
	VTRRESPN	Counts the number of TCAP packages received containing package type ANSI Response or ITU-T End.
	VTRUNIDR	Counts the number of TCAP packages received with package type ANSI or ITU-T Unidirectional.
	VTRPABRT	Counts the number of TCAP packages received with ANSI or ITU-T Abort with P-Abort cause.
	VTRUABRT	Counts the number of TCAP packages received with ANSI or ITU-T Abort with U-Abort cause.
	VTRINVKL	Counts the number of TCAP components received with ANSI Invoke (Last).
	VTRINVNL	Counts the number of TCAP components with ANSI Invoke (Non-Last) or ITU-T.
	VTRRTRSL	Counts the number of TCAP components with ANSI or ITU-T Return Result (Last).
	VTRRTRNL	Counts the number of TCAP components with ANSI or ITU-T Return Result (Non-Last).
	VTRERROR	Counts the number of TCAP components with ANSI or ITU-T Return Error.

Note 1: Refer to *UCS DMS-250 Operational Measurements Reference Manual* for more information.

Note 2: ITU protocol is not currently supported by NetworkBuilder.

—continued—

Step 2: Familiarize yourself with related OA&M Operational measurements (continued)

Table 2-4
NetworkBuilder CAIN-related OMs (continued)

Group	Register	Description
VTCAPRCV (continued)	VTRREJCT	Counts the number of TCAP components with ANSI or ITU-T Reject.
	VTRDLOGP	Counts the number of TCAP components containing a TCAP dialog portion.
VTCAPSNT (note 2)		Tuple 1 CAIN02 is pegged when TCAP messages, packages, and components sent using VAMP TCAP for CAIN.
	VTSMESGS	Counts all TCAP messages using VAMP TCAP regardless of transport mechanism.
	VTSQWPRM	Counts the number of TCAP packages sent containing ANSI Query with Permission or ITU-T Begin.
	VTSQWOPM	Counts the number of TCAP packages sent using ANSI Query without permission.
	VTSCWPRM	Counts the number of TCAP packages sent that contain package type ANSI Conversation with Permission or ITU-T Continue.
	VTSCWOPM	Counts the number of TCAP packages sent using ANSI Conversation without permission.
	VTSRESPN	Counts the number of TCAP packages sent containing package type ANSI Response or ITU-T End.
	VTSUNIDR	Counts the number of TCAP packages sent with package type ANSI or ITU-T Unidirectional.
	VTSPABRT	Counts the number of TCAP packages sent with ANSI or ITU-T Abort with P-Abort cause.
	VTSUABRT	Counts the number of TCAP packages sent with ANSI or ITU–T Abort with U-Abort cause.
	VTSINVKL	Counts the number of TCAP components sent with ANSI Invoke (Last).

Note 1: Refer to *UCS DMS-250 Operational Measurements Reference Manual* for more information.

Note 2: ITU protocol is not currently supported by NetworkBuilder.

—continued—

Step 2: Familiarize yourself with related OA&M Operational measurements (continued)

Table 2-4
NetworkBuilder CAIN-related OMs (continued)

Group	Register	Description
VTCAPSNT (continued)	VTSINVNL	Counts the number of TCAP components sent with ANSI Invoke (Non-Last) or ITU-T.
	VTSRTRSL	Counts the number of TCAP components sent with ANSI or ITU-T Return Result (Last).
	VTSRTRNL	Counts the number of TCAP components sent with ANSI or ITU-T Return Result (Non-Last).
	VTSERROR	Counts the number of TCAP components sent with ANSI or ITU-T Return Error.
	VTSREJCT	Counts the number of TCAP components sent with ANSI or ITU-T Reject.
	VTSDLOGP	Counts the number of TCAP components sent containing a TCAP dialog portion.
Note 1: Refer to *UCS DMS-250 Operational Measurements Reference Manual* for more information. ***Note 2:*** ITU protocol is not currently supported by NetworkBuilder.		
—end—		

The following table lists the OMs pegged by the switch in association with Bellcore *TR-NWT-000533* IN/1:

Step 2: Familiarize yourself with related OA&M Operational measurements (continued)

Table 2-5
NetworkBuilder TR-NWT-000533 IN/1-related OMs

Group	Register	Description
CAINMGR2		An extension OM group of CAINMSGR. Reports the number of CAIN TCAP messages received from the SCP related to call processing.
	TR533R2	Extension register of register TR533R.
CAINMGS2		An extension OM group of CAINMSGS. Reports the number of TCAP messages sent to the SCP for CAIN processing.
	TR533S2	Extension register of register TR533S.
CAINMSGR		Provides OMs for messages received by the switch.
	TR533R	Counts the number of TCAP messages which are received by the switch related to Bellcore's *TR-NWT-000533* Toll-Free Services.
CAINMSGS		Provides OMs for messages sent to the SCP from the switch.
	TR533S	Counts the number of TCAP messages sent by the switch related to Bellcore's *TR-NWT-000533* Toll-Free Services.
TFREE533		Provides reporting information pertaining to Bellcore's *TR-NWT-000533* Toll-Free Services.
	N00CALL	Counts the number of toll-free calls attempted.
	NOMATCHT	Counts the number of toll-free calls which access the TOLLFREE trigger table without locating a matching tuple.
	IGNOREDT	Counts the number of toll-free calls which access the TOLLFREE trigger table with an action of IGNORE.
	BLOCKEDT	Counts the number of toll-free calls which access the TOLLFREE trigger table with an action of BLOCK.
	LECROUTE	Counts the number of toll-free call routing responses that indicate LEC routing will be used.
	IXCROUTE	Counts the number of toll-free call routing responses that indicate IXC routing will be used.

Note 3: Refer to *UCS DMS-250 Operational Measurements Reference Manual* for more information.

Note 4: ITU protocol is not currently supported by NetworkBuilder.

—continued—

Step 2: Familiarize yourself with related OA&M Operational measurements (continued)

Table 2-5
NetworkBuilder TR-NWT-000533 IN/1-related OMs (continued)

Group	Register	Description
	PLAYANN	Counts the number of toll-free call Play Announcement responses received.
	NOTIFREQ	Counts the number of toll-free call requests for **`Termination_Information`** components.
TFREE533 (continued)	VACTBLK	Counts the number of toll-free call attempts that are blocked due to ACG controls for vacant codes (control cause = #01).
	OOBBLK	Counts the number of toll-free call attempts that are blocked due to ACG controls for non-purchased NPAs (control cause = #02).
	SOLVDBLK	Counts the number of toll-free call attempts that are blocked due to ACG controls for SCP overload controls (control cause = #03).
	MASSBLK	Counts the number of toll-free call attempts that are blocked due to ACG controls for mass calling controls (control cause = #04).
	SMSBLK	Counts the number of toll-free call attempts that are blocked due to ACG controls for SMS initiated controls (control cause = #05).
	VACTOVFL	Counts the number of toll-free call attempts that are to be blocked due to ACG controls for 10-digit vacant codes, but cannot due to ACG control list overflow.
	VAC6OVFL	Counts the number of toll-free call attempts that are to be blocked due to ACG controls for 6-digit vacant codes, but cannot due to ACG control list overflow.
	OOBOVFL	Counts the number of toll-free call attempts that are to be blocked due to ACG controls for non-purchased NPAs, but cannot due to ACG control list overflow.

Note 3: Refer to *UCS DMS-250 Operational Measurements Reference Manual* for more information.

Note 4: ITU protocol is not currently supported by NetworkBuilder.

—continued—

Step 2: Familiarize yourself with related OA&M Operational measurements (continued)

Table 2-5
NetworkBuilder TR-NWT-000533 IN/1-related OMs (continued)

Group	Register	Description
	MASSOVFL	Counts the number of toll-free call attempts that are to be blocked due to ACG controls for mass calling, but cannot due to ACG control list overflow.
	SMSOVFL	Counts the number of toll-free call attempts that are to be blocked due to ACG SMS initiated controls, but cannot due to ACG control list overflow.
TFREE533 (continued)	SCPOVFL	Counts the number of toll-free call attempts that are to be blocked due to ACG SCP overload controls, but cannot due to ACG control list overflow.
	SPAREBLK	Counts the number of query attempts blocked due to ACG spare value control.
	SPAROVFL	Counts the number of messages received by the switch containing a ACG spare value control that overflows.
	IXCROUT2	Extension register of register IXCROUTE.
	LECROUT2	Extension register of register LECROUTE.
	N00CALL2	Extension register of register N00CALL.
VAMPACG		Tuple 2 IN1 measures the use of the VAMP ACG control system for applications in table VAMPTRID for Bellcore *TR-NWT-000533* IN/1.
	VACGADD	Counts ACG controls that have been added to the control list for the application.
	VACGEXP	Counts ACG controls that have expired on the duration timer.
	VACGREM	Counts ACG controls that have been removed from the duration timer.
	VACGUPD	Counts ACG controls that have been updated. A control is updated when VAMP receives an **ACG** component that contains a control that already exists in the control list for that application.

Note 3: Refer to *UCS DMS-250 Operational Measurements Reference Manual* for more information.

Note 4: ITU protocol is not currently supported by NetworkBuilder.

—continued—

Step 2: Familiarize yourself with related OA&M Operational measurements (continued)

Table 2-5
NetworkBuilder TR-NWT-000533 IN/1-related OMs (continued)

Group	Register	Description
	VACGQCAC	Counts the queries for an application that have been checked against the ACG controls in the control list.
	VACGBLK	Counts an applications queries that have been blocked by an ACG control in the control list.
	VACGRSET	Counts **`ACG_Global_Ctrl_Restore`** messages received by VAMP for an application.
VPTRUSAG		Tuple 2 IN1 measures the use of the transaction identifier blocks (TRID) for Bellcore *TR-NWT-000533* IN/1.
	TRIDSEIZ	Measures seizures of TRID.
	TRIDUSED	Tracks the highest number of TRID blocks in use at one time.
	TRIDOVFL	Measures failed attempts to seize TRID blocks.
	TRIDCMPS	Tracks the highest number of component identifier (COMP) blocks allocated against a single TRID block.
	TRIDMSGS	Tracks the highest number of message buffers (MESG) allocated against a single TRID block.
	COMPSEIZ	Measures seizures of COMP blocks.
	COMPUSED	Tracks highest number of COMP blocks in use at any one time.
	COMPOVFL	Measures failed attempts to seize COMP blocks.
	COMPT1TM	Measures number of COMP blocks for which T1 timing was requested.
	COMPT1TO	Measures number of COMP blocks for which T1 timer timed out without receiving a response.
	MESGSEIZ	Measures seizures of MESG blocks.
	MESGUSED	Tracks highest number of MESG blocks in use at any one time.
	MESGOVFL	Measures failed attempts to seize MESG blocks.

Note 3: Refer to *UCS DMS-250 Operational Measurements Reference Manual* for more information.

Note 4: ITU protocol is not currently supported by NetworkBuilder.

—continued—

Step 2: Familiarize yourself with related OA&M Operational measurements (continued)

Table 2-5
NetworkBuilder TR-NWT-000533 IN/1-related OMs (continued)

Group	Register	Description
VTCAPERR (note 2)		Tuple 2 IN1 counts the number of protocol errors detected in TCAP messages, packages, and components received using VAMP TCAP for Bellcore *TR-NWT-000533* IN/1. This OM group decodes the peg count of VAMP TCAP detected errors.
	VTETPBAD	Counts the number of bad or incorrect transaction portion errors.
	VTETPPKG	Counts the number of unrecognized package type errors.
VTCAPERR (continued)	VTETPPRM	Counts the number of "permission to release" errors when transactions are terminated. This OM group's peg count also includes query or last conversation package errors sent "without permission."
	VTETPUCS	Counts the number of unexpected component sequence errors.
	VTETPUID	Counts the number of unrecognized transaction ID errors.
	VTETPNID	Counts the number of missing respond ID errors.
	VTEDPBAD	Counts the number of bad or incorrect dialog portion errors.
	VTEDPDLG	Counts the number of unrecognized dialog ID errors.
	VTEDPINC	Counts the number of inconsistent or missing dialog portion errors.
	VTECPBAD	Counts the number of bad or incorrect component portion errors.
	VTECPCMP	Counts the number of unrecognized component errors.
	VTECPDID	Counts the number of duplicate invoke ID errors.
	VTECPCID	Counts the number of unrecognized correlation ID errors.
	VTECPOPC	Counts the number of unrecognized operation or code errors.
	VTECPUPS	Counts the number of unexpected parameter set/sequence errors.

Note 3: Refer to *UCS DMS-250 Operational Measurements Reference Manual* for more information.

Note 4: ITU protocol is not currently supported by NetworkBuilder.

—continued—

Step 2: Familiarize yourself with related OA&M Operational measurements (continued)

Table 2-5
NetworkBuilder TR-NWT-000533 IN/1-related OMs (continued)

Group	Register	Description
	VTEPMBAD	Counts the number of bad or incorrect parameter errors.
	VTEPMPID	Counts the number of unrecognized parameter ID errors.
	VTEPMMND	Counts the number of missing mandatory parameter errors.
	VTEPMCND	Counts the number of missing conditional parameter errors.
	VTEPMERR	Counts the number of incorrect parameter data errors.
	VTEUCOMM	Counts the number of unexpected communication errors.
VTCAPERR (continued)	VTENETWK	Counts the number of network errors, when an undeliverable VAMP message is returned.
	VTENRSRC	Counts the number of lack of resources or decoding software errors using VAMP TCAP.
VTCAPRCV (note 2)	VTEOTHER	Counts the number of other errors including internal or miscellaneous errors. Tuple 2 IN1 is pegged when TCAP messages, packages, and components are received using VAMP TCAP for Bellcore *TR-NWT-000533* IN/1.
	VTRMESGS	Counts all TCAP messages received using VAMP TCAP regardless of transport mechanism.
	VTRQWPRM	Counts the number of TCAP packages received with ANSI Query with Permission or ITU-T Begin.
	VTRQWOPM	Counts the number of TCAP packages received using ANSI Query without permission.
	VTRCWPRM	Counts the number of TCAP packages received that contain package type ANSI Conversation with Permission or ITU-T Continue.
	VTRCWOPM	Counts the number of TCAP packages received using ANSI Conversation without permission.

Note 3: Refer to *UCS DMS-250 Operational Measurements Reference Manual* for more information.

Note 4: ITU protocol is not currently supported by NetworkBuilder.

—continued—

Step 2: Familiarize yourself with related OA&M Operational measurements (continued)

Table 2-5
NetworkBuilder TR-NWT-000533 IN/1-related OMs (continued)

Group	Register	Description
	VTRRESPN	Counts the number of TCAP packages received containing package type ANSI Response or ITU-T End.
	VTRUNIDR	Counts the number of TCAP packages received with package type ANSI or ITU-T Unidirectional.
	VTRPABRT	Counts the number of TCAP packages received with ANSI or ITU-T Abort with P-Abort cause.
	VTRUABRT	Counts the number of TCAP packages received with ANSI or ITU-T Abort with U-Abort cause.
VTCAPRCV (continued)	VTRINVKL	Counts the number of TCAP components received with ANSI Invoke (Last).
	VTRINVNL	Counts the number of TCAP components with ANSI Invoke (Non-Last) or ITU-T.
	VTRRTRSL	Counts the number of TCAP components with ANSI or ITU-T Return Result (Last).
	VTRRTRNL	Counts the number of TCAP components with ANSI or ITU-T Return Result (Non-Last).
	VTRERROR	Counts the number of TCAP components with ANSI or ITU-T Return Error.
	VTRREJCT	Counts the number of TCAP components with ANSI or ITU-T Reject.
	VTRDLOGP	Counts the number of TCAP components containing a TCAP dialog portion.
VTCAPSNT (note 2)		Tuple 2 IN1 is pegged when TCAP messages, packages, and components sent using VAMP TCAP for Bellcore *TR-NWT-000533* IN/1.
	VTSMESGS	Counts all TCAP messages using VAMP TCAP regardless of transport mechanism.

Note 3: Refer to *UCS DMS-250 Operational Measurements Reference Manual* for more information.

Note 4: ITU protocol is not currently supported by NetworkBuilder.

—continued—

Step 2: Familiarize yourself with related OA&M Operational measurements (continued)

Table 2-5
NetworkBuilder TR-NWT-000533 IN/1-related OMs (continued)

Group	Register	Description
	VTSQWPRM	Counts the number of TCAP packages sent containing ANSI Query with Permission or ITU-T Begin.
	VTSQWOPM	Counts the number of TCAP packages sent using ANSI Query without permission.
	VTSCWPRM	Counts the number of TCAP packages sent that contain package type ANSI Conversation with Permission or ITU-T Continue.
	VTSCWOPM	Counts the number of TCAP packages sent using ANSI Conversation without permission.
VTCAPSNT (continued)	VTSRESPN	Counts the number of TCAP packages sent containing package type ANSI Response or ITU-T End.
	VTSUNIDR	Counts the number of TCAP packages sent with package type ANSI or ITU-T Unidirectional.
	VTSPABRT	Counts the number of TCAP packages sent with ANSI or ITU-T Abort with P-Abort cause.
	VTSUABRT	Counts the number of TCAP packages sent with ANSI or ITU–T Abort with U-Abort cause.
	VTSINVKL	Counts the number of TCAP components sent with ANSI Invoke (Last).
	VTSINVNL	Counts the number of TCAP components sent with ANSI Invoke (Non-Last) or ITU-T.
	VTSRTRSL	Counts the number of TCAP components sent with ANSI or ITU-T Return Result (Last).
	VTSRTRNL	Counts the number of TCAP components sent with ANSI or ITU-T Return Result (Non-Last).
	VTSERROR	Counts the number of TCAP components sent with ANSI or ITU-T Return Error.

Note 3: Refer to *UCS DMS-250 Operational Measurements Reference Manual* for more information.

Note 4: ITU protocol is not currently supported by NetworkBuilder.

—continued—

Step 2: Familiarize yourself with related OA&M Operational measurements (end)

Table 2-5
NetworkBuilder TR-NWT-000533 IN/1-related OMs (continued)

Group	Register	Description
	VTSREJCT	Counts the number of TCAP components sent with ANSI or ITU-T Reject.
	VTSDLOGP	Counts the number of TCAP components sent containing a TCAP dialog portion.
Note 3: Refer to *UCS DMS-250 Operational Measurements Reference Manual* for more information. ***Note 4:*** ITU protocol is not currently supported by NetworkBuilder.		
—end—		

Step 2: Familiarize yourself with related OA&M Data schema

The following table lists the data schema tables related to NetworkBuilder:

Note: AXXESS agents require different data schema tables. Refer to *UCS DMS-250 CAIN/FlexDial Interactions* for more information.

Table 2-6
NetworkBuilder-related data schema

Table	Description
Agent provisioning	
CALLATTR	CALL ATTRIBUTES. Provision a PRI call attribute as CAIN-capable or T_CAIN-capable
TRKGRP	TRUNK GROUP. Identifies the trunk group as CAIN-capable. Also provides an index into table TRKFEAT for AXXESS agents.
TRKFEAT	TRUNK GROUP FEATURES. Identifies the features available on an AXXESS agent, such as CAIN or TCAIN-capability and related CAIN group.
Announcements and tones	
CAINRSRC	CAIN RESOURCE. Maps the resources identified by the SCP in a **`Send_To_Resource`** or **`Analyze_Route`** (`callBranding` extension parameter) message to an announcement resource available on the switch.
CCS7 connectivity	
C7GTT	CCS7 GLOBAL TITLE TRANSLATION. Maps a translation type (defined in table C7GTTYPE) to a CCS7 network address.
C7GTTYPE	CCS7 GLOBAL TITLE TYPE. Maps a CCS7-defined translation to a network-defined global title translation type.
C7LINK	CCS7 LINK. Makes the association between the physical equipment of the link and the logical view of the link as a member of a linkset.
C7LKSET	CCS7 LINK SET. Defines the characteristics of a linkset. A linkset is a set of links used as a group. Each link carries traffic between the origination point code and a destination point code. The table also defines attributes that are common to all links in the link set. The links are defined in table C7LINK.
C7LOCSSN	CCS7 LOCATION SUBSYSTEM NUMBER. Defines the subsystems located on the switch.
Note: Refer to *UCS DMS-250 Data Schema Reference Manual* for more data schema information.	

—continued—

Step 2: Familiarize yourself with related OA&M Data schema (continued)

Table 2-6
NetworkBuilder-related data schema (continued)

Table	Description
C7NETSSN	CCS7 NETWORK SUBSYSTEM NUMBER. Provides the set of remote point codes (PC) and subsystems, at the remote PCs, where messages are routed by the SCCP. A PC is a node in the CCS7 network that may be an SSP, an STP, or an SCP. SCCP routes messages to subsystems at the PC, including SCCP management (SCMG) or SCCP itself, for further global title translation.
C7NETWK	CCS7 NETWORK. Describes the signaling networks in use in a switching office.
C7RPLSSN	CCS7 REPLICATE SUBSYSTEM. Provides the set of remote subsystem replicate pairs. It has a one part key, the subsystem name. For each subsystem a list of PC pairs at which the replicated subsystems reside must be given.
C7RSSCRN	CCS7 REMOTE SUBSYSTEM CONCERNED NODE. Provides a list of concerned nodes for a remote subsystem point code combination. The table has a two part key. The first part is the PC and the second part is the subsystem name. The PC and subsystem combination must be datafilled in table C7NETSSN.
C7RTESET	CCS7 ROUTE SET. Associates linksets used as possible routes for each signaling point in the network. An office point code identifies a signaling point within any network. Each office point code must have a routeset. The information in this table records which routes and linksets can carry the signaling information to the destination signaling point. This table is also used for alternate routing decisions.
VAMPTRID General	VAMP TRANSACTION IDENTIFIERS. Provisions the key resources used in Carrier AIN messaging, including transaction and component identifiers and message buffers.
CAINPARM	CAIN PARAMETERS. Provides default CAIN data for the office. The following data is provided: • determines the global title to be used for requery in the event that a query is blocked due to an ACG control being encountered (ACG_OVERFLOW_GT) • determines the treatment to be applied to a call that is blocked by an ACG control when the error action provisioned in the applicable trigger table is set to TREAT (ACG_TREATMENT)
Note: Refer to *UCS DMS-250 Data Schema Reference Manual* for more data schema information.	
—continued—	

Determines if TRTMTCD and COMPCODE fields in CDR to be zapped to Zero for calls terminated by DISCONNECT message (TRTMTCD_COMPCODE_ZAPPED_ZERO)

Step 2: Familiarize yourself with related OA&M Data schema (continued)

Table 2-6
NetworkBuilder-related data schema (continued)

Table	Description
CAINPARM (continued)	• indicates the type of global title encoding to be used for the CAIN_ADDR_GT global title type (ADDR_GT_FORMAT) • determines whether a network busy condition sets GNCT or RTTE treatment (ALLOW_RTTE_TRTMT) • maximum number of conversations allowed during a call (CAIN_CONVERSATION_LIMIT) • default global title value (CAIN_DEFAULT_GT) • when unable to connect to SCP due to T1 timeout, routes to the alternate SCP defined here (CAIN_DEFAULT_OVERFLOW_GT) • office subscription group (CAIN_OFFICE_GROUP) • determines whether certain messages or parameters are allowed to be sent in outgoing CAIN TCAP packages (CAIN_PROTOCOL_STREAM) • determines the encoding format to use for parameter whose encodings have changed between software releases (CAIN_PROTOCOL_VERSION) • maximum number of times a subscriber can reset dialing (CAIN_STR_RESET_ALLOWED) • SCP request time-out (CAIN_T1_TIMEOUT) • CAIN900-series log generation (CAIN900_LOGS_ENABLED) • indicates the type of global title encoding to be used for the CAIN_CLID_GT global title type (CLID_GT_FORMAT) • default SNPA (DEFAULT_SNPA) • indicates the type of global title encoding to be used for the CAIN_FEAT_GT global title type (FEAT_GT_FORMAT) • controls whether the *Specific_Feature_Code*, *Customized_Dialing_Plan*, and *Specific_Digit_String* triggers are allowed on the second call leg of an SS7Inter-IMT RLT call (INFOANALYZED_FOR_RLT) • controls how the `OverflowRoutingNo` and ***`CalledPartyID`*** parameters are translated (INTL_XLA_TYPE)
Note: Refer to *UCS DMS-250 Data Schema Reference Manual* for more data schema information.	
—continued—	

Step 2: Familiarize yourself with related OA&M Data schema (continued)

Table 2-6
NetworkBuilder-related data schema (continued)

Table	Description
CAINPARM (continued)	• indicates whether a new called number should be evaluated against the trigger criteria from table OFFCCODE (LNP_FOR_RX_SELECTOR) • CAIN301 log generation (LNP_LOGS_ENABLED) • indicates whether to use a limited set of parameters in the **`Info_Analyzed`** message for LNP queries (LNP_PARAMETER_SET) • determines whether certain LNP parameters are allowed to be sent in outgoing packages (LNP_PROTOCOL_STREAM) • determines the encoding format to use for LNP parameters (LNP_PROTOCOL_VERSION) • CLLI routing requirements (MATCH_TERMRTE_CLLI) • maximum number of times the switch can send a **`Failure_Outcome`** message per call (MAX_FAILURE_OUTCOMES) • maximum number of times a call can send a TDP-Request and/or EDP-Request to the SCP (MAX_NUM_SERIAL_TRIGGERS) • controls whether a CAIN303 log is generated or not (NB_ROUTING LOG) • specifies the announcement ID which when received from an SCP causes the CAIN303 log to be generated (NB_ROUTING_ANNC_ID) • number of extended call condense blocks available (NUM_CAIN_ECCBS) • CAIN extension blocks available (NUM_CAIN_EXT_BLOCKS) • Framework extension blocks available (NUM_FRAMEWORK_EXT_BLOCKS) • indicates the number of CAIN furnish AMA extension blocks to be allocated for a call (NUM_FURNISHEDAMA_EXT_BLOCKS) • number of send notification extension blocks allowed for the given UCS DMS-250 (NUM_SEND_NOTIFICATION_EXT_BLOCKS) • number of Send_To_Resource extension blocks available (NUM_STR_EXT_BLOCKS)
Note: Refer to *UCS DMS-250 Data Schema Reference Manual* for more data schema information.	
—continued—	

Step 2: Familiarize yourself with related OA&M Data schema (continued)

Table 2-6
NetworkBuilder-related data schema (continued)

Table	Description
CAINPARM (continued)	• number of T_CAIN extension blocks available (NUM_T_CAIN_EXT_BLOCKS) • O_No_Answer timer value (O_NO_ANSWER_TIMER) • indicates the type of global title encoding to be used for the CAIN_OFCD_GT global title type (OFCD_GT_FORMAT) • determines whether call screening encounters the *Off_Hook_Delay* trigger with the (OFFHKDEL_TRIG_AFTER_TREAT) • indicates method to store the digits collected by the *O_Feature_Requested* trigger and FLEXDIAL for later retrieval by both CAIN and FLEXDIAL (OFTRREQ_FLEXTYPE_MAP) • controls PrivateFacilityGID encoding of the ***UserID*** parameter (PRIVATE_FACILITY_GROUP_USERID) • controls the sending of the ***BusyCause*** parameter in the Network_Busy EDP-Request message (RESTRICT_NETBUSY_BUSYCAUSE) • send CIC value in table TRKGRP to the SCP (SEND_CARRIER_FROM_TRKGRP) • connection type to an IP (STR_CONNECTION_TYPE) • determines the maximum time duration in which the IP must respond to an ISDN FACILITY message with the `cancelIPResource` (TDISC_TIMER) • *Timeout* event timer value (TIMEOUT_TIMER) ▪ Determines if TRTMTCD and COMPCODE fields in CDR to be zapped to Zero for calls terminated by DISCONNECT message (TRTMTCD_COMPCODE_ZAPPED_ZERO.)
Note: Refer to UCS DMS-250 Data Schema Reference Manual for more data schema information.	

Continued

Step 2: Familiarize yourself with related OA&M Data schema (continued)

Table 2-6
NetworkBuilder-related data schema

Table	Description
CAINPARM (continued)	• determines the maximum time duration of the STR-Connection (TSTRC_TIMER)
CAINXDFT	CAIN EXTENSION PARAMETER DEFAULTS. Provides default values for extension parameters. When a relevant extension parameter is missing in an SCP response, NetworkBuilder call processing uses the values defined in table CAINXDFT. The default values for the extension parameters are associated with a call's CAIN group.
CAINSTS	CAIN SERVING TRANSLATION SCHEME. Provides a list of STS values that when matched, prevent the OFFCCODE trigger table lookup from being performed.
CAINPRT	CAIN PRETRANSLATOR. Defines CAINPRT collectibles used by the O_Feature_Requested handler.
CNPREXLA	CAIN PRETRANSLATOR NAME. Maps extension parameter values for the pretranslator which can be returned by the SCP to a valid pretranslator name in table STDPRTCT.
CNPRTNUM	CAIN PRETRANSLATOR NUMBER. Assigns a number to each CAINPRT collectible in table CAINPRT.
INWATNPA	INWATS NPA. Identifies INWATS toll free calls. Assigns NPA codes to receive INWATS toll free processing for call–by–call feature terminations to a PRI PBX.
OFCENG	OFFICE ENGINEERING. Provides office engineering parameters. The following parameters are used by NetworkBuilder: • determines the number of call processing timers allocated for use by the switch (NUMCPWAKE) • stores information received in an incoming IAM message for later use in a call (NUM_ISUP_EXT_BLOCKS) • stores information for use by **`Merge_Call`** processing (NUMPERMEXT)
Routing	
TANDMRTE	TANDEM ROUTING. Provisions routing through tandem switches within the IEC network to reach the required terminating switch.
Note: Refer to *UCS DMS-250 Data Schema Reference Manual* for more data schema information.	

—continued—

Step 2: Familiarize yourself with related OA&M Data schema (continued)

Table 2-6
NetworkBuilder-related data schema (continued)

Table	Description
TERMRTE	TERMINATION ROUTING. Provision to route to a terminating switch directly connected to the current switch.
TERMLRN	TERMINATING LOCATION ROUTING NUMBER. Provides up to 16 terminating Location Routing Numbers (LRNs) assigned to the switch. Supports the UCS DMS-250's function of recognizing when the LRN for the call is that assigned to the UCS DMS-250 and, when such is the case, routing the call based on the contents of the Generic Address Parameter (GAP).
SCP simulator	
CAINCONV	CAIN CONVERSATION. Controls SCP simulator interaction during TCAP conversation with the switch.
CAINKEY	CAIN KEY. Determines a range of possible responses for a given three-part key. The range of possible responses is represented by an option vector of indexes into table CAINMTCH.
CAINMTCH	CAIN MATCHING. Determines possible responses from the SCP simulator
CAINRESP	CAIN RESPONSE. Contains response data to return to the switch. The simulator's encoder takes this data and builds a Transaction Capabilities Application Part (TCAP) message.
CAINREXT	CAIN RESPONSE EXTENSION PARAMETERS. Contains the extension parameters used to build a response message.
CAINUID	CAIN USER IDENTIFICATION. Provides symbolic names for trunk groups and switch identifiers used in the simulator. It is similar to table CLLI in function. The use of symbolic names rather than numbers provides enhanced clarity when datafilling the simulator tables.
IN1RESP	IN/1 RESPONSE. Provides data needed to build a response for IN/1 queries by using the SCP simulator.
Subscription	
CAINGRP	CAIN GROUP. Defines a CAIN group, including the group number and trigger sets.
Note: Refer to *UCS DMS-250 Data Schema Reference Manual* for more data schema information.	

—continued—

Step 2: Familiarize yourself with related OA&M Data schema (continued)

Table 2-6
NetworkBuilder-related data schema (continued)

Table	Description
STDPRTCT	STANDARD PRETRANSLATOR CONTROL. Table STDPRTCT is the first table indexed for digit pretranslation when the incoming or two-way trunk group associated with the call is assigned a standard pretranslator name. Call processing then indexes the appropriate STDPRT subtable. The CAINGRP option identifies the CAIN group for address-based subscription.
ANISCUSP	AUTOMATIC NUMBER IDENTIFICATION SCREENING CUSTOMER PROFILE. Assigns a CAIN group to a particular ANI. CAIN/FlexDial Interaction does not support subscription through table ANISCUSP.
UNIPROF	UNIVERSAL PROFILES. Assigns a CAIN group to a particular profile.
AUTHCODU	AUTHCODE DATABASE. Assigns a CAIN group to a particular authorization code. CAIN/FlexDial Interaction does not support subscription through table AUTHCODU.
AUTHCDU2	AUTHCODE DATABASE 2. Assigns a CAIN group to a particular authorization code. CAIN/FlexDial Interaction does not support subscription through table AUTHCDU2.
AUTHCDU3	AUTHCODE DATABASE 3. Assigns a CAIN group to a particular authorization code. CAIN/FlexDial Interaction does not support subscription through table AUTHCDU3.
AUTHCDU4	AUTHCODE DATABASE 4. Assigns a CAIN group to a particular authorization code. CAIN/FlexDial Interaction does not support subscription through table AUTHCDU4.
AUTHCDU5	AUTHCODE DATABASE 5. Assigns a CAIN group to a particular authorization code. CAIN/FlexDial Interaction does not support subscription through table AUTHCDU5.
CAINPARM	CAIN PARAMETERS. Parameter CAIN_OFFICE_GROUP assigns a CAIN group to an office.
CALLATTR	CALL ATTRIBUTES. Assigns a CAIN group to a PRI call attribute.
TRKGRP	TRUNK GROUP. Assigns a CAIN group to a particular non-AXXESS, non-PRI agent. CAIN/FlexDial Interaction does not support subscription through table TRKGRP.
Note: Refer to *UCS DMS-250 Data Schema Reference Manual* for more data schema information.	
—continued—	

Step 2: Familiarize yourself with related OA&M Data schema (end)

Table 2-6
NetworkBuilder-related data schema (continued)

Table	Description
TRKFEAT	TRUNK GROUP FEATURES. Assigns a CAIN group to a particular AXXESS feature set.
Trigger	
CUSTDP	CUSTOMIZED DIALING PLAN. Defines trigger criteria for *Customized_Dialing_Plan*.
NETBUSY	NETWORK BUSY. Defines trigger criteria for *Network_Busy*.
OCLDBUSY	ORIGINATING CALLED PARTY BUSY. Defines trigger criteria for *O_Called_Party_Busy*.
OFFCCODE	OFFICE CODE. Defines the trigger criteria for *Office_Code*.
OFFHKDEL	OFFHOOK DELAY. Defines trigger criteria for *Offhook_Delay*.
OFFHKIMM	OFF HOOK IMMEDIATE. Defines trigger criteria for *Off_Hook_Immediate*.
OFTRREQ	ORIGINATING FEATURE REQUESTED. Defines trigger criteria for *O_Feature_Requested*.
OIECREO	ORIGINATING IEC REORIGINATION. Defines trigger criteria for *O_IEC_Reorigination*.
ONOANSWR	ORIGINATING NO ANSWER. Defines trigger criteria for *O_No_Answer*.
PRIBCHNL	PRI B-CHANNEL. Defines trigger criteria for *PRI_B-Channel*.
SIOTRK	SHARED INTEROFFICE TRUNK. Defines trigger criteria for *Shared_Interoffice_Trunk*.
SPECDIG	SPECIFIC DIGIT STRING. Defines trigger criteria for *Specific_Digit_String*.
SPECFEAT	SPECIFIC FEATURE CODE. Defines trigger criteria for *Specific_Feature_Code*.
TERMATT	TERMINATION ATTEMPT. Defines trigger criteria for *Termination_Attempt*.
TOLLFREE	TOLL-FREE. Defines trigger criteria for Bellcore's *TR-NWT-000533* toll-free service specifications.

Note: Refer to *UCS DMS-250 Data Schema Reference Manual* for more data schema information.

—end—

Step 2: Familiarize yourself with related OA&M Treatments (end)

The following table lists the treatments related to NetworkBuilder:

Table 2-7
NetworkBuilder-related treatments

Treatment	Description
AIND	AIN DISCONNECT. Applied when the SCP determines a call should be disconnected (for example, **`Disconnect`** or **`Send_To_Resource`** with a ***`DisconnectFlag`*** is received).
AINF	AIN FINAL. Applied when the switch detects fatal application errors or you datafill a trigger action of BLOCK.
MLNP	MISROUTED LOCAL NUMBER PORTABILITY. Applied when the final switch on the network receives an SS7 RELEASE message with a value of 26 from the LEC. This is an indication of a routing failure of the ported number.

Refer to *UCS DMS-250 Data Schema Reference Manual*, tables TMTMAP and CSEMAP for a full list of treatment codes.

Step 2: Familiarize yourself with related OA&M Billing (continued)

CAUTION
Changes may affect site functionality
Changes to the billing system require updates to engineering parameters that may affect site functionality. Any changes to the billing system may affect downstream processing of billing records. Nortel Networks recommends that only experienced personnel make changes to the billing system.

The new billing system that was introduced in the UCS06 software release requires you to provision the billing system that best fits your needs; choose between Internal or Flexible billing records.

Internal billing

The internal billing systems offers the following:

- The billing data collected by the switch is formatted using an optimized formatter.
- You can identify one CDR template that best fits your traffic.

Flexible billing

The Flexible billing system offers the following:

- The billing data collected by the switch is formatted using the flexible billing formatters to populate the billing records.
- Through datafill, you can provision certain call processing criteria that indicates a specific template to be used during CDR population.
- When call processing does not specify a template you can
  - identify a default template for use (controlled by office parameter FCDR_CDR_TEMPLT, table OFCENG)
  - or allow the switch to perform a bestfit analysis on the call data and choose the most appropriate CDR template available.
- You can specify the size of event records, OSRs, and CDRs.

Step 2: Familiarize yourself with related OA&M Billing (continued)

- CDR fields can be populated in either read left-to-right order (format used in software releases previous to UCS06) or in a read right-to-left order (called NORMAL, due to bit ordering within the switch). The word layout is controlled by office parameter FCDR_CDR_WORD_LAYOUT (table OFCENG). Refer to the *UCS DMS-250 Billing Records Application Guide* for more information.
- CDR sizes can be fixed to a particular size (regardless of the template) or set to a variable size (dependent on the size of the template). CDR size is controlled by office parameter FCDR_CDR_SIZE (table OFCENG). Refer to the *UCS DMS-250 Billing Records Application Guide* for more information.
- You can define the call data type for CDRs on answer (controlled by office parameter FCDR_ANSCDR_CDT for CM or FCDR_ANSCDR_SBSCOT for FP, in table OFCVAR).
- By purchasing the UBFR0001 SOC option, you can create your own billing templates. Optimize your billing system by creating new, smaller templates specifically designed for certain call scenarios. Refer to the *UCS DMS-250 Billing Records Application Guide* for more information.

CDR templates

For CDR template information and provisioning refer to the *UCS DMS-250 Billing Records Application Guide*.

Note 1: LNP calls require the UCS08, UCS09, UCS11, UCS12, UCS13, or UCS14 CDR template. Refer to *UCS DMS-250 Local Number Portability Application Guide*.

Note 2: Nortel Networks recommends the use of the Flexible billing system to create CDR templates for CAIN/FlexDial Interactions. Refer to *UCS DMS-250 CAIN/FlexDial Interactions* for more information on CAIN/FlexDial interactions. Template creation requires the UBFR0001 SOC option. Refer to the *UCS DMS-250 Billing Records Application Guide* for more information on the billing systems.

CDR field descriptions

Table 2-8 lists CDR fields populated by NetworkBuilder call processing. Refer to the *UCS DMS-250 Billing Records Application Guide* for more specific CDR field information.

Step 2: Familiarize yourself with related OA&M Billing (continued)

Note: Field population may be affected by LNP or CAIN/FlexDial interactions. Refer to *UCS DMS-250 Local Number Portability Application Guide* and *UCS DMS-250 CAIN/FlexDial Interactions* for more information.

Table 2-8
CDR field descriptions

Field name	Field description
ACCTCD	ACCOUNT CODE. This field is populated with the account code digits collected for the call.
ACG	AUTOMATIC CODE GAPPING. This field indicates a call has encountered an ACG control.
ALTBILL	ALTERNATE BILLING NUMBER. This field contains an alternate billing number (for example, the CallingPartyID or ANI).
AMABAFMD	AUTOMATIC MESSAGE ACCOUNTING BELLCORE AUTOMATIC MESSAGE ACCOUNTING FORMAT MODULE. This field contains the BAF encoded module codes.
AMASIZE	AUTOMATIC MESSAGE ACCOUNTING SIZE. This field contains the size of the AMABAFMD field in bytes.
ANISP	AUTOMATIC NUMBER IDENTIFICATION SPILL. This contains the ANI for MF originating agencies, Charge Number for SS7 originating agencies, or Calling Line ID for PRI originating agencies.
ANSTYPE	ANSWER TYPE. This field contains the type of answer detected.
BILLNUM	BILLING NUMBER. This field identifies the billing number for a call.
CAINCT	CARRIER AIN CALL TYPE. This field contains the CAIN call type received from the SCP response within the callType extension parameter.

—continued—

Step 2: Familiarize yourself with related OA&M Billing (continued)

Table 2-8
CDR field descriptions (continued)

Field name	Field description
CALLDUR	CALL DURATION. This field contains the duration of a call, measured in 10ms ticks. Call duration is measured as the time between called party answer and on-hook by either called party or calling party.
CALLEDNO	CALLED NUMBER. This field is populated with the translated called party digits for a call.
CARRSEL	CARRIER SELECTION. This indicates how the carrier selection was derived.
CIC	CARRIER IDENTIFICATION CODE. This field identifies the long distance carrier for the call.
CLGPTYNO	CALLING PARTY NUMBER. This field identifies the calling party number of an originating SS7 call.
CN1REQ	CAIN FIRST REQUEST. This field contains the first NetworkBuilder trigger/event that sends a TDP/EDP request message to an SCP (Service Control Point).
CN2REQ	CAIN SECOND REQUEST. This field contains the next unique NetworkBuilder trigger/event that sends a TDP/EDP request message to an SCP (Service Control Point).
CN3REQ	CAIN THIRD REQUEST. This field contains the last unique NetworkBuilder trigger/event that sends a TDP/EDP request message to an SCP (Service Control Point).
CN1TREQ	CAIN FIRST REQUEST TOTALS. This field contains the number of times the CN1REQ trigger/event sends a TDP/EDP request message to an SCP (Service Control Point) during the life of a call.

—continued—

Step 2: Familiarize yourself with related OA&M Billing (continued)

Table 2-8
CDR field descriptions (continued)

Field name	Field description
CN2TREQ	CAIN SECOND REQUEST TOTALS. This field contains the number of times the CN2REQ trigger/event sends a TDP/EDP request message to an SCP (Service Control Point) during the life of a call.
CN3TREQ	CAIN THIRD REQUEST TOTALS. This field contains the number of times the CN3REQ trigger/event sends a TDP/EDP request message to an SCP (Service Control Point) during the life of a call.
CNPREDIG	CALLED PARTY PREFIX DIGITS. This field identifies one of the following as the translated called number's prefix digits. 0 = No prefix digits 1 = 0 Prefix 2 = 01 Prefix 3 = 011 Prefix 4 = 1 Prefix 5–7 = Not used
CNTOTREQ	CAIN REQUEST TOTALS. This field contains the total number of times TDP/EDP request messages were sent to an SCP (Service Control Point) during the life of a call.
	—continued—

Step 2: Familiarize yourself with related OA&M Billing (continued)

Table 2-8
CDR field descriptions (continued)

Field name	Field description
COSINDEX	CLASS OF SERVICE INDEX. Identifies the index into table COSUS that was used to perform class of service screening.
DIALEDNO	DIALED NUMBER. This field contains the dialed number or a hotline number for hotline calls.
HEXID	HEXADECIMAL IDENTIFIER. This field identifies the beginning of a Bellcore Automatic Message Accounting Format (BAF) record and indicates whether there are errors in the record that require special handling. This field is used for the CDR to BAF conversion. For the CAIN framework, the value for the AMASetHexABIndicator parameter populates this field.
INFODIG	INFORMATION DIGITS. This field contains the information digits.
LATA	Local Access and Transport Area. This field contains the LATA when the switch sends the LATA.
LNPCHECK	LOCAL NUMBER PORTABILITY CHECK. This is an integer field which indicates the use of LNP functionality in the call.

—continued—

Step 2: Familiarize yourself with related OA&M Billing (continued)

Table 2-8
CDR field descriptions (continued)

Field name	Field description
MLTCOSID	MULTI-CLASS OF SERVICE IDENTIFIER. This field contains the index, within table MULTICOS, used to perform multiple COS screening.
OPART	ORIGINATION PARTITION. This field contains the subscriber's region and can be used to determine if the subscriber is in the home region.
ORIGLRN	ORIGINATING LOCATION ROUTING NUMBER. This field contains an originating LRN.
ORIGPVN	ORIGINATING PRIVATE NUMBER This field contains the customer-defined dialing plan number assigned to the station or the user originating the VPN call.
PINDIGS	PERSONAL IDENTIFICATION NUMBER DIGITS. This field contains the subscriber's PINDIGS.
PORTEDNO	PORTED NUMBER. This contains the called party address from the LNP GAP if an LRN is present in an incoming SS7 IAM, or the CPA from the LNP GAP if the SCP returns an LRN, or the CPA from the LNP SCP if the LNP SCP returns an LRN.
PRESIND	PRESENTATION RESTRICTION INDICATOR. This field indicates if the calling party number can be presented to the called party on SS7 calls.
	—continued—

Step 2: Familiarize yourself with related OA&M Billing (end)

Table 2-8
CDR field descriptions (continued)

Field name	Field description
PRJCODE	PROJECT CODE. This field indicates the project code value in the Carrier AIN TCAP message received from the SCP.
RTELIST	ROUTE LIST. This field indicates the number (index) obtained from the routing table used to route the call.
RTEINDEX	ROUTE INDEX. This field indicates the number (index) obtained from the routing table used to route the call.
RTETAB	ROUTE TABLE. This field identifies the table used to route the call.
SCPBILL	SERVICE CONTROL POINT BILLING. This field is used to correlate SCP and switch billing records.
SLPID	SERVICE LINE PROVIDER IDENTIFICATION. This field indicates the Automatic Message Accounting Service Line Provider ID (AMAslpID) value in the Carrier AIN TCAP message received from the SCP.
TERMPVN	TERMINATING PRIVATE NUMBER. This field contains the customer-defined dialing plan number assigned to the called party.
TERMLRN	TERMINATING LOCATION ROUTING NUMBER. This field contains the 10-digit terminating LRN for the call when it matches one of the LRNs assigned the switch.
TPART	TERMINATION PARTITION. This field contains the termination region.
TRMTCD	TREATMENT CODE. This field identifies the treatment code applied to the call.
—end—	

Step 2: Familiarize yourself with related OA&M Commands (continued)

The NetworkBuilder platform offers several tools for troubleshooting and tracking purposes. Refer to *UCS DMS-250 Commands Reference Manual* for more commands information.

ACGCNTRL

ACGCNTRL as part of VAMP allows you to view ACG controls currently in effect for each application. It can also be used to add and remove controls.

The following commands are accessible through ACGCNTRL:

- LIST
- REMOVE
- RESET
- ADD
- GLOBAL
- HELP
- QUIT

CAINTEST

CAINTEST is the Carrier Advanced Intelligent Network (CAIN) message query test tool. You can create and send test TCAP queries to the SCP. The SCP responds as if an actual call were being made and does not recognize any difference between a CAINTEST-generated message and a call-generated message. SCP queries and responses can be displayed for maintenance and verification purposes.

The following commands are accessible through CAINTEST:

- CLRPARM
- HELP
- LISTPARM
- QUIT
- RESPORD
- SEND
- SETAPPL
- SETFAM
- SETMSG

Step 2: Familiarize yourself with related OA&M Commands (continued)

- SETPARM
- SETQUERY
- SETRESP
- SETTRANS
- SHOWFLDS
- TIMEOUT

Note: Refer to *UCS DMS-250 Local Number Portability Application Guide* for information on utilizing CAINTEST for LNP queries.

SCP simulator

Use the simulator to test NetworkBuilder functionality. You can datafill the simulator to provide NetworkBuilder testing, as well as robustness testing. Through datafill the simulator provides the following SCP services:

- accepts and decodes TCAP messages
- detects and reports protocol errors

  Note: When the error does not require an SCP response, the simulator may generate a CAIN901 log. (CAIN900 series logs are enabled and disabled by parameter CAIN900_LOGS_ENABLED table CAINPARM).

- checks incoming message parameters against the database you have built using tables CAINKEY and CAINMTCH
- determines if additional data is required to process the call
- determines appropriate response messages, including error cases
- encodes response messages in TCAP message format, using tables CAINRESP, IN1RESP, and CAINMTCH
- sends the TCAP message to the switch
- continues to collect data until datafill requirements are met
- validates (using table CAINCONV) data gathered during TCAP conversational digit collection and sends an appropriate message
- supports event processing

Step 2: Familiarize yourself with related OA&M Commands (continued)

Robustness testing

Perform robustness testing by datafilling errors that can occur in communication between the CAIN framework and the SCP. You can datafill (in table CAINRESP) the following for robustness testing:

- nonrestricted messages with normally invalid package and component types
- messages with invalid parameters
- parameters with invalid data

SOC

Nortel uses Software Optionality Control (SOC) to define and deliver software in product computing-module loads (PCL). Nortel categorizes all functionality in a PCL as either base or optional. Base functionality is available for immediate use. Optional functionality is grouped into commercial units called SOC options, which can be purchased by operating companies. SOC options correspond to functional groups and functions and are controlled by Nortel-supplied passwords.

SOC is the tool for managing options in a PCL. These options reside in the software. When an operating company purchases an option, SOC allows the company to monitor and control its use. You can order, activate, and use these options without a software reload or restart.

The following commands are accessible through SOC:

- ASSIGN
- DBAUDIT
- DELETE
- REMOVE
- RESET
- RESET_AUDIT
- SELECT
- VALIDATE

TRAVER

The TRAVER (translation verification) tool simulates a call from a user specified originating trunk to a user-specified address. TRAVER examines and displays translation and routing data for a single call leg.

Step 2: Familiarize yourself with related OA&M Commands (end)

TRAVER performs the following:

- verifies the translation tables
- aids in debugging and analyzing translation and routing datafill.
- helps determine reasons for unexpected results and changes required to achieve the expected results.

TRAVER is capable of displaying the following:

- tables used to translate and route a call
- treatment
- CAIN subscription method and group
- tuple (from the appropriate trigger table) where trigger criteria was met
- limited messaging parameters

VPTRACE

VPTRACE allows you to enable or disable the tracing of VAMP messages (logged in VAMP 90x logs). If called with no parameter, the command displays the current status of tracing (enabled or disabled).

CAINSCPT

When the CAINSCPT tool is activated, the user is able to perform the following tasks:

- idle the specified trid(s)
- mark the specified trid(s) in use
- display information associated with the given trids
- detail information associated with the simulator's timeout wheel

CAINSCPT supports the SCP Simulator.

Step 3: Be familiar with agent setup messaging

To better understand NetworkBuilder call processing, you need to be familiar with the setup messaging for NetworkBuilder-supported originating agencies.

Note: Agent setup messaging is handled differently for LNP. Refer to *UCS DMS-250 Local Number Portability Application Guide* for more information.

Note: Agent setup messaging is handled differently for AXXESS agents. Refer to *UCS DMS-250 CAIN/FlexDial Interactions* for more information.

SS7 messaging

The switch recognizes CCS7 messages from:

- out-of-network agents
- in-network communications

Out-of-network agents

NetworkBuilder supports FGD, Inter-IMT, Global-IMT, and AXXESS originations using CCS7 messaging.

NetworkBuilder uses the following ISUP parameters from the received IAM (Initial Address Message):

- Called Party Number (CPA) – contains the number to use in translations and routing; also identifies the nature of address and numbering plan.
- Calling Party Number (CPN) – identifies the calling party and the associated nature of address and numbering plan.
- Carrier Identification Parameter (CIP) – identifies the carrier for the call.
- Carrier Selection (CSI) – contains carrier selection information.
- Charge Number (CGN) – contains the ANI, nature of address, and numbering plan.
- Forward Call Indicator (FCI) – used to indicate when an LNP check has been completed.
- Generic Address (GAP) – used to relay different types of addresses; identifies the associated nature of address and numbering plan.
- Generic Digits (GD) – used to relay different digit types; identifies the digit type and the digits.
- Jurisdiction Information Parameter (JIP) – contains the originating switch's location routing number (LRN).

Step 3: Be familiar with agent setup messaging (continued)

- Network Information (NETINFO) – used to relay different network identities; identifies the external network ID, network customer group ID and network class of service.
- Originating Line Information (OLI) – contains the information digits for the subscriber.
- Transit Network Selector (TNS) – identifies the carrier for the call.

Note: Additional parameters may be used with AXXESS agents. Refer to *UCS DMS-250 CAIN/FlexDial Interactions* for more information.

Generic digits parameter

In order to provide direct termination routing with NetworkBuilder between switches, NetworkBuilder adds the GD to the IAM. NetworkBuilder calls use four to six octets to deliver the terminating switch ID and trunk group.

Note: Other features may deliver the GD; the GD may be larger than four octets depending on the content.

The format of the GD parameter is controlled by the INCREASED_SWITCH_ID parameter and the SWID option in table TRKGRP.

When the INCREASED_SWITCH_ID is set to N or the SWID option in table TRKGRP is present, the format is:

- Octet 1
  - — Bits 8, 7, and 6 – encoding scheme (always 000 for NetworkBuilder calls, indicating binary encoded digits)
  - — Bits 5 through 1 – digit type (always 01001 for NetworkBuilder calls, indicating the SWID and terminating trunk group number are encoded)
- Octet 2
  - — Bit 8 – Calling party identifier (CPI) provided boolean
  - — Bits 7 through 1 – SWID
- Octet 3
  - — Bits 8 through 1 – 8 least significant binary digits of the terminating trunk group number

  Note: Octet 4 is read before octet 3.
- Octet 4

Step 3: Be familiar with agent setup messaging (continued)

— Bits 8 through 1 – 8 most significant binary digits of the terminating trunk group number, when required

The following figure shows the GD encoding format and an example of the GD for the previous figure. (The SWID is 15 and the terminating trunk group number is 101. The trunk group number, 101, only requires one octet.)

Figure 2-5
Generic Digits (GD) parameter encoding format

GD parameter format for routing information

MSB LSB

8	7	6	5	4	3	2	1	OCTET
Encoding scheme			Digit type					1
C	Terminating SWID							2
Terminating trunk group number								3
Terminating trunk group number								4

C= CPI provided booleam.

GD parameter format example (see previous figure)

MSB LSB

8	7	6	5	4	3	2	1	OCTET
0	0	0	0	1	0	0	1	1
0	0	0	0	1	1	1	1	2
0	1	1	0	0	1	0	1	3
0	0	0	0	0	0	0	0	4

When the INCREASED_SWITCH_ID is set to Y and the SWID option in table TRKGRP is not present, the format is:

- Octet 1
  — Bits 8, 7, and 6 – encoding scheme (always 000 for NetworkBuilder calls, indicating binary encoded digits)

Step 3: Be familiar with agent setup messaging (continued)

— Bits 5 through 1 – digit type (always 01001 for NetworkBuilder calls, indicating the SWID and terminating trunk group number are encoded)

- Octet 2
  - — Bit 8 – Calling party identifier (CPI) provided boolean
  - — Bits 7 through 1 – spare (reserved for future use)
- Octet 3
  - — Bits 8 through 1 – 8 least significant binary digits of the terminating trunk group number

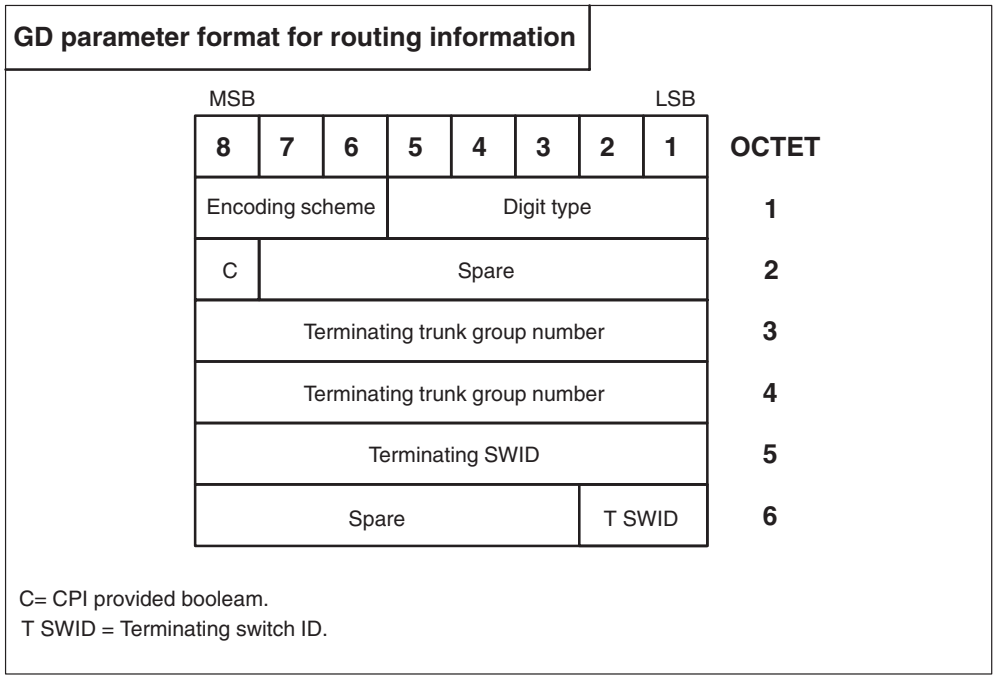
  Note: Octet 4 is read before octet 3.

- Octet 4
  - — Bits 8 through 1 – 8 most significant binary digits of the terminating trunk group number, when required
- Octet 5 and Octet 6 bits 2 through 1
  - — Bits 8 through 1 of Octet 5 – 8 least significant binary digits of the terminating SWID
  - — Bits 2 through 1 of Octet 6 – 2 most significant binary digits of the terminating SWID
- Octet 6 bits 8 through 3
  - — Bits 8 through 3 – spare (reserved for future use)

The following figure shows the GD encoding format when with the increased switch ID.

Step 3: Be familiar with agent setup messaging (continued)

Figure 2-6
Generic Digits (GD) parameter encoding format with increased switch ID



In-network communications

NetworkBuilder interacts with the following messages:

- ACM (address complete message) – sent in a backward direction when called party information is complete
- ANM (answer message)
  - — backward direction to indicate call has been answered
- REL (release)
  - — Either direction to notify next switch to release circuit involved in the call
  - — Cause Indicator (CI) – contains reason for call release
- RES (resume)
  - — backward direction from LEC to cancel a timed disconnect (follows SUS)
- SUS (suspend)
  - — backward direction from LEC to initiate a timed disconnect

Step 3: Be familiar with agent setup messaging (end)

PRI SETUP

NetworkBuilder interacts with PRI SETUP message. The following information elements are processed by NetworkBuilder software for the originating call model:

- Bearer Capability
- Calling Party Number
- Called Party Number

The following information element is processed by NetworkBuilder software for the terminating call model:

- Display Information

Originator outpulsing

The following data may be outpulsed to the switch:

- called party number
- ANI
- information digits
- CIC
- OPART/TPART
- Display information

Step 4: Define CCS7 connectivity

Terminology

The following terms are related to CCS7 connectivity:

- OSI-related
  - — Common Channel Signaling No. 7 (CCS7) – A digital, message-based network signaling standard defined by the CCITT which separates call signaling information from voice channels so that interoffice signaling is exchanged over a separate signaling link.
  - — Integrated Services Digital Network (ISDN) – A network that provides end-to-end digital connectivity using CCS7 to support a wide range of voice and data services to the end-user.
  - — ISDN User Part (ISUP) – The signaling portion of SS7 required to provide voice and non-voice service in ISDN networks.
  - — Message Transfer Part (MTP) – Serves as a transport system providing transfer of signaling messages between nodes in the network. Concerned about the availability of routing to a node identified by a point code (PC); point code to point code messaging; link and point code status management; not concerned with contents of the message.
  - — Signaling System No. 7 (SS7) – An ANSI standard protocol used by networks.
  - — Signaling Connection Control Part (SCCP) – Software protocol acting as an interface between OSI layers. Provides additional functions to the MTP level to enhance the routing capabilities. Concerned with end-to-end application messaging; global title addressing; subsystem status management; not concerned with contents of message. Used for non-trunk related services; provides the global title translation function.
  - — Transaction Capabilities Application Part (TCAP) – A service that provides a common protocol for remote operations across the Common Channel Signaling No. 7 (CCS7) network. The protocol consists of message formatting, content rules, and exchange procedures. TCAP provides the ability for the service switching point (SSP) to communicate with the service control point (SCP). TCAP is also used by the integrated services digital network (ISDN) layer facility message to transport service information for transaction signaling; not associated with an active call over primary rate interface D (data) channels.
- global title – The application's address (translated to a PC + SSN).
- point code (PC) – Address of a node in the CCS7 network.

- subsystem number (SSN) – Address of a subsystem (application) at a node in the CCS7 network (for example, CAINTEST).
- signaling link – The communication channel between two nodes.
- link set – Two or more redundant links between two nodes.
- route – Signaling path from one node to another (may go through multiple nodes).
- route set – Set of all routes from one node to another.

CCS7 tables

This section describes the tables used to define CCS7 connectivity.

C7GTT

Table C7GTT (Common Channel Signaling No. 7 Global Title Table) maps translation types to CCS7 network addresses. Each global title corresponds to the address of an application. The CCS7 network address resulting from global title translations can correspond to an error, a point code, a subsystem number, a PC and an SSN, or a PC and a new GT. If SSN only is supplied, the local PC is assumed.

C7GTTYPE

Table C7GTTYPE (CCS7 Global Title Type) lists the GT supported by this node. Maps a CCS7-defined translation to a network-defined global title translation type.

C7LINK

Table C7LINK (CCS7 Link) makes the association between the physical equipment of the link and the logical view of the link as a member of a linkset.

C7LKSET

Table C7LKSET (CCS7 Link Set) defines the characteristics of a linkset. A linkset is a set of links used as a group. Each link carries traffic between the origination point code and a destination point code. The table also defines attributes that are common to all links in the link set. The links are defined in table C7LINK.

C7LOCSSN

Table C7LOCSSN (CCS7 Local Subsystem) identifies the subsystems residing on the switch by name and a subsystem number (SSN).

Step 4: Define CCS7 connectivity (continued)

C7NETSSN

Table C7NETSSN (CCS7 Network Subsystem Number) lists the remote point codes and remote subsystems that are of importance to the current node. Identifies a PC name (from table C7RTESET) and lists remote subsystems at this PC.

C7NETWRK

Table C7NETWRK (CCS7 Network) describes the signaling networks in use in a switching office.

C7RPLSSN

Table C7RPLSSN (CCS7 Replicate Subsystem) provides the set of remote subsystem replicate pairs. It has a one part key, the subsystem name. For each subsystem a list of PC pairs at which the replicated subsystems reside must be given.

C7RSSCRN

Table C7RSSCRN (CCS7 Remote Subsystem Concerned Node) provides a list of concerned nodes for a remote subsystem point code combination. The table has a two part key. The first part is the PC and the second part is the subsystem name. The PC and subsystem combination must be datafilled in table C7NETSSN.

C7RTESET

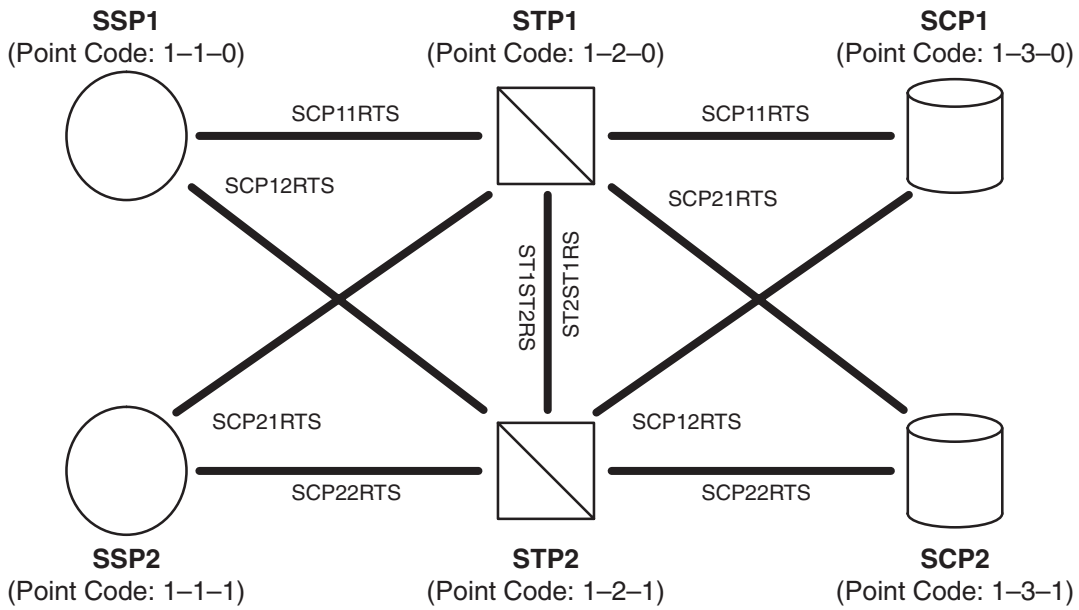
Table C7RTESET (CCS7 Route Set) logically associates linksets that are used as possible routes for each signaling point in the network. An office point code identifies a signaling point within any network. Each office point code must have a routeset. The information in this table records contains the routes and linksets that can carry the signaling information to the destination signaling point. This table is also used for alternate routing decisions.

Datafill example

This section shows the datafill for an example network containing two switches, a mated STP pair, and a mated SCP pair (see figure 2-7).

Step 4: Define CCS7 connectivity (continued)

Figure 2-7
Example network



Assumptions:

- Each routeset has one linkset with two links.
- The CAIN subsystem uses SSN 250 across the network.
- All nodes are on the national network.
- One GT type is used to route queries to the SCPs, while the other is used for the in-switch simulator.

Datafill for SSP1

The SSP requires datafill to recognize links to the STPs and to route messages to them for global title translation. The second GT type will be used to route to the internal simulator subsystem.

1 Table C7NETWRK

```
NETNAME NODETYPE PTCODE NI SLSROT TFR MCS CLUSTERS RCTEST MTPRES
-----------------------------------------------------------------
SSPNTWK1 SSP ANSI7 1 1 0 NATL Y Y 3 N Y Y
```

Step 4: Define CCS7 connectivity (continued)

2 Table C7LKSET

```
LINKSET LSTYPE NETNAME FEPC FECLLI SIGLKTST RSTEST INGTEST Q704 CNGSTN
NUMFLAGS MTPRES CHNGSLS
-----------------------------------------------------------------------
STP11LKS ALINK SSPNTWK1 ANSI7 1 2 0 SSP11LKS Y N Y 0 0 1 Y N
STP12LKS ALINK SSPNTWK1 ANSI7 1 2 1 SSP12LKS Y N Y 0 0 1 Y N
```

3 Table C7LINK

```
LINKNAME LINKDATA CLASDATA Q707 LINKOPT
---------------------------------------
STP11LKS 0 LIUBASIC LIU7 101 MTP2 0 0 $
STP11LKS 1 LIUBASIC LIU7 201 MTP2 0 0 $
STP12LKS 0 LIUBASIC LIU7 107 MTP2 0 0 $
STP12LKS 1 LIUBASIC LIU7 207 MTP2 0 0 $
```

4 Table C7RTESET

```
ROUTESET NETNAME TFPBCAST DPC ROUTES
------------------------------------
STP11RTS SSPNTWK1 Y ANSI7 1 2 0 (STP11LKS 0) (STP11LKS 1) $
STP12RTS SSPNTWK1 Y ANSI7 1 2 1 (STP12LKS 0) (STP12LKS 1) $
```

5 Table C7NETSSN

```
PCNAME SSNAMES
------------------------------------------
STP11RTS $
STP12RTS $
```

6 Table C7LOCSSN

```
SSNAME SSNUMBER MININST REPLINFO TFMI PCNAMES
---------------------------------------------
CAIN     250 1 N N STP11RTS STP12RTS $
CAINTEST 251 1 N N $
```

7 Table C7GTTYPE

```
GTTNAME GTTYPE GTTID
--------------------------------
CAINCLID ANSI7 80 (CAIN_CLID_GT) $
CAINADDR ANSI7 81 (CAIN_ADDR_GT) $
CAINFEAT ANSI7 82 (CAIN_FEAT_GT) $
E800BELLCORE ANSI7 83 (E800BELLCORE) S
```

8 Table C7GTT

```
GTTKEY GTTRSLT
-----------------------------------
CAINCLID 0 9 PCONLY  (STP11RTS 0) (STP12RTS 1) $ GT
CAINADDR 0 9 SSNONLY (CAINTEST) $
CAINFEAT 0 9 SSNONLY (CAINTEST) $
E800BELLCORE 0 9 SSNONLY (CAINTEST) S
```

Datafill for STP1

The STP requires datafill for links to both SSPs, both SCPs, and the other STP element in the mated pair for reliability.

ATTENTION

This is sample datafill for a Nortel STP.

1 Table C7NETWRK

```
NETNAME NODETYPE PTCODE NI SLSROT TFR MCS CLUSTERS RCTEST MTPRES
-----------------------------------------------------------------
STPNTWK1 STP ANSI7 1 2 0 NATL Y Y 3 N Y Y
```

2 Table C7LKSET

```
LINKSET LSTYPE NETNAME FEPC FECLLI SIGLKTST RSTEST INGTEST Q704 CNGSTN
NUMFLAGS MTPRES CHNGSLS
-----------------------------------------------------------------------
SSP11LKS ALINK STPNTWK1 ANSI7 1 1 0 STP11LKS Y N Y 0 0 1 Y N
SSP21LKS ALINK STPNTWK1 ANSI7 1 1 1 STP21LKS Y N Y 0 0 1 Y N
SCP11LKS ALINK STPNTWK1 ANSI7 1 3 0 STP11LKS Y N Y 0 0 1 Y N
SCP21LKS ALINK STPNTWK1 ANSI7 1 3 1 STP21LKS Y N Y 0 0 1 Y N
ST1ST2LS CLINK STPNTWK1 ANSI7 1 2 1 ST2ST1LS Y N Y 0 0 1 Y N
```

3 Table C7LINK

```
LINKNAME LINKDATA CLASDATA Q707 LINKOPT
---------------------------------------
SSP11LKS 0 LIUBASIC LIU7 101 MTP2 0 0 $
SSP11LKS 1 LIUBASIC LIU7 201 MTP2 0 0 $
SSP21LKS 0 LIUBASIC LIU7 107 MTP2 0 0 $
SSP21LKS 1 LIUBASIC LIU7 207 MTP2 0 0 $
SCP11LKS 0 LIUBASIC LIU7 103 MTP2 0 0 $
SCP11LKS 1 LIUBASIC LIU7 203 MTP2 0 0 $
SCP21LKS 0 LIUBASIC LIU7 109 MTP2 0 0 $
SCP21LKS 1 LIUBASIC LIU7 209 MTP2 0 0 $
ST1ST2LS 0 LIUBASIC LIU7 105 MTP2 0 0 $
ST1ST2LS 1 LIUBASIC LIU7 205 MTP2 0 0 $
```

4 Table C7RTESET

Step 4: Define CCS7 connectivity (continued)

```
ROUTESET NETNAME TFPBCAST DPC ROUTES
------------------------------------
SSP11RTS STPNTWK1 Y ANSI7 1 1 0 (SSP11LKS 0) (SSP11LKS 1) $
SSP21RTS STPNTWK1 Y ANSI7 1 1 1 (SSP21LKS 0) (SSP21LKS 1) $
SCP11RTS STPNTWK1 Y ANSI7 1 3 0 (SCP11LKS 0) (SCP11LKS 1) $
SCP21RTS STPNTWK1 Y ANSI7 1 3 1 (SCP21LKS 0) (SCP21LKS 1) $
ST1ST2RS STPNTWK1 Y ANSI7 1 2 1 (ST1ST2LS 0) (ST1ST2LS 1) $
```

5 In addition to global title routing data, the STP requires datafill for the subsystems on the SSPs and SCPs, so that SCCP maintenance messages can be passed back and forth. In this example, the SCPs are in a load-share arrangement.

6 Table C7NETSSN

```
PCNAME SSNAMES
------------------------------------------
SSP11RTS (CAIN 250) $
SSP21RTS (CAIN 250) $
SCP11RTS (CAIN 250) $
SCP21RTS (CAIN 250) $
ST1ST2RS $
```

7 Table C7RPLSSN

```
SSNAME REPLIST
--------------
CAIN (SCP11RTS SCP21RTS N) $
```

8 Table C7RSSCRN

```
PCSSN PCNAMES
-------------
SSP11RTS CAIN (ST1ST2RS) $
SSP21RTS CAIN (ST1ST2RS) $
SCP11RTS CAIN (ST1ST2RS) $
SCP21RTS CAIN (ST1ST2RS) $
```

9 Table C7GTTYPE

```
GTTNAME GTTYPE GTTID
------------------------------------
CAINCLID ANSI7 80 $
```

10 Table C7GTT

```
GTTKEY GTTRSLT
--------------
CAINCLID 0 9 PCSSN  (SCP11RTS CAIN 0) (SCP21RTS CAIN 0) $ SSN
```

Step 4: Define CCS7 connectivity (continued)

Datafill for SCP1

The SCP requires datafill only for the links to the STPs, as no global title routing is performed. This datafill is simulated, as there is no DMS-based SCP available for NetworkBuilder.

> **ATTENTION**
> This datafill is simulated using switch tables to show how an SCP may appear.

1 Table C7NETWRK

```
NETNAME NODETYPE PTCODE NI SLSROT TFR MCS CLUSTERS RCTEST MTPRES
-------------------------------------------------------------------
SCPNTWK1 SCP ANSI7 1 3 0 NATL Y Y 3 N Y Y
```

2 Table C7LKSET

```
LINKSET LSTYPE NETNAME FEPC FECLLI SIGLKTST RSTEST INGTEST Q704 CNGSTN
NUMFLAGS MTPRES CHNGSLS
-------------------------------------------------------------------
STP11LKS ALINK SCPNTWK1 ANSI7 1 2 0 SCP11LKS Y N Y 0 0 1 Y N
STP12LKS ALINK SCPNTWK1 ANSI7 1 2 1 SCP12LKS Y N Y 0 0 1 Y N
```

3 Table C7LINK

```
LINKNAME LINKDATA CLASDATA Q707 LINKOPT
---------------------------------------
STP11LKS 0 LIUBASIC LIU7 101 MTP2 0 0 $
STP11LKS 1 LIUBASIC LIU7 201 MTP2 0 0 $
STP12LKS 0 LIUBASIC LIU7 107 MTP2 0 0 $
STP12LKS 1 LIUBASIC LIU7 207 MTP2 0 0 $
```

4 Table C7RTESET

```
ROUTESET NETNAME TFPBCAST DPC ROUTES
------------------------------------
STP11RTS SCPNTWK1 Y ANSI7 1 2 0 (STP11LKS 0) (STP11LKS 1) $
STP12RTS SCPNTWK1 Y ANSI7 1 2 1 (STP12LKS 0) (STP12LKS 1) $
```

5 Table C7NETSSN

```
PCNAME SSNAMES
---------------------------------------------
STP11RTS $
STP12RTS $
```

6 Table C7LOCSSN

Step 4: Define CCS7 connectivity (continued)

```
SSNAME SSNUMBER MININST REPLINFO TFMI PCNAMES
-------------------------------------------
CAIN     250 1 N N STP11RTS STP12RTS $
```

Datafill for SSP2

The SSP requires datafill to recognize links to the STPs and to route messages to them for global title translation. The second GT type will be used to route to the internal simulator subsystem.

1 Table C7NETWRK

```
NETNAME NODETYPE PTCODE NI SLSROT TFR MCS CLUSTERS RCTEST MTPRES
-------------------------------------------------------------
SSPNTWK2 SSP ANSI7 1 1 1 NATL Y Y 3 N Y Y
```

2 Table C7LKSET

```
LINKSET LSTYPE NETNAME FEPC FECLLI SIGLKTST RSTEST INGTEST Q704 CNGSTN
NUMFLAGS MTPRES CHNGSLS
-----------------------------------------------------------------
STP21LKS ALINK SSPNTWK2 ANSI7 1 2 0 SSP21LKS Y N Y 0 0 1 Y N
STP22LKS ALINK SSPNTWK2 ANSI7 1 2 1 SSP22LKS Y N Y 0 0 1 Y N
```

3 Table C7LINK

```
LINKNAME LINKDATA CLASDATA Q707 LINKOPT
--------------------------------------
STP21LKS 0 LIUBASIC LIU7 101 MTP2 0 0 $
STP21LKS 1 LIUBASIC LIU7 201 MTP2 0 0 $
STP22LKS 0 LIUBASIC LIU7 107 MTP2 0 0 $
STP22LKS 1 LIUBASIC LIU7 207 MTP2 0 0 $
```

4 Table C7RTESET

```
ROUTESET NETNAME TFPBCAST DPC ROUTES
-----------------------------------
STP21RTS SSPNTWK2 Y ANSI7 1 2 0 (STP21LKS 0) (STP21LKS 1) $
STP22RTS SSPNTWK2 Y ANSI7 1 2 1 (STP22LKS 0) (STP22LKS 1) $
```

5 Table C7NETSSN

```
PCNAME SSNAMES
-----------------------------------------
STP21RTS $
STP22RTS $
```

6 Table C7LOCSSN

```
SSNAME SSNUMBER MININST REPLINFO TFMI PCNAMES
-------------------------------------------
CAIN     250 1 N N STP21RTS STP22RTS $
CAINTEST 251 1 N N $
```

7 Table C7GTTYPE

```
GTTNAME GTTYPE GTTID
-----------------------------------
CAINCLID ANSI7 80 (CAIN_CLID_GT) $
CAINADDR ANSI7 81 (CAIN_ADDR_GT) $
CAINFEAT ANSI7 82 (CAIN_FEAT_GT) $
E800BELLCORE ANSI7 83 (E800BELLCORE) $
```

8 Table C7GTT

```
GTTKEY GTTRSLT
---------------
CAINCLID 0 9 PCONLY  (STP21RTS 1) (STP22RTS 0) $ GT
CAINADDR 0 9 SSNONLY (CAINTEST) $
CAINFEAT 0 9 SSNONLY (CAINTEST) $
E800BELLCORE 0 9 SSNONLY (CAINTEST) $
```

Datafill for STP2

The STP requires datafill for links to both SSPs, both SCPs, and the other STP element in the mated pair for reliability.

ATTENTION

This is sample datafill for a Nortel STP.

1 Table C7NETWRK

```
NETNAME NODETYPE PTCODE NI SLSROT TFR MCS CLUSTERS RCTEST MTPRES
-----------------------------------------------------------------
STPNTWK2 STP ANSI7 1 2 1 NATL Y Y 3 N Y Y
```

2 Table C7LKSET

```
LINKSET LSTYPE NETNAME FEPC FECLLI SIGLKTST RSTEST INGTEST Q704 CNGSTN
NUMFLAGS MTPRES CHNGSLS
-----------------------------------------------------------------------
SSP12LKS ALINK STPNTWK1 ANSI7 1 1 0 STP12LKS Y N Y 0 0 1 Y N
SSP22LKS ALINK STPNTWK1 ANSI7 1 1 1 STP22LKS Y N Y 0 0 1 Y N
SCP12LKS ALINK STPNTWK1 ANSI7 1 3 0 STP12LKS Y N Y 0 0 1 Y N
SCP22LKS ALINK STPNTWK1 ANSI7 1 3 1 STP22LKS Y N Y 0 0 1 Y N
ST2ST1LS CLINK STPNTWK1 ANSI7 1 2 0 ST1ST2LS Y N Y 0 0 1 Y N
```

3 Table C7LINK

Step 4: Define CCS7 connectivity (continued)

```
LINKNAME LINKDATA CLASDATA Q707 LINKOPT
---------------------------------------
SSP12LKS 0 LIUBASIC LIU7 101 MTP2 0 0 $
SSP12LKS 1 LIUBASIC LIU7 201 MTP2 0 0 $
SSP22LKS 0 LIUBASIC LIU7 107 MTP2 0 0 $
SSP22LKS 1 LIUBASIC LIU7 207 MTP2 0 0 $
SCP12LKS 0 LIUBASIC LIU7 103 MTP2 0 0 $
SCP12LKS 1 LIUBASIC LIU7 203 MTP2 0 0 $
SCP22LKS 0 LIUBASIC LIU7 109 MTP2 0 0 $
SCP22LKS 1 LIUBASIC LIU7 209 MTP2 0 0 $
ST2ST1LS 0 LIUBASIC LIU7 105 MTP2 0 0 $
ST2ST1LS 1 LIUBASIC LIU7 205 MTP2 0 0 $
```

4 Table C7RTESET

```
ROUTESET NETNAME TFPBCAST DPC ROUTES
------------------------------------
SSP12RTS STPNTWK2 Y ANSI7 1 1 0 (SSP11LKS 0) (SSP11LKS 1) $
SSP22RTS STPNTWK2 Y ANSI7 1 1 1 (SSP21LKS 0) (SSP21LKS 1) $
SCP12RTS STPNTWK2 Y ANSI7 1 3 0 (SCP11LKS 0) (SCP11LKS 1) $
SCP22RTS STPNTWK2 Y ANSI7 1 3 1 (SCP21LKS 0) (SCP21LKS 1) $
ST2ST1RS STPNTWK2 Y ANSI7 1 2 0 (ST1ST2LS 0) (ST1ST2LS 1) $
```

5 In addition to global title routing data, the STP requires datafill for the subsystems on the SSPs and SCPs, so that SCCP maintenance messages can be passed back and forth. In this example, the SCPs are in a load-share arrangement.

6 Table C7NETSSN

```
PCNAME SSNAMES
------------------------------------------
SSP12RTS (CAIN 250) $
SSP22RTS (CAIN 250) $
SCP12RTS (CAIN 250) $
SCP22RTS (CAIN 250) $
ST2ST1RS $
```

7 Table C7RPLSSN

```
SSNAME REPLIST
--------------
CAIN (SCP12RTS SCP22RTS N) $
```

8 Table C7RSSCRN

```
PCSSN PCNAMES
-------------
SSP12RTS CAIN (ST2ST1RS) $
SSP22RTS CAIN (ST2ST1RS) $
SCP12RTS CAIN (ST2ST1RS) $
SCP22RTS CAIN (ST2ST1RS) $
```

9 Table C7GTTYPE

```
GTTNAME GTTYPE GTTID
-----------------------------------
CAINCLID ANSI7 80 $
```

10 Table C7GTT

```
GTTKEY GTTRSLT
---------------
CAINCLID 0 9 PCSSN  (SCP12RTS CAIN 0) (SCP22RTS CAIN 0) $ SSN
```

Datafill for SCP2

The SCP requires datafill only for the links to the STPs, as no global title routing is performed.

ATTENTION

This datafill is simulated using switch tables to show how an SCP may appear.

1 Table C7NETWRK

```
NETNAME NODETYPE PTCODE NI SLSROT TFR MCS CLUSTERS RCTEST MTPRES
-------------------------------------------------------------------
SCPNTWK2 SCP ANSI7 1 3 0 NATL Y Y 3 N Y Y
```

2 Table C7LKSET

```
LINKSET LSTYPE NETNAME FEPC FECLLI SIGLKTST RSTEST INGTEST Q704 CNGSTN
NUMFLAGS MTPRES CHNGSLS
-------------------------------------------------------------------------
STP21LKS ALINK SCPNTWK2 ANSI7 1 2 0 SCP21LKS Y N Y 0 0 1 Y N
STP22LKS ALINK SCPNTWK2 ANSI7 1 2 1 SCP22LKS Y N Y 0 0 1 Y N
```

3 Table C7LINK

```
LINKNAME LINKDATA CLASDATA Q707 LINKOPT
---------------------------------------
STP21LKS 0 LIUBASIC LIU7 101 MTP2 0 0 $
STP21LKS 1 LIUBASIC LIU7 201 MTP2 0 0 $
STP22LKS 0 LIUBASIC LIU7 107 MTP2 0 0 $
STP22LKS 1 LIUBASIC LIU7 207 MTP2 0 0 $
```

4 Table C7RTESET

```
ROUTESET NETNAME TFPBCAST DPC ROUTES
-------------------------------------
STP21RTS SCPNTWK2 Y ANSI7 1 2 0 (STP21LKS 0) (STP21LKS 1) $
STP22RTS SCPNTWK2 Y ANSI7 1 2 1 (STP22LKS 0) (STP22LKS 1) $
```

5 Table C7NETSSN

Step 4: Define CCS7 connectivity (continued)

```
PCNAME SSNAMES
------------------------------------------
STP21RTS $
STP22RTS $
```

6 Table C7LOCSSN

```
SSNAME SSNUMBER MININST REPLINFO TFMI PCNAMES
------------------------------------------------
CAIN     250 1 N N STP21RTS STP22RTS $
```

Initializing the network

1 Datafill all tables in the order shown on each node.

2 Bring up the LPP (LIM and LIU7s) on each node.

3 Bring up the links/linksets. Do this by aligning one linkset at a time from both ends at once.

4 Bring up the routesets; again, do it one routeset at a time from both ends at once.

5 Bring up the local subsystems on the SSP and SCP nodes.

6 Bring up the remote subsystems on the STPs for each routeset to an SSP/SCP.

7 Initiate manual test queries from each SSP.

Directly connecting the SSP and SCP

1 In general, the SSP datafill will look more like the STP datafill. However, an SSP can only route queries generated on that SSP, unless it is a combined SSP/STP using the INODE product.

2 Datafill one routeset and linkset to each SCP, with multiple links in each linkset for traffic capacity and redundancy. The linksets will be FLINKs in this case.

3 The C7NETSSN table will have actual subsystem names and numbers for the SCP subsystems.

4 Table C7RPLSSN must be datafilled if a mated pair of SCPs will be used with global title routing to both SCPs (either in loadsharing mode, or in a primary/backup configuration).

5 The C7GTT table datafill will use PCSSN routes with an SSN result; each route choice will contain the subsystem name used in table C7NETSSN to represent the SCP's subsystem.

6 After bringing up the local SSP subsystem, bring up the remote SCP subsystems on the routesets to the SCPs.

Step 4: Define CCS7 connectivity
ACG_OVERFLOW_GT (end)

The ACG_OVERFLOW_GT parameter (table CAINPARM) defines the global title type value used to identify the destination SCP when the ACGOVFLGT option is not datafilled in the trigger table. The following global title types are supported: CAIN_CLID_GT, CAIN_ADDR_GT, and CAIN_FEAT_GT. The default is NIL.

Define the ACG_OVERFLOW_GT parameter

At the CI prompt

7 Enter table CAINPARM

8 Replace the parameter by typing:

>REP ACG_OVERFLOW_GT parmval
where

parmval the global title value (CAIN_CLID, CAIN_ADDR, CAIN_FEAT) or wildcard (NIL)

Sample entry: **>REP ACG_OVERFLOW_GT CAIN_ADDR**

The ACG_OVERFLOW_GT parameter is defined.

Step 4: Define CCS7 connectivity CAIN_DEFAULT_GT (end)

The CAIN_DEFAULT_GT parameter (table CAINPARM) defines the global title type value used to identify the destination SCP when the GT option is not datafilled in the trigger table. The following global title types are supported: CAIN_CLID_GT, CAIN_ADDR_GT, and CAIN_FEAT_GT. The default is CAIN_CLID_GT.

Define the CAIN_DEFAULT_GT parameter

At the CI prompt

1 Enter table CAINPARM.

2 Replace the parameter by typing:

>REP CAIN_DEFAULT_GT parmval
where

parmval the global title value (CAIN_CLID, CAIN_ADDR, CAIN_FEAT)

Sample entry: **>REP CAIN_DEFAULT_GT CAIN_ADDR**

The CAIN_DEFAULT_GT parameter is defined.

Step 4: Define CCS7 connectivity CAIN_DEFAULT_OVERFLOW_GT (end)

The CAIN_DEFAULT_OVERFLOW_GT parameter (table CAINPARM) specifies the default SCP (global title type value) to use for overflow queries. The default is NIL.

Define the CAIN_DEFAULT_OVERFLOW_GT parameter

At the CI prompt

1 Enter table CAINPARM.

2 Replace the parameter by typing:

>REP CAIN_DEFAULT_OVERFLOW_GT parmval
where

parmval the global title value (CAIN_CLID, CAIN_ADDR, CAIN_FEAT) or wildcard (NIL)

Note: When CAIN_DEFAULT_OVERFLOW_GT is set to NIL, then overlow SCP requerying is not performed unless the T1OVFLGT option is datafilled in the trigger table tuple that triggered the query. This NIL value is the default value for the CAIN_DEFAULT_OVERFLOW_GT parameter.

Sample entry: **>REP CAIN_DEFAULT_OVERFLOW_GT CAIN_ADDR**

The CAIN_DEFAULT_OVERFLOW_GT parameter is defined.

Step 5: Define resource allocation requirements

Define the following resources for NetworkBuilder processing capabilities:

- CAIN extension blocks
- T_CAIN extension blocks
- CAIN framework extension blocks
- VAMPTRID resources
- CAIN extended call condense blocks
- CAIN STR extension blocks
- ISUP extension blocks
- O_No_Answer and Timeout timers
- CAIN send notification extension blocks
- CAIN Furnish AMA extension blocks
- Permanent extension blocks

Note: Refer to Appendix B, "Engineering guidelines," for information on engineering these parameters.

Step 5: Define resource allocation requirements CAIN extension blocks

NUM_CAIN_EXT_BLOCKS

An extension block is a storage mechanism used to store feature data required for a single call. The CAIN extension block is used to store data needed for triggering, routing, and connecting to a resource.

ATTENTION

The switch requires CAIN extension blocks in order to query the SCP. The default is 260.

When a TDP is encountered, trigger criteria is met, and an action of QUERY or FEAT is provisioned, the CAIN framework determines if a CAIN extension block has been allocated. If not, a new CAIN extension block is allocated. CAIN extension blocks normally remain allocated until a call is answered. However, some scenarios, such as fatal application errors, may release the extension block early.

Fatal application errors

The fatal application error in the following table can occur during CAIN extension block allocation.

CAIN Extension block allocation fatal application errors

Error type	Log generated	Reported to SCP?	Error action performed
Unable to allocate extension block (Note 1)	CAIN200	No	ERRACT in trigger table (Note 2)
Note 1: More extension blocks need to be allocated; there were not enough extension blocks to handle call volume. ***Note 2:*** The OFFCCODE trigger table does not provision ERRACT, it defaults to the ROUTE error action. Trigger tables OFFHKIMM, SPECFEAT, CUSTDP, SPECDIG, and TERMATT provision an error action. AINF is used for all other triggers.			
—end—			

Associated logs

CAIN200, AUD621

Step 5: Define resource allocation requirements CAIN extension blocks (end)

Associated OMs

EXT

Provision the number of CAIN extension blocks available to CAIN call processing

At the CI prompt

1 Enter table CAINPARM.

2 Replace the defined parameter by typing:

>REP NUM_CAIN_EXT_BLOCKS parmval
where

parmval is the number of CAIN extension blocks (0–32,767).

Sample entry: **>REP NUM_CAIN_EXT_BLOCKS 300**

ATTENTION

NUM_CAIN_EXT_BLOCKS requires at least a Cold Restart when decreasing the number of extension blocks available. Changes to the parameter take effect after a Cold Restart.

CAIN extension blocks are provisioned for NetworkBuilder call processing.

Step 5: Define resource allocation requirements T_CAIN extension blocks

NUM_T_CAIN_EXT_BLOCKS

An extension block is a storage mechanism used to store feature data required for a single call. The T_CAIN extension block is used to store data needed for processing by the terminating call model. This extension block will be released following the required processing in the **T_Null** PIC.

ATTENTION

The switch requires T_CAIN extension blocks in order to query the SCP. The default is 260.

Fatal application errors

The fatal application error in the following table can occur during T_CAIN extension block allocation.

T_CAIN Extension block allocation fatal application errors

Error type	Log generated	Reported to SCP?	Error action performed
Unable to allocate extension block (Note 1)	CAIN200	No	ERRACT in trigger table
Note: More extension blocks need to be allocated; there were not enough extension blocks to handle call volume.			
—end—			

Associated logs

CAIN200, AUD665

Associated OMs

EXT

Provision the number of T_CAIN extension blocks available to NetworkBuilder call processing

At the CI prompt

1 Enter table CAINPARM.

Step 5: Define resource allocation requirements T_CAIN extension blocks (end)

2 Replace the defined parameter by typing:

>REP NUM_T_CAIN_EXT_BLOCKS parmval
where

parmval is the number of T_CAIN extension blocks (0–32,767).

Sample entry: **>REP NUM_T_CAIN_EXT_BLOCKS 300**

ATTENTION

NUM_T_CAIN_EXT_BLOCKS requires at least a Cold Restart when decreasing the number of extension blocks available. Changes to the parameter take effect after a Cold Restart.

T_CAIN extension blocks are provisioned for the terminating call model.

Step 5: Define resource allocation requirements CAIN framework extension blocks

NUM_FRAMEWORK_EXT_BLOCKS

An extension block is a storage mechanism used to store feature data required for a single call. The framework extension block is used to store data, such as state and transaction identifiers, needed by the CAIN messaging framework.

It is possible to have two CAIN extension blocks active for a given call, one for the originating call model and one for the terminating call model. Consider the potential impact of allocating multiple extension blocks per call when provisioning NUM_FRAMEWORK_EXT_BLOCKS.

ATTENTION

This parameter should be engineered when NetworkBuilder is introduced for the first time.

ATTENTION

The switch requires framework extension blocks in order to query the SCP. The default is 1800.

When a TDP is encountered, trigger criteria is met, and an action of QUERY or FEAT is provisioned, the CAIN framework determines if a framework extension block has been allocated. If not, a new framework extension block is allocated. Framework extension blocks normally remain allocated until a call is answered. The default is 1800.

Fatal application errors

The fatal application error in the following table can occur during CAIN framework extension block allocation.

Step 5: Define resource allocation requirements CAIN framework extension blocks (end)

CAIN framework extension block allocation fatal application errors

Error type	Log generated	Reported to SCP?	Error action performed
Unable to allocate extension block (Note 1)	CAIN200	No	ERRACT in trigger table (Note 2)
Note 1: More extension blocks need to be allocated; there were not enough extension blocks to handle call volume. ***Note 2:*** The OFFCCODE trigger table does not provision ERRACT, it defaults to the ROUTE error action. Trigger tables OFFHKIMM, SPECFEAT, CUSTDP, SPECDIG, and TERMATT provision an error action. AINF is used for all other triggers.			
—end—			

Associated logs

CAIN200, AUD620

Associated OMs

EXT

Define the NUM_FRAMEWORK_EXT_BLOCKS parameter

At the CI prompt

1 Enter table CAINPARM.

2 Replace the parameter by typing:

>REP NUM_FRAMEWORK_EXT_BLOCKS parmval
where

parmval the number of CAIN framework extension blocks (0 to 32,767)

Sample entry: **>REP NUM_FRAMEWORK_EXT_BLOCKS 2000**

Framework extension blocks are provisioned for NetworkBuilder call processing.

Step 5: Define resource allocation requirements VAMPTRID resources

VAMPTRID resources

Table VAMPTRID (Variable AIN Messaging Platform Transaction Identifiers) provides the following resources used in NetworkBuilder messaging.

- transaction identifier – used to establish and maintain a TCAP communication session between two applications
- component identifier – used to identify and correlate individual operations/requests between applications within the context of a transaction
- message buffers – used as internal workspace for encoding and decoding messages
- ACG blocks – used by applications that use the VAMP framework and ACG controls

NetworkBuilder requires one transaction identifier, one component identifier or two component identifiers when using EDPs, and two message buffers for each query generated by a NetworkBuilder call.

Fatal application errors

The fatal application error in the following table can occur during VAMPTRID resource allocation.

VAMPTRID resource allocation fatal application errors

Error type	Log generated	Reported to SCP?	Error action performed
Unable to allocate VAMPTRID resources (Note 1)	CAIN200 VAMP202	No	ERRACT in trigger table (Note 2)
Note 1: More VAMPTRID resources need to be allocated; there were not enough blocks to handle call volume. ***Note 2:*** The OFFCCODE trigger table does not provision ERRACT, it defaults to the ROUTE error action. Trigger tables OFFHKIMM, SPECFEAT, CUSTDP, SPECDIG, and TERMATT provision an error action. AINF is used for all other triggers.			
—end—			

Step 5: Define resource allocation requirements VAMPTRID resources (end)

Associated logs

CAIN200, VAMP201, VAMP202, VAMP203, VAMP601, VAMP602, VAMP603

Associated OMs

CAINOM, CAINAGOM, VPTRUSAG

Provisioning VAMPTRID resources

At the CI prompt

1 Enter table VAMPTRID.

2 Position on the tuple (CAIN02, IN1):

>POS cain02
Example of a MAP response:

```
CAIN02 0 0 0 0
```

3 Replace the defined parameter by typing:

>REP CAIN02 trids comps mesgs acgbs
where

trids	is the number of transaction identifier blocks.
comps	is the number of component identifier blocks.
mesgs	is the number of message buffer blocks.
acgbs	is the number of ACG blocks.

Sample entry: **>REP CAIN02 30 30 30 512**

ATTENTION

Changing VAMPTRID resources requires at least a Cold Restart when decreasing the number of blocks available. Changes to the table take effect after a Cold Restart.

VAMPTRID resources are provisioned for NetworkBuilder call processing.

Step 5: Define resource allocation requirements CAIN extended call condense blocks

NUM_CAIN_ECCBS Office Parameter

NUM_CAIN_ECCBS is an office parameter in the table CAINPARM. This office parameter is used to allocate pools of extended call condense blocks (ECCBS) for CAIN. These extended call condense blocks, which store call data, are allocated during origination on an agent that has the CAIN option, and during termination on an agent that has the TCAIN option. During call processing, this block stores data that must survive for the entire call, such as various CAIN group subscriptions.

Extended call condense blocks remain active for the entire call, including reorigination. They are not deallocated until the originator is released. Because of this extended life, these blocks are allocated as a percentage of the NCCBS office parm in table OFCENG.

The potential impact on ECCB allocation should be considered when defining agents as CAIN or TCAIN capable. Only one ECCB is required for the call. Refer to Appendix B, "Engineering guidelines," for more information on engineering the NUM_CAIN_ECCBS parameter for ECCB allocation.

ATTENTION

The switch requires extended call condense blocks in order to query the SCP. The default is 260.

Fatal application errors

The fatal application error in the following table can occur during extended call condense block allocation.

Extended call condense block allocation fatal application errors

Error type	Log generated	Reported to SCP?	Error action performed
Unable to allocate extended call condense block (Note 1)	CAIN200	No	N/A (call proceeds without CAIN interaction
Note: More extended call condense blocks need to be allocated; there were not enough call condense blocks to handle call volume.			
—end—			

Step 5: Define resource allocation requirements CAIN extended call condense blocks (end)

Associated logs

CAIN200, AUD621

Associated OMs

EXT

Define the NUM_CAIN_ECCBS parameter

At the CI prompt

1 Enter table CAINPARM.

2 Replace the parameter by typing:

>REP NUM_CAIN_ECCBS parmval
where

parmval the number of extended call condense blocks (0 to 65,535))

Sample entry: **>REP NUM_CAIN_ECCBS 2000**

ATTENTION

NUM_CAIN_ECCBS requires at least a Cold Restart when decreasing the number of extension blocks available. Changes to the parameter take effect after a Cold Restart.

Extended call condense blocks are provisioned for NetworkBuilder originating or terminating call model.

Step 5: Define resource allocation requirements CAIN STR extension blocks

NUM_STR_EXT_BLOCKS

An extension block is a storage mechanism used to store feature data required for a single call. The STR extension block is used to store data needed for managing the connection to an SSP resource.

When a **`Send_To_Resource`** or **`Connect_To_Resource`** message is received, the CAIN user interaction framework determines if an STR extension block has been allocated. If not, a new STR extension block is allocated. STR extension blocks normally remain allocated until a call is routed. However, some scenarios (such as fatal application errors) may release the extension block early.

ATTENTION

The switch requires CAIN STR extension blocks for **`Send_To_Resource`** or **`Connect_To_Resource`** conversational processing. The default is 260.

Fatal application errors

The fatal application error in the following table can occur during STR extension block allocation.

STR extension block allocation fatal application errors

Error type	Log generated	Reported to SCP?	Error action performed
Unable to allocate extension block (Note 1)	CAIN200	No	ERRACT in trigger table (Note 2)
Note 1: More extension blocks need to be allocated; there were not enough extension blocks to handle call volume. ***Note 2:*** The OFFCCODE trigger table does not provision ERRACT, it defaults to the ROUTE error action. Trigger tables OFFHKIMM, SPECFEAT, CUSTDP, SPECDIG, and TERMATT provision an error action. AINF is used for all other triggers.			
—end—			

Associated logs

CAIN200, AUD621

Step 5: Define resource allocation requirements CAIN STR extension blocks (end)

Associated OMs

EXT

Define the NUM_STR_EXT_BLOCKS parameter

At the CI prompt

1 Enter table CAINPARM.

2 Replace the parameter by typing:

>REP NUM_STR_EXT_BLOCKS parmval
where

parmval the number of STR extension blocks (0 to 32,767)

Sample entry: **>REP NUM_STR_EXT_BLOCKS 260**

ATTENTION

NUM_STR_EXT_BLOCKS requires at least a Cold Restart when decreasing the number of extension blocks available. Changes to the parameter take effect after a Cold Restart.

STR extenstion blocks are provisioned for NetworkBuilder call processing.

Step 5: Define resource allocation requirements ISUP extension blocks (end)

NUM_ISUP_EXT_BLKS Office Parameter

NUM_ISUP_EXT_BLKS is a non-NetworkBuilder specific office parameter in table OFCENG. This office parameter is used to store information received in an incoming IAM message for later use in a call. It is allocated upon receipt of an IAM message, and is typically deallocated when the call is answered.

During a STR-Connection to an IP, the ISUP extension block is not deallocated at the local switch when the IP answers. The extension block is needed when the switch establishes a second call leg following an STR-Connection. Once a second call leg is established, the ISUP extension block is deallocated when the called party answers. The default is 1000.

Because the average holding time may be longer for a STR-Connection, the number of ISUP extension blocks may need to be increased.

Associated logs

CAIN200, AUD621

Associated OMs

EXT

Define the NUM_ISUP_EXT_BLKS parameter

At the CI prompt

1 Enter table OFCENG.

2 Replace the parameter by typing:

>REP NUM_ISUP_EXT_BLKS parmval
where

parmval the number of ISUP extension blocks (0–32767)

Sample entry: **>REP NUM_ISUP_EXT_BLKS 1000**

ISUP extension blocks are provisioned for NetworkBuilder STR-Connections.

Step 5: Define resource allocation requirements
CAIN No Answer Timers (end)

NUMCPWAKE Office Parameter

NUMCPWAKE is a non-NetworkBuilder specific office parameter in table OFCENG. This office parameter is used to determine the number of call processing timers allocated for use by the switch. NetworkBuilder O_No_Answer and Timeout timers use timers from this pool. The O_No_Answer timer is started at the ***O_Term_Seized*** event and deallocated at the ***O_Answer*** event or when the call ends. The Timeout timer is started at the ***O_Answer*** event and deallocated at the ***O_Disconnect*** event.

Because the O_No_Answer and Timeout timers cause more of these timer resources to be consumed, this office parameter needs to be increased when provisioning O_No_Answer or Timeout services on the SSP.

Associated logs

AUD621

Associated OMs

None

Define the NUMCPWAKE parameter

At the CI prompt

1 Enter table OFCENG.

2 Replace the parameter by typing:

>REP NUMCPWAKE parmval
where

parmval the number of no answer timers (0 to 32767)

Sample entry: **>REP NUMCPWAKE 300**

No answer timers are provisioned for NetworkBuilder call processing.

Step 5: Define resource allocation requirements CAIN send notification extension blocks (end)

NUM_SEND_NOTIFICATION_EXT_BLOCKS

The NUM_SEND_NOTIFICATION_EXT_BLOCKS parameter in table CAINPARM indicates the number of send notification extension blocks which are allowed for the given UCS DMS-250. The default is 0.

Associated logs

AUD665

Associated OMs

EXT

Provision the number of send notification extension blocks allowed

At the CI prompt

1 Enter table CAINPARM.

2 Replace the defined parameter by typing:

>REP NUM_SEND_NOTIFICATION_EXT_BLOCKS parmval

where

parmval is the number of send notification extension blocks (0–32,767).

Sample entry: **>REP NUM_SEND_NOTIFICATION_EXT_BLOCKS 300**

Send notification extension blocks are provisioned.

Step 5: Define resource allocation requirements CAIN Furnish AMA extension blocks (end)

NUM_FURNISHAMA_EXT_BLOCKS

The NUM_FURNISHAMA_EXT_BLOCKS parameter in table CAINPARM indicates the number of CAIN Furnish_AMA extension blocks to be allocated for the UCS DMS-250 switch. The default is 0.

Associated logs

CAIN100

Associated OMs

none

Provision the number of Furnish AMA extension blocks allowed

At the CI prompt

1 Enter table CAINPARM.

2 Replace the defined parameter by typing:

>REP NUM_FURNISHAMA_EXT_BLOCKS parmval

where

parmval is the number of Furnish AMA extension blocks (0–32,767).

Sample entry: **>REP NUM_FURNISHAMA_EXT_BLOCKS 300**

CAIN Furnish AMA extension blocks are provisioned.

Step 5: Define resource allocation requirements Permanent extension blocks

NUMPERMEXT

An extension block is a storage mechanism used to store feature data required for a call. The switch uses permanent extension blocks to store data needed for **Connect_To_Resource** interactions and three-way (three subscribers) **Merge_Call** processing.

When the switch receives a **Connect_To_Resource** or a **Merge_Call** message, call processing determines if the switch has allocated PORTPERM extension blocks. If the switch has not allocated any PORTPERM extemsion blocks, the switch allocates four new PORTPERM extension blocks. PORTPERM extension blocks normally remain allocated until the **Connect_To_Resource** interaction or **Merge_Call** processing is complete. If the **Merge_Call** processing transitions the call into call configuration 10, the switch holds PORTPERM extension blocks until one party releases. If the call transitions to any other call configuration, the switch releases the PORTPERM extension blocks immediately. Certain scenarios, such as fatal application errors, can release the extension blocks early.

Associated OMs

EXT

Provision the number of CAIN extension blocks available to CAIN call processing

At the CI prompt

1 Enter table OFCENG.

2 Replace the defined parameter by typing:

>REP NUMPERMEXT parmval
where

parmval is the number of permanent extension blocks (1 to 16000).

Sample entry: **>REP NUMPERMEXT 100**

ATTENTION

NUMPERMEXT requires at least a Cold Restart when decreasing the number of extension blocks available. Changes to the parameter take effect after a Cold Restart.

Step 5: Define resource allocation requirements Permanent extension blocks (end)

You have provisioned permanent extension blocks for NetworkBuilder call processing.

Step 6: Datafill required agents as CAIN/T_CAIN-capable

Before any NetworkBuilder processing takes place, the originating or terminating agency must be provisioned as CAIN or T_CAIN-capable.

NetworkBuilder supports originating agencies: DAL, FGD, SS7 Inter-IMT, SS7 Global-IMT, AXXESS, and PRI.

Note: Originating FGB trunks are supported for the *Office_Code* trigger. Refer to the *UCS DMS-250 Local Number Portability Application Guide* for more information.

NetworkBuilder supports terminating agencies: DAL, FGB, FGC, FGD, IMT, AXXESS, and PRI.

ATTENTION

When the CAIN option is not provisioned for the originating agency, or TCAIN option is not provisioned for the terminating agency, the call is not evaluated for the corresponding NetworkBuilder services.

Note: Datafill requirements are handled differently for AXXESS agents. Refer to *UCS DMS-250 CAIN/FlexDial Interactions* for more information.

Datafill requirements

Before any NetworkBuilder services can be used, you must provision the originating or terminating agent as CAIN-capable using the CAIN or TCAIN option in table TRKGRP (Trunk Group), TRKFEAT (Trunk Features), or CALLATTR (Call Attributes).

TRKGRP

Provision DAL, FGD, SS7 Inter-IMT, or SS7 Global-IMT originating agents in table TRKGRP. Provision DAL, FGB, FGC, FGD, IMT terminating agents in table TRKGRP. CAIN and TCAIN provisioning is set up in the OPTION refinement before using NetworkBuilder services.

Note: Originating FGB trunks are supported for the *Office_Code* trigger. Refer to the *UCS DMS-250 Local Number Portability Application Guide* for more information.

TRKFEAT

Provision AXXESS originating and terminating agents in table TRKFEAT. NetworkBuilder provisioning is set up in the ORIGOPTS or TERMOPTS refinement before using NetworkBuilder services.

Step 6: Datafill required agents as CAIN/T_CAIN-capable (continued)

CALLATTR

Although PRI originating and terminating agents are datafilled in table TRKGRP, you must provision the PRI call attributes for NetworkBuilder services. CAIN and TCAIN provisioning is set up in the OPTION refinement of table CALLATTR.

Step 6: Datafill required agents as CAIN/T_CAIN-capable Non-PRI, non-AXXESS originating agents (end)

NetworkBuilder supports the following originating agencies: DAL, FGD, SS7 Inter-IMT, and SS7 Global-IMT, AXXESS, and PRI.

Note: Datafill requirements are handled differently for AXXESS agents. Refer to *UCS DMS-250 CAIN/FlexDial Interactions* for more information.

At the CI prompt

1 Enter table TRKGRP.

2 Position on TRKGRP requiring NetworkBuilder originating call model capabilities.

3 Change the tuple to reflect NetworkBuilder capability and add the CAIN option by using the following format:

>REP grpkey grptyp trafsno padgrp nccls grpinfo
where

grpkey is the CLLI of the trunk requiring NetworkBuilder services.
grptyp is the trunk group type.
trafsno is the traffic separation software number.
padgrp is the pad group name.
nccls is the no circuit class type used when routing list is exhausted.
grpinfo contains multiple subfields, including the OPTION subfield, where
CAIN is the option used to provision the originating agency as CAIN-capable.

DAL sample entry: **>REP dal221twdtls dal 30 npdgp ncit 0 2w dal midl 16 7 16 16 s 7 nil id 0 7 111 manual 213 0 6113311 rte2 0 3_1khz y 1 n y none 00 160 CAIN**

FGD sample entry: **>REP ean862c7lp00 eant 50 npdgp ncof 0 2w ean midl 16 7 16 16 eapt 10 4 214 ucs2eaeo nil 214 111 manual 0 rte1 0 1 voice_data 160 mccs reorgval id24_on CAIN**

SS7-Inter-IMT sample entry: **>REP imt762c7lp00 imt 40 npdgp ncit 0 2w imt midl 16 7 16 16 ucs2ucs nil c n none 4 always i3pa 111 0 inter n voice data none 4 160 214 0 CAIN**

SS7-Global-IMT sample entry: **>REP imt762c7xx02 imt 40 npdgp ncit 0 2w imt midl 16 7 16 16 ucs2ucs nil c n none 4 always addr 111 0 global n y cgpa n voice_data none 4 160 214 0 CAIN**

The agent is now capable of using NetworkBuilder call processing in the originating call model.

Step 6: Datafill required agents as CAIN/T_CAIN-capable Non-PRI, non-AXXESS terminating agents (end)

NetworkBuilder supports the following terminating agencies: DAL, FGB, FGC, FGD, IMT, AXXESS, and PRI.

Note: Datafill requirements are handled differently for AXXESS agents. Refer to *UCS DMS-250 CAIN/FlexDial Interactions* for more information.

At the CI prompt

1 Enter table TRKGRP.

2 Position on TRKGRP requiring NetworkBuilder terminating call model capabilities.

3 Change the tuple to reflect NetworkBuilder capability and add the TCAIN option by using the following format:

>REP grpkey grptyp trafsno padgrp nccls grpinfo
where

grpkey is the CLLI of the trunk requiring NetworkBuilder services.
grptyp is the trunk group type.
trafsno is the traffic separation software number.
padgrp is the pad group name.
nccls is the no circuit class type used when routing list is exhausted.
grpinfo contains multiple subfields, including the OPTION subfield, where
TCAIN is the option used to provision the terminating agency as TCAIN-capable.

DAL sample entry: **>REP dal221twdtls dal 30 npdgp ncit 0 2w dal midl 16 7 16 16 s 7 nil id 0 7 111 manual 213 0 6113311 rte2 0 3_1khz y 1 n y none 00 160 TCAIN**

FGB sample entry: **>REP ont544twdtwk onat 50 npdgp ncof 0 2w ont midl 16 7 16 16 7 7 ot 0 0 214 s 7 fgbc 111 manual 214 0 6112211 none 0 y 1 none 00 immediate voice_data 160 TCAIN**

FGC sample entry: **>REP onl420ogdpsz onal 127 npdgp ncof 0 og onl midl 16 7 16 214 1 n on 0 s 5 111 manual 214 0 6112211 none 0 y 1 none 00 160 TCAIN**

FGD sample entry: **>REP ean862c7lp00 eant 50 npdgp ncof 0 2w ean midl 16 7 16 16 eapt 10 4 214 ucs2eaeo nil 214 111 manual 0 rte1 0 1 voice_data 160 mccs reorgval id24_on TCAIN**

IMT sample entry: **>REP imt762c7xx02 imt 40 npdgp ncit 0 2w imt midl 16 7 16 16 ucs2ucs nil c n none 4 always addr 111 0 global n y cgpa n voice_data none 4 160 214 0 TCAIN**

The agent is now capable of using NetworkBuilder call processing in the terminating call model.

Step 6: Datafill required agents as CAIN/T_CAIN-capable AXXESS originating and terminating agents (end)

Note: Datafill requirements are handled differently for AXXESS agents. Refer to *UCS DMS-250 CAIN/FlexDial Interactions* for more information.

At the CI prompt

1 Enter table TRKFEAT.

2 Position on feature set requiring NetworkBuilder originating and/or terminating call model capabilities.

3 Change the tuple to reflect NetworkBuilder capability and add the CAIN option by using the following format:

>REP key origopts termopts
where

key	is the name of the feature set (up to 16 alphanumeric characters) indexed by table TRKGRP.
origopts	is the option(s) you want to enable on the originations (CAIN).
termopts	is the option(s) you want to enable on the terminations (CAIN).

AXXESS sample entry: **>REP trkfeat_2323 CAIN $ onnettrk CAIN**

The AXXESS agent is now capable of using NetworkBuilder call processing in the originating and terminating call models.

Step 6: Datafill required agents as CAIN/T_CAIN-capable PRI originating call attributes (continued)

At the CI prompt

1 Enter table CALLATTR.

2 Position on the CATTRIDX requiring NetworkBuilder originating call model capabilities.

3 Change the tuple to reflect NetworkBuilder capability and add the CAIN option using the following format:

>REP cattridx customer prtnm cos zerompos custarea
where

cattridx is the index datafilled in table LTCALLS' CALLATTR field.
customer is the customer type.
prtnm is the pretranslator name [index into table STDPRTCT (Standard Pretranslator Control)].
cos is the class of service screening (index into table TRKCOS).
zerompos is the route choice (index into table POSITION).
custarea contains multiple subfields, including the OPTION subfield, where
CAIN is the option used to provision the originating agency as CAIN-capable.

Sample entry: **>REP 21 ucs pri 0 none 611 auth 1 6112211 0 5026112211 anidelv CAIN $**

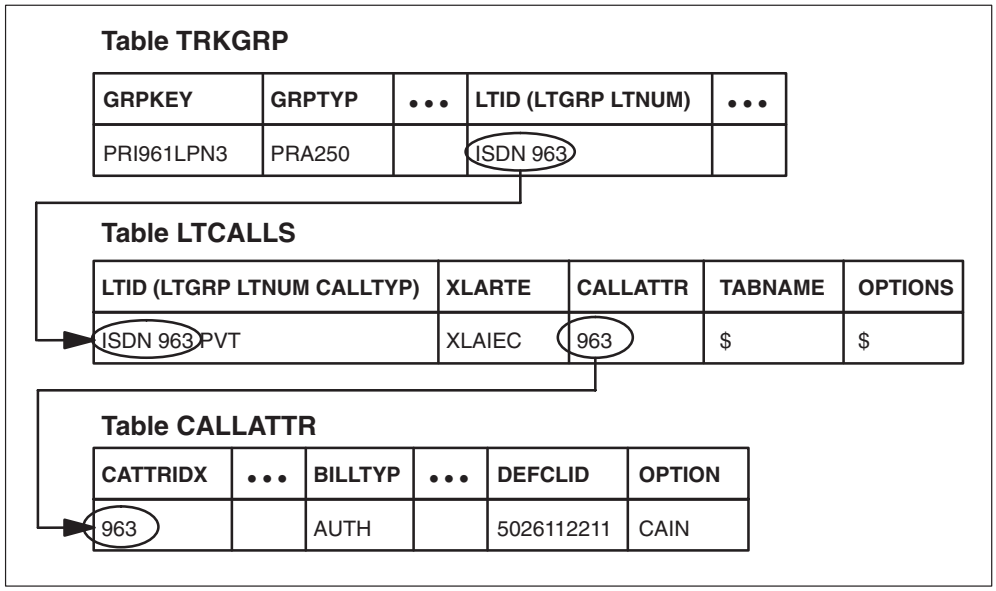
4 Check table LTCALLS datafill for required index into table CALLATTR (field CALLATTR). Figure 2-8 below shows the relationship between tables CALLATTR, LTCALLS, and TRKGRP.

5 Check table TRKGRP datafill for required index into table LTCALLS (field GRPINFO, subfield LTID). Figure 2-8 below shows the relationship between tables CALLATTR, LTCALLS, and TRKGRP.

PRI call attribute is now capable of using NetworkBuilder call processing in the originating call model.

Step 6: Datafill required agents as CAIN/T_CAIN-capable PRI originating call attributes (end)

Figure 2-8
Relationship between CALLATTR, LTCALLS, and TRKGRP



Step 6: Datafill required agents as CAIN/T_CAIN-capable PRI terminating call attributes (continued)

At the CI prompt

1 Enter table CALLATTR.

2 Position on the CATTRIDX requiring NetworkBuilder terminating call model capabilities.

3 Change the tuple to reflect NetworkBuilder capability and add the TCAIN option using the following format:

>REP cattridx customer prtnm cos zerompos custarea

where

cattridx is the index datafilled in table LTCALLS' CALLATTR field.

customer is the customer type.

prtnm is the pretranslator name [index into table STDPRTCT (Standard Pretranslator Control)].

cos is the class of service screening (index into table TRKCOS).

zerompos is the route choice (index into table POSITION).

custarea contains multiple subfields, including the OPTION subfield, where

TCAIN is the option used to provision the terminating agency as CAIN-capable.

Sample entry: **>REP 21 ucs pri 0 none 611 auth 1 6112211 0 5026112211 anidelv TCAIN $**

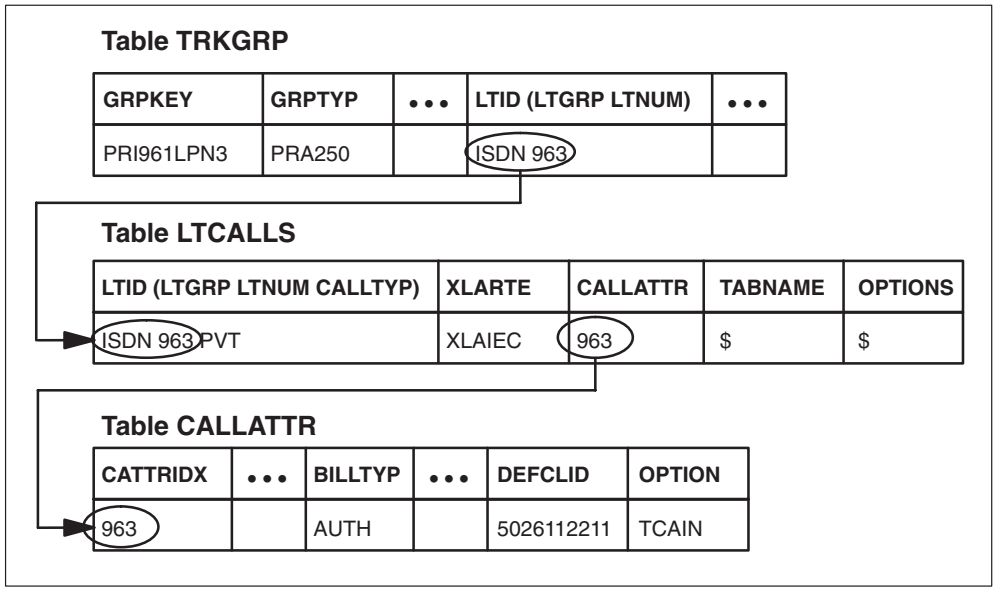
4 Check table LTCALLS datafill for required index into table CALLATTR (field CALLATTR). Figure 2-8 below shows the relationship between tables CALLATTR, LTCALLS, and TRKGRP.

5 Check table TRKGRP datafill for required index into table LTCALLS (field GRPINFO, subfield LTID). Figure 2-8 below shows the relationship between tables CALLATTR, LTCALLS, and TRKGRP.

PRI call attribute is now capable of using NetworkBuilder call processing in the terminating call model.

Step 6: Datafill required agents as CAIN/T_CAIN-capable PRI terminating call attributes (end)

Figure 2-9
Relationship between CALLATTR, LTCALLS, and TRKGRP



Step 7: Define CAIN groups and enable trigger sets

When you define CAIN groups, consider these two questions:

1 What TDPs are best suited to the SCP service?

2 Which calls require the SCP service?

What TDPs are best suited to the SCP service?

TDPs identify places within a call (as defined by the call model) where the switch can temporarily suspend call processing and off-load call processing to the SCP. You should choose the TDP based on the type of service the SCP is providing.

N00 time-of-day routing case study

For this case study, an N00 service provider needs the ability to route to different locations depending on the time of day. You need to determine which TDP is best suited for off-loading the processing of this call to enable your SCP to provide this service.

What TDP should be used?

- ***Origination_Attempt***? No, ***Origination_Attempt*** is not a good choice because:
  - — no digit collection has occurred. The switch has not determined whether this in an N00 call. Your SCP service provides routing data to the switch. You should allow existing switch software perform digit collection.
  - — time-of-day routing requires SS7 FGD, SS7 Inter-IMT, SS7 Global-IMT, or AXXESS originations. *Off_Hook_Immediate* (for DAL originations only) is the only trigger available at ***Origination_Attempt***.
- ***Info_Collected***? Yes, ***Info_Collected*** is a good choice because:
  - — digit collection has occurred, but the in-switch translations have not. Since your SCP service is providing translations, there is no need for the switch to also perform the translation.
  - — even when screening fails, N00 calls are allowed to query. In this case, the switch has set treatment, but has not applied it to the call.
- ***Info_Analyzed***? No, ***Info_Analyzed*** is not a good choice because:
  - — translations have already been performed. Since the SCP service is providing translations for the N00 number, switch resources are wasted if the translation is performed twice (once at the switch or data control point [DCP] and then again at the SCP)
  - — if ANI screening failed, the call was sent to treatment and is no longer available to query the SCP

Step 7: Define CAIN groups and enable trigger sets (continued)

Note: Calls may query when ANI screening fails, if the network allows casual use.

- ***Network_Busy***, ***O_Called_Party_Busy***, ***O_No_Answer***? No, these triggers are not good choices, because:
  - — each TDP occurs after translations has identified a routing index. Since the SCP is providing translation services, a query needs to happen earlier in the call model.
  - — these TDPs are not invoked for every call. The SCP service is needed for every call to a specific N00 number.

CAINGRP

Datafill table CAINGRP (CAIN Group) to define the groups available for originating and terminating NetworkBuilder services subscription. You can set subscription to a CAIN group in the following tables:

- STDPRTCT – provides NetworkBuilder subscription for addresses
- AUTHCODU, AUTHCDU2, AUTHCDU3, AUTHCDU4, and AUTHCDU5 – provide NetworkBuilder subscription for authorization codes
- ANISCUSP, or ANIVAL and UNIPROF – provide NetworkBuilder subscription for ANIs
- TRKGRP – provides NetworkBuilder subscription for DAL, FGD, SS7 Inter-IMT, and SS7 Global-IMT originating agencies, as well as, DAL, FGB, FGC, FGD, and IMT terminating agencies
- TRKFEAT – provides NetworkBuilder subscription for originating and terminating AXXESS agencies
- CALLATTR – provides NetworkBuilder subscription for originating and terminating PRI call attributes
- CAINPARM (parameter CAIN_OFFICE_GROUP) – provides office-wide NetworkBuilder subscription

CAIN group subscription can also be determined by the SCP when the `cainGroup` extension parameter is returned in an **`Analyze_Route`** message.

Note: CAIN group subscription is handled differently for AXXESS agents. Refer to *UCS DMS-250 CAIN/FlexDial Interactions* for more information.

Each CAIN group enables one or more triggers in the originating or terminating call model and provides an index into the triggering tables.

Step 7: Define CAIN groups and enable trigger sets (continued)

Defining a CAIN group

At the CI prompt

1 Enter table CAINGRP.

Note 1: Table CAINGRP provisions the CAIN group name and identifies the triggers that are evaluated for the associated customer, agency, or office subscription. Agent is the only supported subscription method for the terminating call model triggers.

Note 2: Table CAINGRP must be datafilled prior to defining CAIN subscription.

2 Create a CAIN group by using the following format:

>ADD caingrp msgset proto trigrset options extparms textparms $
where

caingrp is the name of the CAIN group (0–16 alphanumeric characters).
grpnum is the number associated with the CAIN group (0 to 4095).
msgset is the message set (CAIN02, IN1).
proto is the message protocol (TCAP_SCCP).
trigrset is the trigger (TOLLFREE) when the **msgset** is IN1.
is a multiple-entry vector comprised of three subfields (PIC, TDP, TRIGGER) when the **msgset** is CAIN02.
PIC is the point in call defined for this CAIN group.
TDP is the trigger detection point defined for this CAIN group and PIC.
TRIGGER is the trigger being defined for this CAIN group, PIC, and TDP.

The following CAIN trigger sets are supported in the NetworkBuilder originating call model when the **msgset** is CAIN02:

- O_NULL ORIGATT OFFHKIMM
- COLLINFO OFTRREQ OFTRREQ
- COLLINFO INFOCOLL OFFHKDEL
- COLLINFO INFOCOLL SIOTRK
- COLLINFO INFOCOLL PRIBCHNL
- ANLZINFO INFOANLZ SPECFEAT
- ANLZINFO INFOANLZ CUSTDP
- ANLZINFO INFOANLZ SPECDIG
- ANLZINFO INFOANLZ OFFCCODE
- SELROUTE NETBUSY NETBUSY
- SEND_CALL OCLDBUSY OCLDBUSY
- O_ALERTG ONOANSWR ONOANSWR
- O_ACTIVE OMIDCALL OIECREO

Step 7: Define CAIN groups and enable trigger sets (end)

The following CAIN trigger set is supported in the NetworkBuilder terminating call modell when the **msgset** is CAIN02:

- T_NULL TERMATT TERMATT

Note: A CAINGRP may subscribe to more than one trigger.

options is an option vector (OANSTIME, OVRREORG, NIL).

extparms identifies the extension parameters to be sent in the query message of the originating call model.

textparms identifies the extension parameters to be sent in the query message of the terminating call model.

Sample entry: **>ADD time_of_day 99 cain02 tcap_sccp collinfo infocoll siotrk collinfo infocoll pribchnl $ $ $ $**

Sample entry: **>ADD termgrp 99 cain02 tcap_sccp t_null termatt termatt $ $ $ termtrkinfo $**

Sample entry: **>ADD tfreegrp 100 in1 tcap_sccp tollfree $ $ $ $**

A CAIN group is defined.

Step 8: Choose the type of subscription

Once you've identified the TDPs that require specific services provided by the SCP, you need to determine the calls that require the service.

Any call originating on a supported, provisioned agency (DAL, FGD, SS7 Inter-IMT, SS7 Global-IMT, AXXESS, or PRI) can subscribe to NetworkBuilder originating call model services.

Any call terminating on a supported, provisioned agency (DAL, FGB, FGC, FGD, SS7 Inter-IMT, SS7 Global-IMT, AXXESS, or PRI) can subscribe to NetworkBuilder terminating call model services.

To allow subscription to originating and terminating NetworkBuilder capabilities exclusive of each other, the CAINGRP option in table TRKGRP or CALLATTR is used to subscribe to originating CAIN groups and the TCAINGRP option is used to subscribe to terminating CAIN groups.

Note: Subscription to NetworkBuilder services is handled differently for AXXESS agents. Refer to *UCS DMS-250 CAIN/FlexDial Interactions* for more information.

Subscription methods are evaluated at **O_Null** and **Collect_Information**. Up to six subscription groups (in order of precedence) are stored by NetworkBuilder call processing for the remaining PICs. The six types of subscription are:

- SCP-returned CAIN group – returned by the SCP. Whenever the `cainGroup` extension parameter is returned by the SCP, the previously stored CAIN group is overwritten by the CAIN group provided by the SCP. Requires the CAIN0601 SOC option.

  Note: This is the only type of subscription that can change after leaving **Collect_Information**.

- Address subscription – subscriber subscription provisioned in table (STDPRTCT) identifies the CAIN group used by the address.
- Authorization code subscription – subscriber subscription provisioned in one of the authcode tables (AUTHCODU, AUTHCDU2, AUTHCDU3, AUTHCDU4, or AUTHCDU5) identifies the CAIN group used by the authcode.
- Automatic number identification (ANI) subscription – subscriber subscription provisioned in one of the ANI tables (ANISCUSP, or ANIVAL and UNIPROF) identifies the CAIN group used by the ANI.
- Agent subscription – agent subscription provisioned in table TRKGRP, table TRKFEAT, or table CALLATTR identifies the originating or terminating CAIN group used by the agent.

Step 8: Choose the type of subscription (continued)

- Default office subscription – office-wide subscription provisioned in table CAINPARM identifies the CAIN group used by the office.

The following table shows the order that triggers are evaluated when an CAIN group for is stored for each subscription method.

Step 8: Choose the type of subscription (continued)

Table 2-9
Trigger evaluation order

		Subscription method					
TDP	**Trigger**	**SCP**	**Addr**	**Auth**	**ANI**	**Agent**	**Office**
Origination_Attempt	*Off_Hook_Immediate*	na	na	na	na	1	2
O_Feature_Requested	*O_Feature_Requested*	na	1	2	3	4	5
Info_Collected	*Tollfree_Service* (note 3)	1 (note 2)	2	3	4	5	6
	Offhook_Delay	7 (note 2)	8	9	10	11	12
	Shared_Interoffice_Trunk	13 (note 2)	14	15	16	17	18
	PRI_B-Channel	13 (note 2)	14	15	16	17	18
Info_Analyzed	*Specific_Feature_Code*	1	5	9	13	17	21
	Customized_Dialing_Plan	2	6	10	14	18	22
	Specific_Digit_String	3	7	11	15	19	23
	Office_Code	4	8	12	16	20	24
Network_Busy	*Network_Busy*	1	2	3	4	5	6
O_Called_Party_Busy	*O_Called_Party_Busy*	1	2	3	4	5	6

Note 1: Only the ADDR CAIN group is re-evaluated upon reorigination.
Note 2: An SCP-returned CAIN group, specified before reorigination, is available at these TDPs after reorigination.
Note 3: NetworkBuilder supports Bellcore's *TR-NWT-000533* toll-free service specifications. The ability to subscribe to mulitple CAIN groups allows for a service integration of the CAIN triggers and the IN/1 *Tollfree_Service* trigger defined in *TR-NWT-000533*.

—continued—

Step 8: Choose the type of subscription (continued)

Table 2-9
Trigger evaluation order (continued)

		Subscription method					
TDP	**Trigger**	**SCP**	**Addr**	**Auth**	**ANI**	**Agent**	**Office**
O_No_Answer	*O_No_Answer*	1	2	3	4	5	6
O_Mid_Call	*O_IEC_Reorigination*	1	2	3	4	5	6
T_Null	*Termination_Attempt*	na	na	na	na	1	na

Note 1: Only the ADDR CAIN group is re-evaluated upon reorigination.
Note 2: An SCP-returned CAIN group, specified before reorigination, is available at these TDPs after reorigination.
Note 3: NetworkBuilder supports Bellcore's *TR-NWT-000533* toll-free service specifications. The ability to subscribe to mulitple CAIN groups allows for a service integration of the CAIN triggers and the IN/1 *Tollfree_Service* trigger defined in *TR-NWT-000533*.

—end—

SCP-returned CAIN groups

ATTENTION
The CAIN0601 SOC option is required for SCP-returned CAIN groups. Refer to Volume 5, Chapter 5, "NetworkBuilder SOC functionality," for more information.

By returning a CAIN group from the SCP through the `cainGroup` extension parameter in an **`Analyze_Route`** message, you gain greater control over the processing of a call. For example,

- New services can be enabled after a query.
- Routing can be controlled when the SCP returns an **`Analyze_Route`** message containing the CAIN routing parameters and the `cainGroup` extension parameter.

Step 8: Choose the type of subscription (continued)

- The new CAIN group can control enabling of the busy triggers (*Network_Busy*, *O_Called_Party_Busy*, and *O_No_Answer*).
- The new CAIN group is used with reoriginated calls.

General subscription rules

The following rules apply to NetworkBuilder subscription:

- Up to six CAIN groups (one for each subscription type) are stored by NetworkBuilder call processing for use during the call.
- A CAIN group returned by the SCP in the `cainGroup` extension parameter always overwrites the previously stored SCP-returned CAIN group.
- Addresses collected at ***O_Feature_Requested*** are overwritten by subsequent addresses collected. The last address collected is used for subscription.
- ANIs and authcodes returned by the SCP are never used to determine subscription.
- When a TDP is encountered, call processing checks the CAIN group for subscription to the appropriate trigger set.

  Note: A trigger set consists of a PIC, TDP, and trigger when the protocol is CAIN02 and consists of a trigger only when the protocol is IN/1.

  CAIN group subscription is always checked in the following order for the originating call model:

  — SCP-returned CAIN group

  — Address-provisioned CAIN group

  — Authcode-provisioned CAIN group

  — ANI-provisioned CAIN group

  — Originating agent-provisioned CAIN group

  — Default office-provisioned CAIN group

  For the terminating call model only agent-provisioned CAIN groups are supported.
- When trigger criteria is met and the datafilled trigger action is QUERY, FEAT, NEXTRTE, NEXTCNRTE, BLOCK, LEAVE_TDP, or CONT_NOTRIG call processing performs the action without checking remaining subscription types.

Step 8: Choose the type of subscription (continued)

- When trigger criteria is not met or the datafilled trigger action is IGNORE, call processing checks the next subscribed trigger at the current TDP. Once all triggers have been checked, call processing returns to the first trigger and checks the next subscription method.
- Reorigination does not reset the SCP-returned CAIN group.
- Since each call has only one switch-determined ANI, originating agent, and office, the CAIN groups stored for these subscription methods are never re-evaluated.
- Since more than one authcode may be collected for a call, call processing may re-evaluate authcode subscription during **Collect_Information** for a new CAIN group.
- Since more than one address can be collected for a call during ***O_Feature_Requested*** and Virtual IP processing, call processing may re-evaluate address subscription for a new CAIN group.

What type of subscription is best suited to the SCP service?

Consider the time-of-day example from Step 7 that allows the SCP to control destination routing based on the time of day. What type of subscription is appropriate for this service?

- Address? Yes, address subscription is a good choice because:
  - — it allows the SCP to discriminate within its service logic based on the address.
- ANI or authorization code? No, ANI or authcode subscription are not good choices because:
  - — these types of subscription should mainly be used for services subscribed to by individual subscribers.
  - — Access to N00 numbers is not based on subscriber data. Therefore, it is unnecessary for the switch to identify every subscriber allowed to dial an N00 number, which would require a large amount of datafill.
- Agent? Yes, agent subscription is a good choice because:
  - — it allows the switch to subscribe to the service for all calls originating from a specific agent. Therefore, the switch could subscribe to all agents that are likely to carry N00 traffic (for example, FGD).
- Office? Yes, office subscription is a good choice because:
  - — it allows the switch to subscribe to the service on all calls originating on NetworkBuilder capable agents.
- SCP-returned group? No, using an SCP-returned CAIN group would not work, because use of this subscription type requires a previous query.

Step 8: Choose the type of subscription (continued)

Subscription uses

The following table lists the subscription type, the recommended trigger, and the associated NetworkBuilder service.

Subscription uses

Subscription type	NetworkBuilder service	Recommended triggers
SCP-returned CAIN group	Network Select Enhanced Services	
	• Network Forwarding	*Network_Busy*
	• Call Forwarding	*O_Called_Party_Busy*, *O_No_Answer*
	• Rerouting on Busy Signal	*O_Called_Party_Busy*
	• Rerouting on No Answer	*O_No_Answer*
	• Network Queuing (Cancel_Resource_Event)	*Network_Busy*, *O_Called_Party_Busy*, *O_No_Answer*
	• Reorigination Control	*O_IEC_Reorigination*, *Offhook_Delay*, *Shared_Interoffice_Trunk*
Address	N00 Services	
	• Follow Me, Find Me, Do Not Disturb Me	*Offhook_Delay*, *Shared_Interoffice_Trunk*, *PRI_B-Channel*, *Specific_Digit_String*
	• Call Screening	*Offhook_Delay*, *Shared_Interoffice_Trunk*, *PRI_B-Channel*, *Specific_Digit_String*
	• Customized Call Branding	*Offhook_Delay*, *Shared_Interoffice_Trunk*, *PRI_B-Channel*, *Specific_Digit_String*

—continued—

Step 8: Choose the type of subscription (continued)

Subscription uses (continued)

Subscription type	NetworkBuilder service	Recommended triggers
Address (cont)	• Universal Access Authorization	*Offhook_Delay*, *Shared_Interoffice_Trunk*, *PRI_B-Channel*, *Specific_Digit_String*
Authorization code	VPN Services	
	• OFFNET Overflow	*Specific_Feature_Code*, *Customized_Dialing_Plan*
	• Forced ONNET	*Specific_Feature_Code*, *Customized_Dialing_Plan*
	• Alternate Billing Numbers	*Specific_Feature_Code*, *Customized_Dialing_Plan*
	• Origination/Termination Screening	*Specific_Feature_Code*, *Customized_Dialing_Plan*
	• Customized Announcements	*Specific_Feature_Code*, *Customized_Dialing_Plan*
	• Black Box Screening	*Offhook_Delay*, *Shared_Interoffice_Trunk*, *PRI_B-Channel*, *Specific_Feature_Code*, *Customized_Dialing_Plan*, *Specific_Digit_String*
	• GVNS	*Specific_Feature_Code*, *Customized_Dialing_Plan*
	Dial 1+ Services	
	• Speed Dial	*O_Feature_Requested*, *Offhook_Delay*, *Shared_Interoffice_Trunk*, *PRI_B-Channel*, *Specific_Feature_Code*, *Specific_Digit_String*

—continued—

Step 8: Choose the type of subscription (continued)

Subscription uses (continued)

Subscription type	NetworkBuilder service	Recommended triggers
Authorization code (cont)	• Hotline	*O_Feature_Requested*, *Offhook_Delay*, *Shared_Interoffice_Trunk*, *PRI_B-Channel*, *Specific_Digit_String*
	• Intra-LATA Presubscription Screening	*O_Feature_Requested*
	• Account Code Screening	*O_Feature_Requested*, *Offhook_Delay*, *Shared_Interoffice_Trunk*, *PRI_B-Channel*, *Specific_Digit_String*
	• Bill to Office	*O_Feature_Requested*, *Offhook_Delay*, *Shared_Interoffice_Trunk*, *PRI_B-Channel*, *Specific_Digit_String*
	Dial 1+ Services (cont)	
	• Prepaid services	*O_Feature_Requested* or *Offhook_Delay*, *Shared_Interoffice_Trunk* using IP interaction
	• CIC Routing and Branding	*Off_Hook_Immediate*, *O_Feature_Requested*, *Offhook_Delay*, *Shared_Interoffice_Trunk*, *PRI_B-Channel*, *Specific_Feature_Code*, *Customized_Dialing_Plan*, *Network_Busy*, *O_Called_Party_Busy*, *O_No_Answer*

—continued—

Step 8: Choose the type of subscription (continued)

Subscription uses (continued)

Subscription type	NetworkBuilder service	Recommended triggers
Authorization code (cont)	• Chat Lines	*Offhook_Delay*, *Shared_Interoffice_Trunk*, *PRI_B-Channel*
ANI	N00 Services	
	• Follow Me, Find Me, Do Not Disturb Me	*Offhook_Delay*, *Shared_Interoffice_Trunk*, *PRI_B-Channel*, *Specific_Digit_String*
	• Call Screening	*Offhook_Delay*, *Shared_Interoffice_Trunk*, *PRI_B-Channel*, *Specific_Digit_String*
	• Customized Call Branding	*Offhook_Delay*, *Shared_Interoffice_Trunk*, *PRI_B-Channel*, *Specific_Digit_String*
	• Universal Access Authorization	*Offhook_Delay*, *Shared_Interoffice_Trunk*, *PRI_B-Channel*, *Specific_Digit_String*
	Subscriber Screening	
	• Authorization Code Screening	*Offhook_Delay*, *Shared_Interoffice_Trunk*, *PRI_B-Channel*, *Specific_Digit_String*
	• Account Code Screening	*Offhook_Delay*, *Shared_Interoffice_Trunk*, *PRI_B-Channel*, *Specific_Digit_String*
	• Enhanced Travel Card Services	*O_Feature_Requested*
	Dial 1+ Services	

—continued—

Step 8: Choose the type of subscription (continued)

Subscription uses (continued)

Subscription type	NetworkBuilder service	Recommended triggers
ANI (cont)	• Speed Dial	*O_Feature_Requested*, *Offhook_Delay*, *Shared_Interoffice_Trunk*, *PRI_B-Channel*, *Specific_Digit_String*
	• Hotline	*O_Feature_Requested*, *Offhook_Delay*, *Shared_Interoffice_Trunk*, *PRI_B-Channel*, *Specific_Digit_String*
	• Intra-LATA Presubscription Screening	*O_Feature_Requested*
	• Account Code Screening	*O_Feature_Requested*, *Offhook_Delay*, *Shared_Interoffice_Trunk*, *PRI_B-Channel*, *Specific_Digit_String*
	• Bill to Office	*O_Feature_Requested*, *Offhook_Delay*, *Shared_Interoffice_Trunk*, *PRI_B-Channel*, *Specific_Digit_String*
	• Prepaid services	*O_Feature_Requested* or *Offhook_Delay*, *Shared_Interoffice_Trunk* using IP interaction
	Dial 1+ Services (cont)	
	• CIC Routing and Branding	*Off_Hook_Immediate*, *O_Feature_Requested*, *Offhook_Delay*, *Shared_Interoffice_Trunk*, *PRI_B-Channel*, *Specific_Feature_Code*, *Customized_Dialing_Plan*, *Network_Busy*, *O_Called_Party_Busy*, *O_No_Answer*

—continued—

Step 8: Choose the type of subscription (continued)

Subscription uses (continued)

Subscription type	NetworkBuilder service	Recommended triggers
ANI (cont)	• Black Box Screening	*Off_Hook_Immediate*, *O_Feature_Requested*, *Offhook_Delay*, *Shared_Interoffice_Trunk*, *PRI_B-Channel*, *Specific_Feature_Code*, *Customized_Dialing_Plan*, *Network_Busy*, *O_Called_Party_Busy*, *O_No_Answer*
	• Chat Lines	*Offhook_Delay*, *Shared_Interoffice_Trunk*, *PRI_B-Channel*
Agent	N00 Services	
	• Follow Me, Find Me, Do Not Disturb Me	*Offhook_Delay*, *Shared_Interoffice_Trunk*, *PRI_B-Channel*, *Specific_Digit_String*
	• Call Screening	*Offhook_Delay*, *Shared_Interoffice_Trunk*, *PRI_B-Channel*, *Specific_Digit_String*
	• Customized Call Branding	*Offhook_Delay*, *Shared_Interoffice_Trunk*, *PRI_B-Channel*, *Specific_Digit_String*
	• Determine carrier of N00 number	*Tollfree_Service*
	VPN Services	
	• OFFNET Overflow	*Specific_Feature_Code*, *Customized_Dialing_Plan*
	• Forced ONNET	*Specific_Feature_Code*, *Customized_Dialing_Plan*
Agent (cont)	• Alternate Billing Numbers	*Specific_Feature_Code*, *Customized_Dialing_Plan*

—continued—

Step 8: Choose the type of subscription (continued)

Subscription uses (continued)

Subscription type	NetworkBuilder service	Recommended triggers
	• Origination/Termination Screening	*Specific_Feature_Code*, *Customized_Dialing_Plan*
	• Customized Announcements	*Specific_Feature_Code*, *Customized_Dialing_Plan*
	• Universal Access Authorization	*Offhook_Delay*, *Shared_Interoffice_Trunk*, *PRI_B-Channel*, *Specific_Digit_String*
	• GVNS	*Specific_Feature_Code*, *Customized_Dialing_Plan*,
	• Caller ID Delivery	*Termination_Attempt*
	Customized Dialing Services	
	• Hotline Calls	*Off_Hook_Immediate*
	• Automatic queries with all digit collection, authorization, and route determination performed by the SCP (Billing data and routing information is returned to the UCS DMS-250 switch for call completion.)	*Off_Hook_Immediate*
	• Automatic Call Blocking	*Off_Hook_Immediate*
	Dial 1+ Services	
	• Intra-LATA Presubscription Screening	*O_Feature_Requested*

—continued—

Step 8: Choose the type of subscription (continued)

Subscription uses (continued)

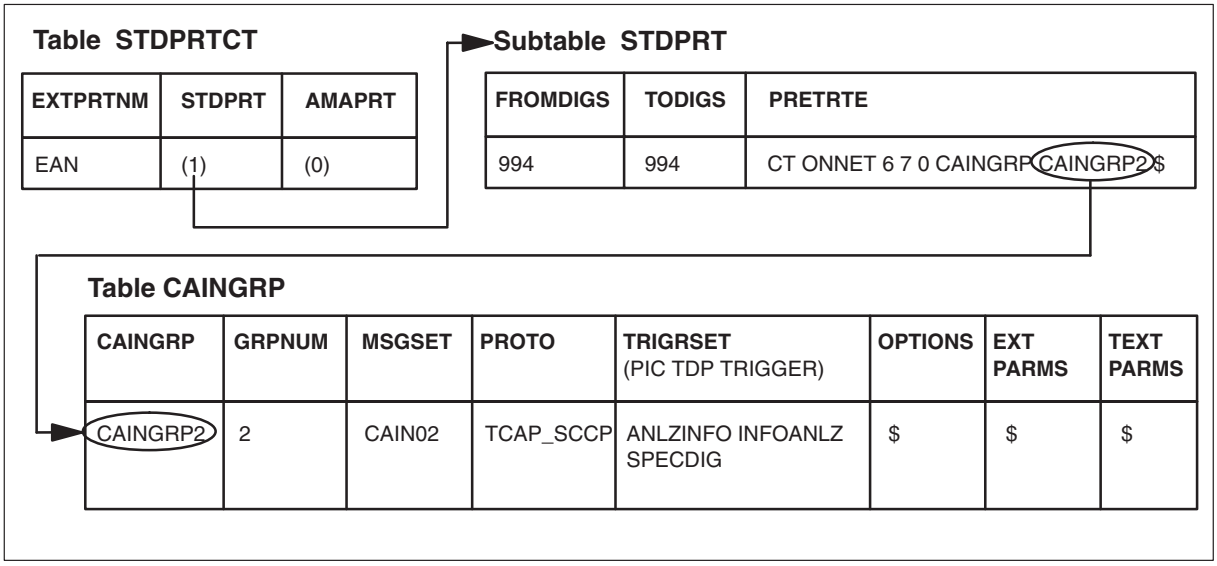
Subscription type	NetworkBuilder service	Recommended triggers
Agent (cont)	• CIC Routing and Branding	*Off_Hook_Immediate*, *O_Feature_Requested*, *Offhook_Delay*, *Shared_Interoffice_Trunk*, *PRI_B-Channel*, *Specific_Feature_Code*, *Customized_Dialing_Plan*, *Network_Busy*, *O_Called_Party_Busy*, *O_No_Answer*
	• Chat Lines	*Offhook_Delay*, *Shared_Interoffice_Trunk*, *PRI_B-Channel*
Office	Default feature set for an office. Used when no other subscription group applies to a call.	*Off_Hook_Immediate*, *O_Feature_Requested*, *Tollfree_Service*, *Offhook_Delay*, *Shared_Interoffice_Trunk*, *PRI_B-Channel*, *Specific_Feature_Code*, *Customized_Dialing_Plan*, *Specific_Digit_String*, *Network_Busy*, *O_Called_Party_Busy*, *O_No_Answer*
	Local Number Portability (LNP) • Ported Number Determination	*Office_Code*
—end—		

Step 8: Choose the type of subscription Address

Subscribing to originating NetworkBuilder services (address)

The following example shows the interaction between tables STDPRTCT and CAINGRP for originating CAIN groups.

Figure 2-10
Address subscription-originating CAINGRP interaction



Note: Note: Only the CT, IP, and IN pretranslator selectors are supported. NetworkBuilder software does not support resolution of address feature functionality, such as speed-dial numbers. The datafilled pretranslator is used for address collection on all reoriginations.

At the CI prompt

1 Enter table STDPRTCT.

2 Position on the pretranslator.

Note: In this example, an existing address subscribes to NetworkBuilder. New addresses can also subscribe to NetworkBuilder services.

3 Enter subtable STDPRT.

>SUB STDPRT

4 Position on the address digits requiring originating NetworkBuilder services.

Step 8: Choose the type of subscription Address (end)

5 Subscribe to NetworkBuilder using the following format:

>REP fromdigs todigs prertsel callfeat mindigs maxdigs nopredig stdprtopt
where

fromdigs is the “from” digits.
todigs is the “to” digits.
pretsel is the pretranslation route selector.
callfeat is the calltype features.
mindigs is the minimum number of digits.
maxdigs is the maximum number of digits.
nopredig is the number of prefix digits.
stdprtopt is a multiple-entry vector requiring 2 subfields: CAINGRP and CAIN_GROUP_TYPE, where
CAINGRP is the option.
CAIN_GROUP_TYPE is the CAIN group (defined in table CAINGRP) providing subscription services to the address.

Sample entry: **>REP 994 994 ct onnet 6 7 0 CAINGRP caingrp2 $**

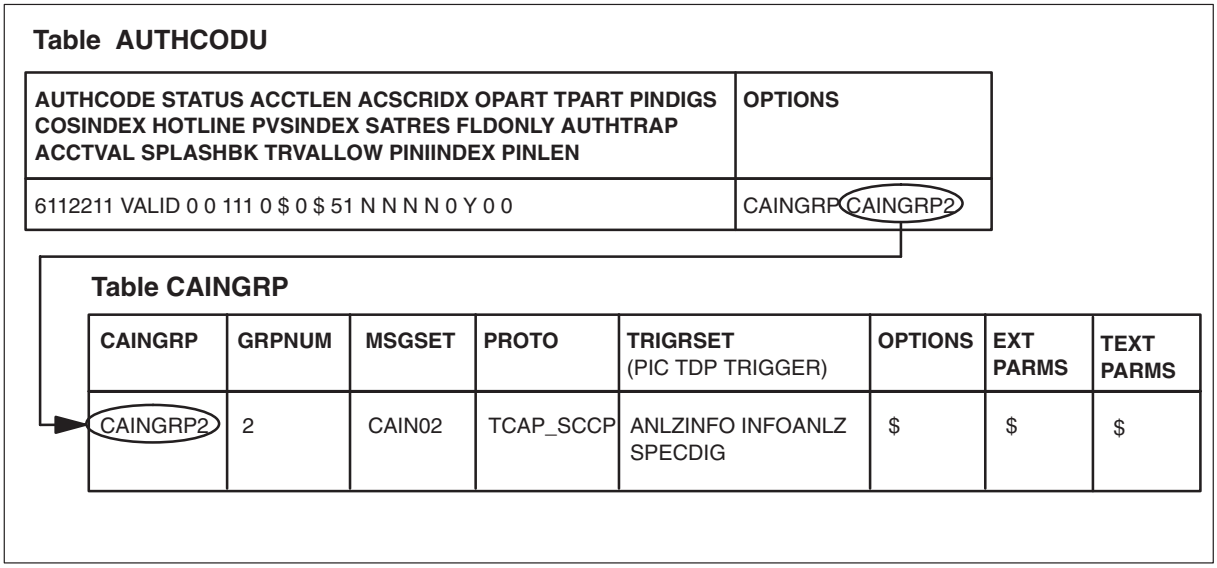
The address has subscribed to originating NetworkBuilder services.

Step 8: Choose the type of subscription Authorization codes

Subscribing to originating NetworkBuilder services (authorization codes)

The following example shows the interaction between tables AUTHCODU and CAINGRP for originating CAIN groups.

Figure 2-11
Authcode subscription-originating CAINGRP interaction



Note: The following procedure uses table AUTHCODU. Tables AUTHCDU2, AUTHCDU3, AUTHCDU4, AND AUTHCDU5 may be used instead.

At the CI prompt

1 Enter table AUTHCODU.

2 Position on the tuple requiring originating NetworkBuilder services.

Note: In this example, an existing authcode subscribes to NetworkBuilder. New authcodes can also subscribe to NetworkBuilder services.

3 Subscribe to NetworkBuilder using the following format:

>REP authcode status acctlen acscridx opart tpart pindigs mltcosid hotline pvsindex satres fldonly authtrap acctval splashbk trvallow pinindex pinlen options
where

authcode is the authcode requiring NetworkBuilder services.
status is the authcode status.
acctlen is the account code length.
acscridx is the account code screening index.

Step 8: Choose the type of subscription Authorization codes (end)

opart is the originating partition number.
tpart is the terminating partition number.
pindigs is the personal identification number digits.
mltcosid is the index into table MULTICOS.
hotline is the hotline number associated with the authcode.
pvsindex is the private speed index.
satres is the satellite restriction.
fldonly is the authcode filed only field.
authtrap is the authorization trap.
acctval is the account code validation.
splashbk is the splashback class.
trvallow is the traveling authcode allowed field.
pinindex is the personal identification number index.
pinlen is the personal identification number length.
options is multiple-entry vector requiring 2 subfields: CAINGRP and CAIN_GROUP_TYPE, where
CAINGRP is the option.
CAIN_GROUP_TYPE is the CAIN group (defined in table CAINGRP) providing subscription services to the authcode.

Sample entry: **>REP 6112211 valid 0 0 111 0 $ 0 $ 51 n n n n 0 y 0 0 CAINGRP caingrp2**

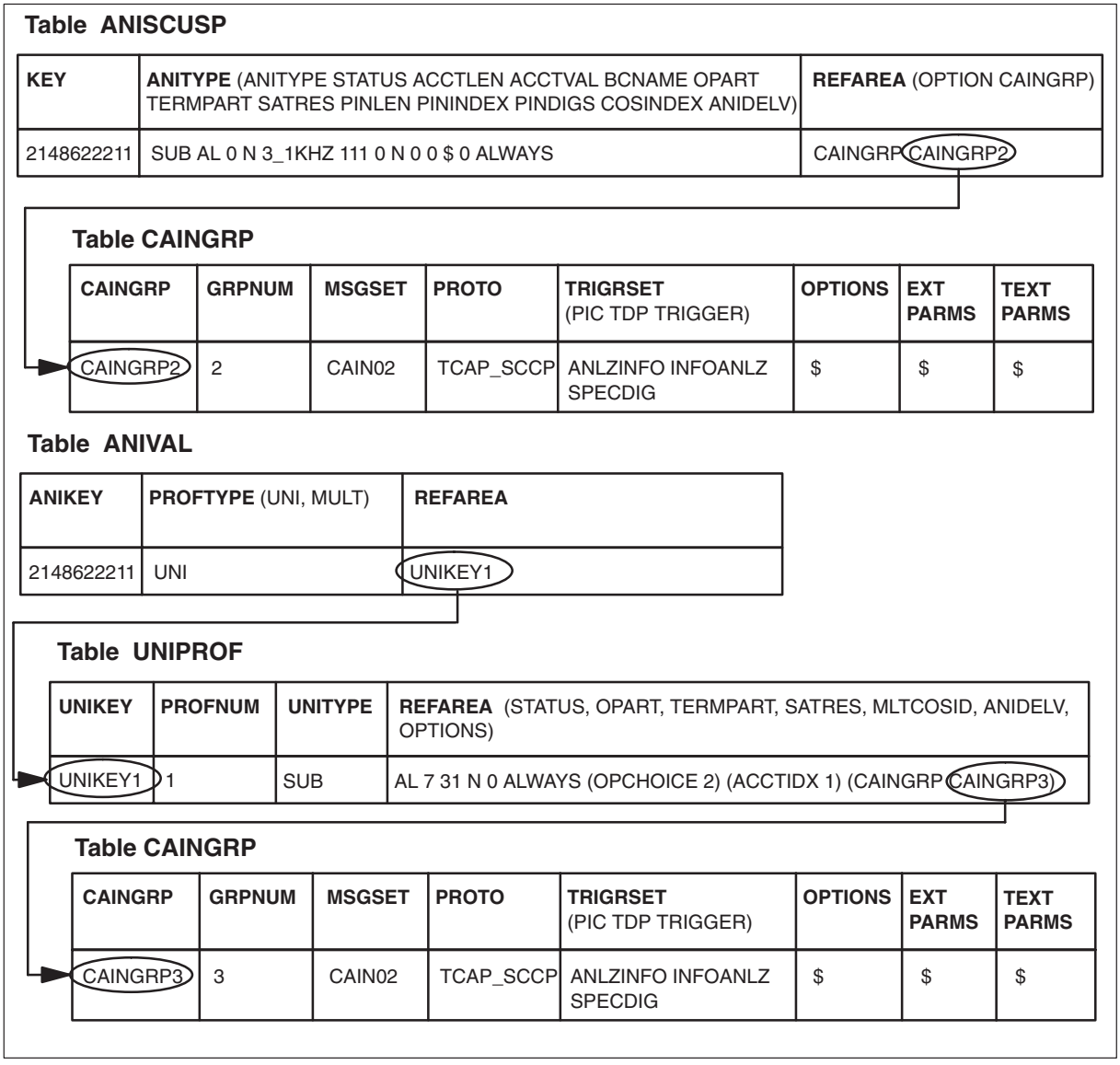
The authorization code has subscribed to originating NetworkBuilder services.

Step 8: Choose the type of subscription ANI (automatic number identification)

Subscribing to originating NetworkBuilder services (ANI)

The following example shows the interaction between tables ANISCUSP or ANIVAL and CAINGRP.

Figure 2-12
ANI subscription-originating CAINGRP interaction



Step 8: Choose the type of subscription ANI (automatic number identification) (continued)

Define the CAINGRP in table ANISCUSP

At the CI prompt

1 Enter table ANISCUSP.

2 Position on the tuple requiring originating NetworkBuilder services.

Note: In this example, an existing ANI subscribes to NetworkBuilder. New ANIs can also subscribe to NetworkBuilder services.

3 Subscribe to NetworkBuilder using the following format:

>REP key anitype refarea
where

key is the ANI digits requiring NetworkBuilder services.
anitype is the ANI type (NPA, NXX, SUB).
refarea contains multiple subfields including the OPTIONS subfield, where
CAINGRP is the option.
CAIN_GROUP_TYPE is the CAIN group (defined in table CAINGRP) providing subscription services to the ANI.

Sample entry: **>REP 2148622211 sub al 0 n 3_1khz 111 0 n 0 0 $ 0 always CAINGRP caingrp2**

The ANI has subscribed to originating NetworkBuilder services.

Define the CAINGRP in table ANIVAL

At the CI prompt

1 Enter table ANIVAL.

2 Position on the tuple requiring originating NetworkBuilder services.

Note: In this example, an existing ANI subscribes to NetworkBuilder. New ANIs can also subscribe to NetworkBuilder services.

3 Subscribe to NetworkBuilder using the following format:

>REP anikey proftype refarea
where

anikey is the ANI digits requiring NetworkBuilder services.
proftype is the profile type (UNI, MULT).
refarea is the profile index into table UNIPROF or table MULTPROF

Sample entry: **>REP 2148622211 uni unikey1**

Step 8: Choose the type of subscription ANI (automatic number identification) (end)

At the CI prompt

1 Enter table UNIPROF.

2 Position on the tuple indicated by table ANIVAL requiring originating NetworkBuilder services.

Note: In this example, an existing ANI subscribes to NetworkBuilder. New ANIs can also subscribe to NetworkBuilder services.

3 Subscribe to NetworkBuilder using the following format:

>REP key unikey profnum unitype refarea
where

unikey	is the profile name for the ANI.
profnum	is the unique profile number (0–4095).
unitype	is the ANI type (NPA, NXX, SUB).
refarea	contains STATUS subfield for ANI types NPA and NXX. Contains multiple subfields for ANI type SUB including the OPTIONS subfield, where
	CAINGRP is the option.
	CAIN_GROUP_TYPE is the CAIN group (defined in table CAINGRP) providing subscription services to the ANI.

Sample entry: **>REP unikey1 1 sub al 7 31 N 0 always opchoice 2 acctidx 1 CAINGRP caingrp3**

The ANI has subscribed to originating NetworkBuilder services.

Step 8: Choose the type of subscription
Non-PRI, non-AXXESS originating agent (continued)

Subscribing to originating NetworkBuilder services (DAL, FGD, SS7 Inter-IMT, SS7 Global-IMT originating agencies)

Note: CAIN group subscription is handled differently for AXXESS agents. Refer to *UCS DMS-250 CAIN/FlexDial Interactions* for more information.

The following example shows the interaction between tables TRKGRP and CAINGRP for originating CAIN groups.

Figure 2-13
Trunk subscription-originating CAINGRP interaction

Table TRKGRP

GRPKEY	**GRPTYP**	**TRAFSNO**	**PADGRP**	**NCCLS**	**GRPINFO** (COS DIR PRTNM SELSEQ ODSFLTR ORIGFLTR TDSCFLTR ANSWFLTR DIALTONE DIGSOUTP RETOFFHK TRAFCLS AUTHDIAL FASTIDGT OPART RECALLDT SNPA TIMEBIAS VAUTHFLD ZEROMPOS ZONE BCNAME AUTHFRST ADIN AIOD ONNETTRK PANIVAL PANIINFO TSUSR OPTION)
DAL221TWDTLS	DAL	30	NPDGP	NCIT	0 2W DAL IDL 16 7 16 16 S 7 NIL ID 0 7 111 MANUAL 214 0 6113311 RTE2 0 3_1KHZ Y 1 N Y NONE 00 160 (CAIN) (CAINGRP CAINGRP1)
EAN862C7LP00	EANT	40	NPDGP	NCOF	0 2W EAN MIDL 16 7 16 16 EAPT 10 4 214 UCS2EAEO NIL 214 111 MANUAL 0 RTE1 0 1 VOICE_DATA 160 MCCS REORGVAL ID24_ON (CAIN) (CAINGRP CAINGRP2)
EAN671TWMFWK	EANT	0	NPDGP	NCIT	0 2W EAN MIDL 16 7 16 16 EAPT 5 5 214 MILIDX NIL 214 111 MANUAL 0 NONE 0 1 3_1KHZ 160 (ALTTRTMT) (CAIN) (CAINGRP CAINGRP2)
IMT762TWMFWK	IMT	40	NPDGP	NCIT	0 2W IMT MIDL 16 7 16 16 UCS2UCS NIL C N NONE 4 ALWAYS I3PA 111 0 INTER N VOICE DATA NONE 4 160 214 0 (CAIN) (CAINGRP CAINGRP2)
IMT762TWMFWK	IMT	40	NPDGP	NCIT	0 2W IMT MIDL 16 7 16 16 UCS2UCS NIL C N NONE 4 ALWAYS ADDR 111 0 GLOBAL N Y VOICE_DATA NONE 4 160 214 0 (CAIN) (CAINGRP CAINGRP2)

Table CAINGRP

CAINGRP	**GRP NUM**	**MSGSET**	**PROTO**	**TRIGRSET** (PIC TDP TRIGGER)	**OPTIONS**	**EXT PARMS**	**TEXT PARMS**
CAINGRP1	1	CAIN02	TCAP_SCCP	O_NULL ORIGATT OFFHKIMM	$	$	$
CAINGRP2	2	CAIN02	TCAP_SCCP	COLLINFO INFOCOLL SIOTRK	$	$	$

Step 8: Choose the type of subscription Non-PRI, non-AXXESS originating agent (end)

At the CI prompt

1 Enter table TRKGRP.

2 Position on the tuple requiring originating NetworkBuilder services.

Note: In this example, an existing trunk group subscribes to NetworkBuilder. New trunk groups can also subscribe to NetworkBuilder services.

ATTENTION

The CAIN and CAINGRP options are independent. Therefore, subscription to a CAIN group using the CAINGRP option does not require the CAIN option. However, all calls requiring NetworkBuilder services must originate from a CAIN-provisioned agency.

3 Subscribe to NetworkBuilder using the following format:

>REP grpkey grptyp trafsno padgrp nccls grpinfo
where

grpkey is the CLLI of the trunk requiring NetworkBuilder services.
grptyp is the trunk group type.
trafsno is the traffic separation software number.
padgrp is the pad group name.
nccls is the no circuit class type used when routing list is exhausted.
grpinfo contains multiple subfields, including the OPTION subfield, where
CAINGRP is the option.
CAIN_GROUP_TYPE is the CAIN group (defined in table CAINGRP) providing subscription services to the trunk group.

DAL sample entry: **>REP dal221twdtls dal 30 npdgp ncit 0 2w dal midl 16 7 16 16 s 7 nil id 0 7 111 manual 213 0 6113311 rte2 0 3_1khz y 1 n y none 00 160 CAIN CAINGRP caingrp1**

FGD sample entry: **>REP ean671twmfwk eant 0 npdgp ncit 0 2w ean midl 16 7 16 16 eapt 5 5 214 milidx nil 214 111 manual 0 none 0 1 3_1khz 160 alttrtmt CAIN CAINGRP caingrp2**

SS7-Inter-IMT sample entry: **>REP imt762c7lp00 imt 40 npdgp ncit 0 2w imt midl 16 7 16 16 ucs2ucs nil c n none 4 always i3pa 111 0 inter n voice data none 4 160 214 0 CAIN caingrp caingrp2**

SS7-Global-IMT sample entry: **>REP imt762c7xx02 imt 40 npdgp ncit 0 2w imt midl 16 7 16 16 ucs2ucs nil c n none 4 always addr 111 0 global n y cgpa n voice_data none 4 160 214 0 CAIN caingrp caingrp2**

The agent has subscribed to originating and terminating NetworkBuilder services.

Step 8: Choose the type of subscription
Non-PRI, non-AXXESS terminating agent (continued)

Subscribing to terminating NetworkBuilder services (DAL, FGB, FGC, FGD, IMT terminating agencies)

The following example shows the interaction between tables TRKGRP and CAINGRP for subscription to terminating CAIN groups by non-PRI and non-AXXESS terminating agents.

Figure 2-14
Trunk subscription-terminating CAINGRP interaction

Table TRKGRP

GRPKEY	GRPTYP	TRAFSNO	PADGRP	NCCLS	GRPINFO (COS DIR PRTNM SELSEQ ODSFLTR ORIGFLTR TDSCFLTR ANSWFLTR DIALTONE DIGSOUTP RETOFFHK TRAFCLS AUTHDIAL FASTIDGT OPART RECALLDT SNPA TIMEBIAS VAUTHFLD ZEROMPOS ZONE BCNAME AUTHFRST ADIN AIOD ONNETTRK PANIVAL PANIINFO TSUSR OPTION)
DAL221TWDTLS	DAL	30	NPDGP	NCIT	0 2W DAL IDL 16 7 16 16 S 7 NIL ID 0 7 111 MANUAL 214 0 6113311 RTE2 0 3_1KHZ Y 1 N Y NONE 00 160 (TCAIN) (TCAINGRP TERMGRP1)
ONT544TWDTWK	ONAT	50	NPDGP	NCOF	0 2W ONAT MIDL 16 7 16 16 7 7 OT 0 0 214 S 7 FGBC 111 MANUAL 214 0 6112211 NONE 0 Y 1 NONE 00 IMMEDIATE VOICE_DATA 160 (TCAIN) (TCAINGRP TERMGRP1)
ONL420OGDPGO	ONAL	127	NPDGP	NCOF	0 OG ONL MIDL 16 7 16 214 1 N ON 0 S 5 111 MANUAL 214 0 6112211 NONE 0 Y 1 NONE 00 160 (TCAIN) (TCAINGRP TERMGRP1)
EAN671TWMFWK	EANT	0	NPDGP	NCIT	0 2W EAN MIDL 16 7 16 16 EAPT 5 5 214 MILIDX NIL 214 111 MANUAL 0 NONE 0 1 3_1KHZ 160 (ALTTRTMT) (TCAIN) (TCAINGRP TERMGRP1)
IMT762TWMFWK	IMT	40	NPDGP	NCIT	0 2W IMT MIDL 16 7 16 16 UCS2UCS NIL C N NONE 4 ALWAYS ADDR 111 0 GLOBAL N Y VOICE_DATA NONE 4 160 214 0 (TCAIN) (TCAINGRP TERMGRP1)

Table CAINGRP

CAINGRP	GRP NUM	MSGSET	PROTO	TRIGRSET (PIC TDP TRIGGER)	OPTIONS	EXT PARMS	TEXT PARMS
TERMGRP1	1	CAIN02	TCAP_SCCP	T_NULL TERMATT TERMATT	$	$	$

Step 8: Choose the type of subscription Non-PRI, non-AXXESS terminating agent (continued)

At the CI prompt

1 Enter table TRKGRP.

2 Position on the tuple requiring terminating NetworkBuilder services.

Note: In this example, an existing trunk group subscribes to NetworkBuilder. New trunk groups can also subscribe to NetworkBuilder services.

ATTENTION

The TCAIN and TCAINGRP options are independent. Therefore, subscription to a CAIN group using the TCAINGRP option does not require the TCAIN option. However, all calls requiring terminating NetworkBuilder services must terminate to a TCAIN-provisioned agency.

3 Subscribe to NetworkBuilder using the following format:

>REP grpkey grptyp trafsno padgrp nccls grpinfo
where

grpkey is the CLLI of the trunk requiring NetworkBuilder services.
grptyp is the trunk group type.
trafsno is the traffic separation software number.
padgrp is the pad group name.
nccls is the no circuit class type used when routing list is exhausted.
grpinfo contains multiple subfields, including the OPTION subfield, where
TCAINGRP is the option.
CAIN_GROUP_TYPE is the CAIN group (defined in table CAINGRP) providing subscription services to the trunk group.

DAL sample entry: **>REP dal221twdtls dal 30 npdgp ncit 0 2w dal midl 16 7 16 16 s 7 nil id 0 7 111 manual 213 0 6113311 rte2 0 3_1khz y 1 n y none 00 160 TCAIN TCAINGRP termgrp1**

FGB sample entry: **>REP ont544twdtwk onat 50 npdgp ncof 0 2w ont midl 16 7 16 16 7 7 ot 0 0 214 s 7 fgbc 111 manual 214 0 6112211 none 0 y 1 none 00 immediate voice_data 160 TCAIN TCAINGRP termgrp1**

FGC sample entry: **>REP on420ogdpsz onal 127 npdgp ncof 0 og onl midl 16 7 16 214 1 n on 0 s 5 111 manual 214 0 6112211 none 0 y 1 none 00 160 TCAIN TCAINGRP termgrp1**

FGD sample entry: **>REP ean671twmfwk eant 0 npdgp ncit 0 2w ean midl 16 7 16 16 eapt 5 5 214 milidx nil 214 111 manual 0 none 0 1 3_1khz 160 alttrtmt TCAIN TCAINGRP termgrp1**

Step 8: Choose the type of subscription Non-PRI, non-AXXESS terminating agent (end)

IMT sample entry: **>REP imt762c7xx02 imt 40 npdgp ncit 0 2w imt midl 16 7 16 16 ucs2ucs nil c n none 4 always addr 111 0 global n y cgpa n voice_data none 4 160 214 0 TCAIN TCAINGRP termgrp1**

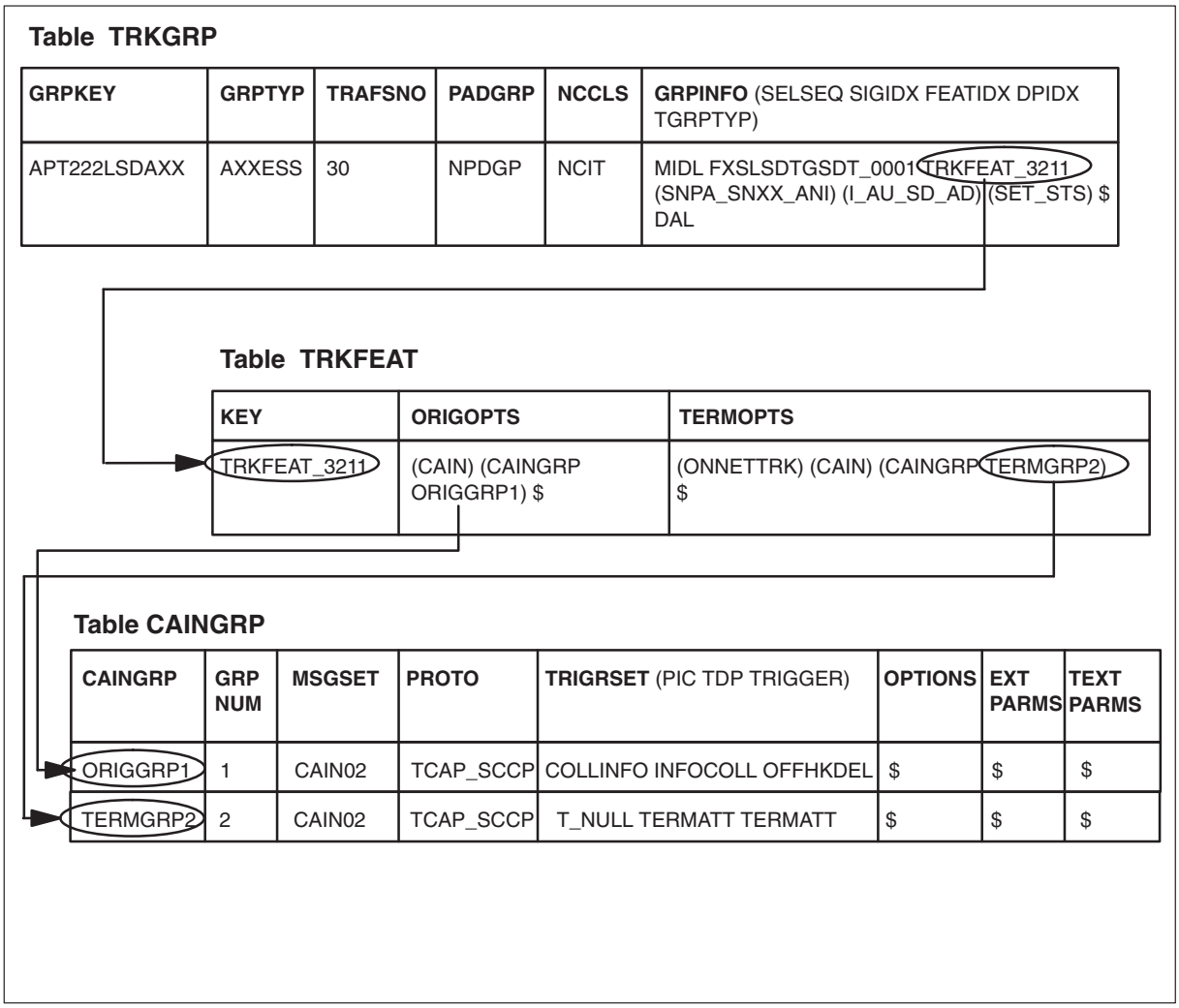
The agent has subscribed to terminating NetworkBuilder services.

Step 8: Choose the type of subscription AXXESS originating and terminating agent

Subscribing to originating and terminating NetworkBuilder services (AXXESS agency)

The following example shows the interaction between tables TRKGRP, TRKFEAT and CAINGRP for subscription to CAIN groups by AXXESS agents.

Figure 2-15
Trunk subscription – CAINGRP interaction



Step 8: Choose the type of subscription AXXESS originating and terminating agent (end)

At the CI prompt

1 Enter table TRKFEAT.

2 Position on feature set requiring NetworkBuilder originating and/or terminating call model capabilities.

Note: In this example, an existing trunk group subscribes to NetworkBuilder. New trunk groups can also subscribe to NetworkBuilder services.

ATTENTION

The CAIN and CAINGRP options are independent. Therefore, subscription to a CAIN group using the CAINGRP option does not require the CAIN option. However, calls on AXXESS agents requiring NetworkBuilder services must originate or terminate to a CAIN-provisioned agent.

3 Change the tuple to reflect NetworkBuilder capability and add the CAINGRP option by using the following format:

>REP key origopts termopts
where

key is the name of the feature set (up to 16 alphanumeric characters) indexed by table TRKGRP.

origopts is the option(s) you want to enable on the originations (CAINGRP).

termopts is the option(s) you want to enable on the terminations (CAINGRP).

AXXESS sample entry: **>REP trkfeat_2323 CAIN CAINGRP origgrp1 $ onnettrk CAIN CAINGRP termgrp2**

The AXXESS agent has subscribed to originating and terminating NetworkBuilder services.

Step 8: Choose the type of subscription PRI originating agent

Subscribing to originating NetworkBuilder services (PRI call attributes)

The following example shows the interaction between tables CALLATTR and CAINGRP for originating CAIN groups.

Figure 2-16
PRI call attribute subscription-originating CAINGRP interaction

Table CALLATTR

CATTRIDX	CUSTOMER	PRTNM	COS	ZEROMPOS	CUSTAREA (OPART BILLTYP ADIN VAUTHFLD AUTHDIAL DEFCLID OPTION)
21	UCS	PRI	0	NONE	611 AUTH 1 6112211 0 5026112211 (ANIDELV) (CAIN) (CAINGRP CAINGRP3)

Table CAINGRP

CAINGRP	GRP NUM	MSGSET	PROTO	TRIGRSET (PIC TDP TRIGGER)	OPTIONS	EXT PARMS	TEXT PARMS
CAINGRP3	3	CAIN02	TCAP_SCCP	COLLINFO INFOCOLL PRIBCHNL	$	$	$

At the CI prompt

1 Enter table CALLATTR.

2 Position on the tuple requiring originating NetworkBuilder services.

Note 1: In this example, an existing PRI call attribute subscribes to NetworkBuilder. New call attributes can also subscribe to originating NetworkBuilder services.

Note 2: Table TRKGRP's LTID subfield (field GRPINFO) indexes table LTCALLS' CALLATTR field, which in turn indexes table CALLATTR. Figure 2-8 on page 2-125 shows the relationship between tables CALLATTR, LTCALLS, and TRKGRP.

ATTENTION

The CAIN and CAINGRP options are independent. Therefore, subscription to a CAIN group using the CAINGRP option does not require the CAIN option. However, all calls requiring NetworkBuilder services must originate from a CAIN-provisioned agency.

Step 8: Choose the type of subscription PRI originating agent (end)

3 Subscribe to NetworkBuilder by using the following format:

>REP cattridx customer prtnm cos zerompos custarea
where

cattridx is the index datafilled in table LTCALLS' CALLATTR field.
customer is the customer type.
prtnm is the pretranslator name (index into table STDPRTCT).
cos is the class of service screening (index into table TRKCOS).
zerompos is the route choice (index into table POSITION).
custarea contains multiple subfields, including the OPTION subfield, where
CAINGRP is the option.
CAIN_GROUP_TYPE is the CAIN group (defined in table CAINGRP) providing subscription services to the call attribute.

Sample entry: **>REP 21 ucs pri 0 none 611 auth 1 6112211 0 5026112211 anidelv CAIN CAINGRP caingrp3**

The PRI agents accessing this call attribute have subscribed to originating NetworkBuilder services.

Step 8: Choose the type of subscription PRI terminating agent

Subscribing to terminating NetworkBuilder services (PRI call attributes)

The following example shows the interaction between tables CALLATTR and CAINGRP for terminating CAIN groups.

Figure 2-17
PRI call attribute subscription-terminating CAINGRP interaction

Table CALLATTR

CATTRIDX	CUSTOMER	PRTNM	COS	ZEROMPOS	CUSTAREA (OPART BILLTYP ADIN VAUTHFLD AUTHDIAL DEFCLID OPTION)
21	UCS	PRI	0	NONE	611 AUTH 1 6112211 0 5026112211 (ANIDELV) (TCAIN) (TCAINGRP TERMGRP2)

Table CAINGRP

CAINGRP	GRP NUM	MSGSET	PROTO	TRIGRSET (PIC TDP TRIGGER)	OPTIONS	EXT PARMS	TEXT PARMS
TERMGRP2	3	CAIN02	TCAP_SCCP	T_NULL TERMATT TERMATT	$	$	$

At the CI prompt

1 Enter table CALLATTR.

2 Position on the tuple requiring terminating NetworkBuilder services.

Note 3: In this example, an existing PRI call attribute subscribes to NetworkBuilder. New call attributes can also subscribe to terminating NetworkBuilder services.

Note 4: Table TRKGRP's LTID subfield (field GRPINFO) indexes table LTCALLS' CALLATTR field, which in turn indexes table CALLATTR. Figure 2-8 on page 2-125 shows the relationship between tables CALLATTR, LTCALLS, and TRKGRP.

ATTENTION

The TCAIN and TCAINGRP options are independent. Therefore, subscription to a CAIN group using the TCAINGRP option does not require the TCAIN option. However, all calls requiring terminating NetworkBuilder services must terminate to a TCAIN-provisioned agency.

Step 8: Choose the type of subscription PRI terminating agent (end)

3 Subscribe to NetworkBuilder by using the following format:

>REP cattridx customer prtnm cos zerompos custarea
where

cattridx is the index datafilled in table LTCALLS' CALLATTR field.
customer is the customer type.
prtnm is the pretranslator name (index into table STDPRTCT).
cos is the class of service screening (index into table TRKCOS).
zerompos is the route choice (index into table POSITION).
custarea contains multiple subfields, including the OPTION subfield, where
TCAINGRP is the option.
CAIN_GROUP_TYPE is the CAIN group (defined in table CAINGRP) providing subscription services to the call attribute.

Sample entry: **>REP 21 ucs pri 0 none 611 auth 1 6112211 0 5026112211 anidelv TCAIN TCAINGRP termgrp2**

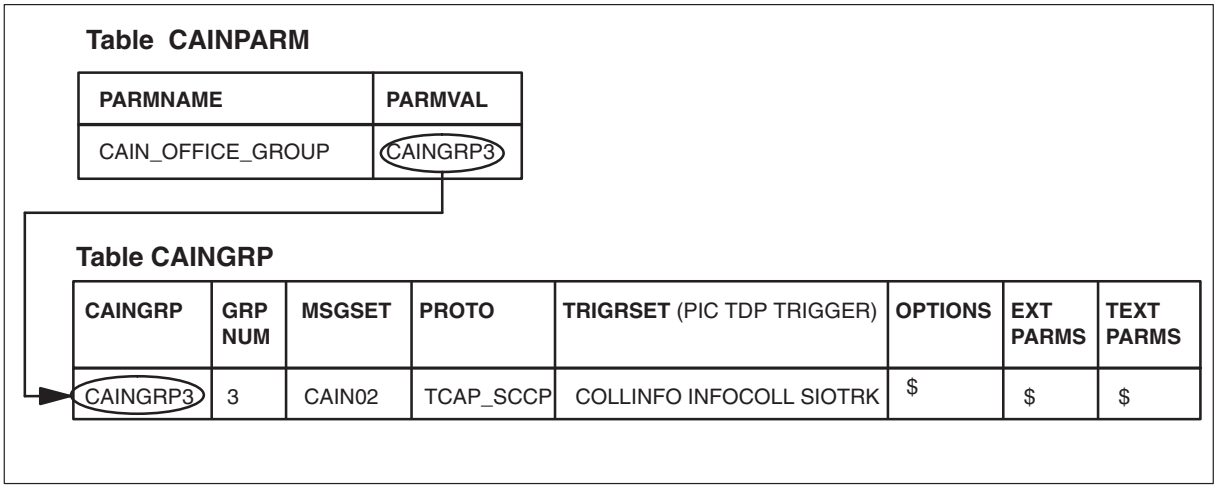
The PRI agents accessing this call attribute have subscribed to terminating NetworkBuilder services.

Step 8: Choose type of subscription
Office (end)

Subscribing to originating NetworkBuilder services (office)

The following example shows the interaction between tables TRKGRP and CAINGRP for originating CAIN groups.

Figure 2-18
Office subscription-originating CAINGRP interaction



At the CI prompt

1 Enter table CAINPARM.

2 Replace the defined parameter by typing:

>REP CAIN_OFFICE_GROUP cain_group
where

cain_group is an originating CAIN group defined in table CAINGRP providing subscription services to the office.

Sample entry: **>REP CAIN_OFFICE_GROUP caingrp3**

The office has subscribed to originating NetworkBuilder services.

Step 9: Define the trigger criteria

ATTENTION

Triggers require the corresponding CAIN0500-series of SOC options. The *Office_Code* trigger requires CAIN0700 SOC option. The SS7 Inter-IMT and SS7 Global-IMT agents require CAIN0604 and CAIN0605 SOC options respectively, in order to trigger. Refer to Volume 5, Chapter 5, "NetworkBuilder SOC functionality," for more information.

When NetworkBuilder call processing encounters a TDP, the following steps have been completed:

1 Data required to process the call as a NetworkBuilder call (at the current PIC, TDP, and trigger) has been gathered.

2 A subscriber (authorization code or ANI), address, originating agency, or office is recognized as a NetworkBuilder subscriber.

3 Table CAINGRP subscribes to the current trigger set.

Note: A trigger set identifies the PIC, TDP, and trigger.

Trigger evaluations

NetworkBuilder evaluates call data against the triggering criteria defined in the associated trigger table (identified by the trigger set in table CAINGRP). Trigger tables identify the following:

- conditions required to meet trigger criteria
- action to take once trigger criteria is met
- error action, when applicable
- available triggering options

The following CAIN triggers are supported in the NetworkBuilder originating call model when the protocol is CAIN02:

- *Off_Hook_Immediate* (associated trigger table: OFFHKIMM)
- *O_Feature_Requested* (associated trigger table: OFTRREQ)
- *Offhook_Delay* (associated trigger table: OFFHKDEL)
- *Shared_Interoffice_Trunk* (associated trigger table: SIOTRK)
- *PRI_B-Channel* (associated trigger table: PRIBCHNL)
- *Specific_Feature_Code* (associated trigger table: SPECFEAT)
- *Customized_Dialing_Plan* (associated trigger table: CUSTDP)

Step 9: Define the trigger criteria (continued)

- *Specific_Digit_String* (associated trigger table: SPECDIG)
- *Office_Code* (associated trigger table: OFFCCODE)
- *Network_Busy* (associated trigger table: NETBUSY)
- *O_Called_Party_Busy* (associated trigger table: OCLDBUSY)
- *O_No_Answer* (associated trigger table: ONOANSWR)
- *O_IEC_Reorigination* (associated trigger table: OIECREO)

The following Bellcore *TR-NWT-000533* IN/1 trigger is supported in the NetworkBuilder originating call model when the protocol is IN/1:

- *Tollfree_Service* (associated trigger table: TOLLFREE)

The following NetworkBuilder trigger is supported in the terminating call model when the protocol is CAIN02:

- *Termination_Attempt* (associated trigger table: TERMATT)

The following digit types are supported in the NetworkBuilder originating call model:

- INFO (information digits)
- ADIN (authcode database index)
- ANI (automatic number identification)
- CLID (calling line identification)
- XLAADDR (address digits for the call after pretranslations are performed)
- ADDR (originally collected address digits)
- CIC (carrier identification code)

The following table lists each trigger and the digit types available for trigger criteria evaluation.

Table 2-10
Digit types for trigger criteria evaluation

Trigger	INFO	ADIN	ANI/CLID	XLAADDR	ADDR	CIC
OFFHKIMM						
OFTRREQ	X		X		X	X
TOLLFREE (note 2)					X	
OFFHKDEL	X	X	X		X	X
SIOTRK	X	X	X		X	X
PRIBCHNL		X	X		X	
SPECFEAT				X	X	
CUSTDP				X	X	
SPECDIG	X	X	X	X	X	X
OFFCCODE (note 2)				X		
NETBUSY	X		X	X	X	X
OCLDBUSY	X		X	X	X	X
ONOANSWR	X		X	X	X	X
OIECREO	X	X	X		X	X
TERMATT				X		

Note 1: Digits are evaluated in the order shown (INFO, ADIN, ANI/CLID, XLAADDR, ADDR, CIC).
Note 2: The digit type is implied but not provisioned.

—end—

Trigger tables

Define the trigger criteria in the trigger tables. By datafilling the trigger tables, you are instructing the switch on how to proceed with processing the call.

OFFHKIMM

Table OFFHKIMM (Off_Hook_Immediate) is used when trigger *Off_Hook_Immediate* is evaluated.

Step 9: Define the trigger criteria (continued)

Trigger criteria is not used for *Off_Hook_Immediate*. Calls trigger when a call subscribes to a CAIN group that enables *Off_Hook_Immediate.* Call processing performs the datafilled ACTION (valid actions are: IGNORE, BLOCK, QUERY, QUERYSCU, LEAVE_TDP, and CONT_NOTRIG) for the tuple (datafilled in table OFFHKIMM) associated with the call.

If a fatal application error occurs once a query is started, call processing uses the datafilled error action. Valid error actions are: ROUTE and TREAT. For more information, refer to Chapter 3, "O_Null PIC."

OFTRREQ

Table OFTRREQ (O_Feature_Requested) is used when *O_Feature_Requested* is evaluated.

The trigger criteria consists of a digit type (information digits [INFO], automatic number identification [ANI], address [ADDR], or carrier identification code [CIC]) and a defined digit range. NetworkBuilder evaluates the call against the datafilled trigger criteria. When the criteria is met, call processing performs the datafilled trigger action (valid actions are: IGNORE, BLOCK, FEAT, LEAVE_TDP, and CONT_NOTRIG). For more information, refer to Chapter 4, "Collect_Information PIC."

Note: Table OFTRREQ does not define error actions. The switch applies AINF treatment when an error occurs.

TOLLFREE

Table TOLLFREE (Tollfree_Service) is used when the Bellcore *TR-NWT-000533* defined IN/1 *Tollfree_Service* trigger is evaluated.

The trigger criteria consists of an address [ADDR] (implied but not provisioned) in a defined digit range. NetworkBuilder evaluates the call against the datafilled trigger criteria. When the criteria is met, call processing performs the datafilled trigger action (valid actions are: IGNORE, BLOCK, QUERY). For more information, refer to Chapter 4, "Collect_Information PIC."

Note: Table TOLLFREE does not define error actions. The switch applies AINF treatment when an error occurs.

OFFHKDEL

Table OFFHKDEL (Offhook_Delay) is used when trigger *Offhook_Delay* is evaluated.

The trigger criteria consists of a digit type (information digits [INFO], authcode database index [ADIN], automatic number identification [ANI],

address [ADDR], or carrier identification code [CIC]) and a defined digit range. NetworkBuilder evaluates the call against the datafilled trigger criteria. When the criteria is met, call processing performs the datafilled trigger action (valid actions are: IGNORE, BLOCK, QUERY, LEAVE_TDP, and CONT_NOTRIG). For more information, refer to Chapter 4, "Collect_Information PIC."

Note: Table OFFHKDEL does not define error actions. The switch applies AINF treatment when an error occurs.

SIOTRK

Table SIOTRK (Shared_Interoffice_Trunk) is used when trigger *Shared_Interoffice_Trunk* is evaluated.

The trigger criteria consists of a digit type (information digits [INFO], authcode database index [ADIN], automatic number identification [ANI], address [ADDR], or carrier identification code [CIC]) and a defined digit range. NetworkBuilder evaluates the call against the datafilled trigger criteria. When the criteria is met, call processing performs the datafilled trigger action (valid actions are: IGNORE, BLOCK, QUERY, QUERYSCU, LEAVE_TDP, and CONT_NOTRIG). For more information, refer to Chapter 4, "Collect_Information PIC."

Note: Table SIOTRK does not define error actions. The switch applies AINF treatment when an error occurs.

PRIBCHNL

Table PRIBCHNL (PRI_B-Channel) is used when trigger *PRI_B-Channel* is evaluated.

The trigger criteria consists of a digit type (authcode database index [ADIN], calling line identifier [CLID], or address [ADDR]) and a defined digit range. NetworkBuilder evaluates the call against the datafilled trigger criteria. When the criteria is met, call processing performs the datafilled trigger action (valid actions are: IGNORE, BLOCK, QUERY, QUERYSCU, LEAVE_TDP, and CONT_NOTRIG). For more information, refer to Chapter 4, "Collect_Information PIC."

Note: Table PRIBCHNL does not define error actions. The switch applies AINF treatment when an error occurs.

SPECFEAT

Table SPECFEAT (Specific_Feature_Code) is used when trigger *Specific_Feature_Code* is evaluated.

Step 9: Define the trigger criteria (continued)

The trigger criteria consists of a digit type (translated address [XLAADDR] or address [ADDR]) and a defined digit range. NetworkBuilder evaluates the call against the datafilled trigger criteria. When the criteria is met, call processing performs the datafilled trigger action (valid actions are: IGNORE, BLOCK, QUERY, LEAVE_TDP, and CONT_NOTRIG).

When the datafilled ACTION is QUERY, call processing uses the error action when fatal application errors occur. Valid error actions are: ROUTE and TREAT. For more information, refer to Chapter 5, "Analyze_Information PIC."

CUSTDP

Table CUSTDP (Customized_Dialing_Plan) is used when trigger *Customized_Dialing_Plan* is evaluated.

The trigger criteria consists of a digit type (translated address [XLAADDR] or address [ADDR]) and a defined digit range. NetworkBuilder framework evaluates the call against the datafilled trigger criteria. When the criteria is met, call processing performs the datafilled trigger action (valid actions are: IGNORE, BLOCK, QUERY, LEAVE_TDP, and CONT_NOTRIG).

When the datafilled ACTION is QUERY, call processing uses the error action when fatal application errors occur. Valid error actions are: ROUTE and TREAT. For more information, refer to Chapter 5, "Analyze_Information PIC."

SPECDIG

Table SPECDIG (Specific_Digit_String) is used when trigger *Specific_Digit_String* is evaluated.

The trigger criteria consists of a digit type (information digits [INFO], authcode database index [ADIN], automatic number identification [ANI], translated address [XLAADDR], address [ADDR], or carrier identification code [CIC]) and a defined digit range. NetworkBuilder evaluates the call against the datafilled trigger criteria. When the criteria is met, call processing performs the datafilled trigger action (valid actions are: IGNORE, BLOCK, QUERY, LEAVE_TDP, and CONT_NOTRIG).

When the datafilled ACTION is QUERY, call processing uses the error action when fatal application errors occur. Valid error actions are: ROUTE and TREAT. For more information, refer to Chapter 5, "Analyze_Information PIC."

Step 9: Define the trigger criteria (continued)

OFFCCODE

Table OFFCCODE (Office_Code) is used when trigger *Office_Code* is evaluated.

The trigger criteria consists of a translated address [XLAADDR] (implied but not datafilled) in a defined digit range. NetworkBuilder evaluates the call against the datafilled trigger criteria. When the criteria is met, call processing performs the datafilled trigger action (valid actions are: IGNORE, BLOCK, QUERY, LEAVE_TDP, and CONT_NOTRIG).

Note: Table OFFCCODE does not define error actions, the switch defaults to ROUTE. For more information, refer to Chapter 5, "Analyze_Information PIC."

NETBUSY

Table NETBUSY (Network_Busy) is used when trigger *Network_Busy* is evaluated.

The trigger criteria consists of the routing status (routes available [RTEAVAIL], routes done [RTESDONE], terminating route generalized no circuit [TERMRTE_GNCT]), a digit type (information digits [INFO], automatic number identification [ANI], translated address [XLAADDR], address [ADDR], or carrier identification code [CIC]) and a defined digit range. NetworkBuilder evaluates the call against the datafilled trigger criteria. When the criteria is met, call processing performs the datafilled trigger action (valid actions are: IGNORE, QUERY, NEXTRTE, NEXTCNRTE, LEAVE_TDP, and CONT_NOTRIG). For more information, refer to Chapter 6, "Select_Route PIC."

Note: Table NETBUSY does not define error actions. The switch applies AINF when an error occurs.

OCLDBUSY

Table OCLDBUSY (O_Called_Party_Busy) is used when trigger *O_Called_Party_Busy* is evaluated.

The trigger criteria consists of the routing status (routes available [RTEAVAIL] or routes done [RTESDONE]), a digit type (information digits [INFO], automatic number identification [ANI], translated address [XLAADDR], address [ADDR], or carrier identification code [CIC]) and a defined digit range. NetworkBuilder evaluates the call against the datafilled trigger criteria. When the criteria is met, call processing performs the datafilled trigger action (valid actions are: IGNORE, QUERY, NEXTRTE, NEXTCNRTE, LEAVE_TDP, and CONT_NOTRIG). For more information, refer to Chapter 7, "Send_Call PIC."

Step 9: Define the trigger criteria (continued)

Note: Table OCLDBUSY does not define error actions. The switch applies AINF when an error occurs.

ONOANSWR

Table ONOANSWR (O_No_Answer) is used when trigger *O_No_Answer* is evaluated.

The trigger criteria consists of the routing status (routes available [RTEAVAIL] or routes done [RTESDONE]), a digit type (information digits [INFO], automatic number identification [ANI], translated address [XLAADDR], address [ADDR], or carrier identification code [CIC]) and a defined digit range. NetworkBuilder evaluates the call against the datafilled trigger criteria. When the criteria is met, call processing performs the datafilled trigger action (valid actions are: IGNORE, QUERY, NEXTRTE, NEXTCNRTE, LEAVE_TDP, and CONT_NOTRIG). For more information, refer to Chapter 8, "O_Alerting PIC."

Note: Table ONOANSWR does not define error actions. The switch applies AINF when an error occurs.

OIECREO

Table OIECREO (O_IEC_Reorigination) is used when trigger *O_IEC_Reorigination* is evaluated.

The trigger criteria consists of a digit type (information digits [INFO], authcode database index [ADIN], automatic number identification [ANI], address [ADDR], or carrier identification code [CIC]) and a defined digit range. NetworkBuilder evaluates the call against the datafilled trigger criteria. When the criteria is met, call processing performs the datafilled trigger action (valid actions are: IGNORE, BLOCK, QUERY, LEAVE_TDP, CONT_NOTRIG, and COLLINFO). For more information, refer to Chapter 9, "O_Active and O_Suspended PICs."

Note: Table OIECREO does not define error actions. The switch applies AINF when an error occurs.

TERMATT

Table TERMATT (Termination_Attempt) is used when trigger *Termination_Attempt* is evaluated in the terminating call model.

The trigger criteria consists of a digit type (translated address [XLADDR]) and a defined digit range. NetworkBuilder evaluates the call against the datafilled trigger criteria. When the criteria is met, call processing performs the datafilled trigger action (valid actions are: IGNORE, BLOCK, QUERY,

LEAVE_TDP, and CONT_NOTRIG). For more information, refer to Chapter 10, "T_Null PIC."

When the datafilled ACTION is QUERY, call processing uses the error action when fatal application errors occur. Valid error actions are: ROUTE and TREAT. For more information, refer to Chapter 10, "T_Null PIC."

Trigger actions

You can datafill the following trigger actions:

- IGNORE – NetworkBuilder continues checking the remaining subscription methods for the current trigger. If datafill does not enable the trigger, call processing continues through the call model.

  You can datafill IGNORE in any trigger table.

- BLOCK – The switch applies AINF treatment to the call.

  Note: For TOLLFREE, the TRTMT option is checked. The provisioned treatment is applied to the call, or AINF is applied to the call if a treatment is not provisioned.

  You can datafill BLOCK in the following trigger tables: OFFHKIMM, OFTRREQ, OFFHKDEL, SIOTRK, PRIBCHNL, SPECFEAT, TOLLFREE, CUSTDP, SPECDIG, OFFCCODE, OIECREO, and TERMATT.

- QUERY – The switch builds the appropriate query message and sends it to the SCP. The SCP analyzes the call and assists the switch with call processing.

  You can datafill QUERY in the following trigger tables: OFFHKIMM, TOLLFREE, OFFHKDEL, SIOTRK, PRIBCHNL, SPECFEAT, CUSTDP, SPECDIG, OFFCCODE, NETBUSY, OCLDBUSY, ONOANSWR, OIECREO, and TERMATT.

- FEAT – Invokes a feature processor, allowing data validation services, and queries the SCP (through an **`O_Feature_Requested`** query). The SCP analyzes the call and assists the switch with call processing.

  You can datafill FEAT in table OFTRREQ.

- QUERYSCU – This action is only used by the programmable service node software. Refer to the *UCS DMS-250 Programmable Service Node (PSN) Application Guide* for more information.

  You can datafill QUERYSCU in the following trigger tables: OFFHKIMM, SIOTRK, PRIBCHNL, CUSTDP and SPECDIG.

- NEXTRTE – Call processing routes using the next routing option according to the following precedences:

Step 9: Define the trigger criteria (continued)

1. The switch route advances and attempts the next route available in the route list.

2. The switch performs a CAIN routing parameter advance and attempts to route accordingly.

Note: Provision NEXTRTE only when the criteria is datafilled as RTEAVAIL or TERMRTE_GNCT.

You can datafill NEXTRTE in table NETBUSY, OCLDBUSY, and ONOANSWR.

- NEXTCNRTE – Call processing attempts to route using the next CAIN routing parameter.

  Note: Provision NEXTCNRTE only when the criteria is datafilled as RTEAVAIL for OCLDBUSY and ONOANSWR, or TERMRTE_GNCT for NETBUSY.

- LEAVE_TDP – Call processing exits the TDP with no further evaluation.

  You can datafill LEAVE_TDP in the following trigger tables: OFFHKIMM, OFTRREQ, OFFHKDEL, SIOTRK, PRIBCHNL, SPECFEAT, CUSTDP, SPECDIG, OFFCCODE, NETBUSY, OCLDBUSY, ONOANSWR, OIECREO, and TERMATT.

- CONT_NOTRIG – Call processing exits the TDP and prevents any further NetworkBuilder interaction for the call. This prevention is reset upon reorigination.

  You can datafill CONT_NOTRIG in the following trigger tables: OFFHKIMM, OFTRREQ, OFFHKDEL, SIOTRK, PRIBCHNL, SPECFEAT, CUSTDP, SPECDIG, OFFCCODE, NETBUSY, OCLDBUSY, ONOANSWR, OIECREO, and TERMATT.

- COLLINFO – Call processing exits the TDP with no further evaluation. Reorigination is handled by normal switch features.

  You can datafill COLLINFO in table OIECREO.

When a “no match” is found in the trigger tables, call processing continues checking the remaining subscription methods and digit types.

Note: Refer to Volume 3, Chapter 10, “Incoming CAIN messages,” for more information on route advancing.

Step 9: Define the trigger criteria (continued)

Table 2-11
Trigger actions

Trigger	IGNORE	BLOCK	QUERY	QUERYSCU	FEAT	NEXTRTE	NEXTCNRTE	LEAVE_TDP	CONT_NOTRIG	COLLINFO
OFFHKIMM *Off_Hook_Immediate*	X	X	X	X				X	X	
OFTRREQ *O_Feature_Requested*	X	X			X			X	X	
TOLLFREE *Tollfree_Service* (Note)	X	X	X							
OFFHKDEL *OffHook_Delay*	X	X	X					X	X	
SIOTRK *Shared_Interoffice_Trunk*	X	X	X	X				X	X	
PRIBCHNL *PRI_B-Channel*	X	X	X	X				X	X	
SPECFEAT *Specific_Feature_Code*	X	X	X					X	X	
CUSTDP *Customized_Dialing_Plan*	X	X	X	X				X	X	
SPECDIG *Specific_Digit_String*	X	X	X	X				X	X	
OFFCCODE *Office_Code*	X	X	X					X	X	
NETBUSY *Network_Busy*	X		X			X	X	X	X	
OCLDBUSY *O_Called_Party_Busy*	X		X			X	X	X	X	
ONOANSWR *O_No_Answer*	X		X			X	X	X	X	
OIECREO *O_IEC_Reorigination*	X	X	X					X	X	X
TERMATT *Termination_Attempt*	X	X	X					X	X	

Note: Bellcore *TR-NWT-000533* defined IN/1 *Tollfree_Service* trigger

—end—

Datafilling trigger options

NetworkBuilder provides the following trigger options within the trigger tables: digit buffering, global title selection, pretranslator selection, reorigination, and T1 timeout overflow. However, all trigger options are not available for all triggers.

Step 9: Define the trigger criteria (continued)

Digit buffering

The digit buffering option (BUFFER) enables the switch to store dialed digits for use during conversation with the SCP. When the BUFFER option is set, the switch starts the user interaction framework (UIF), which stores the dialed digits.

The following rules apply to digit buffering:

- Initial digit collection can be initiated by the switch prior to any SCP query. A maximum of 24 digits can be stored.
- Digit buffering always occurs during conversation. A maximum of 24 digits can be stored.
- When a **`Send_To_Resource`** or **`Connect_To_Resource`** message requests digits, any buffered digits are processed and may be provided to the SCP in a **`Resource_Clear`** or **`CTR_Clear`** message.
  - — When an interruptible resource with digit collection request is returned in a **`Send_To_Resource`** or **`Connect_To_Resource`** message, the switch delivers any buffered digits to the SCP without playing the resource.
  - — When an uninterruptible resource with digit collection request is returned in a **`Send_To_Resource`** or **`Connect_To_Resource`** message, the switch discards the buffered digits, plays the resources, and then collects digits.
- The BUFFER option impacts real time efficiency.
- The switch only uses the buffered digits for interaction with the SCP.
- Buffering is not available at *O_No_Answer*.

Global titles

The GT (global title) option, the T1OVFLGT option, and the ACGOVFLGT option identify the global title value the switch uses to connect to an SCP. Global titles available: CAIN_CLID_GT, CAIN_ADDR_GT, and CAIN_FEAT_GT.

Note: The TOLLFREE trigger table utilizes the E800BELLCORE global title translation type for **`Start`** (tollfree_service) queries. The TOLLFREE trigger table does not support the GT, T1OVFLGT, or ACGOVFLGT trigger options. The E800BELLCORE global title translation type is implied rather than provisioned.

When the GT option is not provisioned, the switch uses the global title provisioned in parameter CAIN_DEFAULT_GT (table CAINPARM). When the T1OVFLGT option is not provisioned, the switch uses the global title

provisioned in parameter CAIN_DEFAULT_OVERFLOW_GT (table CAINPARM). When the ACGOVFLGT option is not provisioned, the switch uses the global title provisioned in parameter ACG_OVERFLOW_GT (table CAINPARM).

CAIN_CLID_GT The calling line identification global title translation type (CAIN_CLID_GT) allows convenient TCAP traffic segregation based on the caller's geographic location, as indicated by the caller's CLID, ANI, or home number plan area (NPA), as datafilled on the originating agency.

The contents of the global title address is dependent on the nature of address (of the Calling Party ID from the IAM of originating SS7 FGD, SS7 Inter-IMT, and SS7 Global-IMT), as follows:

- When the nature of address is NATL, the global title address is the same as the address transmitted in the ***CallingPartyID*** parameter.
- When the nature of address is INTL, UNK, or if the Calling Party ID is not included in the IAM, the switch derives a 3-digit NPA from in-switch datafill for use with this global title type.

Data is delivered to the appropriate SCP as follows:

1 The switch delivers a 10-digit CLID or ANI or a 3-digit NPA as the global title to the STP.

2 The STP translates the global title and routes the message to the appropriate SCP.

The switch obtains the global title translation type for the local type CAIN_CLID_GT in tables C7GTT and C7GTTYPE. The encoding scheme is determined by the CLID_GT_FORMAT parameter in table CAINPARM.

CAIN_ADDR_GT The address global title translation type (CAIN_ADDR_GT) allows traffic segregation based on the called party's identification.

The contents of the global title address is the same as the address transmitted in the ***CollectedAddressInfo*** parameter, when present. Otherwise, the address in the ***CalledPartyID*** is used. When neither parameter is available, a zero (0) is used.

Data is delivered to the appropriate SCP as follows:

1 The switch delivers the address digits as the global title to the STP.

Step 9: Define the trigger criteria (continued)

2 The STP translates the global title and routes the message to the appropriate SCP.

The switch obtains the global title translation type for the local type CAIN_ADDR_GT in tables C7GTT and C7GTTYPE. The encoding scheme is determined by the ADDR_GT_FORMAT parameter in table CAINPARM.

CAIN_FEAT_GT The feature global title translation type (CAIN_FEAT_GT) is used to identify unique SCPs set up for specific applications.

The contents of the global title address is provisioned in the switch. The encoding scheme is determined by the FEAT_GT_FORMAT parameter in table CAINPARM.

Pretranslator selection

The pretranslator option (PRTNUM) identifies a specific pretranslator for use with address collection through *O_Feature_Requested*. Datafilling the pretranslator option identifies a specific pretranslator to be used as an index in table STDPRTCT at *O_Feature_Requested*, and is only available at *O_Feature_Requested*.

Reorigination identifier

The reorigination option (REORIG) allows a reoriginated call to query after reorigination address collection without consulting trigger criteria at *O_Feature_Requested*, and is only available at *O_Feature_Requested*.

Associated logs

CAIN902, CAIN903

Associated OMs

CAINTRIG, CAINAGOM

NetworkBuilder parameters

Set the following trigger-related NetworkBuilder parameters:

- trigger at ***Info_Analyzed*** for SS7 Inter-IMT RLT calls (INFOANALYZED_FOR_RLT)
- maximum number of serial triggers (MAX_NUM_SERIAL_TRIGGERS)
- *O_No_Answer* timer (O_NO_ANSWER_TIMER)
- Timeout timer (TIMEOUT_TIMER)

Step 9: Define the trigger criteria INFOANALYZED_FOR_RLT (end)

The INFOANALYZED_FOR_RLT parameter in table CAINPARM determines whether *Specific_Feature_Code*, *Customized_Dialing_Plan*, and *Specific_Digit_String* triggers at the ***Info_Analyzed*** TDP are allowed to trigger on the second call leg of an SS7 RLT Inter-IMT call. The default is N.

Define the INFOANALYZED_FOR_RLT

At the CI prompt

1 Enter table CAINPARM.

2 Replace the parameter by typing:

>REP INFOANALYZED_FOR_RLT parmval
where

parmval Y or N

Sample entry: **>REP INFOANALYZED_FOR_RLT Y**

The INFOANALYZED_FOR_RLT parameter is provisioned for NetworkBuilder call processing.

Step 9: Define the trigger criteria Maximum number of serial triggers

The MAX_NUM_SERIAL_TRIGGERS parameter in table CAINPARM allows the user to define the maximum number of combined TDP- and EDP-Requests that can be sent to the SCP during a single call. The default is 3.

Note: The *TR-NWT-000533* defined *Tollfree_Service* trigger is not controlled by nor does it impact the count maintained by NetworkBuilder to limit the maximum number of serial triggers in a single call.

ATTENTION
If the MAX_NUM_SERIAL_TRIGGERS parameter is set to 0 (zero), calls using the CAIN protocol will not query the SCP.

The following table shows how a call is processed when the maximum number of serial triggers and events are reached.

Table 2-12
MAX_NUM_SERIAL_TRIGGERS call processing

Trigger action	NetworkBuilder call handling	Action performed
BLOCK	Processed as normal	AINF treatment applied
IGNORE	Processed as normal	Check next CAIN group (if applicable) or continue through call model
QUERY with ERRACT of TREAT (Note 1)	Treated as a fatal application error	AINF treatment applied

Note 1: Field ERRACT is provisioned in the trigger tables (OFFHKIMM, SPECFEAT, CUSTDP, SPECDIG, or TERMATT).
Note 2: If the trigger table includes an error action, the error action is taken. At ONOANSWR the DP is ignored and the originator continues to hear ringing tone. At NETBUSY and OCLDBUSY the DP is ignored. Table OFFCCODE does not include any error actions to provision at this time, it defaults to ROUTE. In all other cases, AINF treatment is applied.
Note 3: A trigger action of FEAT is only available in table OFTRREQ.

—continued—

Step 9: Define the trigger criteria
Maximum number of serial triggers (continued)

Table 2-12
MAX_NUM_SERIAL_TRIGGERS call processing (continued)

Trigger action	NetworkBuilder call handling	Action performed
QUERY with ERRACT of ROUTE (Note 1)	Treated as a fatal application error	Check next CAIN group (if applicable) or continue through call model
QUERY (Note 2)	Treated as a fatal application error	AINF treatment applied
FEAT (Note 3)	Treated as a fatal application error	AINF treatment applied
LEAVE_TDP	Processed as normal	Exit the current trigger detection point and continue through the call model
CONT_NOTRIG	Processed as normal	Exit the current trigger detection point and disallow any further NetworkBuilder interaction for the call

Note 1: Field ERRACT is provisioned in the trigger tables (OFFHKIMM, SPECFEAT, CUSTDP, SPECDIG, or TERMATT).
Note 2: If the trigger table includes an error action, the error action is taken. At ONOANSWR the DP is ignored and the originator continues to hear ringing tone. At NETBUSY and OCLDBUSY the DP is ignored. Table OFFCCODE does not include any error actions to provision at this time, it defaults to ROUTE. In all other cases, AINF treatment is applied.
Note 3: A trigger action of FEAT is only available in table OFTRREQ.

—end—

Associated logs

CAIN302

Define the maximum number of serial triggers allowed to query

At the CI prompt

1 Enter table CAINPARM.

2 Replace the defined parameter by typing:

>REP MAX_NUM_SERIAL_TRIGGERS parmval
where

parmval is the maximum number of serial triggers allowed (0–10).

Step 9: Define the trigger criteria
Maximum number of serial triggers (end)

Sample entry: **>REP MAX_NUM_SERIAL_TRIGGERS 5**

The maximum number of serial triggers is provisioned for NetworkBuilder call processing.

Step 9: Define the trigger criteria O_No_Answer timer (end)

The O_NO_ANSWER_TIMER parameter defines the time, in seconds, the switch should wait for call answer before evaluating the *O_No_Answer* trigger or *O_No_Answer* event. This parameter is used when the OANSTIME option is not datafilled for the CAIN group and no ***ONoAnswerTimer*** parameter is received in the **`Request_Report_BCM_Event`**. The default is 18.

Note: This parameter is used when the *O_No_Answer* trigger or event is enabled.

Define the O_NO_ANSWER_TIMER

At the CI prompt

1 Enter table CAINPARM.

2 Replace the parameter by typing:

>REP O_NO_ANSWER_TIMER parmval
where

parmval a number, in seconds (1 to 120)

Sample entry: **>REP O_NO_ANSWER_TIMER 120**

The O_No_Answer timer is provisioned for NetworkBuilder call processing.

Step 9: Define the trigger criteria
Timeout timer (end)

The TIMEOUT_TIMER parameter defines the time, in minutes, that a call can be active before encountering the *Timeout* event. This parameter is used when no ***TimeoutTimer*** parameter is received in the **Request_Report_BCM_Event** component. The default is 30.

Note 1: This parameter is used when the *Timeout* event is enabled.

Note 2: The Timeout timer can be restarted if the *Timeout* event is requested again after the switch sends a **CTR_Clear** message at the ***O_Mid_Call*** EDP. The timer's duration is determined by the ***TimeoutTimer*** parameter received in the second **Request_Report_BCM_Event** component or the value of this TIMEOUT_TIMER parameter in table CAINPARM.

Define the TIMEOUT_TIMER

At the CI prompt

1 Enter table CAINPARM.

2 Replace the parameter by typing:

>REP TIMEOUT_TIMER parmval
where

parmval a number, in minutes (1 to 180)

Sample entry: **>REP TIMEOUT_TIMER 120**

The Timeout timer is provisioned for NetworkBuilder call processing.

Step 10: Be familiar with NetworkBuilder digit collection

The CAIN_STR_RESET_ALLOWED parameter defines the maximum number of times the user can reset dialing during a **Send_To_Resource** or **Connect_To_Resource** digit collection interaction. The default is 0.

When the maximum number is exceeded, a fatal application error occurs and the switch sends a **Resource_Clear** or **CTR_Clear** message to the SCP. The **Resource_Clear** or **CTR_Clear** message contains a ***ClearCause*** of invalidCode and is sent in a response package.

Fatal application errors

The following error may be detected for CAIN_STR_RESET_ALLOWED:

Table 2-13
CAIN_STR_RESET_ALLOWED fatal application errors

Error type	Log generated	Reported to SCP?	Error action performed
Maximum resets allowed exceeded	CAIN200	Yes (Note)	AIND treatment is applied
Note: A **Resource_Clear** or **CTR_Clear** message is sent (within a response package) and the ***ClearCause*** parameter is set to invalidCode.			
—end—			

Associated logs

CAIN200

Define the CAIN_STR_RESET_ALLOWED parameter

At the CI prompt

1 Enter table CAINPARM.

2 Replace the parameter by typing:

>REP CAIN_STR_RESET_ALLOWED parmval
where

parmval the number of times reset can occur (0 to 255)

Sample entry: **>REP CAIN_STR_RESET_ALLOWED 10**

The number of times reset dialing can occur during an STR-Connection is provisioned for NetworkBuilder call processing.

Step 10: Be familiar with NetworkBuilder digit collection (continued)

NetworkBuilder activates the user interaction framework (UIF) during *O_Feature_Requested* digit collection or during **Send_To_Resource** or **Connect_To_Resource** digit collection.

Note: Digit collection is handled differently for AXXESS agents. Refer to *UCS DMS-250 CAIN/FlexDial Interactions* for more information.

User interaction framework

The UIF provides digit collection in one of the following ways:

- the *O_Feature_Requested* trigger specifies collectibles through datafill
- the *O_Feature_Requested* trigger also allows collectibles to be determined by the digits the user dials through interaction with table CAINPRT
- an SCP requests digits

The UIF also controls resource play. Internal resources are available on the switch. External resources may be available on an IP.

Digit collection

The UIF is unaware of digit context and collects digits as requested by the SCP. The UIF accepts octothorpes (#) and asterisks (*) as regular digits, but only when they are the first digits in the digit string.

Digit buffering

The digit buffering option (BUFFER) enables the switch to store dialed digits for use during conversation with the SCP. When the BUFFER option is set, the switch starts the user interaction framework (UIF), which stores the dialed digits.

The following rules apply to NetworkBuilder digit buffering available when the call has met trigger criteria with an action of QUERY or FEAT:

- Initial digit collection can be initiated by the switch prior to any SCP query. A maximum of 24 digits can be stored.
- Digit buffering always occurs during conversation. A maximum of 24 digits can be stored.
- When a **Send_To_Resource** or **Connect_To_Resource** message requests digits, any buffered digits are processed and may be provided to the SCP in a **Resource_Clear** or **CTR_Clear** message.
  - — When an interruptible resource with digit collection request is returned in a **Send_To_Resource** or **Connect_To_Resource** message, the switch delivers any buffered digits to the SCP without playing the resource.

Step 10: Be familiar with NetworkBuilder digit collection (continued)

— When an uninterruptible resource with digit collection request is returned in a **`Send_To_Resource`** or **`Connect_To_Resource`** message, the switch discards the buffered digits, plays the resources, and then collects digits.

- The BUFFER option impacts real time efficiency.
- The switch only uses the buffered digits for interaction with the SCP.
- Buffering is not available at *O_No_Answer*.

Ambiguous dialing

Ambiguous dialing may occur when multiple digit streams are being collected and the number of digits required is set to variable. For example:

Digit Collection stream #1 (0 to 10 digits) variable
Digit Collection stream #2 (0 to 7 digits) variable

— If the user enters the following without waiting for prompts:
555-1234 + 7777777

— The switch may ambiguously collect the digits as:
555-123-4777 + 7777

Both of these digit streams functionally fulfill the variable requests, but the digit stream the subscriber intended is unknown.

To ensure correct digit collection without encountering ambiguous collection scenarios, consider the following:

- Instruct the subscriber to enter an octothorpe (#) to signal end of dialing.
- Advise the subscriber to wait for the collection prompt.
- Uninterruptible prompts do not necessarily reduce the risk of ambiguous dialing, as the subscriber may become confused when they have already entered digits.
- The switch discards buffered digits when an uninterruptible prompt is received. Therefore, uninterruptible prompts can reduce ambiguous dialing only while buffering digits.

Step 10: Be familiar with NetworkBuilder digit collection (continued)

- The `pretranslatorName` extension parameter in a **`Send_To_Resource`** message can be used to minimize ambiguous dialing. This optional parameter contains an index into the table CNPREXLA. Table CNPREXLA provides a pretranslator name used by the switch, through table STDPRTCT, to pretranslate the collected address digits. (All pretranslators found in table STDPRTCT supported by NetworkBuilder are allowed.) The pretranslator performs call typing (such as, ONNET, OFFNET) and defines the minimum and maximum number of digits to collect. Once the address digits are collected, the switch will buffer any remaining digits for the next collectible. The ***`FlexParameterBlock`*** parameter's minimum and maximum digit information defines the number of billing digits to collect. When it can be determined (based on the pretranslator name), the ADDRESS digits returned to the SCP are assigned the appropriate nature of address (NOA), not UNKNOWN. Using the pretranslator to resolve ambiguous dialing applies only when the address is the first collectible requested by the SCP. Any misdialing errors or other dialing ambiguities continue to be handled by the SCP.

  Note: The new pretranslator name returned in the `pretranslatorName` extension parm is used for the rest of the call, including reorigination.

Reset dialing

At any time after a digit is dialed, the subscriber can reset dialing for that digit stream by pressing the asterisk (*). The switch, upon receiving *, resets the current digit collection stream and plays any applicable resource.

You can determine the number or resets allowed for each **`Send_To_Resource`** or **`Connect_To_Resource`** digit collection interaction by provisioning parameter CAIN_STR_RESETS_ALLOWED (table CAINPARM). A maximum of 255 resets are allowed.

End of dialing

The caller can end digit collection by pressing the octothorpe (#) after dialing one digit.

Interdigit timing

During digit collection, interdigit timing depends on the type of digit string requested by the SCP.

When the SCP requests a variable number of digits, the switch uses normal interdigit timing before and after the first digit is received to determine end of dialing. End of dialing occurs when the required number of digits are received, a timeout occurs, or when the subscriber dials #.

Step 10: Be familiar with NetworkBuilder digit collection (end)

When the SCP requests a fixed number of digits (from 1 to 24), the switch uses normal interdigit timing before and after the first digit is received. End of dialing occurs when a timeout occurs, the requested number of digits are received, or the subscriber dials #.

Permanent signal (PSIG) timing is calculated for each interruptible prompt as it is played. The following formula is used:

PSIG = PromptDuration + AgentDatafill

- PromptDuration
  - — Tone prompts – obtained from table TONES MAXDURN fields
  - — DRAM prompts – obtained from table ANNS CYTIME field multiplied by the MAXCYC field
- AgentDatafill
  - — PTS agents – PSPDSEIZ value from table TRKSGRP
  - — PRI agents – PSPDSEIZ value from table TRKSGRP
  - — SS7 FGD, SS7 Inter-IMT, SS7 Global-IMT agents – SNDRPSIG value from table TRKGRP

Associated OMs

ANNATT, ANNOVFL, CAINUIF, CAINTRIG, CAINAGOM

Step 11: Define the messaging-related parameters

Define the following messaging-related office parameters:

- ADDR_GT_FORMAT
- CAIN_CONVERSATION_LIMIT
- CAIN_PROTOCOL_STREAM
- CAIN_PROTOCOL_VERSION
- CAIN_T1_TIMEOUT
- CLID_GT_FORMAT
- DEFAULT_SNPA
- FEAT_GT_FORMAT
- INTL_XLA_TYPE
- LNP_FOR_RX_SELECTOR
- LNP_PARAMETER_SET
- LNP_PROTOCOL_STREAM
- LNP_PROTOCOL_VERSION
- MAX_FAILURE_OUTCOMES
- OFCD_GT_FORMAT
- PRIVATE_FACILITY_GROUP_USERID
- RESTRICT_NETBUSY_BUSYCAUSE
- SEND_CARRIER_FROM_TRKGRP
- STR_CONNECTION_TYPE
- TDISC_TIMER
- TSTRC_TIMER

Step 11: Define the messaging-related parameters ADDR_GT_FORMAT

NetworkBuilder supports both the 0001 and 0010 global title indicators. The ADDR_GT_FORMAT parameter in table CAINPARM is used to indicate which global title indicator to use for the CAIN_ADDR_GT global title.

When the ADDR_GT_FORMAT parameter is set to IMPLICIT, a global title indicator of 0010 is used. When ADDR_GT_FORMAT parameter is set to ENHANCED, a global title indicator of 0001 is used.

GT encoding information

Within the Called Party Address portion of the SCCP header, the first byte specifies Address Indicator information. Remaining bytes contain the subsystem (SSN), point code (PC), and global title (GT). The following table describes the Address Indicator byte.

Table 2-14
Address Indicator byte format

Bit	Value
0	0 – SSN not included
	1 – SSN included
1	0 – PC not included
	1 – PC included
5–2	0000 – No GT included
	0001 – GT included (Type Encoded format)
	GT Type Encoded format:
	Byte 1 – GT Type
	Byte 2 – Numbering Plan and Encoding Scheme
	Remaining bytes – GT Address Digits
	0010 – GT included (Implicit format)
	GT Implicit format:
	Byte 1 – GT Type
	Remaining bytes – GT Address Digits
	Note: Value 0001 and 0010 are controlled by the ADDR_GT_FORMAT parameter.

—continued—

Step 11: Define the messaging-related parameters ADDR_GT_FORMAT (end)

Table 2-14
Address Indicator byte format (continued)

Bit	Value
6	0 – Routing based on GT 1 – Routing based on destination PC
7	Network Indicator
	—end—

Define the ADDR_GT_FORMAT parameter

At the CI prompt

1 Enter table CAINPARM.

2 Replace the defined parameter by typing:

>REP ADDR_GT_FORMAT parmval
where

parmval is the global title indicator format (IMPLICIT, ENHANCED).

Sample entry: **>REP ADDR_GT_FORMAT ENHANCED**

The global title indicator format for CAIN_ADDR_GT is provisioned for NetworkBuilder call processing.

Step 11: Define the messaging-related parameters CAIN_CONVERSATION_LIMIT

The CAIN_CONVERSATION_LIMIT parameter indicates the maximum number of **Send_To_Resource** or **Connect_To_Resource** conversations (defined as Play Announcement and Collect Digits or Flex Parameter Block and initiated by the SCP) allowed during a call. The default is five SCP-initiated conversations per call.

When the maximum number is exceeded, a fatal application error occurs and the switch sends a **Resource_Clear** or **CTR_Clear** message to the SCP. The **Resource_Clear** or **CTR_Clear** message contains a ***ClearCause*** of taskRefused and is sent in a response package.

When the parameter is set to 15, unlimited (no maximum) **Send_To_Resource** or **Connect_To_Resource** conversations are allowed.

Fatal application errors

The following error may be detected for a CAIN_CONVERSATION_LIMIT:

Table 2-15
CAIN_CONVERSATION_LIMIT fatal application errors

Error type	Log generated	Reported to SCP?	Error action performed
Maximum conversation limit exceeded	CAIN200	Yes (Note)	AINF treatment is applied
Note: A **Resource_Clear** or **CTR_Clear** message is sent (within a response package) and the ***ClearCause*** parameter is set to taskRefused.			
—end—			

Associated logs

CAIN200

Step 11: Define the messaging-related parameters CAIN_CONVERSATION_LIMIT (end)

Define the CAIN_CONVERSATION_LIMIT parameter

At the CI prompt

1 Enter table CAINPARM.

2 Replace the parameter by typing:

>REP CAIN_CONVERSATION_LIMIT parmval
where

parmval the number of conversation messages allowed (0 to 15).

Sample entry: **>REP CAIN_CONVERSATION_LIMIT 5**

The number of conversation messages allowed is provisioned for NetworkBuilder call processing.

Step 11: Define the messaging-related parameters CAIN_PROTOCOL_STREAM

Previous to UCS08, NetworkBuilder controlled the protocol being used for processing through the CAIN_PROTOCOL_VERSION parameter in table CAINPARM. This office parameter consisted of two parts, the stream and the version (for example, UCS05_V0). Starting in UCS08, the protocol control is separated into two parameters: CAIN_PROTOCOL_STREAM and CAIN_PROTOCOL_VERSION.

Note: The trigger tables have an option, STREAM, that provides the ability to control the protocol stream on a per-trigger tuple basis. This simplifies the connections to multiple SCPs. The trigger tables include: OFFHKIMM, OFTRREQ, OFFHKDEL, SIOTRK, PRIBCHNL, SPECFEAT, CUSTDP, SPECDIG, NETBUSY, OCLDBUSY, ONOANSWR, OIECREO, and TERMATT.

The value of the CAIN_PROTOCOL_STREAM parameter determines whether certain messages or parameters are allowed to be sent in outgoing NetworkBuilder TCAP packages. The default is UCS05.

Changes from the previous release of NetworkBuilder TCAP protocol messages and parameters are not accessible until the parameter is changed to indicate the new protocol stream. New streams will include the features of previous streams. New messages can still be sent.

Table 2-16
Protocol version control based on stream designation

Stream	Controlled parameters
UCS05	The parameters allowed in outgoing switch messages by later protocol version streams are not supported for this stream designation.
UCS06	The ***VerticalServiceCode*** parameter is allowed in the **Info_Analyzed** message.
Note: The `adin`, `cainGroup`, and `origTrunkInfo` extension parameters are only encoded when provisioned in the EXTPARMS or TEXTPARMS field of table CAINGRP. When datafilled, these parameters are encoded and delivered to the SCP regardless of the value of CAIN_PROTOCOL_STREAM.	
—continued—	

Step 11: Define the messaging-related parameters CAIN_PROTOCOL_STREAM (continued)

Table 2-16
Protocol version control based on stream designation (continued)

Stream	Controlled parameters
	The `adin`, `cainGroup`, `origTrunkInfo`, `reorigCall`, and `treatment` extension parameters are allowed in the **`Info_Analyzed`** message.
UCS07	The `lnpReceived` extension parameter is allowed in the **`Info_Analyzed`**, **`Info_Collected`**, and **`O_Feature_Requested`** messages.
	If the ***`ChargeNumber`*** parameter has been populated with an ANI, then the Nature of Address (NOA) is set to NATL instead of I2ANI, and the ***`ChargePartyStationType`*** parameter contains the Information Digits from the call.
UCS08	The `subscriptionInfo` extension parameter is allowed in all TDP-Request messages.
	The `cainPRT` extension parameter is allowed in the **`O_Feature_Requested`** message.
	The ***`JurisdictionInfo`*** parameter is allowed in the **`Info_Analyzed`** message.
	The `jurisdictionInfo` extension parameter is allowed in all TDP-Request messages except **`Info_Analyzed`**.
	The `collectedAddress` extension parameter is allowed in the **`Network_Busy`**, **`O_Called_Party_Busy`**, and **`O_No_Answer`** TDP-Request messages.
UCS09	Support for the outgoing `switchID` extension parameter is added.
	Support for the outgoing and incoming `accountCode` extension parameter is added.
	Support for the outgoing and incoming `billingNumber` extension parameter is added.
	Support for the outgoing and incoming `pinDigits` extension parameter is added.
Note: The `adin`, `cainGroup`, and `origTrunkInfo` extension parameters are only encoded when provisioned in the EXTPARMS or TEXTPARMS field of table CAINGRP. When datafilled, these parameters are encoded and delivered to the SCP regardless of the value of CAIN_PROTOCOL_STREAM.	

—continued—

Step 11: Define the messaging-related parameters CAIN_PROTOCOL_STREAM (end)

Table 2-16
Protocol version control based on stream designation (continued)

Stream	Controlled parameters
UCS11	Support for the outgoing ***Lata*** parameter is added.
UCS17	The AIN `ExtensionParameter` parameter is included in the **Info_Analyzed** message as well as the **Info_Collected** message.
Note: The `adin`, `cainGroup`, and `origTrunkInfo` extension parameters are only encoded when provisioned in the EXTPARMS or TEXTPARMS field of table CAINGRP. When datafilled, these parameters are encoded and delivered to the SCP regardless of the value of CAIN_PROTOCOL_STREAM.	
—end—	

Note: When ONP is performed, the new load contains existing datafill. Default datafill is not provided for new triggers. Therefore, only the previously defined datafill is available until you add new datafill.

Define the CAIN_PROTOCOL_STREAM parameter

At the CI prompt

1 Enter table CAINPARM.

2 Replace the parameter by typing:

>REP CAIN_PROTOCOL_STREAM parmval
where

parmval the protocol stream (UCS05, UCS06, UCS07, UCS08, UCS09, UCS11)

Sample entry: **>REP CAIN_PROTOCOL_STREAM UCS11**

The CAIN_PROTOCOL_STREAM parameter is provisioned for NetworkBuilder call processing.

Step 11: Define the messaging-related parameters CAIN_PROTOCOL_VERSION

Previous to UCS08, NetworkBuilder controlled the protocol being used for processing through the CAIN_PROTOCOL_VERSION parameter in table CAINPARM. This office parameter consisted of two parts, the stream and the version (for example, UCS05_V0). Starting in UCS08, the protocol control is separated into two parameters: CAIN_PROTOCOL_STREAM and CAIN_PROTOCOL_VERSION.

Note: The trigger tables have an option, STREAM, that provides the ability to control the protocol stream on a per-trigger tuple basis. This simplifies the connections to multiple SCPs. The trigger tables include: OFFHKIMM, OFTRREQ, OFFHKDEL, SIOTRK, PRIBCHNL, SPECFEAT, CUSTDP, SPECDIG, NETBUSY, OCLDBUSY, ONOANSWR, OIECREO, and TERMATT.

The value of the CAIN_PROTOCOL_VERSION parameter determines the encoding format to use for parameter whose encodings have changed between software releases.

Encoding changes from the previous release are not accessible until the CAIN_PROTOCOL_VERSION parameter is changed to indicate the new protocol version. New versions will include the features of previous versions.

This parameter indicates that available parameters are formatted using the specified version (V0–V5) definitions. The default is V0.

Table 2-17
Protocol version control based on version designation

Version	Controlled parameters
V0	The `univeralAccess` extension parameter allows up to 10 digits to be sent to the SCP.
V1	The `univeralAccess` extension parameter allows up to 24 digits to be sent to the SCP.
V2	The population of the Nature of Address (NOA) and Numbering Plan of the ***`CollectedAddressInfo`*** parameter conforms to *GR-1298-CORE* requirements. In previous versions the ***`CollectedAddressInfo`*** NOAs and Numbering Plans were set to unknown.

—continued—

Step 11: Define the messaging-related parameters CAIN_PROTOCOL_VERSION (continued)

Table 2-17
Protocol version control based on version designation (continued)

Version	Controlled parameters
V3	The ***UserID*** parameter takes the TrunkGroup format as specified in *GR-1299-CORE*. The ***AccessCode*** parameter sent in outgoing messages is not populated by the account code, and therefore not sent. The ***CollectedDigits*** parameter sent in outgoing messages is not populated by PIN digits, and therefore not sent. The digit identifiers supported for the incoming ***AMADigitsDialedWC*** parameter are altered. Refer to Volume 3, Chapter 12, “Incoming CAIN message parameters,” for more information. The NOA values supported for the outgoing ***CalledPartyID*** parameter are altered. Refer to Volume 3, Chapter 12, “Incoming CAIN message parameters,” for more information. Non-standard values (7KHZ audio and multiRate) are removed from the outgoing ***BearerCapability*** parameter, Speech is sent instead. If the ***AMALineNumber*** parameter is received in an **Analyze_Route** message, it is ignored. If the ***DPConverter*** parameter is received in an **Send_To_Resource** message, it is ignored. If the ***ExtendedRinging*** parameter is received in an **Send_To_Resource** message, it is ignored. If the **ACG** message is received as a non-first component, it is ignored. If the **Send_Notification** message is received as a non-first component, it is ignored.
V4	Support added for additional NOAs in outgoing ***ChargeNumber*** parameter.

—continued—

Step 11: Define the messaging-related parameters CAIN_PROTOCOL_VERSION (end)

Table 2-17
Protocol version control based on version designation (continued)

Version	Controlled parameters
	Support added for standard NOAs in outgoing ***CollectedAddressInfo*** parameter.
V5	Population of outgoing ***CallingPartyID*** parameter is modified because the switch retains the ***CallingPartyID*** parameter the switch received in an SCP response message.
	—end—

Note: When ONP is performed, the new load contains existing datafill. Default datafill is not provided for new triggers. Therefore, only the previously defined datafill is available until you add new datafill.

Define the CAIN_PROTOCOL_VERSION parameter

At the CI prompt

1 Enter table CAINPARM.

2 Replace the parameter by typing:

>REP CAIN_PROTOCOL_VERSION parmval
where

parmval the protocol version (V0, V1, V2, V3, V4,V5)

Sample entry: **>REP CAIN_PROTOCOL_VERSION V5**

The CAIN protocol version is provisioned for NetworkBuilder call processing.

Step 11: Define the messaging-related parameters CAIN_T1_TIMEOUT

The CAIN_T1_TIMEOUT parameter, in table CAINPARM, allows the user to define the maximum time (in seconds) the UCS DMS-250 switch waits for a response from the SCP before assuming an error has occurred. This parameter protects the switch from waiting indefinitely for the SCP to respond when the SCP or network fails. The default is 2.

Under normal conditions, the T1 timer should never expire while waiting for an SCP response. Set CAIN_T1_TIMEOUT to a time limit greater than a reasonable amount of time required for an SCP response.

Note: A good estimate is twice the mean response time measured when the network is under a normal busy-hour load.

ATTENTION

Setting the CAIN_T1_TIMEOUT to a value too low may cause SCP responses to be discarded, thereby increasing network load. The switch reports T1 timeout errors to the SCP in an error message. In addition, the switch still receives and performs error processing on SCP responses arriving after a T1 timeout. Error processing may result in another error message being sent to the SCP.

ATTENTION

Setting the CAIN_T1_TIMEOUT to zero (0) prevents launching of query messages to the SCP. A fatal application error is recorded in this case.

ATTENTION

Engineering of certain switch resources related to NetworkBuilder calls and TCAP messaging may depend on the value datafilled in CAIN_T1_TIMEOUT. This value affects the holding time calculation for switch resources.

Fatal application errors

The following error is detected in the case of CAIN_T1_TIMEOUT:

Step 11: Define the messaging-related parameters CAIN_T1_TIMEOUT (end)

CAIN_T1_TIMEOUT fatal application errors

Error type	Log generated	Reported to SCP?	Error action performed
CAIN_T1_TIMEOUT (Note 1)	CAIN200	Yes	ERRACT in trigger table (Note 2)
User abandon occurs before a T1 timeout	CAIN201	Yes	None

Note 1: The SCP did not respond within the provisioned amount of time. Refer to the CAINPARM CAIN_T1_TIMEOUT section of *UCS DMS-250 Data Schema Reference Manual* for more information.
Note 2: Field ERRACT is provisioned in the trigger tables (OFFHKIMM, SPECFEAT, CUSTDP, SPECDIG, or TERMATT). The OFFCCODE trigger table does not provision ERRACT, it defaults to the ROUTE error action. Trigger tables that don't provision ERRACT default to AINF treatment, when required.

—end—

Associated logs

CAIN200, CAIN201

Associated OMs

CAINOM, CAINAGOM, VPTRUSAG

Define the CAIN_T1_TIMEOUT parameter

At the CI prompt

1 Enter table CAINPARM.

2 Replace the defined parameter by typing:

>REP CAIN_T1_TIMEOUT parmval
where

parmval is timeout value (in seconds).

Sample entry: **>REP CAIN_T1_TIMEOUT 5**

The CAIN_T1_TIMEOUT parameter is defined.

Step 11: Define the messaging-related parameters CLID_GT_FORMAT

CAIN supports both the 0001 and 0010 global title indicators. The CLID_GT_FORMAT parameter in table CAINPARM is used to indicate which global title indicator to use for the CAIN_CLID_GT global title.

When the CLID_GT_FORMAT parameter is set to IMPLICIT, a global title indicator of 0010 is used. When CLID_GT_FORMAT parameter is set to ENHANCED, a global title indicator of 0001 is used.

GT encoding information

Table 2-18
Address Indicator byte format

Bit	Value
0	0 – SSN not included
	1 – SSN included
1	0 – PC not included
	1 – PC included
5–2	0000 – No GT included
	0001 – GT included (Type Encoded format)
	GT Type Encoded format:
	Byte 1 – GT Type
	Byte 2 – Numbering Plan and Encoding Scheme
	Remaining bytes – GT Address Digits
	0010 – GT included (Implicit format)
	GT Implicit format:
	Byte 1 – GT Type
	Remaining bytes – GT Address Digits
	Note: Value 0001 and 0010 are controlled by the CLID_GT_FORMAT parameter.
6	0 – Routing based on GT
	1 – Routing based on destination PC
7	Network Indicator
	—end—

Step 11: Define the messaging-related parameters CLID_GT_FORMAT (end)

Define the CLID_GT_FORMAT parameter

At the CI prompt

1 Enter table CAINPARM.

2 Replace the defined parameter by typing:

>REP CLID_GT_FORMAT parmval
where

parmval is the global title indicator format (IMPLICIT, ENHANCED).

Sample entry: **>REP CLID_GT_FORMAT ENHANCED**

The global title indicator format for CAIN_CLID_GT is provisioned for NetworkBuilder call processing.

Step 11: Define the messaging-related parameters DEFAULT_SNPA (end)

The DEFAULT_SNPA parameter is used as a default value for ***CallingPartyID*** parameter population.

Define a default SNPA

At the CI prompt

1 Enter table CAINPARM.

2 Replace the defined parameter by typing:

>REP DEFAULT_SNPA parmval
where

parmval is the default SNPA (000–999).

Sample entry: **>REP DEFAULT_SNPA 972**

A default SNPA is provisioned for NetworkBuilder call processing.

Step 11: Define the messaging-related parameters FEAT_GT_FORMAT

NetworkBuilder supports both the 0001 and 0010 global title indicators. The FEAT_GT_FORMAT parameter in table CAINPARM is used to indicate which global title indicator to use for the CAIN_FEAT_GT global title.

When the FEAT_GT_FORMAT parameter is set to IMPLICIT, a global title indicator of 0010 is used. When FEAT_GT_FORMAT parameter is set to ENHANCED, a global title indicator of 0001 is used.

GT encoding information

Table 2-19
Address Indicator byte format

Bit	Value
0	0 – SSN not included
	1 – SSN included
1	0 – PC not included
	1 – PC included
5–2	0000 – No GT included
	0001 – GT included (Type Encoded format)
	GT Type Encoded format:
	Byte 1 – GT Type
	Byte 2 – Numbering Plan and Encoding Scheme
	Remaining bytes – GT Address Digits
	0010 – GT included (Implicit format)
	GT Implicit format:
	Byte 1 – GT Type
	Remaining bytes – GT Address Digits
	Note: Value 0001 and 0010 are controlled by the FEAT_GT_FORMAT parameter.
6	0 – Routing based on GT
	1 – Routing based on destination PC
7	Network Indicator
	—end—

Step 11: Define the messaging-related parameters FEAT_GT_FORMAT (end)

Define the FEAT_GT_FORMAT parameter

At the CI prompt

1 Enter table CAINPARM.

2 Replace the defined parameter by typing:

>REP FEAT_GT_FORMAT parmval
where

parmval is the global title indicator format (IMPLICIT, ENHANCED).

Sample entry: **>REP FEAT_GT_FORMAT ENHANCED**

The global title indicator format for CAIN_FEAT_GT is provisioned for NetworkBuilder call processing.

Step 11: Define the messaging-related parameters
INTL_XLA_TYPE (end)

The INTL_XLA_TYPE parameter in table CAINPARM is used in translating the ***CalledPartyID*** and the ***GenericAddressList***'s `OverflowRoutingNo` when the Nature of Address (NOA) is international (INTL). The SCP does not need to use partitioned (PART) NOAs when sending the ***CalledPartyID*** and `OverflowRoutingNo` parameter. The INTL NOA can be sent and the value set in the INTL_XLA_TYPE parameter specifies the international translation system to be used to translate the number.

When provisioned with the default value of "IN", the international translation type is used in translating the number for routing through table CCTR.

When provisioned with the default value of "IP", the international partitioned translation type is used in translating the number for routing through tables CTRTE, CTCODE, CTHEAD, and STS2CCDB.

The "IN" and "IP" values act as if the translation type was defined in standard translations through the translation system (TRANSYS) in table STDPRTCT.

Define the INTL_XLA_TYPE parameter

At the CI prompt

1 Enter table CAINPARM.

2 Replace the parameter by typing:

>REP INTL_XLA_TYPE parmval
where

parmval the international translations type (IN, IP)

Sample entry: **>REP INTL_XLA_TYPE IN**

The international translations type is provisioned for NetworkBuilder call processing.

Step 11: Define the messaging-related parameters LNP_EVALUATE_AFTER_OTC_CIC (end)

The LNP_EVALUATE_AFTER_OTC_CIC parameter in table CAINPARM allows a subsequent CAIN query after receiving an ANALYZE_ROUTE message with a CIC other than 0110.

When the LNP_EVALUATE_AFTER_OTC_CIC parameter is set to Y and the CIC returned in the ANALYZE_ROUTE is datafilled as OTC in table CIC_ROUTE, the DMS evaluates subsequent triggers.

Define the LNP_EVALUATE_AFTER_OTC_CIC parameter

At the CI prompt

1 Enter table CAINPARM.

2 Replace the parameter by typing:

```
>REP LNP_EVALUATE_AFTER_OTC_CIC parmval
```

where

parmval
is the parameter value (Y, N)

Sample entry:
>REP LNP_EVALUATE_AFTER_OTC_CIC Y

The LNP_EVALUATE_AFTER_OTC_CIC parameter is provisioned for NetworkBuilder call processing.

Step 11: Define the messaging-related parameters LNP_FOR_RX_SELECTOR (end)

The LNP_FOR_RX_SELECTOR parameter in table CAINPARM is used to indicate if NetworkBuilder interactions are valid for CAIN-capable calls using the RX selector.

When the LNP_FOR_RX_SELECTOR parameter is set to "Y", the called number or the STS changed by the RX selector is evaluated to determine if NetworkBuilder interactions are required to route the call. If NetworkBuilder interactions are required, an LNP query may be performed to correctly route the call based on the called number or the STS.

When the LNP_FOR_RX_SELECTOR parameter is set to "N", the called number or the STS is not evaluated for NetworkBuilder interactions. However, if the called number is changed by the RX selector any previous LNP data is reset.

Define the LNP_FOR_RX_SELECTOR parameter

At the CI prompt

1 Enter table CAINPARM.

2 Replace the parameter by typing:

>REP LNP_FOR_RX_SELECTOR parmval
where

parmval the parameter value (Y, N)

Sample entry: **>REP LNP_FOR_RX_SELECTOR Y**

The LNP_FOR_RX_SELECTOR parameter value is provisioned for NetworkBuilder call processing.

Step 11: Define the messaging-related parameters LNP_PARAMETER_SET (end)

The LNP_PARAMETER_SET parameter in table CAINPARM is used to enhance realtime processing of LNP calls by controlling whether a complete set or minimum set of parameters are sent in an *Office_Code* **Info_Analyzed** message.

Define the LNP_PARAMETER_SET parameter

At the CI prompt

1 Enter table CAINPARM.

2 Replace the parameter by typing:

>REP LNP_PARAMETER_SET parmval
where

parmval is the parameter value (COMPLETE_SET or MINIMUM_SET)

Note: When the LNP_PARAMETER_SET parameter is set to COMPLETE_SET, the ***UserID***, ***BearerCapability***, ***CalledPartyID***, ***TriggerCriteriaType***, ***CallingPartyID***, ***Carrier***, ***ChargeNumber***, and ***ChargePartyStationType*** parameters are sent in the **Info_Analyzed** message. When the parameter is set to MINIMUM_SET, the parameters sent in **Info_Analyzed** are ***UserID***, ***BearerCapability***, ***CalledPartyID***, and ***TriggerCriteriaType***.

Sample entry: **>REP LNP_PARAMETER_SET minimum_set**

The LNP parameter set value is provisioned for NetworkBuilder call processing

Step 11: Define the messaging-related parameters LNP_PROTOCOL_STREAM (end)

The LNP_PROTOCOL_STREAM parameter in table CAINPARM controls the set of parameters that are sent for a particular LNP query. If set to UCS07, then only the ***CalledPartyID*** parameter is returned in response to *Office_Code* queries. When set to UCS08, the ***ForwardCallIndicator*** parameter with bit M, ***GenericAddressList*** parameter containing a `PortedDialedNo`, and a ***CalledPartyID*** parameter must be returned in the response to interpret the ***CalledPartyID*** as a location routing number (LRN). The default is UCS07.

Define the LNP_PROTOCOL_STREAM parameter

At the CI prompt

1 Enter table CAINPARM.

2 Replace the parameter by typing:

>REP LNP_PROTOCOL_STREAM parmval
where

parmval the protocol stream (UCS07, UCS08)

Sample entry: **>REP LNP_PROTOCOL_STREAM UCS08**

The LNP_PROTOCOL_STREAM parameter is provisioned for NetworkBuilder call processing.

Step 11: Define the messaging-related parameters LNP_PROTOCOL_STREAM (end)

The LNP_PROTOCOL_VERSION parameter in table CAINPARM controls the encoding of outgoing LNP parameters. The default is V1.

Note: Only one version of the LNP encoding is possible (V1).

Define the LNP_PROTOCOL_VERSION parameter

At the CI prompt

1 Enter table CAINPARM.

2 Replace the parameter by typing:

>REP LNP_PROTOCOL_VERSION parmval
where

parmval the protocol version (V1)

Sample entry: **>REP LNP_PROTOCOL_VERSION V1**

The LNP_PROTOCOL_VERSION parameter is provisioned for NetworkBuilder call processing.

Step 11: Define the messaging-related parameters MAX_FAILURE_OUTCOMES

The MAX_FAILURE_OUTCOMES parameter, in table CAINPARM, allows you to define the maximum number of times per call the switch can send a **Failure_Outcome** message to the SCP.

Fatal application errors

The switch detects the following error for the MAX_FAILURE_OUTCOMES parameter.

Table 2-20
MAX_FAILURE_OUTCOMES fatal application errors

Error type	Log generated	Reported to SCP?	Error action performed
Maximum number of **Failure_Outcome** messages exceeded	CAIN200	Yes (Note)	Close transaction
Note: The switch sends a **Close** message with a ***CloseCause*** of unexpectedCommunication.			
—end—			

Associated logs

CAIN907, CAIN200

Associated OMs

CAINMSGS

Define the MAX_FAILURE_OUTCOMES parameter

At the CI prompt

1 Enter table CAINPARM.

2 Replace the defined parameter by typing:

>REP MAX_FAILURE_OUTCOMES parmval
where

parmval is the number of **Failure_Outcome** messages allowed (0 to 3). A value of 3 means the switch allows an unlimited number of **Failure_Outcome** messages. The default value is 2.

Sample entry: **>REP MAX_FAILURE_OUTCOMES 1**

Step 11: Define the messaging-related parameters MAX_FAILURE_OUTCOMES (end)

You have defined the MAX_FAILURE_OUTCOMES parameter.

Step 11: Define the messaging-related parameters OFCD_GT_FORMAT

NetworkBuilder supports both the 0001 and 0010 global title indicators. The OFCD_GT_FORMAT parameter in table CAINPARM is used to indicate which global title indicator to use for the CAIN_OFCD_GT global title.

When the OFCD_GT_FORMAT parameter is set to IMPLICIT, a global title indicator of 0010 is used. When OFCD_GT_FORMAT parameter is set to ENHANCED, a global title indicator of 0001 is used.

GT encoding information

Table 2-21
Address Indicator byte format

Bit	Value
0	0 – SSN not included
	1 – SSN included
1	0 – PC not included
	1 – PC included
5–2	0000 – No GT included
	0001 – GT included (Type Encoded format)
	GT Type Encoded format:
	Byte 1 – GT Type
	Byte 2 – Numbering Plan and Encoding Scheme
	Remaining bytes – GT Address Digits
	0010 – GT included (Implicit format)
	GT Implicit format:
	Byte 1 – GT Type
	Remaining bytes – GT Address Digits
	Note: Value 0001 and 0010 are controlled by the OFCD_GT_FORMAT parameter.
6	0 – Routing based on GT
	1 – Routing based on destination PC
7	Network Indicator
	—end—

Step 11: Define the messaging-related parameters OFCD_GT_FORMAT (end)

Define the OFCD_GT_FORMAT parameter

At the CI prompt

1 Enter table CAINPARM.

2 Replace the defined parameter by typing:

>REP OFCD_GT_FORMAT parmval
where

parmval is the global title indicator format (IMPLICIT, ENHANCED).

Sample entry: **>REP OFCD_GT_FORMAT ENHANCED**

The global title indicator format for CAIN_OFCD_GT is provisioned for NetworkBuilder call processing.

Step 11: Define the messaging-related parameters PRIVATE_FACILITY_GROUP_USERID (end)

The PRIVATE_FACILITY_GROUP_USERID parameter, in table CAINPARM, controls PrivateFacilityGID encoding of the UserID parameter for originating DAL trunks only.

Associated logs

Not applicable

Associated OMs

Not applicable

Define the PRIVATE_FACILITY_GROUP_USERID parameter

At the CI prompt

1 Enter table CAINPARM.

2 Replace the defined parameter by typing:

>REP PRIVATE_FACILITY_GROUP_USERID parmval
where

parmval is the value of the PRIVATE_FACILITY_GROUP_USERID parmeter (Y or N). A value of N indicates that the UserID parameter is set to the same value that was sent in the **`Info_Collected`** message. A value of Y indicates the UserID parameter contains the incoming DAL PrivateFacilityGID from the ADNUM field in Table CLLI with Priviate Facility encoding. The default value is N.

Sample entry: **>REP PRIVATE_FACILITY_GROUP_USERID Y**

You have defined the PRIVATE_FACILITY_GROUP_USERID parameter.

Step 11: Define the messaging-related parameters RESTRICT_NETBUSY_BUSYCAUSE (end)

The RESTRICT_NETBUSY_BUSYCAUSE parameter, in table CAINPARM, controls the sending of the BusyCause parameter in the Network_Busy EDP-Request message.

Associated logs

Not applicable

Associated OMs

Not applicable

Define the RESTRICT_NETBUSY_BUSYCAUSE parameter

At the CI prompt

1 Enter table CAINPARM.

2 Replace the defined parameter by typing:

>REP RESTRICT_NETBUSY_BUSYCAUSE parmval
where

parmval is the value of the RESTRICT_NETBUSY_BUSYCAUSE parameter (Y or N). A value of N indicates the BusyCause parameter is included in the Network_Busy message. A value of Y indicates the parameter is not included in the message. The default value is N.

Sample entry: **>REP RESTRICT_NETBUSY_BUSYCAUSE Y**

You have defined the RESTRICT_NETBUSY_BUSYCAUSE parameter.

Step 11: Define the messaging-related parameters SEND_CARRIER_FROM_TRKGRP (end)

When you set the SEND_CARRIER_FROM_TRKGRP parameter to Y, the switch uses the carrier (field DEFCIC) datafilled in table TRKGRP to populate the ***Carrier*** parameter when it builds a query message. The default is N.

Note: The value from table TRKGRP is only used when a CIC is not available from a previously received network message.

Define the SEND_CARRIER_FROM_TRKGRP parameter

At the CI prompt

1 Enter table CAINPARM.

2 Replace the parameter by typing:

>REP SEND_CARRIER_FROM_TRKGRP parmval
where

parmval is the parameter value (Y or N)

Sample entry: **>REP SEND_CARRIER_FROM_TRKGRP Y**

The SEND_CARRIER_FROM_TRKGRP parameter is provisioned for NetworkBuilder call processing.

Step 11: Define the messaging-related parameters STR_CONNECTION_TYPE (end)

The STR_CONNECTION_TYPE paramter in table CAINPARM specifies the type of Intelligent Peripheral Interface (IPI) to be used during STR-connection. The default is NONE.

Note: An STR-connection refers to the connection between an IP and the switch that was initiated by a **`Send_To_Resource`** or **`Connect_To_Resource`** message containing a ***`DestinationAddress`*** parameter.

Two connections are currently supported:

- NONE – Specifies that the switch does not allow an STR-connection to an IP. If the switch receives a **`Send_To_Resource`** or **`Connect_To_Resource`** message containing a ***`DestinationAddress`*** parameter, a **`Resource_Clear`** or **`CTR_Clear`** message is sent to the SCP in a conversation package with a ***`ClearCause`*** value of `taskRefused`.
- CONNECT_ONLY – Specifies that the switch allows only a connection to an IP. Data exchange between the SCP and IP through the switch (using ISDN or SS7 messaging) is not supported. For additional information on IPI, refer to Volume 4, Chapter 2, “STR-Connections.”
- CONNECT_1129_STYLE – Provides the ability to create a voice channel connection between the caller and an IP to allow the IP to use its internal resources and functionality to exchange information with the caller. Provides capability for the switch to interwork an exchange of data between an IP and SCP. Also provides the ability for calls to originate and terminate over the IPI.

Define the STR_CONNECTION_TYPE parameter

At the CI prompt

1 Enter table CAINPARM.

2 Replace the parameter by typing:

>REP STR_CONNECTION_TYPE parmval
where

parmval is the parameter value (NONE, CONNECT_ONLY, CONNECT_1129_STYLE).

Sample entry: **>REP STR_CONNECTION_TYPE CONNECT_1129_STYLE**

The STR-Connection type is provisioned for NetworkBuilder call processing.

Step 11: Define the messaging-related parameters TDISC_TIMER (end)

The TDISC_TIMER (IP Disconnect Timer) parameter in table CAINPARM, allows the user to define the maximum time (in seconds) in which the IP must respond to a FACILITY or FAR message with the `cancelIPResource` operation. The default is 4 seconds.

Associated logs

None

Associated OMs

CAINIP

Define the TDISRC_TIMER parameter

At the CI prompt

1 Enter table CAINPARM.

2 Replace the defined parameter by typing:

>REP TDISC_TIMER parmval
where

parmval is timeout value (in seconds).

Sample entry: **>REP TDISC_TIMER 4**

The IP Disconnect timer is provisioned for NetworkBuilder call processing.

Step 11: Define the messaging-related parameters TSTRC_TIMER (end)

The TSTRC_TIMER (STR-Connection timer) parameter in table CAINPARM, allows the user to define the maximum time (in minutes) allowed for an STR-Connection to an IP before assuming an error has occurred. This parameter protects the switch from waiting indefinitely for the IP to respond. The default is 6. Setting the value to 0 disables the timer.

Associated logs

None

Associated OMs

CAINIP

Define the TSTRC_TIMER parameter

At the CI prompt

1 Enter table CAINPARM.

2 Replace the defined parameter by typing:

>REP TSTRC_TIMER parmval
where

parmval is timeout value (in minutes).

Sample entry: **>REP TSTRC_TIMER 6**

The STR-Connection timer is provisioned for NetworkBuilder call processing.

Step 12: Define the billing-related parameters TRTMTCD_COMPCODE_ZAPPED_ZERO (end)

The TRTMTCD_COMPCODE_ZAPPED_ZERO parameter in table CAINPARM allows the Telcos to zap TRTMTCD and COMPCODE fields in CDR to Zero for calls terminated by DISCONNECT message.
Note: Currently, this parameter is used by TBT (Take-Back & Transfer) application only.

Associated logs

None

Associated OMs

None

Define the TRTMTCD_COMPCODE_ZAPPED_ZERO parameter

At the CI prompt

1. Enter table CAINPARM.
2. Replaced the defined parameter by typing:
>REP TRTMTCD_COMPCODE_ZAPPED_ZERO parmval
where
parmval is the value of TRTMTCD_COMPCODE_ZAPPED_ZERO parameter (Y or N).
A value of N indicates that TRTMTCD and COMPCODE fields in CDR will be populated with appropriate values
depending on the treatment set and the type of call completion for calls terminated by DISCONNECT message.
A value of Y indicates that these CDR fields will be zapped to Zero for calls terminated by DISCONNECT message.
The default value is N.
Sample entry: >REP TRTMTCD_COMPCODE_ZAPPED_ZERO Y
You have defined the TRTMTCD_COMPCODE_ZAPPED_ZERO parameter.

Step 13 Enable or disable log generation

The CAIN900_LOGS_ENABLED parameter enables or disables the generation of the following CAIN900 logs:

- CAIN900 – generated when the SCP simulator sends a response to NetworkBuilder
- CAIN901 – generated when the SCP simulator receives an error, abort, or EDP-Notification message from NetworkBuilder
- CAIN902 – generated to identify the current NetworkBuilder subscription method when evaluating trigger criteria
- CAIN903 – generated to identify the CAIN group and trigger criteria met
- CAIN904 – generated when a collectible on the switch is being overridden
- CAIN905 – generated each time a requested event is reached
- CAIN906 – generated when an *Office_Code* trigger is blocked by the NO_LNP option in table CAINSTS
- CAIN907 – generated when the switch detects an unarmed event while the switch and the SCP are handling a call.

Note: The default is set to allow generation of the CAIN900 and CAIN901 logs only.

Define the CAIN900_LOGS_ENABLED parameter

At the CI prompt

1 Enter table CAINPARM.

2 Replace the parameter by typing:

>REP CAIN900_LOGS_ENABLED parmval
where

parmval a vector of enabled log numbers (up to eight values: 900, 901, 902, 903, 904, 905, 906, 907)

Sample entry: **>REP CAIN900_LOGS_ENABLED 900 901 902 903 904 905 906 907**

CAIN900 logs are enabled.

Step 13: Enable or disable log generation (end)

The LNP_LOGS_ENABLED parameter enables or disables the generation of the CAIN301 log.

- CAIN301 – generated when the switch receives a REL message (cause 26) indicating that a call has been misrouted to a ported number, or receives a REL message (cause 28) indicating that the LNP GAP received is not formatted properly. This log will only be generated if LNP is active on the call.

Define the LNP_LOGS_ENABLED parameter

At the CI prompt

1 Enter table CAINPARM.

2 Replace the parameter by typing:

>REP LNP_LOGS_ENABLED parmval
where

parmval the enabled log number (301)

Sample entry: **>REP LNP_LOGS_ENABLED 301**

CAIN301 log is enabled.

Routing

The switch begins attempting to route the call at the **Select_Route** (refer to Volume 3, Chapter 10, "Incoming CAIN messages," for more information). The following terms are used to define CAIN routing:

- **Route indexes** are determined by the data provided in an **Analyze_Route** response or through normal translations. The index points to the appropriate table that identifies a routing list.
- **Route lists** contain the routes to be attempted by the switch when establishing the call. Route lists are provisioned in tables, such as: TANDMRTE, TERMRTE, HNPACONT, FNPACONT, CTRTE, and OFRT. Each route list can contain multiple routes.
- **Routes** typically consists of a route selector, connection type, and common language location identifier (CLLI).
- **CAIN routing parameters** are provided in an **Analyze_Route** response. The parameters are: ***PrimaryTrunkGroup***, ***AlternateTrunkGroup***, ***SecondAlternateTrunkGroup***, ***Carrier***, ***AlternateCarrier***, ***SecondAlternateCarrier***,***CalledPartyID***, and the ***GenericAddressList*** parameter's `OverflowRoutingNo`.

  Note 1: Parameters ***PrimaryTrunkGroup***, ***AlternateTrunkGroup***, and ***SecondAlternateTrunkGroup*** are used to identify direct termination route indexes into tables TANDMRTE and TERMRTE.

  Note 2: Standard routing parameters, ***Carrier***, ***AlternateCarrier***, and ***SecondAlternateCarrier*** are used to identify route indexes.

  Note 3: Standard routing parameters, ***CalledPartyID*** and/or the `servTranslationScheme` (or `univIdx` for SS7 Global-IMTs), and ***GenericAddressList*** parameter's `OverflowRoutingNo` require in-switch translations to derive the route index. One route index is calculated for each parameter.
- **Serving translation scheme (STS)** extension parameters are provided in an **Analyze_Route** response along with CAIN routing parameters. Each CAIN routing parameter has an associated STS extension parameter. Default STS extension parameters are also datafilled in table CAINXDFT to be used when the SCP fails to return the associated STS extension parameter with the CAIN routing parameter. Additionally, the original STS derived prior to the query message may be used when the STS extension parameter is not returned from the SCP and is not datafilled in table CAINXDFT. Figure 2-19 illustrates the order of precedence used to obtain the STS for each CAIN routing parameter.

Step 14: Define routing preferences (continued)

Figure 2-19
Precedence order for STS extension parameters

Route Parameters Returned by the SCP \ STS Extension Parameters	PRISTS	ALTSTS	SALTSTS	STS	OVLFSTS	table CAINXDFT PRISTS	ALTSTS	SALTSTS	STS	OVLFSTS	Pre Query STS
PrimaryTrunkGroup	1			3		2			4		5
AlternateTrunkGroup		1		3			2		4		5
SecondAlternateTrunkGroup			1	3				2	4		5
CalledPartyID				1					2		3
OverflowRoutingNo				3	1				4	2	5

LEGEND

PRISTS (primaryTrunkGroupSTS)	– STS to be used with the ***PrimaryTrunkGroup*** parameter
ALTSTS (alternateTrunkGroupSTS)	– STS to be used with the ***AlternateTrunkGroup*** parameter
SALTSTS (secondAlternateTrunkGroupSTS)	– STS to be used with the ***SecondAlternateTrunkGroup*** parameter
STS (servTranslationScheme)	– STS to be used with the ***CalledPartyID*** parameter
OVFLSTS (overflowRoutingNo)	– STS to be used with the OverflowRoutingNo parameter

Note: The univIdx extension parameter is used to specify the translations scheme for SS7 Global-IMT agents.

- **Standard route advance** – The switch attempts the first route in route list, and when unable to terminate using the route, attempts to establish the call using the next route in the route list.
- **CAIN routing parameter advance** – The switch attempts to route according to data provided in the next untried CAIN routing parameter. When unable to establish the connection using the data, call processing advances to the next NetworkBuilder parameter to derive a new route list and attempts to route the call.

Routing tables

Routing tables are used to define a routing list. The SCP route response provides an index into the routing tables.

Step 14: Define routing preferences (continued)

TANDMRTE

Datafill table TANDMRTE (Tandem Routing) to route through tandem switches within the IEC network to reach the required terminating switch.

TERMRTE

Datafill table TERMRTE (Termination Routing) to route to the terminating switch directly connected to the current switch.

TERMLRN

Provision table TERMLRN (Terminating Location Routing Number) with up to 16 terminating LRNs assigned to the switch. This supports the UCS DMS-250's function of recognizing when the LRN for the call is that assigned to the UCS DMS-250 and, when such is the case, routing the call based on the directory number (DN). Refer to *UCS DMS-250 Local Number Portability Application Guide* for more information.

Limitations and restrictions

- Direct termination through a tandem switch (using table TANDMRTE) is only supported over IMTs that are datafilled to support UCS-to-UCS ISUP protocol. When any other protocol is used, retranslation occurs at the tandem switch.
- Direct terminating routing to a terminating ISUP92/Q.767 trunk is supported through table TERMRTE only.

Step 14: Define routing preferences ACG overflow treatment (end)

The ACG_TREATMENT parameter in table CAINPARM specifies the treatment to be applied to a call that is blocked by an ACG control when the error action provisioned in the applicable trigger table is set to TREAT.

Note: The error action is taken if the ACG_OVERFLOW_GT parameter in table CAINPARM is set to NIL, or if the query was blocked at the overflow global title.

Define the ACG_TREATMENT parameter

At the CI prompt

1 Enter table CAINPARM.

2 Replace the defined parameter by typing:

>REP ACG_TREATMENT parmval
where

parmval is the treatment.

Sample entry: **>REP ACG_TREATMENT AINF**

The ACG_TREATMENT parameter is provisioned.

Step 14: Define routing preferences
Allow redirect tandem threshold exceeded treatment (end)

At a terminating switch, the ALLOW_RTTE_TRTMT parameter controls the type of treatment set for a direct termination routed call when no idle members for a terminating trunk are encountered.

When the ALLOW_RTTE_TRTMT parameter is set to N, a general no circuit (GNCT) treatment is set and a NO_CIRCUIT_AVAILABLE cause value is returned in a REL message to the querying switch.

When the ALLOW_RTTE_TRTMT parameter is set to Y, a redirect tandem threshold exceeded (RTTE) treatment is set and a TERM_RESOURCE_UNAVAILABLE cause value is returned in a REL message back to the querying switch.

Note: Although an RTTE treatment can be set when the querying and terminating switch are the same, the TERMRTE_GNCT criteria is never hit. Instead, a route busy scenario is encountered.

Define the ALLOW_RTTE_TRTMT

At the CI prompt

1 Enter table CAINPARM.

2 Replace the parameter by typing:

>REP ALLOW_RTTE_TRTMT parmval
where

parmval Y or N

Sample entry: **>REP ALLOW_RTTE_TRTMT N**

The ALLOW_RTTE_TRTMT parameter is defined.

Step 14: Define routing preferences
Enable/disable table CLLI matching for table TERMRTE

When the MATCH_TERMRTE_CLLI parameter (table CAINPARM) is set to MATCH, the TERMTRK field in table TERMRTE must match the ADNUM field in table CLLI for the terminating trunk. By matching, only one route can be defined and is limited to the datafill in table CLLI.

When the MATCH_TERMRTE_CLLI parameter is set to NOMATCH, the TERMTRK field is unrelated to the ADNUM field. By not matching table CLLI, multiple route definition is allowed. Therefore, the SCP needs no knowledge of table CLLI.

ATTENTION

This parameter is set to NOMATCH during ONP. In UCS05, values in table TERMRTE could match values in table CLLICDR.

However, starting in UCS06, values (when the parameter is set to MATCH) will match values in table CLLI. Until the SCP is updated to reflect the conversion to table CLLI, routes may not map as expected.

Also, because fields TERMTRK (table TERMRTE) and ADNUM (table CLLI) must match, inconsistencies may exist due to the UCS06 change from table CLLICDR to table CLLI. If the parameter is set to MATCH, you may experience datafill loss during ONP.

Once the SCP is updated and switch datafill is checked, the parameter can be set to MATCH.

Enable/disable table CLLI matching

Note: When CLLI matching is enabled, the CLLI subfield in table TERMRTE must be datafilled with a CLLI defined in table CLLI. If the CLLI doesn't exist in table CLLI, an error message is generated.

At the CI prompt

1 Enter table CAINPARM.

2 Replace the parameter by typing:

>REP MATCH_TERMRTE_CLLI parmvalue
where

parmval is match or nomatch.

Sample entry: **>REP MATCH_TERMRTE_CLLI nomatch**

Step 14: Define routing preferences
Enable/disable table CLLI matching for table TERMRTE (end)

MATCH_TERMRTE_CLLI is disabled.

Step 14: Define routing preferences
Routing out of the IEC network with direct termination (end)

Routing out of the IEC network with direct termination

Note: Table TERMRTE is used for direct termination routing to trunks in the IEC network.

At the CI prompt

1 Enter table TERMRTE.

2 Provision termination route using the following format:

>ADD termtrk route $
where

termtrk	is the terminating trunk number (0–9999).
route	is multiple-entry vector comprised of three subfields: RTESEL, CONNTYPE, and CLLI, where
	RTESEL is the termination route selector (S).
	CONNTYPE is the connection type (D).
	CLLI is the common language location identifier of the terminating trunk (from table CLLI).

Sample entry: **>ADD 215 s d t20lec $**

Standard termination routing is provisioned.

Step 14: Define routing preferences
Routing out of the IEC network with table termination (end)

Routing out of the IEC network with table termination

Note: Table TERMRTE is used for direct termination routing to trunks in the IEC network.

At the CI prompt

1 Enter table TERMRTE.

2 Provision termination route using the following format:

>ADD termtrk route $
where

termtrk is the terminating trunk number (0–9999).
route is multiple-entry vector comprised of two subfields: RTESEL and EXTRTEID, where:
- RTESEL is the termination route selector (T).
- EXTRTEID is comprised of two subfields: TABID and KEY, where:
  - TABID is the index into an external routing table (OFRT, TOPSAMA, TOPS, EXDGTRTE, TTL4, RRTE, OFR4, OFR3, OFR2).
  - KEY is the key for the table datafilled in subfield TABID.

Sample entry: **>ADD 632 t ofrt 102**

Table termination routing is provisioned.

Step 14: Define routing preferences Routing within the IEC network (end)

Routing within the IEC network

Note: Table TANDMRTE is used for direct termination routing to tandem trunks connecting switches in the IEC network.

At the CI prompt

1 Enter table TANDMRTE.

2 Provision routing using the following format:

>ADD switchid route options $

where

switchid is the switch identifier of the terminating switch (0–127).

route is a multiple-entry vector comprised of three subfields: RTESEL, CONNTYPE, and CLLI

Note: A maximum of nine route choices can be defined in the ROUTE field.

RTESEL	is the termination route selector (S).
CONNTYPE	is the connection type (D).
CLLI	is the common language location identifier of the terminating trunk (from table CLLI).

Sample entry: **>ADD 15 s d imt761c7dr22 $ $**

Tandem routing is provisioned.

Step 15: Define default extension parameters

Datafill table CAINXDFT (CAIN Extension Parameter Default) to define default values for the following extension parameters:

- servTranslationScheme
- callType
- satRestriction
- classOfSvc
- callBranding
- networkBusyActions
- oCalledPartyBusyActions
- oNoAnswerActions
- edpBuffer
- univIdx
- netinfo
- primaryTrunkGroupSTS
- alternateTrunkGroupSTS
- secondAlternateTrunkGroupSTS
- overflowRoutingNoSTS
- strConnectionType
- switchHookFlashEnabledLegs
- in1_Requery

Note: The in1_Requery option is added to table CAINXDFT to support SACREMOT IN/1 query functionality for particular numbers subsequent to the receipt of an **Analyze_Route** message. This is not an actual extension parameter.

You can specify extension parameters for each CAIN group. The switch uses the defaults when the SCP fails to return a necessary extension parameter in the response message.

The following sections provide information about the default extension parameter. Refer to Volume 3, Chapter 12, “Incoming CAIN message parameters,” for additional information.

Step 15: Define default extension parameters (continued)

servTranslationScheme

This default contains a serving translation scheme (STS) that replaces the STS defined at the time of the query.

Range of values

000–999

Provision a default servTranslationScheme parameter

At the CI prompt

1 Enter table CAINXDFT.

Note: This procedure defines one default for a single CAIN group. Each CAIN group can have one set of defaults.

2 Add a default servTranslationScheme parameter value using the following format:

>ADD caingrp STS sts_value
where

caingrp is the CAIN group requiring default STS values.
sts_value is the default STS value.

Sample entry: **>ADD caingrp1 sts 761 $**

A default servTranslationScheme parameter has been provisioned.

callType

This default contains the network call type that is populated into the CDR, but has no impact on call processing.

Range of values

ONNET, OFFNET, FORCED_ONNET, VIRTUAL_ONNET

Provision a default callType parameter

At the CI prompt

1 Enter table CAINXDFT.

Note: This procedure defines one default for a single CAIN group. Each CAIN group can have one set of defaults.

2 Add a default call type parameter using the following format:

>ADD caingrp CALLTYPE calltype
where

caingrp is the CAIN group requiring a default call type.
calltype is the call type (OFFNET, ONNET, FORCED_ONNET, VIRTUAL_ONNET).

Sample entry: **>ADD caingrp2 CALLTYPE onnet**

A default call type parameter has been provisioned.

satRestriction

This default, when present indicates the call should not terminate to a satellite-based trunk; replaces the satellite restriction value determined before the SCP query.

Range of values

Presence or Absence

At the CI prompt

1 Enter table CAINXDFT.

Note: This procedure defines one default for a single CAIN group. Each CAIN group can have one set of defaults.

2 Provision satellite restriction using the following format:

>ADD caingrp SATREST
where

caingrp is the CAIN group requiring satellite restriction.

Sample entry: **>ADD caingrp1 SATREST**

Satellite restriction is provisioned.

classOfSvc

This default provides an index into table MULTICOS for COS screening. When datafilled, COS screening is performed.

Range of values

0–2047

Datafill requirements

Provision a default classOfSvc parameter

At the CI prompt

1 Enter table CAINXDFT.

Note: This procedure defines one default for a single CAIN group. Each CAIN group can have one set of defaults.

2 Add a default class of service using the following format:

>ADD caingrp COS multicos_index
where

Step 15: Define default extension parameters (continued)

caingrp is the CAIN group requiring a default call type.
multicos_index is the index into table MULTICOS, which defines COS screening (0–2047).

Sample entry: **>ADD caingrp1 COS 5**

A default class of service parameter has been provisioned.

callBranding

This default identifies an announcement or tone to be played prior to routing. The parameter contains an index into table CAINRSRC, which identifies the appropriate announcement or tone.

Range of values

0–4095

Provision a default callBranding parameter

At the CI prompt

1 Enter table CAINXDFT.

Note: This procedure defines one default for a single CAIN group. Each CAIN group can have one set of defaults.

2 Add a default branding parameter using the following format:

>ADD caingrp CALLBRND cain_rsrc
where

caingrp is the CAIN group requiring a default call type.
cain_rsrc is the index into table CAINRSRC, which identifies the tone or announcement to be played (0–4095).

Sample entry: **>ADD caingrp1 CALLBRND 9**

A default call branding parameter has been provisioned.

netinfo

This default identifies external network ID, network customer group ID, and network class of service.

Range of values

- EXTNETID range: 0–32767
- NETCGID range: 0–4095
- NCOS range: 0–511

Step 15: Define default extension parameters (continued)

Provision a default netinfo parameter

At the CI prompt

1 Enter table CAINXDFT.

Note: This procedure defines one default for a single CAIN group. Each CAIN group can have one set of defaults.

2 Add a default netinfo parameter using the following format:

>ADD caingrp NETINFO extnetid netcgid ncos
where

caingrp	is the CAIN group requiring a default call type.
extnetid	is the external network ID for the call (0–32767).
netcgid	is the network customer group ID for the call (0–4095).
ncos	is the network class of service for the call (0–511).

Sample entry: **>ADD caingrp1 NETINFO 5 100 25**

A default netinfo parameter has been provisioned.

univIdx

This default identifies the universal translation scheme to use for SS7 Global-IMT calls.

Range of values

NIL, AC, PX, CT, FA, OFC, DN, AM, FT, CC, NSC, CTY, NN, VPN.

Provision a default univIdx parameter

At the CI prompt

1 Enter table CAINXDFT.

Note: This procedure defines one default for a single CAIN group. Each CAIN group can have one set of defaults.

2 Add a default univIdx parameter using the following format:

>ADD caingrp UNIVIDX unividx_scheme
where

caingrp	is the CAIN group requiring a default call type.
unividx_scheme	is the universal translation scheme to use for the call.

Sample entry: **>ADD caingrp1 UNIVIDX FT**

A default univIdx parameter has been provisioned.

networkBusyActions

This default identifies actions for the ***Network_Busy*** EDP.

Step 15: Define default extension parameters (continued)

Range of values

- RTEAVAIL: REQUEST, IGNORE, NEXTRTE.
- RTESDONE: REQUEST, IGNORE.
- TERMRTE_GNCT: REQUEST, IGNORE, NEXTRTE, NEXTCNRTE.

Provision a default networkBusyActions parameter

At the CI prompt

1 Enter table CAINXDFT.

Note: This procedure defines one default for a single CAIN group. Each CAIN group can have one set of defaults.

2 Add a default networkBusyActions parameter using the following format:

>ADD caingrp NETBACT rteavail_action rtesdone_action termrtegnct_action
where

caingrp is the CAIN group requiring a default call type.
rteavail_action is the ***Network_Busy*** EDP action to take when an additional route(s) is available for the call.
rtesdone_action is the ***Network_Busy*** EDP action to take when all possible routes have been attempted for the call.
termrtegnct_action is the ***Network_Busy*** EDP action to take when the route choice determined by direct termination routing through table TERMRTE is busy.

Sample entry: **>ADD caingrp1 NETBACT REQUEST IGNORE REQUEST**

A default networkBusyActions parameter has been provisioned.

oCalledPartyBusyActions

This default identifies actions for the ***O_Called_Party_Busy*** EDP.

Range of values

- RTEAVAIL: REQUEST, IGNORE, NEXTRTE, NEXTCNRTE.
- RTESDONE: REQUEST, IGNORE.

Provision a default oCalledPartyBusyActions parameter

At the CI prompt

1 Enter table CAINXDFT.

Note: This procedure defines one default for a single CAIN group. Each CAIN group can have one set of defaults.

2 Add a default oCalledPartyBusyActions parameter using the following format:

>ADD caingrp OCLDBACT rteavail_action rtesdone_action
where

caingrp is the CAIN group requiring a default call type.
rteavail_action is the ***O_Called_Party_Busy*** EDP action to take when an additional route(s) is available for the call.
rtesdone_action is the ***O_Called_Party_Busy*** EDP action to take when all possible routes have been attempted for the call.

Sample entry: **>ADD caingrp1 OCLDBACT IGNORE REQUEST**

A default oCalledPartyBusyActions parameter has been provisioned.

oNoAnswerActions

This default identifies actions for the ***O_No_Answer*** EDP.

Range of values

- RTEAVAIL: REQUEST, IGNORE, NEXTRTE, NEXTCNRTE.
- RTESDONE: REQUEST, IGNORE.

Provision a default oNoAnswerActions parameter

At the CI prompt

1 Enter table CAINXDFT.

Note: This procedure defines one default for a single CAIN group. Each CAIN group can have one set of defaults.

2 Add a default oNoAnswerActions parameter using the following format:

>ADD caingrp ONOANACT rteavail_action rtesdone_action
where

caingrp is the CAIN group requiring a default call type.
rteavail_action is the ***O_No_Answer*** EDP action to take when an additional route(s) is available for the call.
rtesdone_action is the ***O_No_Answer*** EDP action to take when all possible routes have been attempted for the call.

Sample entry: **>ADD caingrp1 ONOANACT REQUEST REQUEST**

A default oNoAnswerActions parameter has been provisioned.

edpBuffer

This default identifies that digits should be buffered when a **`Network_Busy`** or **`O_Called_Party_Busy`** EDP-Request message is sent to the SCP.

Step 15: Define default extension parameters (continued)

Range of values

Presence or Absence

Provision a default edpbuffer parameter

At the CI prompt

1 Enter table CAINXDFT.

Note: This procedure defines one default for a single CAIN group. Each CAIN group can have one set of defaults.

2 Provision EDP buffering using the following format:

>ADD caingrp EDPBUFFR
where

caingrp is the CAIN group requiring EDP buffering.

Sample entry: **>ADD caingrp1 EDPBUFFR**

EDP buffering is provisioned.

primaryTrunkGroupSTS

This default contains a serving translation scheme (STS) associated with the returned ***PrimaryTrunkGroup*** parameter (it replaces the STS defined at the time of the query).

Range of values

000–999

Provision a default primaryTrunkGroupSTS parameter

At the CI prompt

1 Enter table CAINXDFT.

Note: This procedure defines one default for a single CAIN group. Each CAIN group can have one set of defaults.

2 Add a default primaryTrunkGroupSTS parameter value using the following format:

>ADD caingrp PRISTS sts_value
where

caingrp is the CAIN group requiring default PRISTS values.
sts_value is the default STS value.

Sample entry: **>ADD caingrp1 prists 761 $**

A default primaryTrunkGroupSTS parameter has been provisioned.

Step 15: Define default extension parameters (continued)

alternateTrunkGroupSTS

This default contains a serving translation scheme (STS) associated with the returned ***AlternateTrunkGroup*** parameter (it replaces the STS defined at the time of the query).

Range of values

000–999

Provision a default alternateTrunkGroupSTS parameter

At the CI prompt

1 Enter table CAINXDFT.

Note: This procedure defines one default for a single CAIN group. Each CAIN group can have one set of defaults.

2 Add a default alternateTrunkGroupSTS parameter value using the following format:

>ADD caingrp ALTSTS sts_value
where

caingrp is the CAIN group requiring default ALTSTS values.
sts_value is the default STS value.

Sample entry: **>ADD caingrp1 altsts 761 $**

A default alternateTrunkGroupSTS parameter has been provisioned.

secondAlternateTrunkGroupSTS

This default contains a serving translation scheme (STS) associated with the returned ***SecondAlternateTrunkGroup*** parameter (it replaces the STS defined at the time of the query).

Range of values

000–999

Provision a default secondAlternateTrunkGroupSTS paramter

At the CI prompt

1 Enter table CAINXDFT.

Note: This procedure defines one default for a single CAIN group. Each CAIN group can have one set of defaults.

2 Add a default secondAlternateTrunkGroupSTS parameter value using the following format:

>ADD caingrp SALTSTS sts_value
where

Step 15: Define default extension parameters (continued)

caingrp is the CAIN group requiring default SALTSTS values.
sts_value is the default STS value.

Sample entry: **>ADD caingrp1 saltsts 761 $**

A default secondAlternateTrunkGroupSTS parameter has been provisioned.

overflowRoutingNoSTS

This default contains a serving translation scheme (STS) associated with the ***GenericAddressList*** parameter's returned `OverflowRoutingNo` (it replaces the STS defined at the time of the query).

Range of values

000–999

Provision a default overflowRoutingNoSTS parameter

At the CI prompt

1 Enter table CAINXDFT.

Note: This procedure defines one default for a single CAIN group. Each CAIN group can have one set of defaults.

2 Add a default overflowRoutingNoSTS parameter value using the following format:

>ADD caingrp OVFLSTS sts_value
where

caingrp is the CAIN group requiring default OVFLSTS values.
sts_value is the default STS value.

Sample entry: **>ADD caingrp1 ovflsts 761 $**

A default overflowRoutingNoSTS parameter has been provisioned.

strConnectionType

This default identifies the type of connection protocol to be used when establishing a connection between the switch and an IP if the `strConnectionType` extension parameter is not present in the incoming **`Send_To_Resource`** or **`Connect_To_Resource`** message.

Range of values

NONE, CONNECT_ONLY, CONNECT_1129_STYLE

Provision a default strConnectionType parameter

At the CI prompt

1 Enter table CAINXDFT.

Step 15: Define default extension parameters (continued)

Note: This procedure defines one default for a single CAIN group. Each CAIN group can have one set of defaults.

2 Add a default strConnectionType parameter using the following format:

>ADD caingrp STRCONTP str_conn_type
where

caingrp is the CAIN group requiring a default str connection type.
str_conn_type is the type of connection protocol to use when estabilishing a connection with an IP (NONE, CONNECT_ONLY, CONNECT_1129_STYLE).

Sample entry: **>ADD caingrp1 STRCONTP connect_1129_style**

A default strConnectionType parameter has been provisioned.

switchHookFlashEnabledLegs

This default specifies which call legs can send an **`O_Mid_Call`** EDP-Request message for the *Switch_Hook_Flash* event.

Range of values

Leg0, Leg1 (vector of up to 2)

Provision a default switchHookFlashEnabledLegs parameter

At the CI prompt

1 Enter table CAINXDFT.

Note: This procedure defines one default for a single CAIN group. Each CAIN group can have one set of defaults.

2 Add a default switchHookFlashEnabledLegs parameter using the following format:

>ADD caingrp SHFELEGS shfeleg_type
where

caingrp is the CAIN group requiring a default call leg(s) to for the switch to monitor.
shfeleg_type is the call legid(s) to monitor (Leg0, Leg1).

Sample entry: **>ADD caingrp1 SHFELEGS Leg0**

You have provisioned a default switchHookFlashEnabledLegs parameter.

in1_requery

This default allows the switch to send a SACREMOT IN/1 query after the switch has received an **`Analyze_Route`** message for a NetworkBuilder query.

Step 15: Define default extension parameters (end)

Range of values

Presence or Absence

Provision a default in1_requery

At the CI prompt

1 Enter table CAINXDFT.

Note: This procedure defines one default for a single CAIN group. Each CAIN group can have one set of defaults.

2 Add a default in1_requery using the following format:

>ADD caingrp IN1_REQUERY
where

caingrp is the CAIN group requiring the switch to perform an IN/1 query after the switch receives an Analyze_Route message for a NetworkBuilder query.

Sample entry: **>ADD caingrp1 IN1_REQUERY**

You have provisioned in1_requery.

Step 16: Define NetworkBuilder resources

The switch plays tones and announcements as directed by NetworkBuilder when:

- the PLAY_ANN option in table ONOANSWR identifies a NetworkBuilder resource
- table OFTRREQ datafill identifies a NetworkBuilder resource
- the SCP returns the `callBranding` extension parameter (that contains a NetworkBuilder resource) in an **`Analyze_Route`** message
- the SCP returns a **`Send_To_Resource`**, **`Connect_To_Resource`**, or **`Play_Announcement`** message that identifies the NetworkBuilder resources

Note: The **`Play_Announcement`** message is only supported as a valid response to the Bellcore *TR-NWT-000533* **`Start`** message.

NetworkBuilder resources are datafilled in table CAINRSRC. The NetworkBuilder resource identifier (a number between 0 and 4095) is mapped to an announcement CLLI datafilled in table DRAMTRK or a tone CLLI datafilled in table TONES.

Supported announcement types for resources in table CAINRSRC are standard (STND), calling number announcement trunks (CNAT), and variable phrase standard announcements (VPSA). These types are provisioned in table ANNS.

Note: For more information on messages, refer to the *Digital Recorded Announcement Machine DRAM and EDRAM Guide*.

The ***`StandardAnnouncement`*** parameter for the **`Play_Announcement`** message contains a code corresponding to the announcement element that is to be played. The code is used as an index into table CAINRSRC. In order to be compliant with the Bellcore *TR-NWT-000533* specificiation for the ***`StandardAnnouncement`*** parameter, table CAINRSRC must be provisioned with the following announcement types and matching indexes.

Step 16: Define NetworkBuilder resources (end)

Table 2-22
Announcement type indexes for table CAINRSRC

Index	Announcement type
01	Out of band
02	Vacant code
03	Disconnected number
04	Reorder (120 IPM (impulses per minute))
05	Busy (60 IPM)
06	No circuit available
07	Reorder (announcement)
08	Audible ring

—end—

Datafilling a CAIN resource

At the CI prompt

1 Enter table CAINRSRC.

2 Add a NetworkBuilder resource by entering:

>ADD rsrcid rsrcclli
where

rsrcid is the resource identifier used by NetworkBuilder call processing to identify a resource (0 – 4095)

rsrcclli is a valid CLLI datafilled in table CLLI and either table DRAMTRK or table TONES.

Sample entry: **>ADD 1 tstone**

A NetworkBuilder resource is datafilled.

Step 17: Enable SOC options

Software Optionality Control (SOC) enables software to be defined and delivered in product computing-module loads (PCL). All functionality in a PCL is categorized as either base or optional. Base functionality is available for immediate use. Optional functionality is grouped into commercial units called SOC options, which can be purchased by operating companies. SOC options correspond to functional groups and functions and are controlled by Nortel-supplied passwords.

Refer to Volume 5, Chapter 5, "NetworkBuilder SOC functionality," for more information on NetworkBuilder SOCs and SOC dependencies.

The following order codes are necessary for full implementation of NetworkBuilder.

Table 2-23
NetworkBuilder SOC order codes

Order code	SOC name	Available functionality when SOC is on
CAIN0100	Messages	Usage option for all NetworkBuilder messages to the SCP
CAIN0200	Extension Parms	Extension parameter set
CAIN0300	SCP Simulator	SCP simulator to test NetworkBuilder functionality
CAIN0400	Test Query Tool	Ability to send test messages (CAINTEST)
CAIN0500	CUSTDP Trigger	Controls the Customized_Dialing_Plan trigger
CAIN0501	SPECDIG Trigger	Controls the Specific_Digit_String Trigger
CAIN0502	OFFHKIM Trigger	Controls the Off_Hook_Immediate Trigger
CAIN0503	SIOTRK Trigger	Controls the Shared_Interoffice_Trunk Trigger
CAIN0504	PRIBCHNL Trigger	Controls the PRI_B-Channel Trigger
CAIN0505	ONOANSWER Trigger	Controls the O_No_Answer Trigger
CAIN0506	NETBUSY Trigger	Controls the Network_Busy Trigger
CAIN0507	OCLDBUSY Trigger	Controls the O_Called_Party_Busy Trigger
CAIN0508	OFTRREQ Trigger	Controls the O_Feature_Requested Trigger
CAIN0509	OIECREO Trigger	Controls the O_IEC_Reorigination Trigger
CAIN0510	TERMATT Trigger	Controls the Termination_Attempt Trigger

—continued—

Step 17: Enable SOC options (continued)

Table 2-23
NetworkBuilder SOC order codes (continued)

Order code	SOC name	Available functionality when SOC is on
CAIN0511	SPECFEAT Trigger	Controls the Specific_Feature_Code Trigger
CAIN0512	OFFHKDEL Trigger	Controls the Offhook_Delay Trigger
CAIN0513	TOLLFREE Trigger	Controls the Tollfree_Service Trigger
CAIN0600	Con Digit Collect	Controls NetworkBuilder conversational digit collection
CAIN0601	SCP Trigger Sub	Controls NetworkBuilder SCP Trigger Subscription
CAIN0602	EDPs	Controls Network_Busy, O_Term_Seized, O_Called_Party_Busy, O_Answer, and O_No_Answer EDPs
CAIN0603	STR Connection	Controls the STR connection
CAIN0604	Inter IMT Support	Controls triggering for SS7 Inter-IMTs
CAIN0605	Global IMT Support	Controls triggering for SS7 Global-IMTs
CAIN0606	1129-Style IP	Controls 1129-Style IP interaction
CAIN0607	Virtual IP	Controls Virtual IP interaction
CAIN0609	Term Notification	Controls Termination Notification
CAIN0610	CainPrt Digit Coll	Controls the CAIN Pretranslator Digit Collectible
CAIN0700	LNP QOO	Controls triggering for Local Number Portability
CAIN0800	Mid Call Services 1	Controls the Timeout and O_Disconnect Events
CAIN0801	Mid Call Services 2	Controls Connect_To_Resource message processing at the Timeout event
CAIN0802	Takeback & Transfer	Controls multi-party calls, the *Switch_Hook_Flash* event, and the ***O_Abandon*** EDP
CAIN0900	Auto Code Gapping	Controls Automatic Code Gapping
CAIN0901	Manual Code Gapping	Controls Manual Code Gapping

—end—

Step 17: Enable SOC options (end)

Enabling a SOC option

At the CI prompt

1 Enter the SOC command set by entering:

>SOC

2 Assign a key code using the following format:

>ASSIGN RTU keycode TO soc_option
where

keycode is the password that will enable the SOC.
soc_option is the SOC option you want to enable.

3 Activate the SOC state option:

>ASSIGN STATE ON TO soc_option
where

soc_option is the SOC option you want to enable.

Sample entry: **>ASSIGN STATE ON TO cain0502**

A SOC option is enabled.

O_Null PIC

Call processing enters **O_Null** when the switch detects off-hook.

Note: Call processing enters call configuration 1 (originating two-party setup) when the switch detects off-hook. Refer to Volume 3, Chapter 4, "Call Configuration Model," for more information on call configurations.

Origination_Attempt TDP

Call processing encounters ***Origination_Attempt*** when the following conditions are met:

- The switch is prepared to collect digits.
- Originating agent and time data is captured for population into the CDR.

  Note 1: The originating agency data is populated into the ORIGGRP and ORIGMEM fields of the CDR.

  Note 2: The originating time data is populated into the ORIGAMPM, ORIGDATE, and ORIGTIME fields of the CDR.

When ***Origination_Attempt*** is encountered, NetworkBuilder directs the switch to check the following:

- call subscription to a CAIN group through table TRKGRP or CAINPARM

Note: CAIN group subscription for AXXESS agents is handled differently. Refer to *UCS DMS-250 CAIN/FlexDial Interactions*.

- enabled triggers (Triggers are enabled through table CAINGRP.)

Note: NetworkBuilder software supports the *Off_Hook_Immediate* trigger at ***Origination_Attempt***.

Off_Hook_Immediate trigger

ATTENTION

The *Off_Hook_Immediate* trigger requires the CAIN0502 SOC option. Refer to Volume 5, Chapter 5, "NetworkBuilder SOC functionality," for more information.

Uses

When you provision NetworkBuilder services on the switch, the SCP can make major call processing decisions. You define the services offered on the SCP. Some examples of services that can be developed on the SCP for the *Off_Hook_Immediate* trigger are:

- hotline calls

Note: Hotline calls are routed to predefined connections the moment a call goes off-hook.

- automatically queries with all digit collection, authorization, and route determination performed by the SCP (Billing data and routing information is returned to the switch for call completion.)
- automatic call blocking

Limitations and restrictions

When a call queries from *Off_Hook_Immediate* and the SCP responds with an **`Analyze_Route`** response, the SCP must return a ***`ChargeNumber`*** with a nature of address (NOA) of I2ANI or ANI, or a ***`ChargeNumber`*** with a ***`ChargePartyStationType`***, if interaction during later points in the call model, such as busy triggers, is required. The SCP must provide either the ***`CalledPartyID`*** or ***`OutpulseNumber`*** parameter.

Note 1: If the NOA is set to ANI, INFO digits can only match on 00.

Note 2: Call processing cannot meet digit criteria for a digit type of ADDR if the call queried at *Off_Hook_Immediate* and the SCP responded with an **`Analyze_Route`**.

Supported originating agencies

The *Off_Hook_Immediate* trigger supports originating DAL loop starts and ground starts.

Note 1: DAL TIE originating agencies are not supported.

Note 2: An AXXESS agent can be datafilled to mimic a DAL agent. Refer to *UCS DMS-250 CAIN/FlexDial Interactions* for more information.

Subscribing to the Off_Hook_Immediate trigger

Subscription to the *Off_Hook_Immediate* trigger is available on an originating agency basis (through table TRKGRP) or on an office basis (through table CAINPARM).

Note: Subscription for AXXESS agents is handled differently. Refer to *UCS DMS-250 CAIN/FlexDial Interactions*.

Trigger evaluation

No trigger criteria is datafilled for the *Off_Hook_Immediate* trigger.

Trigger actions

NetworkBuilder call processing supports the following actions for the *Off_Hook_Immediate* trigger:

- BLOCK – AINF treatment is applied.
- IGNORE – NetworkBuilder continues checking the remaining subscription methods for *Off_Hook_Immediate*. If datafill does not enable the trigger, call processing continues through the call model, and the appropriate dialtone (example: field DIALTONE, table TRKGRP) is applied to the call.
- QUERY – Switch suspends dial tone; switch halts digit collection and an **`Origination_Attempt`** query is built and sent to the SCP. The SCP analyzes the call and assists the switch with call processing.
- QUERYSCU – the UCS DMS-250 switch discontinues NetworkBuilder trigger processing. The call enters server mode and becomes a Programmable Service Node (PSN) call and a **`New_Call`** message is sent to the PSN SCU. The call is no longer a NetworkBuilder call. Refer to *UCS DMS-250 Programmable Service Node (PSN) Application Guide* for more information on PSN.

  Note: If an in-switch error occurs before all digits are present the call goes to treatment and does not become a PSN call as it would have under the existing NetworkBuilder functionality.

- LEAVE_TDP – NetworkBuilder call processing exits the ***Origination_Attempt*** TDP with no further evaluation, call processing continues through the call model, and the appropriate dialtone (example: field DIALTONE, table TRKGRP) is applied to the call.

Off_Hook_Immediate trigger (continued)

- CONT_NOTRIG – NetworkBuilder call processing exits the ***Origination_Attempt*** TDP and prevents any further NetworkBuilder interaction for the call. This prevention is reset upon reorigination. Call processing continues through the call model, and the appropriate dialtone (example: field DIALTONE, table TRKGRP) is applied to the call.

Error actions

When a fatal application occurs during an SCP query, one of the following error actions is performed (as datafilled):

- TREAT – AINF treatment is applied.
- ROUTE – The switch discards any SCP-provided data. NetworkBuilder continues checking the remaining subscription methods for *Off_Hook_Immediate*. If datafill does not enable the trigger, call processing continues through the call model and the appropriate dialtone (field DIALTONE, table TRKGRP) is applied to the call.

  Note: The switch discards any previously buffered digits.

Options

Off_Hook_Immediate supports the following options:

- BUFFER – to activate digit buffering while the SCP is queried
- GT – to identify the global title used to identify the SCP handling the query
- T1OVFLGT – to identify the specific SCP to query on T1 overflow
- ACGOVFLGT – to identify the global title to use for requerying when a query is blocked by an ACG control
- VERSION – controls the CAIN protocol version for outgoing messages

Note 1: When you provision the GT option with the CAIN_ADDR_GT global title translation type, the switch assumes zero (0) is the address for the global title.

Note 2: The value provisioned in the VERSION option overrides the version provisioned in the CAIN_PROTOCOL_VERSION parameter of table CAINPARM on a transaction-by-transaction basis. Additionally, the VERSION option applies to both the outgoing and incoming messages within the same transaction. (Subsequent queries will use the setting provisioned in the CAIN_PROTOCOL_VERSION or provisioned setting of the VERSION option in the trigger table performing the subsequent query.)

- STREAM – to control protocol stream on a per-trigger tuple basis. The value of the STREAM option controls the set of parameters that are sent in NetworkBuilder messages.

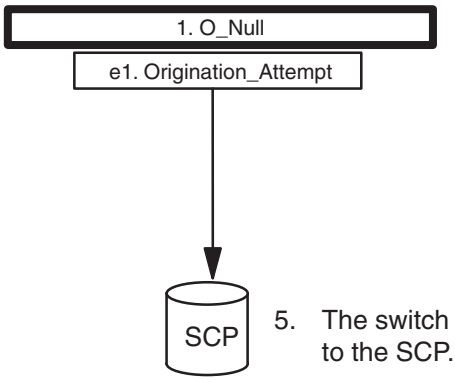
The following figure shows how a call progresses through the call model, encounters **`Origination_Attempt`** and queries the SCP.

DAL origination call

DAL origination (ground or loop start) **Signalling:** FXO and FXS
Datafill:

Table OFFHKIMM

OFTRREQKY	ACTION	OPTION
CAINGRP1	QUERY TREAT	$



1. The switch detects off-hook and enters **O_Null**.
2. The switch encounters **Origination_Attempt**.
3. Datafill enables *Off_Hook_Immediate* for CAINGRP1.
4. Trigger criteria is met in table OFFHKIMM.
5. The switch builds an **`Origination_Attempt`** query and sends it to the SCP.

Origination_Attempt TDP-Request

An **`Origination_Attempt`** TDP-Request (query) is sent to the SCP in a query package, with a component type of Invoke_Last. The following table defines the parameters and usage requirements for the parameters the **`Origination_Attempt`** TDP-Request may contain. Refer to Volume 3, Chapter 8, "Outgoing CAIN message parameters," for detailed descriptions of the request parameters.

Note: Please note that parameters, extension parameters, and their population may differ for AXXESS agents. Refer to *UCS DMS-250 CAIN/FlexDial Interactions* for more information.

Off_Hook_Immediate trigger (continued)

Origination_Attempt TDP-Request message parameters

Parameter	Usage	Definition
UserID	Required	Contains the network identity of the originating agent
BearerCapability	Required	Contains the bearer capability name (field BCNAME) datafilled in table TRKGRP
ChargeNumber (Note 3 and 5)	Optional	Contains the digits in field VAUTHFLD of table TRKGRP, when available
Lata (Note 6)	Optional	Contains the call's Local Access and Transport Area (LATA)
TriggerCriteriaType	Optional	Contains offHookImmediate
CallingPartyID (Note 7)	Optional	Contains the SNPA datafilled in table TRKGRP (for the originating agency), when available, or the default SNPA provisioned in table CAINPARM
Carrier	Optional	Contains the CIC value (with an indication of: Selected CIC presubscribed and not input by calling party) from table TRKGRP
ACGEncountered	Optional	Contains the ACG control encountered when the ACG control expires and a query is allowed to be sent
ExtensionParameter	Optional	Extension parameters require the CAIN0200 SOC option.

Note 1: The parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS08 or higher.
Note 2: The default `jurisdictionInfo` can be datafilled against the agent in table TRKGRP.
Note 3: If the CAIN_PROTOCOL_VERSION is set to V3 or lower, and if the ***ChargeNumber*** has been populated with an ANI, then the NOA is NATL, and the ***ChargePartyStationType*** parameter contains the Information Digits from the call.
Note 4: The parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS09 or higher.
Note 5: When the CAIN_PROTOCOL_VERSION is set to V4 or higher non-standard charge numbers (CARD, AUTH, ACCT, PIN, and N00) no longer populate the ***ChargeNumber*** parameter and all standard NOAs are supported.
Note 6: The parameter is supported only on DAL, FGD, and AXXESS agencies and the parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS11 or higher.
Note 7: Refer to Volume 3, Chapter 8, "Outgoing CAIN message parameters" for information on the population of this parameter when the CAIN_PROTOCOL_VERSION is set to V5 or higher.

—continued—

Off_Hook_Immediate trigger (continued)

Origination_Attempt TDP-Request message parameters (continued)

Parameter	Usage	Definition
cainGroup	Optional	Contains the group number (field GRPNUM, table CAINGRP) for the CAIN group associated with the trigger when field EXTPARM in table CAINGRP contains CAINGRP
adin	Optional	Contains the authorization code database index (field ADIN, table TRKGRP, DAL agencies) associated with the originating agency (filed authcodes only) when field EXTPARM in table CAINGRP contains ADIN ***Note:*** The adin extension parameter is not supported for AXXESS agents.
origTrunkInfo	Optional	Contains the originating trunk group number, trunk type, and trunk member number when field EXTPARM in table CAINGRP contains ORGTINFO
t1Overflow	Optional	Indicates to the SCP that an overflow occurred on the query to the initial SCP
subscriptionInfo (Note 1)	Optional	Contains the digit type the switch triggered on and which subscription method was in use when the query occurred
jurisdictionInfo (Note 1)	Optional	Contains the originating switch's LRN

Note 1: The parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS08 or higher.
Note 2: The default jurisdictionInfo can be datafilled against the agent in table TRKGRP.
Note 3: If the CAIN_PROTOCOL_VERSION is set to V3 or lower, and if the ***ChargeNumber*** has been populated with an ANI, then the NOA is NATL, and the ***ChargePartyStationType*** parameter contains the Information Digits from the call.
Note 4: The parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS09 or higher.
Note 5: When the CAIN_PROTOCOL_VERSION is set to V4 or higher non-standard charge numbers (CARD, AUTH, ACCT, PIN, and N00) no longer populate the ***ChargeNumber*** parameter and all standard NOAs are supported.
Note 6: The parameter is supported only on DAL, FGD, and AXXESS agencies and the parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS11 or higher.
Note 7: Refer to Volume 3, Chapter 8, "Outgoing CAIN message parameters" for information on the population of this parameter when the CAIN_PROTOCOL_VERSION is set to V5 or higher.

—continued—

Off_Hook_Immediate trigger (continued)

Origination_Attempt TDP-Request message parameters (continued)

Parameter	Usage	Definition
collectedAddress (Note 1)	Optional	Contains the address collected from the incoming agent message, through subscriber dialing, through datafilled hotline digits, or through the *O_Feature_Requested* trigger
switchID (Note 4)	Optional	Contains the switch ID
billingNumber (Note 4)	Optional	Contains the non-standard charge number
acgRequery	Optional	When present, indicates to the SCP that the current query is a re-query message resulting from the original query being blocked by an ACG control

Note 1: The parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS08 or higher.
Note 2: The default jurisdictionInfo can be datafilled against the agent in table TRKGRP.
Note 3: If the CAIN_PROTOCOL_VERSION is set to V3 or lower, and if the ***ChargeNumber*** has been populated with an ANI, then the NOA is NATL, and the ***ChargePartyStationType*** parameter contains the Information Digits from the call.
Note 4: The parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS09 or higher.
Note 5: When the CAIN_PROTOCOL_VERSION is set to V4 or higher non-standard charge numbers (CARD, AUTH, ACCT, PIN, and N00) no longer populate the ***ChargeNumber*** parameter and all standard NOAs are supported.
Note 6: The parameter is supported only on DAL, FGD, and AXXESS agencies and the parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS11 or higher.
Note 7: Refer to Volume 3, Chapter 8, “Outgoing CAIN message parameters” for information on the population of this parameter when the CAIN_PROTOCOL_VERSION is set to V5 or higher.

—end—

Once the request message is built, NetworkBuilder formats the message and sends it to the message encoder. Once encoded, the message is sent to the SCP.

SCP response processing

- **Analyze_Route**
- **Continue**
- **Send_To_Resource**

  Note 1: EDPs may be armed through the **Request_Report_BCM_Event** component with **Analyze_Route** and **Continue** messages.

  Note 2: A **Termination_Notification** may be requested through the **Send_Notification** component with the SCP response message.

Note 3: An **ACG** message may be sent with the SCP response message.

Refer to Volume 3, Chapter 10, "Incoming CAIN messages," for specific information regarding response messages; refer to Volume 3, Chapter 12, "Incoming CAIN message parameters," for detailed descriptions of the response parameters; refer to Volume 3, Chapter 3, "Event processing," for EDP specific messaging information.

Analyze_Route message

When an **Analyze_Route** message is received in response to an **Origination_Attempt** query, the following requirements exist:

- The SCP must provide a ***ChargeNumber*** parameter, since no validation is performed before the query is sent.
- The SCP must provide either the ***CalledPartyID*** or ***OutpulseNumber*** parameter.

  Note 1: If the ***CalledPartyID*** is not provided by the SCP, the CALLEDNO field of the CDR will be empty.

  Note 2: When an ***OutpulseNumber*** is received (along with a ***PrimaryTrunkGroup***, ***AlternateTrunkGroup***, or ***SecondAlternateTrunkGroup***), it (the ***OutpulseNumber***) is not translated in-switch. The SCP has identified a route list for direct termination routing. Therefore the ***OutpulseNumber*** is not used for standard route determination. However, once the call leaves the network (as directed by table TERMRTE), the ***OutpulseNumber*** can be used to route the call.

  Note 3: If the **Analyze_Route** message does not contain either the ***CalledPartyID*** or ***OutpulseNumber*** parameter, a fatal application error is detected. NetworkBuilder generates a CAIN200 log, due to missing conditional parameters, and performs the provisioned error action.
- When the SCP does not provide an STS, and no STS is datafilled in table CAINXDFT, then office parameter DEFAULT_STS (table OFCVAR) is used for the remainder of the call.

Send_To_Resource message

When the BUFFER option is set in table OFFHKIMM, dialed digits are buffered and sent to the user interaction framework (UIF) for conversational digit collection.

Any digits (up to 24 digits) dialed, before a conversational **Send_To_Resource** message is received by the switch, are buffered and delivered to the SCP in a **Resource_Clear** message.

Off_Hook_Immediate trigger (continued)

If an interruptible announcement is identified in the conversational **`Send_To_Resource`** message, any digit buffering prior to receipt aborts resource play.

If an uninterruptible announcement is identified, buffered digits are discarded, and digit collection (when required) is started after resource completion.

Continue message

When the switch receives a **`Continue`** message in response to an **`Origination_Attempt`** query, the following occur:

1 The switch discards any previously buffered digits.

2 The permanent signal timer is reset.

3 NetworkBuilder checks the remaining subscription methods for *Off_Hook_Immediate*.

4 Once NetworkBuilder has checked all subscription methods, the switch applies the appropriate dialtone (field DIALTONE, table TRKGRP).

5 The switch initiates digit collection.

Note: If a delay occurs before the **`Continue`** message is returned, the call will appear to have a delayed dialtone.

Fatal application errors

Fatal application errors occur when NetworkBuilder call processing is unable to continue due to an unexpected error. The following table lists fatal application errors that may occur and are associated with the *Off_Hook_Immediate* trigger.

Off_Hook_Immediate fatal application errors

Error type	Log generated	Reported to SCP?	Error action performed
Required parameters missing from **`Analyze_Route`** message in response to **`Origination_Attempt`** query	CAIN200	Yes, the switch sends an error cause set to "missingConditionalParameter"	ERRACT from table OFFHKIMM
Note: Refer to *UCS DMS-250 Logs Reference Manual* for information on the CAIN200 log.			

Datafill

The following figure shows how a subscription table interacts with the *Off_Hook_Immediate* trigger table (OFFHKIMM).

Subscription-OFFHKIMM table interaction

Table TRKGRP

GRPKEY	GRPTYP	TRAFSNO	PADGRP	NCCLS	GRPINFO (COS DIR PRTNM SELSEQ ODSFLTR ORIGFLTR TDSCFLTR ANSWFLTR DIALTONE DIGSOUTP RETOFFHK TRAFCLS AUTHDIAL FASTIDGT OPART RECALLDT SNPA TIMEBIAS VAUTHFLD ZEROMPOS ZONE BCNAME AUTHFRST ADIN AIOD ONNETTRK PANIVAL PANIINFO TSUSR OPTION)
DAL221TWDTLS	DAL	30	NPDGP	NCIT	0 2W DAL IDL 16 7 16 16 S 7 NIL ID 0 7 111 MANUAL 214 0 6113311 RTE2 0 3_1KHZ Y 1 N Y NONE 00 160 (CAIN) (CAINGRP CAINGRP1)

Table CAINGRP

CAINGRP	GRP NUM	MSGSET	PROTO	TRIGRSET (PIC TDP TRIGGER)	OPTIONS	EXT PARMS	TEXT PARMS
CAINGRP1	2	CAIN02	TCAP_SCCP	O_NULL ORIGATT OFFHKIMM	$	$	$

Table OFFHKIMM

OFFHKIMMKY (CAINGRP)	ACTION (TRIGACT ERRACT)	OPTIONS
CAINGRP1	QUERY TREAT	$

Provisioning the Off_Hook_Immediate trigger

At the CI prompt

1 Provision the originating agent as CAIN-capable.

2 Subscribe to a CAIN group (table TRKGRP or CAINPARM).

Off_Hook_Immediate trigger (continued)

Note: CAIN group subscription is handled differently for AXXESS agents. Refer to *UCS DMS-250 CAIN/FlexDial Interactions*.

3 Datafill CAIN group trigger subscription to the (O_NULL ORIGATT OFFHKIMM) trigger set (table CAINGRP).

4 Enter table OFFHKIMM.

5 Define the trigger criteria for a CAIN group by using the following format:

>ADD offhkimmkey action options
where

offhkimmkey is comprised of one subfield: CAINGRP, where
CAINGRP is the CAIN group trigger criteria (from table CAINGRP).

action is comprised of one subfield: TRIGACT, where
TRIGACT is the trigger action taken (BLOCK, IGNORE, QUERY, QUERYSCU, LEAVE_TDP, CONT_NOTRIG).

If you datafill TRIGACT as:	Go to:
BLOCK	step 7
IGNORE	step 7
QUERY	step 6
QUERYSCU	step 7
LEAVE_TDP	step 7
CONT_NOTRIG	step 7

6 Datafill the ERRACT refinement, where

ERRACT is the error action taken when a fatal application error occurs (TREAT, ROUTE).

7 Datafill the OPTIONS refinement, where

options is only allowed when ACTION is QUERY. Enter up to 6 options: BUFFER, GT, T1OVFLGT, ACGOVFLGT, VERSION, or STREAM. When the ACTION is not QUERY enter $.

Sample entry: **>ADD caingrp1 block $**

Sample entry: **>ADD caingrp1 query treat buffer $**

Off_Hook_Immediate trigger (end)

Off_Hook_Immediate trigger is defined.

Associated OMs

CAINTRIG, CAINMSGS, CAINAGOM

Collect_Information PIC

Call processing enters **Collect_Information** once the following are completed:

- The originating agent is identified.
- CDR is started and contains data such as: ORIGTIME, ORIGDATE, ORIGMEM, and ORIGAMPM.
- The resources required to handle the call are allocated.
- Authorize origination is detected.

O_Abandon EDP

Call processing encounters the ***O_Abandon*** EDP during the **Collect_Information**, **Analyze_Information**, **Select_Route**, **Send_Call**, and **O_Alerting** PICs when three requirements are met:

- EDPs are active
- the call is in the second call segment
- the calling party disconnects before the called party answers

The *O_Abandon* event is detected.

O_Feature_Requested TDP

Call processing encounters ***O_Feature_Requested*** when the address is collected.

Note: Once the address is collected, trigger criteria is examined, and a query is sent to the SCP, if required. NetworkBuilder does not wait for completion of the dialing plan.

The switch ignores any normal digit collection after the address is collected.

> **ATTENTION**
> Call processing encounters ***O_Feature_Requested*** whenever the switch collects an address (for example, universal access, speed dial, and subscriber addresses). Therefore, trigger evaluation may occur multiple times within one call.

When ***O_Feature_Requested*** is encountered, NetworkBuilder directs the switch to check for the following:

- call subscription to a CAIN group through table STDPRTCT, authcode tables, ANI tables, TRKGRP, CALLATTR, or CAINPARM

Note: CAIN group subscription for AXXESS agents is handled differently. Refer to *UCS DMS-250 CAIN/FlexDial Interactions*.

- enabled triggers (Triggers are enabled through table CAINGRP.)

Note: NetworkBuilder software supports the *O_Feature_Requested* trigger at ***O_Feature_Requested***.

Info_Collected TDP

Call processing encounters ***Info_Collected*** when the following conditions are met:

- dialing plan is identified and executed
- all digits are collected, but not validated
- authorization code screening is performed (in-switch or off-board, as needed) to identify the dialing plan
- IN/1 screening for account codes and N00 addresses has not been performed
- ANI/PANI screening is performed (as needed) to identify the dialing plan
- preliminary screening may assign treatment to the call (However, treatment is ignored and a query is sent, when applicable. The query contains the `treatment` extension parameter.)
- SETUP message received (PRI originations)
- IAM received (SS7 originations)

When ***Info_Collected*** is encountered, NetworkBuilder directs the switch to check the following:

- call subscription to a CAIN group through table STDPRTCT, authcode tables, ANI tables, TRKGRP, CALLATTR, or CAINPARM
- enabled triggers (Triggers are enabled through table CAINGRP.)

Note: NetworkBuilder software supports the *Tollfree_Service*, *Offhook_Delay*, *Shared_Interoffice_Trunk*, and *PRI-B-Channel* triggers at ***Info_Collected***.

Failed screening

For failed address, ANI, and authorization code screening during **Collect_Information**, the switch performs the following:

1 collect address, ANI, and authcode

2 screen digits

— switch collects more digits if required by the switch

— NetworkBuilder checks the call for a CAIN group

— switch logic is executed to determine an STS

3 Screening results are determined. If screening fails, the switch sets the treatment, but suspends application of the treatment.

4 NetworkBuilder checks the trigger criteria in the appropriate trigger table.

— If trigger criteria is not met, or the action is LEAVE_TDP, CONT_NOTRIG, or IGNORE the switch applies the determined treatment to the call.

— If trigger criteria is met and the action is QUERY or FEAT, a query is sent to the SCP. The query includes the treatment (as set by the switch) in the `treatment` extension parameter.

Note: Extension parameters require the CAIN0200 SOC option.

— If trigger criteria is met, and the action is BLOCK, the switch applies AINF treatment.

— If the SCP returns a **`Disconnect`** with the `treatment` extension parameter, the switch applies the SCP-supplied treatment and disconnects the call.

— If the SCP returns a **`Disconnect`** without the `treatment` extension parameter, the switch applies AIND treatment and disconnects the call.

Subscription

The subscription method is determined in the following order:

1 SCP-returned CAIN groups, enabling the *Tollfree_Service* trigger

2 Address subscription, enabling the *Tollfree_Service* trigger

3 Authorization code subscription, enabling the *Tollfree_Service* trigger

4 ANI subscription, enabling the *Tollfree_Service* trigger

5 Agent subscription, enabling the *Tollfree_Service* trigger

6 Office subscription, enabling the *Tollfree_Service* trigger

7 SCP-returned CAIN groups, enabling the *Offhook_Delay* trigger

8 Address subscription, enabling the *Offhook_Delay* trigger

9 Authorization code subscription, enabling the *Offhook_Delay* trigger

10 ANI subscription, enabling the *Offhook_Delay* trigger

11 Agent subscription, enabling the *Offhook_Delay* trigger

12 Office subscription, enabling the *Offhook_Delay* trigger

13 SCP-returned CAIN groups, enabling the *Shared_Interoffice_Trunk* or *PRI_B-Channel* trigger

14 Address subscription, enabling the *Shared_Interoffice_Trunk* or *PRI_B-Channel* trigger

15 Authorization code subscription, enabling the *Shared_Interoffice_Trunk* or *PRI_B-Channel* trigger

16 ANI subscription, enabling the *Shared_Interoffice_Trunk* or *PRI_B-Channel* trigger

17 Agent subscription, enabling the *Shared_Interoffice_Trunk* or *PRI_B-Channel* trigger

18 Office subscription, enabling the *Shared_Interoffice_Trunk* or *PRI_B-Channel* trigger

Once a subscription method is identified and a *Offhook_Delay*, *Shared_Interoffice_Trunk*, or *PRI_B-Channel* trigger is enabled, NetworkBuilder performs the following steps:

1 Evaluates trigger criteria (datafilled in the trigger table) against gathered data.

2 Performs the datafilled trigger action.

3 If the trigger action is QUERY, an **`Info_Collected`** query message is sent to the SCP.

4 If the trigger action is QUERYSCU, in-switch call processing is temporarily suspended. The call enters server mode and becomes a Programmable Service Node (PSN) call and a **`New_Call`** message is sent to the PSN SCU. Refer to *UCS DMS-250 Programmable Service Node (PSN) Application Guide* for more PSN information.

Note: If an in-switch error occurs before all digits are present the call goes to treatment and does not become a PSN call as it would have under the existing NetworkBuilder functionality.

5 If the trigger action is IGNORE, subscription method determination continues at the point where the trigger was evaluated.

6 If the trigger action is BLOCK, subscription method determination stops, the call is blocked by call processing and AINF treatment is applied.

7 If the trigger action is LEAVE_TDP, call processing exits the current TDP with no further evaluation.

8 If the trigger action is CONT_NOTRIG, call processing exits the current TDP and prevents any further NetworkBuilder interaction for the current call.

O_Abandon event

ATTENTION

The *O_Abandon* event requires the CAIN0802 SOC option. Refer to Volume 5, Chapter 5, "NetworkBuilder SOC functionality," for more information.

Uses

The switch detects the *O_Abandon* event during the second call segment when the calling party disconnects before the called party answer. When you provision NetworkBuilder services on the switch, the SCP can make major call processing decisions. You define the services offered on the SCP. Take-back and transfer is an example of a service that you can develop on the SCP for the *O_Abandon* event. The take-back and transfer service allows a terminating party to transfer or redirect a calling party to a third party.

Supported originating agencies

The *O_Abandon* event supports the following originating agencies:

- DAL
- FGD
- PRI
- SS7 Inter-IMT
- SS7 Global-IMT
- AXXESS

  Note: Refer to *UCS DMS-250 CAIN/FlexDial Interactions* for more information on support for AXXESS agents.

Event evaluation and actions

When the switch encounters the ***O_Abandon*** EDP and the EDP is active, the switch sends the **`O_Abandon`** EDP-Request message.

O_Abandon EDP-Request

The following table defines the **`O_Abandon`** EDP-Request message's parameters and the parameters' requirements.

Table 4-1
O_Abandon message parameters

Parameter	Usage	Definition
UserID	Required	Contains the network identity of the originating agent
BearerCapability	Required	Contains the bearer capability of the call when the message is built
CcID	Optional	Specifies the current call cofiguration of the call
PointInCall	Optional	Contains the current point in call of the call
ExtensionParameter	Optional	No extension parameters are supported for this message
NotificationIndicator	Optional	Identifies the message type as notification or request
—end—		

SCP response processing

NetworkBuilder supports the following messages in response to the **O_Abandon** EDP-Request message:

- **Acknowledge** message
- **Connect_To_Resource** message
- **Disconnect** message
- **Disconnect_Leg** message
- **Merge_Call** message

NetworkBuilder arms EDPs when the switch receives a **Request_Report_BCM_Event** component within the **Acknowledge**, **Merge_Call**, and **Disconnect_Leg** messages.

The switch transitions the call to call configuration 11 (transfer call) when two requirements are met:

- the call is in call configuration 4 or 6
- the switch receives the message in conversation

Refer to Volume 3, Chapter 10, “Incoming CAIN messages,” for specific information regarding response messages; refer to Volume 3, Chapter 12, “Incoming CAIN message parameters,” for detailed descriptions of the

O_Abandon event (end)

response parameters; refer to Volume 3, Chapter 3, “Event processing,” for EDP specific message information.

O_Feature_Requested trigger

ATTENTION

The *O_Feature_Requested* trigger requires the CAIN0508 SOC option. CAINPRT digit collection associated with the *O_Feature_Requested* trigger requires the CAIN0610 SOC option. Refer to Volume 5, Chapter 5 "NetworkBuilder SOC functionality," for more information.

Uses

When you provision NetworkBuilder services on the switch, the SCP can make major call processing decisions. You define the services offered on the SCP. Some examples of services that can be developed on the SCP for the *O_Feature_Requested* trigger are:

- Dial 1+ Services
  - — Speed Dial
  - — Hotline
  - — Intra-LATA Presubscription Screening (ANI screening)
  - — Account Code Screening
  - — Prepaid services
  - — CIC Routing and Branding
- Enhanced Travel Card Services
- Universal Access for Authorization Codes

Supported originating agencies

The *O_Feature_Requested* trigger supports the following originating agencies:

- DAL
- FGD
- PRI
- SS7 Inter-IMT
- SS7 Global-IMT
- AXXESS

  Note: Refer to *UCS DMS-250 CAIN/FlexDial Interactions* for more information on support for AXXESS agents.

O_Feature_Requested trigger (continued)

Subscribing to the O_Feature_Requested trigger

Subscription to the *O_Feature_Requested* trigger is available on an

- address basis (table STDPRTCT, subtable STDPRT)
- authorization code basis (tables AUTHCODU, AUTHCDU2, AUTHCDU3, AUTHCDU4, or AUTHCDU5)
- ANI basis (table ANISCUSP, or tables ANIVAL and UNIPROF)
- agent basis (table TRKGRP or CALLATTR)
- office basis (table CAINPARM)

Note 1: An SCP-returned CAIN subscription group is not available for *O_Feature_Requested* on reoriginated calls.

Note 2: CAIN group subscription for AXXESS agents is handled differently. Refer to *UCS DMS-250 CAIN/FlexDial Interactions*.

Trigger evaluation

NetworkBuilder checks the *O_Feature_Requested* trigger table (OFTRREQ) and evaluates the identified digits against the datafilled range associated with the appropriate CAIN group.

Note: The digit type (information, ANI, address, or CIC) is identified through datafill within the OFTRREQ table.

Trigger actions

NetworkBuilder call processing supports the following actions for the *O_Feature_Requested* trigger:

- BLOCK – AINF treatment is applied.
- IGNORE – NetworkBuilder continues checking the remaining subscription methods for *O_Feature_Requested*. If datafill does not enable the trigger, call processing continues through the call model.
- FEAT – Invokes a feature processor, allowing data validation services, and queries the SCP (through an **`O_Feature_Requested`** query). The SCP analyzes the call and assists the switch with call processing.
- LEAVE_TDP – NetworkBuilder call processing exits the ***O_Feature_Requested*** TDP with no further evaluation.
- CONT_NOTRIG – NetworkBuilder call processing exits the ***O_Feature_Requested*** TDP and prevents any further NetworkBuilder interaction for the call. This prevention is reset upon reorigination.

O_Feature_Requested trigger (continued)

Feature processors

Datafilling a FEAT trigger action allows the user to define the digit collections necessary to complete a call. Datafill order indicates the order of digit collection.

NetworkBuilder software supports three feature processors: CARD, ADDR, and AUTH. The ADDR feature processor immediately instructs the switch to play any resources identified by datafill and to query the SCP. The CARD and AUTH feature processors are used to define digit collection necessary before querying the SCP. Interruptible and uninterruptible resources can be datafilled for all feature processors.

Datafill for the CARD and AUTH features identifies the following:

- collectible digit type (address, authorization code, card, account, or PIN digits). Datafill up to five collectibles.
- resource played for collectible (a resource is identified for each collectible)
- interrupt status of collectible resource (interruptible or not)
- minimum and maximum number of digits the switch needs to collect (The first digit can be * or #.)
- digit collection timer
- extra resources using the ANNC digit type

Note 1: When CARD is identified as the feature processor, a CARD collectible must be identified within the tuple.

Note 2: When AUTH is identified as the feature processor, an AUTH collectible must be identified within the tuple.

O_Feature_Requested trigger (continued)

The following figure shows how a call progresses through the call model, encounters ***O_Feature_Requested***, collects digits (as required by the CARD processor in datafill), and queries the SCP.

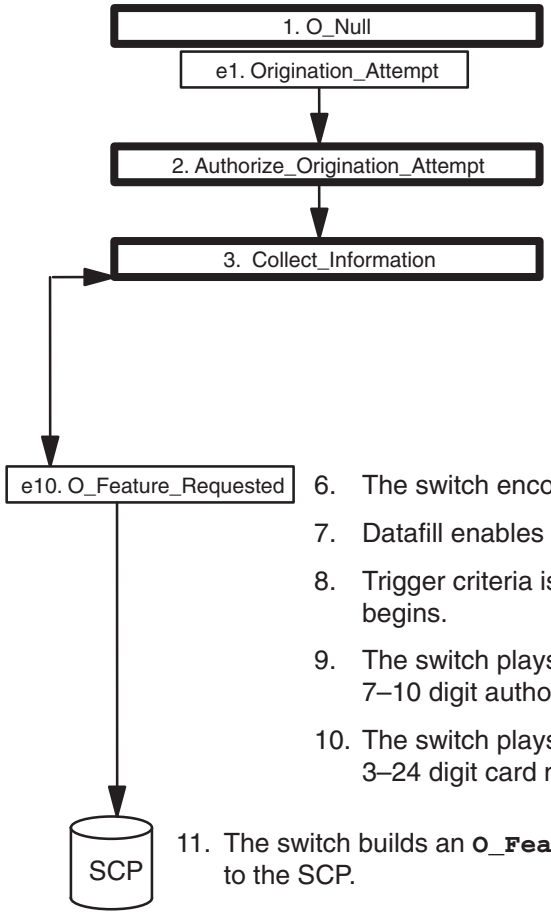
0+ MCCS call

DAL origination
Datafill:

Signaling: DTMF
Dialing plan: 0 + ADDRESS
Address dialed: 0 + (214) 555-1111

Table OFTRREQ

OFTRREQKY	TODIGS	ACTION	OPTION
CAINGRP1 ADDR 0	0	FEAT CARD (ADDR 23 N 7 10 12) (CARD 16 Y 13 24 15) $	$



1. The switch detects off-hook and enters **O_Null**.
2. Call processing enters **Authorize_Origination_Attempt**.
3. Call processing enters **Collect_Information**.
4. *O_Feature_Requested* supports 0+ dialing without authcode. (This functionality is similar to MCCS.)
5. The switch collects the address.
6. The switch encounters ***O_Feature_Requested***.
7. Datafill enables *O_Feature_Requested* trigger.
8. Trigger criteria is met for address and CARD feature processing begins.
9. The switch plays an uninterruptible resource 23 and collects a 7–10 digit authorization code.
10. The switch plays an interruptible resource 16 and collects a 3–24 digit card number.
11. The switch builds an **`O_Feature_Requested`** query and sends it to the SCP.

O_Feature_Requested trigger (continued)

The following figure shows how a call progresses through the call model, encounters ***O_Feature_Requested***, plays an announcement (as required by the ADDR processor in datafill), and queries the SCP.

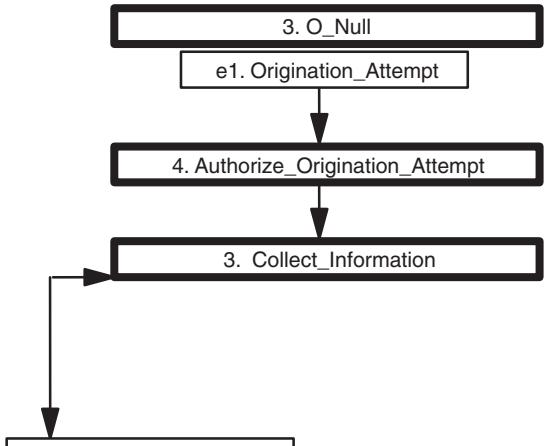
DAL originations

Datafill:

Signaling: DTMF
Dialing plan: (AUTHCODE) + ADDRESS
SNPA: 214
Authcode dialed: 555-1111
Address dialed: (700) 555–2222

Table OFTRREQ

OFTRREQKY	TODIGS	ACTION	OPTION
CAINGRP1 ANI 214	214	FEAT ADDR (ANNC 215 N)	$



1. The switch detects off-hook and enters **O_Null**.
2. Call processing enters **Authorize_Origination_Attempt**.
3. Call processing enters **Collect_Information**.
4. The switch obtains the authorization code (either filed or collected).
5. The switch collects the address.
6. The switch encounters ***O_Feature_Requested***.
7. Datafill enables *O_Feature_Requested* trigger.
8. Trigger criteria is met for ANI (matching on the PANI) and ADDR feature processing begins. (The PANI is the SNPA+Authcode.)
9. The switch plays an uninterruptible resource 215.
10. The switch builds an **`O_Feature_Requested`** query and sends it to the SCP.

O_Feature_Requested trigger (continued)

The following figure shows how a call progresses through the call model, encounters ***O_Feature_Requested***, collects digits (as required by the CARD feature processor in datafill), and queries the SCP.

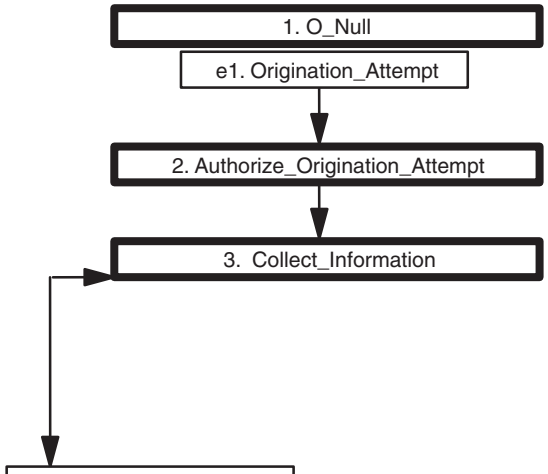
UA MCCS call

SS7 FGD origination
Datafill:

Signaling: SS7
Dialing plan: IAM received, contains CIP
CIC: 333

Table OFTRREQ

OFTRREQKY	TODIGS	ACTION	OPTION
CICGRP1 CIC 333	333	FEAT CARD (ADDR 347 N 7 10 12) (CARD 16 Y 13 24 15) $	$



1. The switch detects seizure and enters **O_Null**.
2. Call processing enters **Authorize_Origination_Attempt**.
3. Call processing enters **Collect_Information**.
4. The switch processes the IAM.
5. The switch encounters ***O_Feature_Requested***.
6. Datafill enables *O_Feature_Requested* trigger.
7. Trigger criteria is met for the CIC and CARD feature processing begins.
8. The switch plays an uninterruptible resource 347 and collects a 7–10 digit address (using DTMF signaling).
9. The switch plays an interruptible resource 16 and collects a 13–24 digit card number (using DTMF signaling).
10. The switch builds an **`O_Feature_Requested`** query and sends it to the SCP.

O_Feature_Requested trigger (continued)

The following figure shows how a call progresses through the call model, encounters ***O_Feature_Requested***, collects digits (as required by the AUTH feature processor in datafill), and queries the SCP.

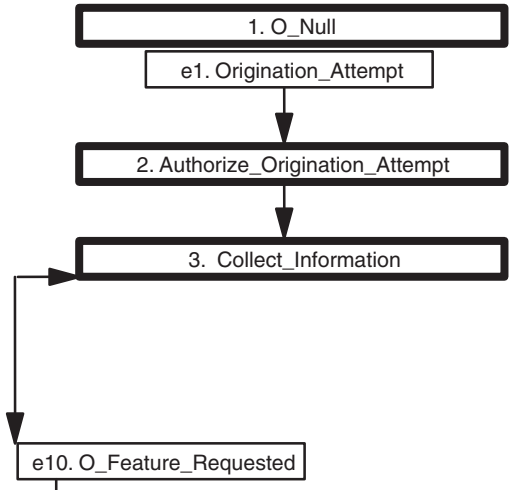
UA Auth call

FGD origination
Datafill:

Signaling: MF at origination
DTMF for digit collection
Dialing plan: KP + II + ANI + ST
KP + ADDR + ST
INFO digits: 12

Table OFTRREQ

OFTRREQKY	TODIGS	ACTION	OPTION
CAINGRP1 INFO 12	12	FEAT AUTH (AUTH 145 Y 7 7 9) (PIN 576 4 4 8) (ADDR 23 Y 7 10 15) (ACCT 15 Y 14 20 15) $	$



1. The switch detects seizure and enters **O_Null**.
2. Call processing enters **Authorize_Origination_Attempt**.
3. Call processing enters **Collect_Information**.
4. The switch collects KP + II + ANI + ST.
5. The switch screens II and ANI digits. ANI screening checks for a CAIN group. The switch does not act upon any screening failures until **Analyze_Infomation**.
6. The switch collects KP + ADDR + ST (ADDR is a UA number).
7. The switch encounters ***O_Feature_Requested***.
8. Datafill enables *O_Feature_Requested* trigger.
9. Trigger criteria is met for the INFO and AUTH feature processing begins.
10. The switch plays an interruptible resource 145 and collects a 7 digit authorization code (using DTMF signaling).
11. The switch plays an interruptible resource 576 and collects a 4 digit PIN number (using DTMF signaling).
12. The switch plays an interruptible resource 23 and collects a 7–10 digits address (using DTMF signaling).
13. The switch plays an interruptible resource 25 and collects a 14–20 digit account code (using DTMF signaling).
14. The switch builds an **`O_Feature_Requested`** query and sends it to the SCP.

O_Feature_Requested trigger (continued)

The following figure shows how a call progresses through the call model, encounters ***O_Feature_Requested***, collects digits (as required by the CARD feature processor in datafill), and queries the SCP.

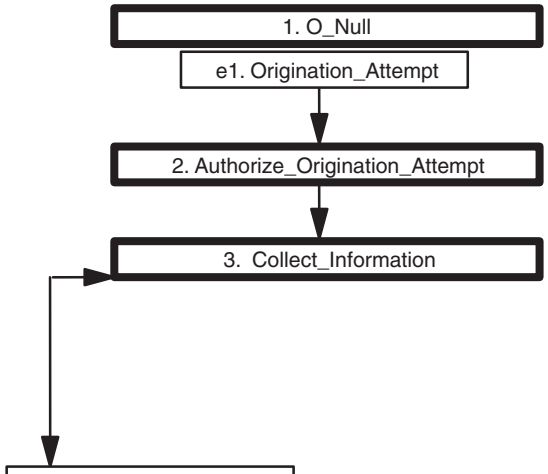
UA MCCS call

PRI origination
Datafill:

Signaling: DTMF
Dialing plan: AUTH + ADDR
Address dialed: (700) 555-2222

Table OFTRREQ

OFTRREQKY	TODIGS	ACTION	OPTION
CAINGRP1 ADDR 700	700	FEAT CARD (CARD 16 Y 13 24 15) (ADDR 347 N 7 10 12) $	$



1. The switch detects seizure and enters **O_Null**.
2. Call processing enters **Authorize_Origination_Attempt**.
3. Call processing enters **Collect_Information**.
4. The switch collects an authorization code.
5. The switch screens the authcode.
6. The switch collects an address.
7. The switch encounters ***O_Feature_Requested***.
8. Datafill enables *O_Feature_Requested* trigger.
9. Trigger criteria is met for the address and CARD feature processing begins.
10. The switch plays an interruptible resource 16 and collects a 13–24 digit card number.
11. The switch plays an uninterruptible resource 347 and collects a 7–10 digit address.
12. The switch builds an **`O_Feature_Requested`** query and sends it to the SCP.

O_Feature_Requested trigger (continued)

The following figure shows how a call progresses through the call model, encounters ***O_Feature_Requested***, collects digits (as required by the CARD feature processor in datafill), and queries the SCP.

UA CAINPRT call

FGD origination
Datafill:

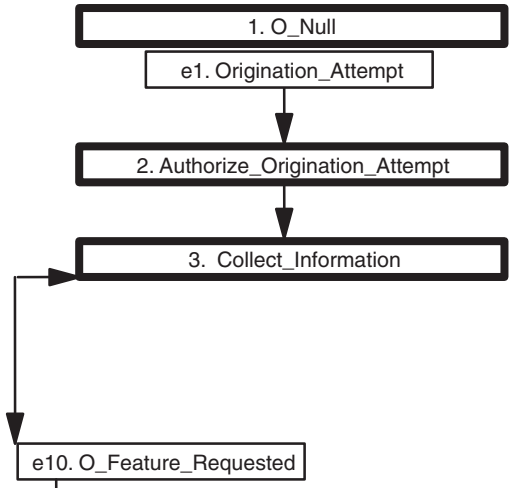
Signaling: MF at origination
DTMF for digit collection
Dialing plan: KP + II + ANI + ST
KP + ADDR + ST
Address dialed: (214)666–1111

Table OFTRREQ

OFTRREQKY	TODIGS	ACTION	OPTION
UA_OFTR_GRP ADDR 214	214	FEAT CARD (CAINPRT UA_PRT 15 10) $	REORIG $

Table CAINPRT

CAINPRTKY	FROMDIGS	TODIGS	COLLTYPE	MIN	MAX	COLLLIST	NEXTCOLL
UA_PRT	0	1	ADDR	11	11	CARD 4 N 16 16 15 $	ACCT_OPT_PIN 27 N 12 $
ACCT_OPT_PIN	445	448	ACCT	12	12	PIN 6 N 4 4 15 $	$



1. The switch detects seizure and enters **O_Null**.
2. Call processing enters **Authorize_Origination_Attempt**.
3. Call processing enters **Collect_Information**.
4. The switch collects KP + II + ANI + ST.
5. The switch screens II and ANI digits. ANI screening checks for a CAIN group. The switch does not act upon any screening failures until **Analyze_Infomation**.
6. The switch collects KP + ADDR + ST (ADDR is a UA number).
7. The switch encounters ***O_Feature_Requested***.
8. Datafill enables *O_Feature_Requested* trigger.
9. Trigger criteria is met for the address and CARD feature processing begins.
10. The switch processes the UA_PRT CAINPRT collectible.

O_Feature_Requested trigger (continued)

Error actions

Error actions are not identified for the *O_Feature_Requested* trigger. The switch applies AINF treatment for fatal application errors.

Options

The *O_Feature_Requested* trigger supports the following options:

- BUFFER – to activate digit buffering while the SCP is queried
- GT – to identify the global title used to identify the SCP handling the query
- PRTNM – to identify the pretranslator call processing should use
- REORIG – to allow reoriginated calls to query
- T1OVFLGT – to identify the specific SCP to query on T1 overflow
- ACGOVFLGT – to identify the global title to use for requerying when a query is blocked by an ACG control
- VERSION – controls the CAIN protocol version for outgoing messages

  Note 1: The value provisioned in the VERSION option overrides the version provisioned in the CAIN_PROTOCOL_VERSION parameter of table CAINPARM on a transaction-by-transaction basis. Additionally, the VERSION option applies to both the outgoing and incoming messages within the same transaction. (Subsequent queries will use the setting provisioned in the CAIN_PROTOCOL_VERSION or provisioned setting of the VERSION option in the trigger table performing the subsequent query.)
- STREAM – to control protocol stream on a per-trigger tuple basis. The value of the STREAM option controls the set of parameters that are sent in NetworkBuilder messages.

Pretranslator selection

Use the pretranslator option (PRTNM) to help reduce the impact of ambiguous dialplans by identifying a specific pretranslator for use with address collection through *O_Feature_Requested*. Datafilling the pretranslator option identifies a specific pretranslator and indexes the pretranslator in table STDPRTCT.

Consider the following: a subscriber auto-dials a 10-digit national address followed by a 14 digit card number. Without the use of the pretranslator option, you would have to datafill a minimum of 3 digits (for feature codes) or a maximum of 18 digits (for international numbers) for the ADDR collectible in table OFTRREQ. The SCP would be left with the responsibility of determining which digits represented the address, since the

first 18 digits dialed (10-digit auto-dial and 8 digits of the card number) would be collected for ADDR.

There are three solutions to this problem:

1 As part of the service, you can specify that the subscriber must dial an octothorpe (#) to indicate end of dialing for the address.

2 As part of the service, you can require the subscriber to wait for digit prompting.

3 You can datafill the pretranslator option in table OFTRREQ to identify a pretranslator to determine the address length.

The pretranslator option is available as a means of determining the address length for the ADDR collectible.

Restrictions

The following restrictions apply to the pretranslator option:

- Only the CT, IP, and IN pretranslator selectors are supported.
- NetworkBuilder software does not support resolution of address feature functionality, such as speed-dial numbers.
- The datafilled pretranslator is used for address collection on all reoriginations.

Reorigination identifier

The reorigination option (REORIG) is only available at *O_Feature_Requested* and forces a reoriginated call to automatically query with no additional trigger criteria evaluations. Reoriginated calls trigger once a new address is collected. The following reorigination behavior and *O_Feature_Requested* interaction apply to the use of the reorigination option:

- Once the address is collected on a reoriginated call, the switch queries the SCP. No other digit collection is performed by the switch. SCP digit requirements are fulfilled through conversational digit collection directed by the SCP.

  Note: Datafill is not consulted again. A query is sent based on the original call. The address sent in the query is the number associated with the reoriginated call.

O_Feature_Requested trigger (continued)

- Upon reorigination, call processing resets any account code or PIN information collected through regular call processing. However, a reoriginated call's query will contain an account code or a PIN collected previously through ***O_Feature_Requested*** processing.

Additional information

The following apply to reoriginated calls at *O_Feature_Requested*:

- When the ADDR collectible is executed on the original call, it is executed on the reoriginated call.
- When the ADDR collectible is not executed on the original call, normal (non-NetworkBuilder) reorigination occurs and the caller is prompted for an address.
- If the pretranslator option is datafilled, the specified pretranslator is used on all reoriginated calls (NetworkBuilder or non-NetworkBuilder).
- When the original call triggers at *O_Feature_Requested* and collects an account code, upon reorigination the account code is no longer available. The ACCTCD field in the CDR (for each reoriginated call) is not populated.
- Upon reorigination, the PINDIGS field (of the CDR) will contain the same value as the original call.
- If a **`Collect_Information`** message is received in response to an **`O_Mid_Call`** TDP-Request message, then ***O_Feature_Requested*** processing is invoked, an account code is collected and sent to the SCP.

O_Feature_Requested TDP-Request

An **`O_Feature_Requested`** TDP-Request (query) is sent to the SCP in a query package, with a component type of Invoke_Last. The following table defines the parameters and usage requirements for the parameters the **`O_Feature_Requested`** TDP-Request may contain. Refer to Volume 3, Chapter 8, "Outgoing CAIN message parameters," for detailed descriptions of the request parameters.

Note: Parameters and extension parameters for LNP queries and AXXESS agents are handled differently. Refer to *UCS DMS-250 Local Number Portability Application Guide* and *UCS DMS-250 CAIN/FlexDial Interactions* for more information.

O_Feature_Requested trigger (continued)

O_Feature_Requested TDP-Request message parameters

Parameter	Usage	Definition
UserID	Required	Contains the network identity of the originating agent.
BearerCapability	Required	Contains the bearer capability of the call when the message is built.
ChargeNumber (Note 6)	Optional	Contains the number that would be used to populate the CDR at this point in call processing. NetworkBuilder call processing populates this field with the data collected from the CARD or AUTH collectibles.
Lata (Note 7)	Optional	Contains the call's Local Access and Transport Area (LATA)
Carrier	Optional	Contains the dialed CIC or the CIC value (with an indication of: Selected CIC presubscribed and not input by calling party) from table TRKGRP.
TriggerCriteriaType	Optional	Contains FeatureActivator.
PointInCall	Required	Contains Collect_Information.

Note 1: The `universalAccess` extension parameter is included in the query (a flat 10 digits) when parameter CAIN_PROTOCOL_VERSION is V0. When CAIN_PROTOCOL_VERSION is set to V1 or higher, the parameter contains up to 24 digits including a nature of address and numbering plan similar to the ***CalledPartyID***.
Note 2: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS07 or higher.
Note 3: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS08 or higher.
Note 4: This parameter is not populated when the CAIN_PROTOCOL_VERSION is set to V3 or higher, and is therefore not sent.
Note 5: This parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS09 or higher.
Note 6: When the CAIN_PROTOCOL_VERSION is set to V4 or higher non-standard charge numbers (CARD, AUTH, ACCT, PIN, and N00) no longer populate the ***ChargeNumber*** parameter and all standard NOAs are supported.
Note 7: The parameter is supported only on DAL, FGD, and AXXESS agencies and the parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS11 or higher.
Note 8: Refer to Volume 3, Chapter 8, "Outgoing CAIN message parameters" for information on the population of this parameter when the CAIN_PROTOCOL_VERSION is set to V5 or higher.

—continued—

O_Feature_Requested trigger (continued)

O_Feature_Requested TDP-Request message parameters (continued)

Parameter	Usage	Definition
FeatureActivatorID	Optional	Identifies the feature activated in table OFTRREQ. 1 = CARD processor 2 = AUTH processor 3 = ADDR processor
CallingPartyID (Note 8)	Optional	Contains one of the following (listed in order of precedence): For SS7 FGD calls: Calling_Party_Address from ISUP message, when available For FGD and AXXESS calls: valid ANI (information digits are not passed in this parameter) ***Note:*** Refer to *UCS DMS-250 CAIN/FlexDial Interactions* for more information on AXXESS agents. For PRI calls: CLID when available Valid SNPA value from table TRKGRP Default SNPA value from table CAINPARM
ChargePartyStationType (Note 2)	Optional	Contains the information digits for the call

Note 1: The `universalAccess` extension parameter is included in the query (a flat 10 digits) when parameter CAIN_PROTOCOL_VERSION is V0. When CAIN_PROTOCOL_VERSION is set to V1 or higher, the parameter contains up to 24 digits including a nature of address and numbering plan similar to the ***CalledPartyID***.
Note 2: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS07 or higher.
Note 3: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS08 or higher.
Note 4: This parameter is not populated when the CAIN_PROTOCOL_VERSION is set to V3 or higher, and is therefore not sent.
Note 5: This parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS09 or higher.
Note 6: When the CAIN_PROTOCOL_VERSION is set to V4 or higher non-standard charge numbers (CARD, AUTH, ACCT, PIN, and N00) no longer populate the ***ChargeNumber*** parameter and all standard NOAs are supported.
Note 7: The parameter is supported only on DAL, FGD, and AXXESS agencies and the parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS11 or higher.
Note 8: Refer to Volume 3, Chapter 8, “Outgoing CAIN message parameters” for information on the population of this parameter when the CAIN_PROTOCOL_VERSION is set to V5 or higher.

—continued—

O_Feature_Requested trigger (continued)

O_Feature_Requested TDP-Request message parameters (continued)

Parameter	Usage	Definition
ACGEncountered	Optional	Contains the ACG control encountered when the ACG control expires and a query is allowed to be sent
AccessCode (Note 4)	Optional	Contains the account code, when available. NetworkBuilder call processing populates this field with the data collected from ACCT collectible.
CollectedAddressInfo	Optional	Contains the address collected from the incoming agent (from the IAM or SETUP messages, subscriber dialing, *O_Feature_Requested* ADDR collectible, or datafilled hotline digits).
CollectedDigits (Note 4)	Optional	Contains the collected PIN digits, when available; when additional billing numbers are required, they are sent in this parameter. The digits will be an undifferentiated stream of digits; the SCP is responsible for separating the digits into different numbers. NetworkBuilder call processing populates this field with the data collected from the PIN collectible.
VerticalServiceCode	Optional	Contains feature codes dialed by the subscriber
ExtensionParameter	Optional	Extension parameters require the CAIN0200 SOC option.

Note 1: The `universalAccess` extension parameter is included in the query (a flat 10 digits) when parameter CAIN_PROTOCOL_VERSION is V0. When CAIN_PROTOCOL_VERSION is set to V1 or higher, the parameter contains up to 24 digits including a nature of address and numbering plan similar to the ***CalledPartyID***.
Note 2: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS07 or higher.
Note 3: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS08 or higher.
Note 4: This parameter is not populated when the CAIN_PROTOCOL_VERSION is set to V3 or higher, and is therefore not sent.
Note 5: This parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS09 or higher.
Note 6: When the CAIN_PROTOCOL_VERSION is set to V4 or higher non-standard charge numbers (CARD, AUTH, ACCT, PIN, and N00) no longer populate the ***ChargeNumber*** parameter and all standard NOAs are supported.
Note 7: The parameter is supported only on DAL, FGD, and AXXESS agencies and the parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS11 or higher.
Note 8: Refer to Volume 3, Chapter 8, “Outgoing CAIN message parameters” for information on the population of this parameter when the CAIN_PROTOCOL_VERSION is set to V5 or higher.

—continued—

O_Feature_Requested trigger (continued)

O_Feature_Requested TDP-Request message parameters (continued)

Parameter	Usage	Definition
`universalAccess` (Note 1)	Optional	Contains the original address when the *O_Feature_Requested* ADDR collectible is executed or the universal access number dialed by the caller to obtain switch dial tone. Since this number is not the actual address for the call, it is not transported in ***CollectedAddressInfo*** or ***CalledPartyID***.
`cainGroup`	Optional	Contains the group number (field GRPNUM, table CAINGRP) for the CAIN group associated with the trigger when field EXTPARM in table CAINGRP contains CAINGRP.
`adin`	Optional	Contains the authorization code database index (field ADIN in table TRKGRP for DAL and FGD agencies, field ADIN in table CALLATTR for PRI agents) associated with the originating agency when field EXTPARM in table CAINGRP contains ADIN ***Note:*** This does not apply to the authcode collected at *O_Feature_Requested*. ***Note:*** The `adin` extension parameter is not supported for AXXESS agents.

Note 1: The `universalAccess` extension parameter is included in the query (a flat 10 digits) when parameter CAIN_PROTOCOL_VERSION is V0. When CAIN_PROTOCOL_VERSION is set to V1 or higher, the parameter contains up to 24 digits including a nature of address and numbering plan similar to the ***CalledPartyID***.
Note 2: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS07 or higher.
Note 3: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS08 or higher.
Note 4: This parameter is not populated when the CAIN_PROTOCOL_VERSION is set to V3 or higher, and is therefore not sent.
Note 5: This parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS09 or higher.
Note 6: When the CAIN_PROTOCOL_VERSION is set to V4 or higher non-standard charge numbers (CARD, AUTH, ACCT, PIN, and N00) no longer populate the ***ChargeNumber*** parameter and all standard NOAs are supported.
Note 7: The parameter is supported only on DAL, FGD, and AXXESS agencies and the parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS11 or higher.
Note 8: Refer to Volume 3, Chapter 8, “Outgoing CAIN message parameters” for information on the population of this parameter when the CAIN_PROTOCOL_VERSION is set to V5 or higher.

—continued—

O_Feature_Requested trigger (continued)

O_Feature_Requested TDP-Request message parameters (continued)

Parameter	Usage	Definition
origTrunkInfo	Optional	Contains the originating trunk group number, trunk type, and trunk member number when field EXTPARM in table CAINGRP contains ORGTINFO.
treatment	Optional	Contains the treatment if set by regular call processing before the query was sent
reorigCall	Optional	Presence of this parameter indicates the call in progress is a result of reorigination
univIdx	Optional	Used by SS7 Global-IMT agents only. Contains the universal translations scheme to be used for the call
netinfo (Note 2)	Optional	Contains external network ID, network customer group ID, and network class of service
t1Overflow	Optional	Indicates to the SCP that an overflow occurred on the query to the initial SCP
lnpReceived (Note 2)	Optional	Presence of this parameter indicates LNP information was received from a previous switch
subscriptionInfo (Note 3)	Optional	Contains the digit type the switch triggered on and which subscription method was in use when the query occurred
cainPRT (Note 3)	Optional	Contains unused digits collected when the CAINPRT is not matched

Note 1: The universalAccess extension parameter is included in the query (a flat 10 digits) when parameter CAIN_PROTOCOL_VERSION is V0. When CAIN_PROTOCOL_VERSION is set to V1 or higher, the parameter contains up to 24 digits including a nature of address and numbering plan similar to the ***CalledPartyID***.
Note 2: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS07 or higher.
Note 3: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS08 or higher.
Note 4: This parameter is not populated when the CAIN_PROTOCOL_VERSION is set to V3 or higher, and is therefore not sent.
Note 5: This parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS09 or higher.
Note 6: When the CAIN_PROTOCOL_VERSION is set to V4 or higher non-standard charge numbers (CARD, AUTH, ACCT, PIN, and N00) no longer populate the ***ChargeNumber*** parameter and all standard NOAs are supported.
Note 7: The parameter is supported only on DAL, FGD, and AXXESS agencies and the parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS11 or higher.
Note 8: Refer to Volume 3, Chapter 8, “Outgoing CAIN message parameters” for information on the population of this parameter when the CAIN_PROTOCOL_VERSION is set to V5 or higher.

—continued—

O_Feature_Requested trigger (continued)

O_Feature_Requested TDP-Request message parameters (continued)

Parameter	Usage	Definition
`jurisdictionInfo` (Note 3)	Optional	Contains the originating switch's LRN
`switchID` (Note 5)	Optional	Contains the switch ID
`accountCode` (Note 5)	Optional	Contains the account code when available
`pinDigits` (Note 5)	Optional	Contains the collected pin digits when available
`billingNumber` (Note 5)	Optional	Contains the non-standard charge number
`acgRequery`	Optional	When present, indicates to the SCP that the current query is a re-query message resulting from the original query being blocked by an ACG control

Note 1: The `universalAccess` extension parameter is included in the query (a flat 10 digits) when parameter CAIN_PROTOCOL_VERSION is V0. When CAIN_PROTOCOL_VERSION is set to V1 or higher, the parameter contains up to 24 digits including a nature of address and numbering plan similar to the ***`CalledPartyID`***.
Note 2: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS07 or higher.
Note 3: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS08 or higher.
Note 4: This parameter is not populated when the CAIN_PROTOCOL_VERSION is set to V3 or higher, and is therefore not sent.
Note 5: This parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS09 or higher.
Note 6: When the CAIN_PROTOCOL_VERSION is set to V4 or higher non-standard charge numbers (CARD, AUTH, ACCT, PIN, and N00) no longer populate the ***`ChargeNumber`*** parameter and all standard NOAs are supported.
Note 7: The parameter is supported only on DAL, FGD, and AXXESS agencies and the parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS11 or higher.
Note 8: Refer to Volume 3, Chapter 8, "Outgoing CAIN message parameters" for information on the population of this parameter when the CAIN_PROTOCOL_VERSION is set to V5 or higher.

—end—

Once the request message is built, NetworkBuilder formats the message and sends it to the message encoder. Once encoded, the message is sent to the SCP.

CDR population

The following table shows the collectible that populates the associated CDR field.

CDR fields

Collectible	CDR field
CARD	BILLNUM
AUTH	BILLNUM
PIN	PINDIGS
ACCT	ACCTCD
ADDR	DIALEDNO
CAINPRT	(Note 2)

Note 1: CDR field population is handled differently for AXXESS agents. Refer to *UCS DMS-250 CAIN/FlexDial Interactions*.
Note 2: The CDR field populated for the CAINPRT collectible corresponds to the collectible type (COLLTYPE field in table CAINPRT) for the CAINPRT collectible matched.

—end—

SCP response processing

The following response messages are supported:

- **`Analyze_Route`**
- **`Send_To_Resource`**
- **`Disconnect`**

Note 1: EDPs may be armed through the **`Request_Report_BCM_Event`** component with the **`Analyze_Route`** message.

Note 2: A **`Termination_Notification`** may be requested through the **`Send_Notification`** component with the SCP response message.

Note 3: An **`ACG`** message may be sent with the SCP response message.

Refer to Volume 3, Chapter 10, “Incoming CAIN messages,” for specific information regarding response messages; refer to Volume 3, Chapter 12, “Incoming CAIN message parameters,” for detailed descriptions of the response parameters; refer to Volume 3, Chapter 3, “Event processing,” for EDP specific message information.

O_Feature_Requested trigger (continued)

Datafill

The following figure shows how a subscription table interacts with the *O_Feature_Requested* trigger table (OFTRREQ).

Subscription-OFTRREQ table interaction

Table CALLATTR

CATTRIDX	CUSTOMER	PRTNM	COS	ZEROMPOS	CUSTAREA (OPART BILLTYP ADIN VAUTHFLD AUTHDIAL DEFCLID OPTION)
21	UCS	PRI	0	NONE	611 AUTH 1 6112211 0 5026112211 (ANIDELV) (CAIN) (CAINGRP CAINGRP1)

Table CAINGRP

CAINGRP	GRP NUM	MSGSET	PROTO	TRIGRSET (PIC TDP TRIGGER)	OPTIONS	EXT PARMS	TEXT PARMS
CAINGRP1	2	CAIN02	TCAP_SCCP	COLLINFO OFTRREQ OFTRREQ	$	$	$

Called Party Number
2221234

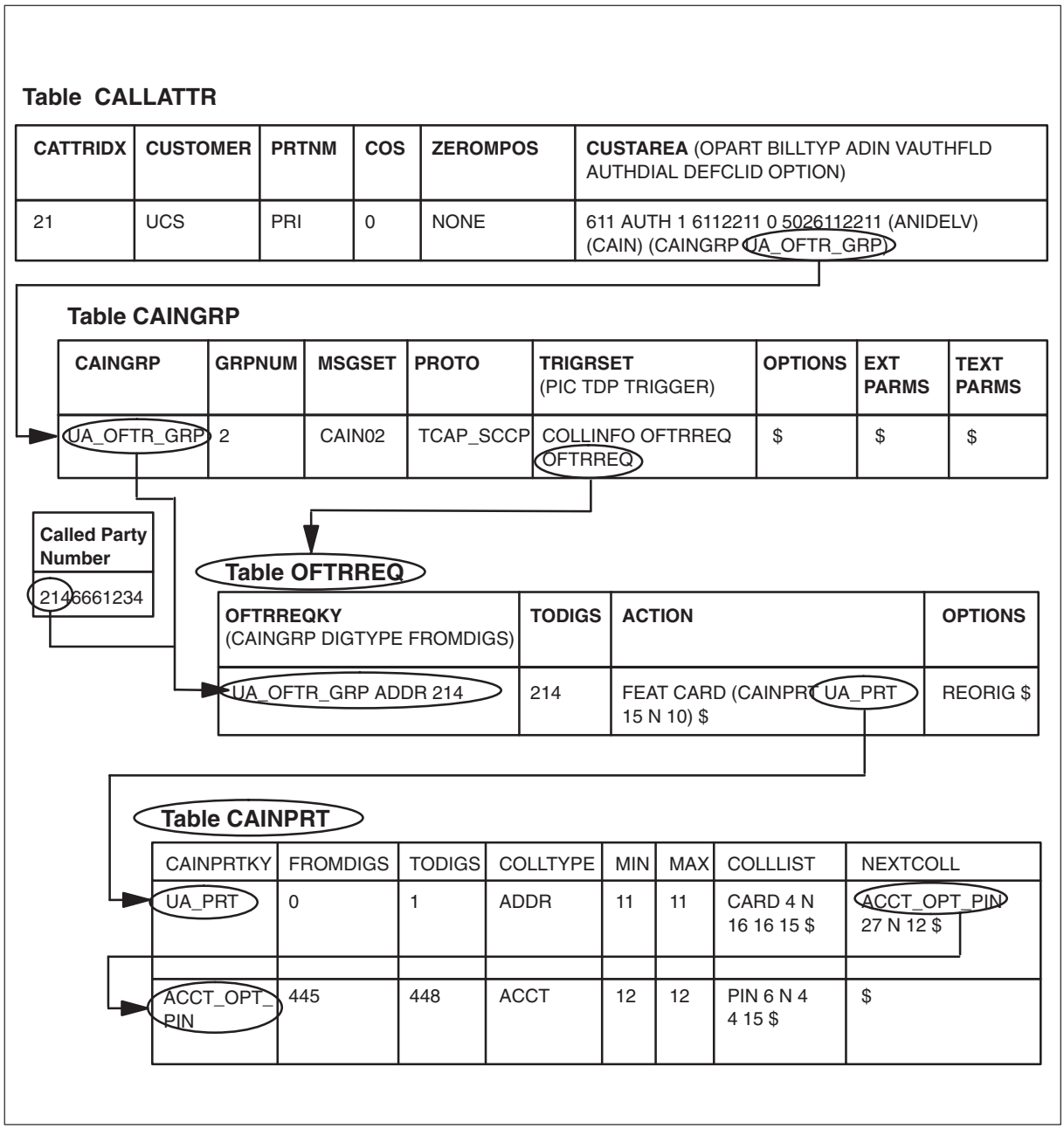
Table OFTRREQ

OFTRREQKY (CAINGRP DIGTYPE FROMDIGS)	TODIGS	ACTION	OPTIONS
CAINGRP1 ADDR 222	9	FEAT CARD (CARD 14 Y 3 6 10) $	$

O_Feature_Requested trigger (continued)

The following figure shows how a subscription table interacts with the *O_Feature_Requested* trigger table (OFTRREQ) and table CAINPRT.

Subscription-OFTRREQ and CAINPRT table interaction



O_Feature_Requested trigger (continued)

Provisioning the O_Feature_Requested trigger

At the CI prompt

1 Provision the originating agent as CAIN-capable

2 Subscribe to a CAIN group (STDPRTCT table, authcode tables, ANI tables, TRKGRP, CALLATTR, or CAINPARM)

Note: CAIN group subscription is handled differently for AXXESS agents. Refer to *UCS DMS-250 CAIN/FlexDial Interactions*.

3 Datafill CAIN group trigger subscription to the (COLLINFO OFTRREQ OFTRREQ) trigger set (table CAINGRP).

4 Enter table OFTRREQ.

5 Define the trigger criteria for a CAIN group by using the following format:

>ADD oftrreqky todigs action options
where

oftrreqky is comprised of three subfields: CAINGRP, DIGTYPE, FROMDIGS, where
- CAINGRP is the CAIN group that enables the trigger (from table CAINGRP).
- DIGTYPE is the digit type being analyzed (INFO, ADDR, ANI, CIC).
- FROMDIGS is the first number used to define the range of digits (0 to 9, *, #).

todigs is the second number used to define the range of digits (0 to 9, *, #).

action is comprised of one subfield: TRIGACT, where
- TRIGACT is the trigger action taken when the required digits are within the FROMDIGS-TODIGS range (BLOCK, IGNORE, FEAT, LEAVE_TDP, CONT_NOTRIG).

If you datafill TRIGACT as:	Go to:
BLOCK	step 10
IGNORE	step 10
FEAT	step 6
LEAVE_TDP	step 10
CONT_NOTRIG	step 10

6 Datafill the FEATURE refinement, where

O_Feature_Requested trigger (continued)

feature is the feature processor required (ADDR, CARD, AUTH)

If you datafill FEATURE as:	Go to:
ADDR	step 7
CARD	step 9
AUTH	step 9

7 Datafill the COLLTYPE, RSRCIND, and INTERRUPT refinements, where

colltype is the collectible type (ANNC, CAINPRT).

Note: The CAINPRT collectible must be datafilled in table CAINPRT.

rsrcind is the index into table CAINRSRC (0 to 4095).

interrupt identifies the resource as interruptible or uninterruptible (Y or N).

8 Go to step 10.

9 Datafill the DPVECTOR refinement, consisting of COLLTYPE, RSRCIND, INTERRUPT, MINDIGS, MAXDIGS, and TIMER refinement, where

Note: Datafill up to five DPVECTORs.

colltype is the collectible type (ACCT, ADDR, ANNC, AUTH, CARD, PIN, or CAINPRT).

Note 1: You must datafill an AUTH collectible for use with the AUTH feature processor.

Note 2: You must datafill a CARD collectible for use with the CARD feature processor.

Note 3: The CAINPRT collectible must be datafilled in table CAINPRT.

rsrcind is the index into table CAINRSRC (0 to 4095).

interrupt identifies the resource as interruptible or uninterruptible (Y or N).

mindigs is the minimum number of digits the switch should collect (0 to 24).

maxdigs is the maximum number of digits the switch should collect (0 to 24).

timer is the number of seconds allowed for the subscriber to dial the first digit (0 to 15).

10 Datafill the OPTIONS field, where

options is only allowed when ACTION is FEAT. Enter up to 8 options: BUFFER, GT, PRTNM, REORIG, T1OVFLGT, ACGOVFLGT, VERSION, or STREAM. If ACTION is not FEAT enter $.

Sample entry: **>ADD caingrp1 214 214 feat annc 23 N BUFFER $**

Sample entry: **>ADD caingrp1 214 214 feat card card 23 N 7 14 5 $ $**

O_Feature_Requested trigger (end)

Sample entry: **>ADD caingrp1 214 214 feat auth auth 23 N 4 18 5 $ $**

Sample entry: **>ADD caingrp1 addr 214 214 ignore $**

Sample entry: **>ADD caingrp1 214 214 feat card card 23 N 4 18 5 prtnm ean reorig**

Sample entry: **>ADD ua_oftr_grp addr 214 214 feat card cainprt ua_prt 15 N 10 $ reorig $**

O_Feature_Requested trigger criteria is defined.

Provisioning table CAINPRT

At the CI prompt

1 Enter table CAINPRT

2 Define the CAINPRT collectible by using the following format:

>ADD cainprtky fromdigs todigs colltype min max colllist nextcoll
where

cainprtky is a valid CAINPRT datafilled in table CNPRTNUM (1 to 16 characters)
fromdigs up to 24 digits (0–9, *, #)
todigs up to 24 digits (0–9, *, #)
colltype is the *O_Feature_Requested* collectible type (ACCT, ADDR, AUTH, CARD, PIN)
min is 0 to 24 digits, valid for all collectible types except ANNC
max is 0 to 24 digits, valid for all collectible types except ANNC
colllist is a vector of up to 4 colltypes
nextcoll is a vector of up to 1 index of another CAINPRT tuple, a resource indicator (0–4095), an interruptible flag and a timer value (0–15)

Sample entry: **>ADD ua_prt 0 1 addr 11 11 card 4 n 16 16 15 $ acct_opt_pin 27 n 12 $**

Sample entry: **>ADD acct_opt_pin 445 448 acct 12 12 pin 6 n 4 4 15 $ $**

Associated OMs

CAINUIF, CAINTRIG, CAINMSGS

Tollfree_Service trigger

ATTENTION

The *Tollfree_Service* trigger requires the CAIN0513 SOC option. Refer to Volume 5, Chapter 5, "NetworkBuilder SOC functionality," for more information.

Uses

When you provision NetworkBuilder services on the switch, the SCP can make major call processing decisions. You define the services offered on the SCP. The *Tollfree_Service* trigger is used to determine the carrier who owns the dialed N00 number.

Supported originating agencies

The *Tollfree_Service* trigger supports the following originating agencies:

- DAL
- FGD
- PRI
- SS7 Inter-IMT
- SS7 Global-IMT
- AXXESS

  Note: Refer to *UCS DMS-250 CAIN/FlexDial Interactions* for more information on support for AXXESS agents.

Subscribing to the Tollfree_Service trigger

Subscription to the *Tollfree_Service* trigger is available on an

- SCP-returned basis
- address basis (table STDPRTCT, subtable STDPRT)
- authorization code basis (tables AUTHCODU, AUTHCDU2, AUTHCDU3, AUTHCDU4, or AUTHCDU5)
- ANI basis (tables ANISCUSP, or ANIVAL and UNIPROF)
- agent basis (tables TRKGRP or CALLATTR)
- office basis (table CAINPARM)

Note 1: An SCP-returned CAIN subscription group is received in **`Analyze_Route`** messages in response to a CAIN query. When a reorigination occurs after an **`Analyze_Route`** is received, the SCP-returned CAIN group is available for any reoriginated call.

Tollfree_Service trigger (continued)

Note 2: CAIN group subscription for AXXESS agents is handled differently. Refer to *UCS DMS-250 CAIN/FlexDial Interactions*.

Note 3: The type of subscription used may affect the ability to subscribe to other NetworkBuilder services.

Although all types of subscription are available, the most likely types of subscription to be utilized for the *Tollfree_Service* trigger are address or agent.

Trigger evaluation

NetworkBuilder checks the *Tollfree_Service* trigger table (TOLLFREE) and evaluates the address [ADDR] digits against the datafilled range associated with the appropriate CAIN group. There must be at least 10 address digits available, regardless of the datafilled trigger action, or no trigger occurs.

Trigger actions

NetworkBuilder call processing supports the following actions for the *Tollfree_Service* trigger:

- BLOCK – Sends the call to the treatment specified in the TRTMT trigger option. If the TRTMT trigger option is not provisioned, the AINF treatment is applied to the call.
- IGNORE – NetworkBuilder exits the current level of CAIN group subscription for *Tollfree_Service* and call processing continues through the call model.
- QUERY – The switch builds a **`Start`** query and sends it to the SCP. The SCP analyzes the call and assists the switch with call processing.

Error actions

Error actions are not identified for the *Tollfree_Service* trigger. The switch applies AINF treatment for fatal application errors.

Options

The *Tollfree_Service* trigger supports the TRTMT option. The TRTMT option provides the switch with a specific treatment value to use when the call is blocked. This option is only datafillable with the BLOCK action.

For PRI calls that activate the *Tollfree_Service* trigger, the Calling Party Number parameter and the LATA parameter can be modified with optional datafill in table CALLATTR. The Calling Party Number parameter contains datafill from the LANI option DIGITS field in table CALLATTR. The LATA parameter is populated with the LATANUM entry in table LATAID from the LATA option in table CALLATTR.

Tollfree_Service trigger (continued)

The following figure shows how a call progresses through the call model, encounters ***Info_Collected*** and queries the SCP.

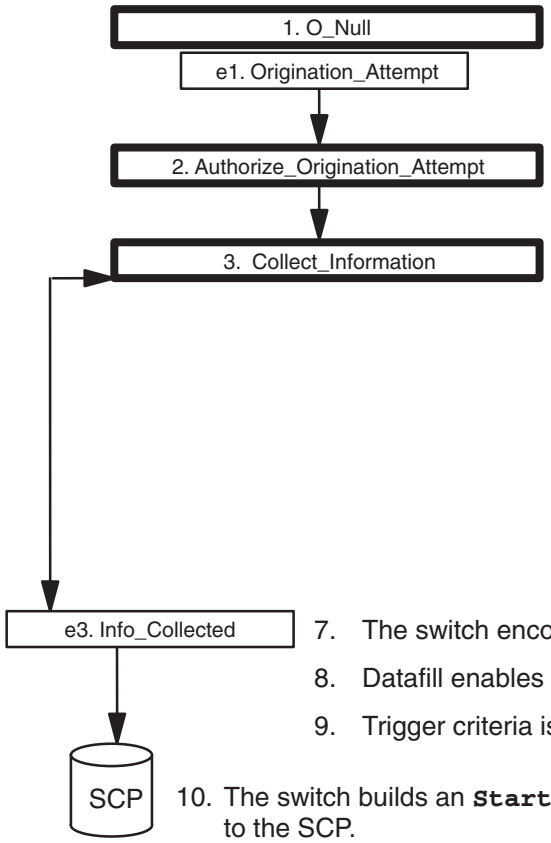
UA 1+ call

FGD origination
Datafill:

Signaling: MF at origination
DTMF for digit collection
Dialing plan: KP + II + ANI + ST
KP + ADDR + ST
ADDR: (888) 611-3311

Table TOLLFREE

TOLLFREEKY	TODIGS	ACTION	OPTION
TFREEGRP1 ADDR 888	888	QUERY	$



1. The switch detects seizure and enters **O_Null**.
2. Call processing enters **Authorize_Origination_Attempt**.
3. Call processing enters **Collect_Information**.
4. The switch collects KP + II + ANI + ST.
5. The switch screens II and ANI digits.
6. The switch collects KP + ADDR + ST.
7. The switch encounters ***Info_Collected***.
8. Datafill enables *Tollfree_Service* trigger.
9. Trigger criteria is met for the ADDR.
10. The switch builds an **`Start`** query and sends it to the SCP.

Tollfree_Service trigger (continued)

Start message

A **Start** message is sent to the SCP with a ***PackageTypeIdentifier*** of Query, with a ***ComponentTypeIdentifier*** of Invoke_Last. The following table defines the parameters and usage requirements for the parameters the **Start** message may contain. Refer to Volume 3, Chapter 8, "Outgoing CAIN message parameters," for detailed descriptions of the request parameters.

Note: Parameters and extension parameters for AXXESS agents are handled differently. Refer to *UCS DMS-250 CAIN/FlexDial Interactions* for more information.

Start message parameters

Parameter	Usage	Definition
ServiceKey	Required	Contains the called party number for the call
Digits	Required	Two required instances of this parameter are sent. The first instance contains the type of digits indicator set `CallingPartyNumber` (ANI), the nature of address, numbering plan, encoding scheme, number of included ANI digits, and the 3, 6, or 10-digit ANI associated with the call. The second instance contains the type of digits indicator set to `LATA`, the nature of address, numbering plan, encoding scheme, number of included LATA digits, and the LATA digits associated with the call.
OriginatingStationType	Required	Contains the II information digits associated with the ANI for the call, or any incoming Originating Line Information (OLI) digits

—end—

Once the request message is built, NetworkBuilder formats the message and sends it to the message encoder. Once encoded, the message is sent to the SCP.

SCP response processing

The following response messages are supported:

- **Play_Announcement**

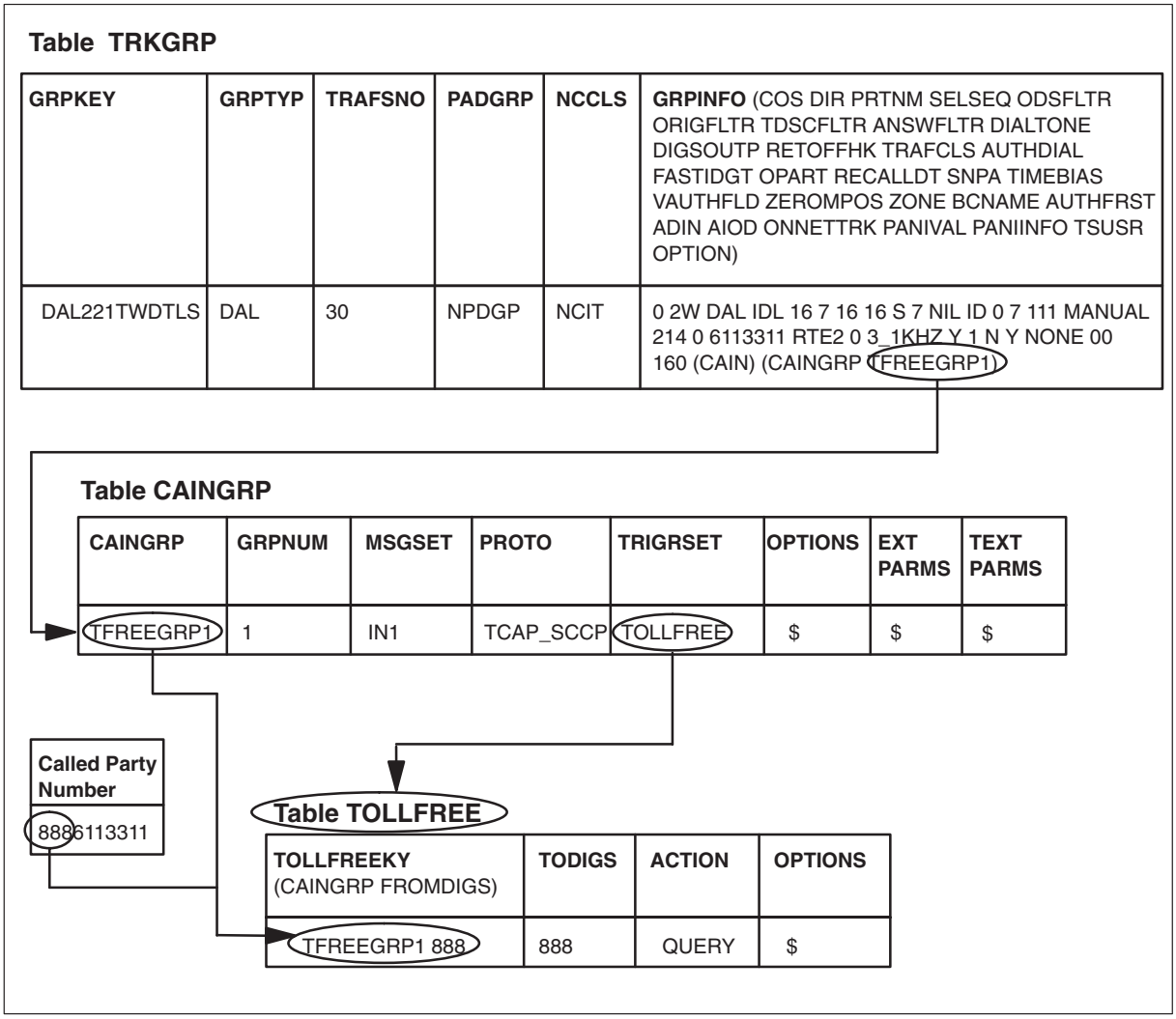
- **`Connect`**

Note: An **`ACG`** or **`Termination`** component may be sent with the **`Connect`** or **`Play_Announcement`** SCP response messages.

Datafill

The following figure shows how a subscription table interacts with the *Tollfree_Service* trigger table (TOLLFREE).

Subscription-TOLLFREE table interaction



Tollfree_Service trigger (end)

Provisioning the Tollfree_Service trigger

At the CI prompt

1 Provision the originating agent as CAIN-capable

2 Subscribe to a CAIN group (table STDPRTCT, authcode tables, ANI tables, TRKGRP, or CAINPARM)

Note: CAIN group subscription is handled differently for AXXESS agents. Refer to *UCS DMS-250 CAIN/FlexDial Interactions*.

3 Datafill CAIN group trigger subscription to the (TOLLFREE) trigger set (table CAINGRP).

4 Enter table TOLLFREE.

5 Define the trigger criteria for a CAIN group by using the following format:

>ADD tollfreeky todigs action options
where

tollfreeky is comprised of three subfields: CAINGRP and FROMDIGS, where
- CAINGRP is the CAIN group that enables the trigger (from table CAINGRP).
- FROMDIGS is the first number used to define the range of digits (0 to 9, *, #).

todigs is the second number used to define the range of digits (0 to 9, *, #).

action is the trigger action taken when the specified digits are within the FROMDIGS-TODIGS range (BLOCK, IGNORE, QUERY).

6 Datafill the OPTIONS, where

options is only allowed when ACTION is BLOCK. Enter TRMT, then enter a treatment code. When ACTION is not BLOCK, enter $.

Sample entry: **>ADD tfreegrp1 888 888 block trtmt ainf $**

Tollfree_Service trigger criteria is defined.

Associated OMs

CAINMSGS, TFREE533, VAMPACG

Offhook_Delay trigger

ATTENTION

The *Offhook_Delay* trigger requires the CAIN0512 SOC option. Refer to Volume 5, Chapter 5, "NetworkBuilder SOC functionality," for more information.

Uses

When you provision NetworkBuilder services on the switch, the SCP can make major call processing decisions. You define the services offered on the SCP. Some examples of services that can be developed on the SCP for the *Offhook_Delay* trigger are:

- Dial 1+ Services
  - — Speed Dial
  - — Hotline
  - — Account Code Screening
  - — Bill to Office
  - — Prepaid services
  - — CIC Routing and Branding
  - — Chat Lines
- VPN Services
  - — Black Box Screening
- N00 Services
  - — Follow Me, Find Me, Do Not Disturb Me
  - — Call Screening
  - — Customized Call Branding
  - — Universal Access Authorization
- Subscriber Screening
  - — Authorization Code Screening
  - — Account Code Screening

Supported originating agencies

The *Offhook_Delay* trigger supports the following originating agencies:

- DAL
- FGD

Offhook_Delay trigger (continued)

- PRI
- SS7 Inter-IMT
- SS7 Global-IMT
- AXXESS

Note: Refer to *UCS DMS-250 CAIN/FlexDial Interactions* for more information on support for AXXESS agents.

Subscribing to the Offhook_Delay trigger

Subscription to the *Offhook_Delay* trigger is available on a

- SCP-returned basis
- address basis (table STDPRTCT, subtable STDPRT)
- authorization code basis (tables AUTHCODU, AUTHCDU2, AUTHCDU3, AUTHCDU4, or AUTHCDU5)
- ANI basis (tables ANISCUSP, or ANIVAL and UNIPROF)
- agent basis (tables TRKGRP or CALLATTR)
- office basis (table CAINPARM)

Note 1: An SCP-returned CAIN subscription group is received in **`Analyze_Route`** messages. When a reorigination occurs after an **`Analyze_Route`** is received, the SCP-returned CAIN group is available for any reoriginated call.

Note 2: CAIN group subscription for AXXESS agents is handled differently. Refer to *UCS DMS-250 CAIN/FlexDial Interactions*.

Trigger evaluation

NetworkBuilder checks the *Offhook_Delay* trigger table (OFFHKDEL) when an address is collected and evaluates the call's 1 to 24 digit address against the datafilled range associated with the appropriate CAIN group.

Note 1: The digit type (INFO, ADIN, ANI, ADDR, CIC) is identified through datafill within the OFFHKDEL table.

Trigger actions

NetworkBuilder call processing supports the following actions for the *Offhook_Delay* trigger:

- BLOCK – Prevents the call from proceeding and applies AINF treatment.

- IGNORE – NetworkBuilder continues checking the remaining subscription methods for *Shared_Interoffice_Trunk* or *PRI_B-Channel*. If datafill does not enable any trigger, call processing continues through the call model.
- QUERY – The switch builds an **`Info_Collected`** query and sends it to the SCP. The SCP analyzes the call and assists the switch with call processing.
- LEAVE_TDP – NetworkBuilder call processing exits the ***Info_Collected*** TDP with no further evaluation.
- CONT_NOTRIG – NetworkBuilder call processing exits the ***Info_Collected*** TDP and prevents any further NetworkBuilder interaction for the call. This prevention is reset upon reorigination.

Error actions

Error actions are not identified for the *Offhook_Delay* trigger. The switch applies AINF treatment for fatal application errors.

Options

The trigger supports the following options:

- BUFFER – to activate digit buffering while the SCP is queried.
- GT – to identify the global title used to identify the SCP handling the query.
- T1OVFLGT – to identify the specific SCP to query on T1 overflow.
- ACGOVFLGT – to identify the global title to use for requerying when a query is blocked by an ACG control
- VERSION – controls the CAIN protocol version for outgoing messages

  Note 1: The value provisioned in the VERSION option overrides the version provisioned in the CAIN_PROTOCOL_VERSION parameter of table CAINPARM on a transaction-by-transaction basis. Additionally, the VERSION option applies to both the outgoing and incoming messages within the same transaction. (Subsequent queries will use the setting provisioned in the CAIN_PROTOCOL_VERSION or provisioned setting of the VERSION option in the trigger table performing the subsequent query.)
- DIGITS – to allow CAIN to restrict the ability to query the SCP unless the number of address digits matches the number defined by this option.

Offhook_Delay trigger (continued)

- STREAM – to control protocol stream on a per-trigger tuple basis. The value of the STREAM option controls the set of parameters that are sent in NetworkBuilder messages.

CAIN parameter OFFHKDEL_TRIG_AFTER_TREAT

The CAIN parameter OFFHKDEL_TRIG_AFTER_TREAT in table CAINPARM gives CAIN the ability to prevent the process of evaluating the *Offhook_Delay* trigger by thte UCS DMS-250 facility if a treatment is detected by the UCS DMS-250 switch.

If the CAIN parameter is datafilled as ALLOWED, CAIN processing would evaluate the trigger. If the CAIN parameter is datafilled as NOT ALLOWED, CAIN processing would stop without evaluating the trigger.

Offhook_Delay trigger (continued)

The following figure shows how a call progresses through the call model, encounters ***Info_Collected*** and queries the SCP.

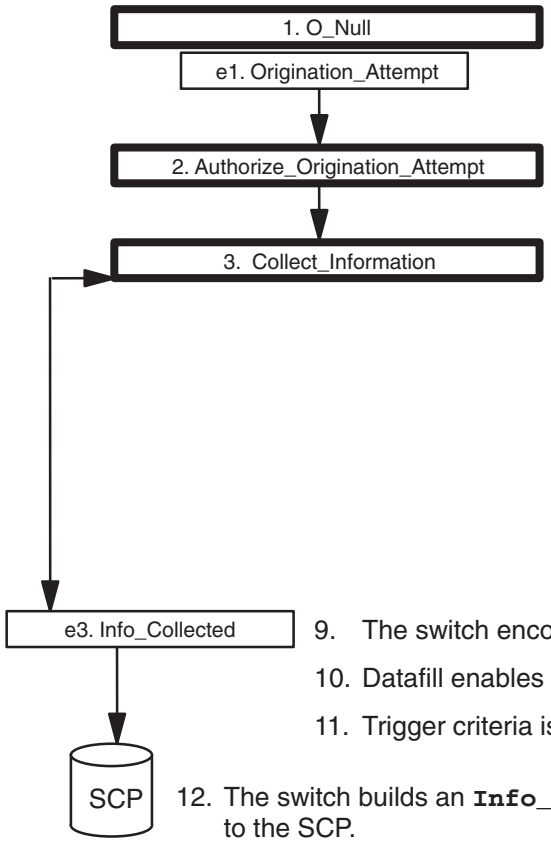
UA Auth call

FGD origination
Datafill:

Signaling: MF at origination
DTMF for digit collection
Dialing plan: KP + II + ANI + ST
KP + ADDR + ST
ADDR: (214) 611-3311

Table OFFHKDEL

OFFHKDELKEY	TODIGS	ACTION	OPTION
CAINGRP1 ADDR 2146113311	2146113311	QUERY	$



1. The switch detects seizure and enters **O_Null**.
2. Call processing enters **Authorize_Origination_Attempt**.
3. Call processing enters **Collect_Information**.
4. The switch collects KP + II + ANI + ST.
5. The switch screens II and ANI digits.
6. The switch collects KP + ADDR + ST.
7. The switch collects DTMF AUTH.
8. The switch collects KP + ADDR + ST.
9. The switch encounters ***Info_Collected***.
10. Datafill enables *Offhook_Delay* trigger.
11. Trigger criteria is met for the ADDR.
12. The switch builds an **`Info_Collected`** query and sends it to the SCP.

Offhook_Delay trigger (continued)

Info_Collected TDP-Request

An **`Info_Collected`** TDP-Request (query) is sent to the SCP in a query package, with a component type of Invoke_Last. The following table defines the parameters and usage requirements for the parameters the **`Info_Collected`** TDP-Request may contain. Refer to Volume 3, Chapter 8, "Outgoing CAIN message parameters," for detailed descriptions of the request parameters.

Note: Parameters and extension parameters for AXXESS agents are handled differently. Refer to *UCS DMS-250 CAIN/FlexDial Interactions* for more information.

Offhook_Delay trigger (continued)

Info_Collected TDP-Request message parameters for Offhook_Delay

Parameter	Usage	Definition
UserID	Required	Contains the network identity of the originating agent
BearerCapability	Required	Contains the bearer capability of the call when the message is built
ChargeNumber(Note 6)	Optional	Contains the billing number that would be used to populate the CDR at this point in call processing
Lata (Note 7)	Optional	Contains the call's Local Access and Transport Area (LATA) For PRI originations, LATA is populated with option LATANUM in table LATAID, from the LATA option in table CALLATTR.
Carrier	Optional	Contains the dialed CIC or the CIC value (with an indication of: Selected CIC presubscribed and not input by calling party) from table TRKGRP. For PRI originations, CARRIER is populated with the CIC value from the DEFCIC option in table CALLATTR.
TriggerCriteriaType	Optional	Contains offHookDelay

Note 1: The `universalAccess` extension parameter is included in the query (a flat 10 digits) when parameter CAIN_PROTOCOL_VERSION is V0. When CAIN_PROTOCOL_VERSION is set to V1 or higher, the parameter contains up to 24 digits including a nature of address and numbering plan similarly to the ***CalledPartyID***.
Note 2: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS07 or higher.
Note 3: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS08 or higher.
Note 4: This parameter is not populated when the CAIN_PROTOCOL_VERSION is set to V3 or higher, and is therefore not sent.
Note 5: The parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS09 or higher.
Note 6: When the CAIN_PROTOCOL_VERSION is set to V4 or higher non-standard charge numbers (CARD, AUTH, ACCT, PIN, and N00) no longer populate the ***ChargeNumber*** parameter and all standard NOAs are supported.
Note 7: The parameter is supported on DAL, FGD, and AXXESS agencies and the parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS11 or higher. With UCS17, PRI calls are also supported.
Note 8: Refer to Volume 3, Chapter 8, "Outgoing CAIN message parameters" for information on the population of this parameter when the CAIN_PROTOCOL_VERSION is set to V5 or higher.

—continued—

Offhook_Delay trigger (continued)

Info_Collected TDP-Request message parameters for Offhook_Delay (continued)

Parameter	Usage	Definition
CallingPartyID (Note 8)	Optional	Contains one of the following (listed in order of precedence): For SS7 FGD, SS7 Inter-IMT, SS7-Global IMT, and AXXESS calls: Calling_Party_Address from ISUP message, when available For FGD and AXXESS calls: valid ANI (information digits are not passed in this parameter) ***Note:*** Refer to *UCS DMS-250 CAIN/FlexDial Interactions* for more information on AXXESS agents. For PRI calls: CLID when available Valid SNPA value from table TRKGRP Default SNPA value from table CAINPARM
ChargePartyStationType (Note 2)	Optional	Contains the information digits for the call
AccessCode (Note 4)	Optional	Contains the account code, when available

Note 1: The `universalAccess` extension parameter is included in the query (a flat 10 digits) when parameter CAIN_PROTOCOL_VERSION is V0. When CAIN_PROTOCOL_VERSION is set to V1 or higher, the parameter contains up to 24 digits including a nature of address and numbering plan similarly to the ***CalledPartyID***.
Note 2: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS07 or higher.
Note 3: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS08 or higher.
Note 4: This parameter is not populated when the CAIN_PROTOCOL_VERSION is set to V3 or higher, and is therefore not sent.
Note 5: The parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS09 or higher.
Note 6: When the CAIN_PROTOCOL_VERSION is set to V4 or higher non-standard charge numbers (CARD, AUTH, ACCT, PIN, and N00) no longer populate the ***ChargeNumber*** parameter and all standard NOAs are supported.
Note 7: The parameter is supported on DAL, FGD, and AXXESS agencies and the parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS11 or higher. With UCS17, PRI calls are also supported.
Note 8: Refer to Volume 3, Chapter 8, “Outgoing CAIN message parameters” for information on the population of this parameter when the CAIN_PROTOCOL_VERSION is set to V5 or higher.

—continued—

Offhook_Delay trigger (continued)

Info_Collected TDP-Request message parameters for Offhook_Delay (continued)

Parameter	Usage	Definition
CollectedAddressInfo	Optional	Contains the address collected from the incoming agent (from the IAM, subscriber dialing, or datafilled hotline digits) For PRI originations, if the called party number information element (CdPN IE) contains a 7–digit called number and the TRGPREFX option is in table CALLATTR, the NPA from the SNPA field in table BCHNLMEM is prepended to the called number.
CollectedDigits (Note 4)	Optional	Contains the collected PIN digits, when available
VerticalServiceCode	Optional	Contains feature codes dialed by the subscriber
ACGEncountered	Optional	Contains the ACG control encountered when the ACG control expires and a query is allowed to be sent
ExtensionParameter (Note 2)	Optional	Extension parameters require the CAIN0200 SOC option.
universalAccess (Note 1)	Optional	Contains the universal access number dialed by the caller to obtain switch dial tone.

Note 1: The universalAccess extension parameter is included in the query (a flat 10 digits) when parameter CAIN_PROTOCOL_VERSION is V0. When CAIN_PROTOCOL_VERSION is set to V1 or higher, the parameter contains up to 24 digits including a nature of address and numbering plan similarly to the ***CalledPartyID***.
Note 2: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS07 or higher.
Note 3: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS08 or higher.
Note 4: This parameter is not populated when the CAIN_PROTOCOL_VERSION is set to V3 or higher, and is therefore not sent.
Note 5: The parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS09 or higher.
Note 6: When the CAIN_PROTOCOL_VERSION is set to V4 or higher non-standard charge numbers (CARD, AUTH, ACCT, PIN, and N00) no longer populate the ***ChargeNumber*** parameter and all standard NOAs are supported.
Note 7: The parameter is supported on DAL, FGD, and AXXESS agencies and the parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS11 or higher. With UCS17, PRI calls are also supported.
Note 8: Refer to Volume 3, Chapter 8, “Outgoing CAIN message parameters” for information on the population of this parameter when the CAIN_PROTOCOL_VERSION is set to V5 or higher.

—continued—

Offhook_Delay trigger (continued)

Info_Collected TDP-Request message parameters for Offhook_Delay (continued)

Parameter	Usage	Definition
`cainGroup`	Optional	Contains the group number (field GRPNUM, table CAINGRP) for the CAIN group associated with the trigger when field EXTPARM in table CAINGRP contains CAINGRP.
`adin`	Optional	Contains the authorization code database index (field ADIN, table TRKGRP for DAL and FGD agencies; field ADIN, table CALLATTR for PRI agencies) associated with the originating agency when field EXTPARM in table CAINGRP contains ADIN. ***Note:*** The `adin` extension parameter is not supported for AXXESS agents.
`origTrunkInfo`	Optional	Contains the originating trunk group number, trunk type, and trunk member number when field EXTPARM in table CAINGRP contains ORGTINFO.
`treatment`	Optional	Contains the treatment set by regular call processing before the query was sent
`reorigCall`	Optional	Presence of this parameter indicates the call in progress is a result of reorigination

Note 1: The `universalAccess` extension parameter is included in the query (a flat 10 digits) when parameter CAIN_PROTOCOL_VERSION is V0. When CAIN_PROTOCOL_VERSION is set to V1 or higher, the parameter contains up to 24 digits including a nature of address and numbering plan similarly to the ***`CalledPartyID`***.
Note 2: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS07 or higher.
Note 3: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS08 or higher.
Note 4: This parameter is not populated when the CAIN_PROTOCOL_VERSION is set to V3 or higher, and is therefore not sent.
Note 5: The parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS09 or higher.
Note 6: When the CAIN_PROTOCOL_VERSION is set to V4 or higher non-standard charge numbers (CARD, AUTH, ACCT, PIN, and N00) no longer populate the ***`ChargeNumber`*** parameter and all standard NOAs are supported.
Note 7: The parameter is supported on DAL, FGD, and AXXESS agencies and the parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS11 or higher. With UCS17, PRI calls are also supported.
Note 8: Refer to Volume 3, Chapter 8, “Outgoing CAIN message parameters” for information on the population of this parameter when the CAIN_PROTOCOL_VERSION is set to V5 or higher.

—continued—

Offhook_Delay trigger (continued)

Info_Collected TDP-Request message parameters for Offhook_Delay (continued)

Parameter	Usage	Definition
univIdx	Optional	Used by SS7 Global-IMT agents only. Contains the universal translations scheme to be used for the call.
netinfo (Note 2)	Optional	Contains external network ID, network customer group ID, and network class of service.
t1Overflow	Optional	Indicates to the SCP that an overflow occurred on the query to the initial SCP.
lnpReceived (Note 2)	Optional	Presence of this parameter indicates LNP information was received from a previous switch.
subscriptionInfo (Note 3)	Optional	Contains the digit type triggered on for the query and the method to which the triggering CAIN group subscribed.
jurisdictionInfo (Note 3)	Optional	Contains the originating switch's LRN.
switchID (Note 5)	Optional	Contains the switch ID
accountCode (Note 5)	Optional	Contains the account code when available
pinDigits (Note 5)	Optional	Contains the collected pin digits when available

Note 1: The universalAccess extension parameter is included in the query (a flat 10 digits) when parameter CAIN_PROTOCOL_VERSION is V0. When CAIN_PROTOCOL_VERSION is set to V1 or higher, the parameter contains up to 24 digits including a nature of address and numbering plan similarly to the ***CalledPartyID***.
Note 2: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS07 or higher.
Note 3: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS08 or higher.
Note 4: This parameter is not populated when the CAIN_PROTOCOL_VERSION is set to V3 or higher, and is therefore not sent.
Note 5: The parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS09 or higher.
Note 6: When the CAIN_PROTOCOL_VERSION is set to V4 or higher non-standard charge numbers (CARD, AUTH, ACCT, PIN, and N00) no longer populate the ***ChargeNumber*** parameter and all standard NOAs are supported.
Note 7: The parameter is supported on DAL, FGD, and AXXESS agencies and the parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS11 or higher. With UCS17, PRI calls are also supported.
Note 8: Refer to Volume 3, Chapter 8, "Outgoing CAIN message parameters" for information on the population of this parameter when the CAIN_PROTOCOL_VERSION is set to V5 or higher.

—continued—

Offhook_Delay trigger (continued)

Info_Collected TDP-Request message parameters for Offhook_Delay (continued)

Parameter	Usage	Definition
`billingNumber` (Note 5)	Optional	Contains the non-standard charge number
`acgRequery`	Optional	When present, indicates to the SCP that the current query is a re-query message resulting from the original query being blocked by an ACG control

Note 1: The `universalAccess` extension parameter is included in the query (a flat 10 digits) when parameter CAIN_PROTOCOL_VERSION is V0. When CAIN_PROTOCOL_VERSION is set to V1 or higher, the parameter contains up to 24 digits including a nature of address and numbering plan similarly to the ***CalledPartyID***.
Note 2: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS07 or higher.
Note 3: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS08 or higher.
Note 4: This parameter is not populated when the CAIN_PROTOCOL_VERSION is set to V3 or higher, and is therefore not sent.
Note 5: The parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS09 or higher.
Note 6: When the CAIN_PROTOCOL_VERSION is set to V4 or higher non-standard charge numbers (CARD, AUTH, ACCT, PIN, and N00) no longer populate the ***ChargeNumber*** parameter and all standard NOAs are supported.
Note 7: The parameter is supported on DAL, FGD, and AXXESS agencies and the parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS11 or higher. With UCS17, PRI calls are also supported.
Note 8: Refer to Volume 3, Chapter 8, “Outgoing CAIN message parameters” for information on the population of this parameter when the CAIN_PROTOCOL_VERSION is set to V5 or higher.

—end—

Once the request message is built, NetworkBuilder formats the message and sends it to the message encoder. Once encoded, the message is sent to the SCP.

SCP response processing

The following response messages are supported:

- **Analyze_Route**
- **Send_To_Resource**
- **Disconnect**

  Note 1: EDPs may be armed through the **Request_Report_BCM_Event** component with the **Analyze_Route** message.

  Note 2: A **Termination_Notification** may be requested through the **Send_Notification** component with the SCP response message.

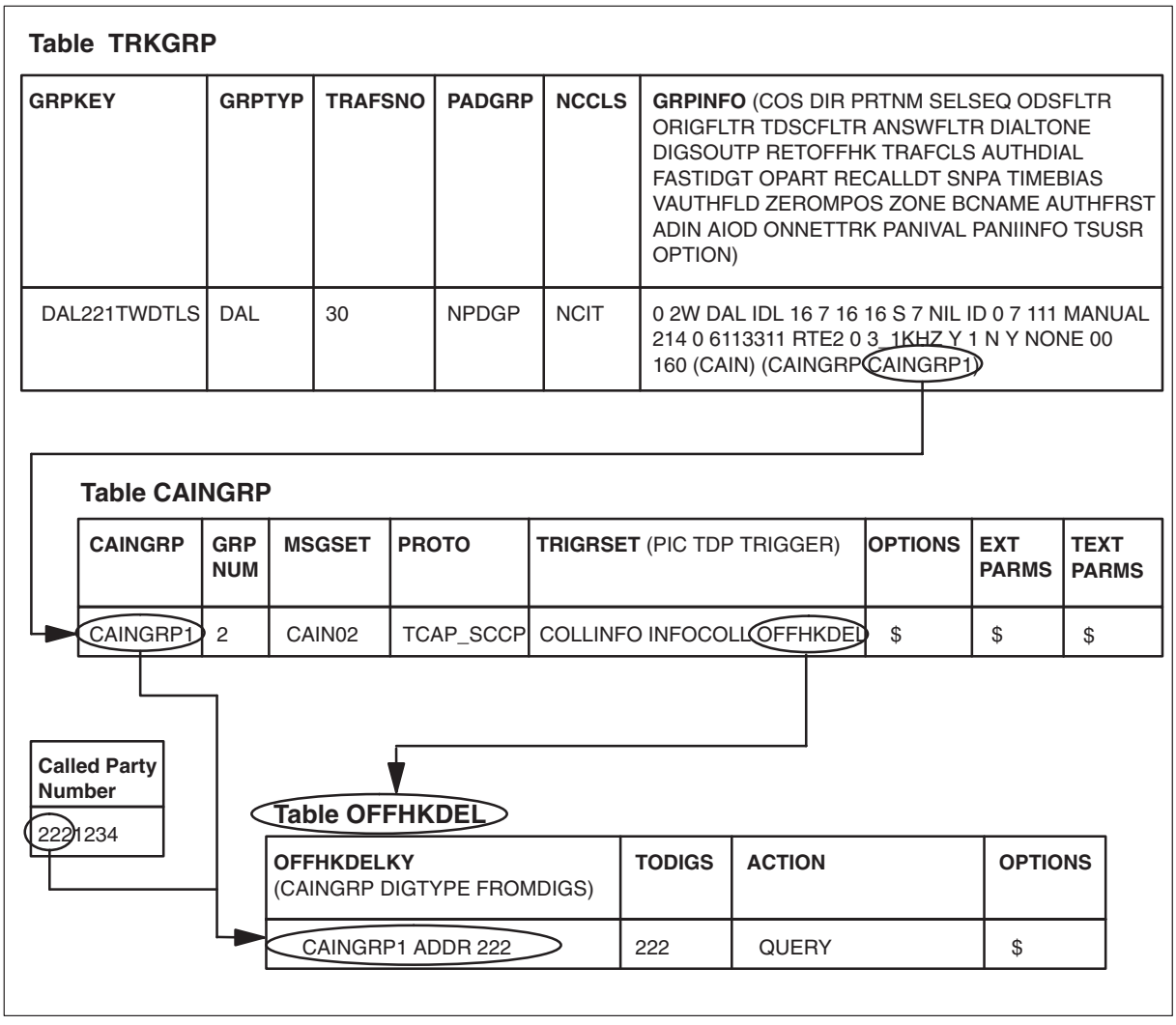
Note 3: An **ACG** message may be sent with the SCP response message.

Refer to Volume 3, Chapter 10, "Incoming CAIN messages," for specific information regarding response messages; refer to Volume 3, Chapter 12, "Incoming CAIN message parameters," for detailed descriptions of the response parameters; refer to Volume 3, Chapter 3, "Event processing," for EDP specific message information.

Datafill

The following figure shows how a subscription table interacts with the *Offhook_Delay* trigger table (OFFHKDEL).

Subscription-OFFHKDEL table interaction



Offhook_Delay trigger (end)

Provisioning the Offhook_Delay trigger

At the CI prompt

1 Provision the originating agent as CAIN-capable

2 Subscribe to a CAIN group (table STDPRTCT, authcode tables, ANI tables, TRKGRP, or CAINPARM)

Note: CAIN group subscription is handled differently for AXXESS agents. Refer to *UCS DMS-250 CAIN/FlexDial Interactions*.

3 Datafill CAIN group trigger subscription to the (COLLINFO INFOCOLL OFFHKDEL) trigger set (table CAINGRP).

4 Enter table OFFHKDEL.

5 Define the trigger criteria for a CAIN group by using the following format:

>ADD offhkdelky todigs action options
where

offhkdelky is comprised of three subfields: CAINGRP, DIGTYPE, FROMDIGS, where
- CAINGRP is the CAIN group that enables the trigger (from table CAINGRP).
- DIGTYPE is the digit type being analyzed (INFO, ADIN, ANI, ADDR,CIC).
- FROMDIGS is the first number used to define the range digits (0 to 9, *, #).

todigs is the second number used to define the range of digits (0 to 9, *, #).

action is the trigger action taken when the specified digits are within the FROMDIGS-TODIGS range (BLOCK, IGNORE, QUERY, LEAVE_TDP, CONT_NOTRIG).

6 Datafill the OPTIONS, where

options is only allowed when ACTION is QUERY. Enter up to 7 options: BUFFER, GT, T1OVFLGT, ACGOVFLGT, VERSION, DIGITS, or STREAM. When the ACTION is not QUERY enter $.

Sample entry: **>ADD caingrp1 addr 222 222 query gt cain_addr_gt $**

Offhook_Delay trigger criteria is defined.

Associated OMs

CAINMSGS, CAINTRIG, CAINAGOM

Shared_Interoffice_Trunk trigger

ATTENTION

The *Shared_Interoffice_Trunk* trigger requires the CAIN0503 SOC option. Refer to Volume 5, Chapter 5, "NetworkBuilder SOC functionality," for more information.

Uses

When you provision NetworkBuilder services on the switch, the SCP can make major call processing decisions. You define the services offered on the SCP. Some examples of services that can be developed on the SCP for the *Shared_Interoffice_Trunk* trigger are:

- Dial 1+ Services
  - — Speed Dial
  - — Hotline
  - — Account Code Screening
  - — Bill to Office
  - — Prepaid services
  - — CIC Routing and Branding
  - — Chat Lines
  - — Black Box Screening
- N00 Services
  - — Follow Me, Find Me, Do Not Disturb Me
  - — Call Screening
  - — Customized Call Branding
  - — Universal Access Authorization
- Subscriber Screening
  - — Authorization Code Screening
  - — Account Code Screening

Supported originating agencies

The *Shared_Interoffice_Trunk* trigger supports the following originating agencies:

- DAL
- FGD

Shared_Interoffice_Trunk trigger (continued)

- SS7 Inter-IMT
- SS7 Global-IMT
- AXXESS

  Note: Refer to *UCS DMS-250 CAIN/FlexDial Interactions* for more information on support for AXXESS agents.

Subscribing to the Shared_Interoffice_Trunk trigger

Subscription to the *Shared_Interoffice_Trunk* trigger is available on a

- SCP-returned basis
- address basis (table STDPRTCT, subtable STDPRT)
- authorization code basis (tables AUTHCODU, AUTHCDU2, AUTHCDU3, AUTHCDU4, or AUTHCDU5)
- ANI basis (table ANISCUSP, or tables ANIVAL and UNIPROF)
- agent basis (table TRKGRP)
- office basis (table CAINPARM)

Note: An SCP-returned CAIN subscription group may be received in **`Analyze_Route`** messages. When a reorigination occurs after an **`Analyze_Route`** is received, the SCP-returned CAIN group is available for any reoriginated call.

Note: CAIN group subscription for AXXESS agents is handled differently. Refer to *UCS DMS-250 CAIN/FlexDial Interactions*.

Trigger evaluation

NetworkBuilder checks the *Shared_Interoffice_Trunk* trigger table (SIOTRK) and evaluates the identified digits against the datafilled range associated with the appropriate CAIN group.

Note: The digit type (information, ANI, address, CIC, or ADIN) is identified through datafill within the SIOTRK table.

Trigger actions

NetworkBuilder call processing supports the following actions for the *Shared_Interoffice_Trunk* trigger:

- BLOCK – AINF treatment is applied.
- IGNORE – NetworkBuilder continues checking the remaining subscription methods for *Shared_Interoffice_Trunk*. If datafill does not enable the trigger, call processing continues through the call model.

- QUERY – The switch builds an **`Info_Collected`** query and sends it to the SCP. The SCP analyzes the call and assists the switch with call processing.
- QUERYSCU – In-switch call processing is temporarily suspended. The call enters server mode and becomes a Programmable Service Node (PSN) call and a **`New_Call`** message is sent to the PSN SCU. Refer to *UCS DMS-250 Programmable Service Node (PSN) Application Guide* for more PSN information.

  Note: If an in-switch error occurs before all digits are present the call goes to treatment and does not become a PSN call as it would have under the existing NetworkBuilder functionality.

- LEAVE_TDP – NetworkBuilder call processing exits the ***Info_Collected*** TDP with no further evaluation.
- CONT_NOTRIG – NetworkBuilder call processing exits the ***Info_Collected*** TDP and prevents any further NetworkBuilder interaction for the call. This prevention is reset upon reorigination.

Trigger criteria

When a query message is built, ***`TriggerCriteriaType`*** is populated by the datafill in field TRIGCRIT (table SIOTRK). Valid datafill values are: STD, SIO_ADDR, SIO_ADIN, SIO_ANI, SIO_INFO, SIO_CIC, SIO_N00, and SIO_INTL.

Error actions

Error actions are not identified for the *Shared_Interoffice_Trunk* trigger. The switch applies AINF treatment for fatal application errors.

Options

The *Shared_Interoffice_Trunk* trigger supports the following options:

- BUFFER – to activate digit buffering while the SCP is queried.
- GT – to identify the global title used to identify the SCP handling the query.
- T1OVFLGT – to identify the specific SCP to query on T1 overflow.
- ACGOVFLGT – to identify the global title to use for requerying when a query is blocked by an ACG control
- VERSION – controls the CAIN protocol version for outgoing messages

Shared_Interoffice_Trunk trigger (continued)

Note 1: The value provisioned in the VERSION option overrides the version provisioned in the CAIN_PROTOCOL_VERSION parameter of table CAINPARM on a transaction-by-transaction basis. Additionally, the VERSION option applies to both the outgoing and incoming messages within the same transaction. (Subsequent queries will use the setting provisioned in the CAIN_PROTOCOL_VERSION or provisioned setting of the VERSION option in the trigger table performing the subsequent query.)

- STREAM – to control protocol stream on a per-trigger tuple basis. The value of the STREAM option controls the set of parameters in NetworkBuilder messages.

Shared_Interoffice_Trunk trigger (continued)

The following figure shows how a call progresses through the call model, encounters ***Info_Collected*** and queries the SCP.

UA Auth call

FGD origination
Datafill:

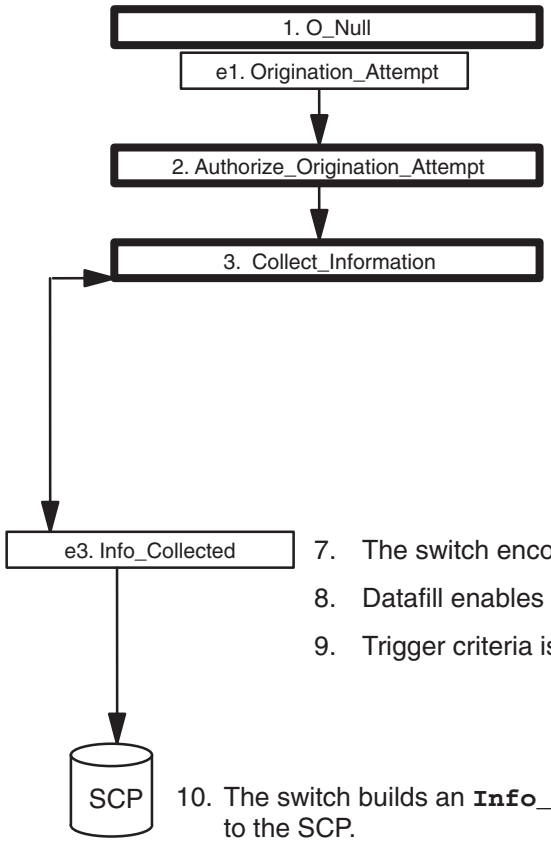
Signaling: MF at origination
DTMF for digit collection

Dialing plan: KP + II + ANI + ST
KP + ADDR + ST

ANI: (214) 611-3311

Table SIOTRK

SIOTRKKY	TODIGS	ACTION	OPTION
CAINGRP1 ANI 2146113311	2146113311	QUERY SIO_ANI	$



1. The switch detects seizure and enters **O_Null**.
2. Call processing enters **Authorize_Origination_Attempt**.
3. Call processing enters **Collect_Information**.
4. The switch collects KP + II + ANI + ST.
5. The switch screens II and ANI digits.
6. The switch collects KP + ADDR + ST (In this case, ADDR is a UA auth number datafilled in table STDPRTCT.).
7. The switch encounters ***Info_Collected***.
8. Datafill enables *Shared_Interoffice_Trunk* trigger.
9. Trigger criteria is met for the ANI.
10. The switch builds an **`Info_Collected`** query and sends it to the SCP.

Shared_Interoffice_Trunk trigger (continued)

Info_Collected TDP-Request

An **Info_Collected** TDP-Request (query) is sent to the SCP in a query package, with a component type of Invoke_Last. The following table defines the parameters and usage requirements for the parameters the **Info_Collected** TDP-Request may contain. Refer to Chapter 14, "Outgoing message parameters," for detailed descriptions of the request parameters.

Note: Parameters and extension parameters for AXXESS agents are handled differently. Refer to *UCS DMS-250 CAIN/FlexDial Interactions* for more information.

Info_Collected TDP-Request message parameters for Shared_Interoffice_Trunk

Parameter	Usage	Definition
UserID	Required	Contains the network identity of the originating agent
BearerCapability	Required	Contains the bearer capability of the call when the message is built
ChargeNumber (Note 6)	Optional	Contains the billing number that would be used to populate the CDR at this point in call processing
Lata (Note 7)	Optional	Contains the call's Local Access and Transport Area (LATA)

Note 1: The `universalAccess` extension parameter is included in the query (a flat 10 digits) when parameter CAIN_PROTOCOL_VERSION is V0. When the CAIN_PROTOCOL_VERSION is set to V1 or higher, the parameter contains up to 24 digits including a nature of address and numbering plan similarly to the ***CalledPartyID***.
Note 2: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS07 or higher.
Note 3: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS08 or higher.
Note 4: This parameter is not populated when the CAIN_PROTOCOL_VERSION is set to V3 or higher, and is therefore not sent.
Note 5: The parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS09 or higher.
Note 6: When the CAIN_PROTOCOL_VERSION is set to V4 or higher non-standard charge numbers (CARD, AUTH, ACCT, PIN, and N00) no longer populate the ***ChargeNumber*** parameter and all standard NOAs are supported.
Note 7: The parameter is supported only on DAL, FGD, and AXXESS agencies and the parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS11 or higher.
Note 8: Refer to Volume 3, Chapter 8, "Outgoing CAIN message parameters" for information on the population of this parameter when the CAIN_PROTOCOL_VERSION is set to V5 or higher.

—continued—

Shared_Interoffice_Trunk trigger (continued)

Info_Collected TDP-Request message parameters for Shared_Interoffice_Trunk (continued)

Parameter	Usage	Definition
Carrier	Optional	Contains the dialed CIC or the CIC value (with an indication of: Selected CIC presubscribed and not input by calling party) from table TRKGRP.
TriggerCriteriaType	Optional	Contains sharedIOTrunk, SIO_CIC, SIO_INFO, SIO_ANI, SIO_ADDR, SIO_ADIN, SIO_N00, or SIO_INTL
CallingPartyID (Note 8)	Optional	Contains one of the following (listed in order of precedence): For SS7 FGD calls: Calling_Party_Address from ISUP message, when available For FGD and AXXESS calls: valid ANI (information digits are not passed in this parameter) ***Note:*** Refer to *UCS DMS-250 CAIN/FlexDial Interactions* for more information on AXXESS agents. Valid SNPA value from table TRKGRP Default SNPA value from table CAINPARM
ChargePartyStationType (Note 2)	Optional	Contains the information digits for the call

Note 1: The `universalAccess` extension parameter is included in the query (a flat 10 digits) when parameter CAIN_PROTOCOL_VERSION is V0. When the CAIN_PROTOCOL_VERSION is set to V1 or higher, the parameter contains up to 24 digits including a nature of address and numbering plan similarly to the ***CalledPartyID***.
Note 2: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS07 or higher.
Note 3: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS08 or higher.
Note 4: This parameter is not populated when the CAIN_PROTOCOL_VERSION is set to V3 or higher, and is therefore not sent.
Note 5: The parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS09 or higher.
Note 6: When the CAIN_PROTOCOL_VERSION is set to V4 or higher non-standard charge numbers (CARD, AUTH, ACCT, PIN, and N00) no longer populate the ***ChargeNumber*** parameter and all standard NOAs are supported.
Note 7: The parameter is supported only on DAL, FGD, and AXXESS agencies and the parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS11 or higher.
Note 8: Refer to Volume 3, Chapter 8, “Outgoing CAIN message parameters” for information on the population of this parameter when the CAIN_PROTOCOL_VERSION is set to V5 or higher.

—continued—

Shared_Interoffice_Trunk trigger (continued)

Info_Collected TDP-Request message parameters for Shared_Interoffice_Trunk (continued)

Parameter	Usage	Definition
AccessCode (Note 4)	Optional	Contains the account code, when available
CollectedAddressInfo	Optional	Contains the address collected from the incoming agent (from the IAM, subscriber dialing, or datafilled hotline digits)
CollectedDigits (Note 4)	Optional	Contains the collected PIN digits, when available
VerticalServiceCode	Optional	Contains feature codes dialed by the subscriber
ACGEncountered	Optional	Contains the ACG control encountered when the ACG control expires and a query is allowed to be sent
ExtensionParameter (Note 2)	Optional	Extension parameters require the CAIN0200 SOC option.
universalAccess (Note 1)	Optional	Contains the original address when the *O_Feature_Requested* ADDR collectible is executed or the universal access number dialed by the caller to obtain switch dial tone. Since this number is not the actual address for the call, it is not transported in ***CollectedAddressInfo*** or ***CalledPartyID***.

Note 1: The universalAccess extension parameter is included in the query (a flat 10 digits) when parameter CAIN_PROTOCOL_VERSION is V0. When the CAIN_PROTOCOL_VERSION is set to V1 or higher, the parameter contains up to 24 digits including a nature of address and numbering plan similarly to the ***CalledPartyID***.
Note 2: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS07 or higher.
Note 3: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS08 or higher.
Note 4: This parameter is not populated when the CAIN_PROTOCOL_VERSION is set to V3 or higher, and is therefore not sent.
Note 5: The parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS09 or higher.
Note 6: When the CAIN_PROTOCOL_VERSION is set to V4 or higher non-standard charge numbers (CARD, AUTH, ACCT, PIN, and N00) no longer populate the ***ChargeNumber*** parameter and all standard NOAs are supported.
Note 7: The parameter is supported only on DAL, FGD, and AXXESS agencies and the parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS11 or higher.
Note 8: Refer to Volume 3, Chapter 8, “Outgoing CAIN message parameters” for information on the population of this parameter when the CAIN_PROTOCOL_VERSION is set to V5 or higher.

—continued—

Shared_Interoffice_Trunk trigger (continued)

Info_Collected TDP-Request message parameters for Shared_Interoffice_Trunk (continued)

Parameter	Usage	Definition
cainGroup	Optional	Contains the group number (field GRPNUM, table CAINGRP) for the CAIN group associated with the trigger when field EXTPARM in table CAINGRP contains CAINGRP.
adin	Optional	Contains the authorization code database index (field ADIN in table TRKGRP) associated with the originating agency when field EXTPARM in table CAINGRP contains ADIN. ***Note:*** The adin extension parameter is not supported for AXXESS agents.
origTrunkInfo	Optional	Contains the originating trunk group number, trunk type, and trunk member number when field EXTPARM in table CAINGRP contains ORGTINFO.
treatment	Optional	Contains the treatment if set by regular call processing before the query was sent
reorigCall	Optional	Presence of this parameter indicates the call in progress is a result of reorigination
univIdx	Optional	Used by SS7 Global-IMT agents only. Contains the universal translations scheme to be used for the call.

Note 1: The universalAccess extension parameter is included in the query (a flat 10 digits) when parameter CAIN_PROTOCOL_VERSION is V0. When the CAIN_PROTOCOL_VERSION is set to V1 or higher, the parameter contains up to 24 digits including a nature of address and numbering plan similarly to the ***CalledPartyID***.
Note 2: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS07 or higher.
Note 3: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS08 or higher.
Note 4: This parameter is not populated when the CAIN_PROTOCOL_VERSION is set to V3 or higher, and is therefore not sent.
Note 5: The parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS09 or higher.
Note 6: When the CAIN_PROTOCOL_VERSION is set to V4 or higher non-standard charge numbers (CARD, AUTH, ACCT, PIN, and N00) no longer populate the ***ChargeNumber*** parameter and all standard NOAs are supported.
Note 7: The parameter is supported only on DAL, FGD, and AXXESS agencies and the parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS11 or higher.
Note 8: Refer to Volume 3, Chapter 8, “Outgoing CAIN message parameters” for information on the population of this parameter when the CAIN_PROTOCOL_VERSION is set to V5 or higher.

—continued—

Shared_Interoffice_Trunk trigger (continued)

Info_Collected TDP-Request message parameters for Shared_Interoffice_Trunk (continued)

Parameter	Usage	Definition
`netinfo` (Note 2)	Optional	Contains external network ID, network customer group ID, and network class of service.
`t1Overflow`	Optional	Indicates to the SCP that an overflow occurred on the query to the initial SCP.
`lnpReceived` (Note 2)	Optional	Presence of this parameter indicates LNP information was received from a previous switch.
`subscriptionInfo` (Note 3)	Optional	Contains the digit type triggered on for the query and the method to which the triggering CAIN group subscribed.
`jurisdictionInfo` (Note 3)	Optional	Contains the originating switch's LRN.
`switchID` (Note 5)	Optional	Contains the switch ID
`accountCode` (Note 5)	Optional	Contains the account code when available
`pinDigits` (Note 5)	Optional	Contains the collected pin digits when available
`billingNumber` (Note 5)	Optional	Contains the non-standard charge number
`acgRequery`	Optional	When present, indicates to the SCP that the current query is a re-query message resulting from the original query being blocked by an ACG control

Note 1: The `universalAccess` extension parameter is included in the query (a flat 10 digits) when parameter CAIN_PROTOCOL_VERSION is V0. When the CAIN_PROTOCOL_VERSION is set to V1 or higher, the parameter contains up to 24 digits including a nature of address and numbering plan similarly to the ***`CalledPartyID`***.
Note 2: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS07 or higher.
Note 3: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS08 or higher.
Note 4: This parameter is not populated when the CAIN_PROTOCOL_VERSION is set to V3 or higher, and is therefore not sent.
Note 5: The parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS09 or higher.
Note 6: When the CAIN_PROTOCOL_VERSION is set to V4 or higher non-standard charge numbers (CARD, AUTH, ACCT, PIN, and N00) no longer populate the ***`ChargeNumber`*** parameter and all standard NOAs are supported.
Note 7: The parameter is supported only on DAL, FGD, and AXXESS agencies and the parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS11 or higher.
Note 8: Refer to Volume 3, Chapter 8, "Outgoing CAIN message parameters" for information on the population of this parameter when the CAIN_PROTOCOL_VERSION is set to V5 or higher.

—end—

Once the request message is built, NetworkBuilder formats the message and sends it to the message encoder. Once encoded, the message is sent to the SCP.

SCP response processing

The following response messages are supported:

- **Analyze_Route**
- **Send_To_Resource**
- **Disconnect**

Note 1: EDPs may be armed through the **Request_Report_BCM_Event** component with the **Analyze_Route** message.

Note 2: A **Termination_Notification** may be requested through the **Send_Notification** component with the SCP response message.

Note 3: An **ACG** message may be sent with the SCP response message.

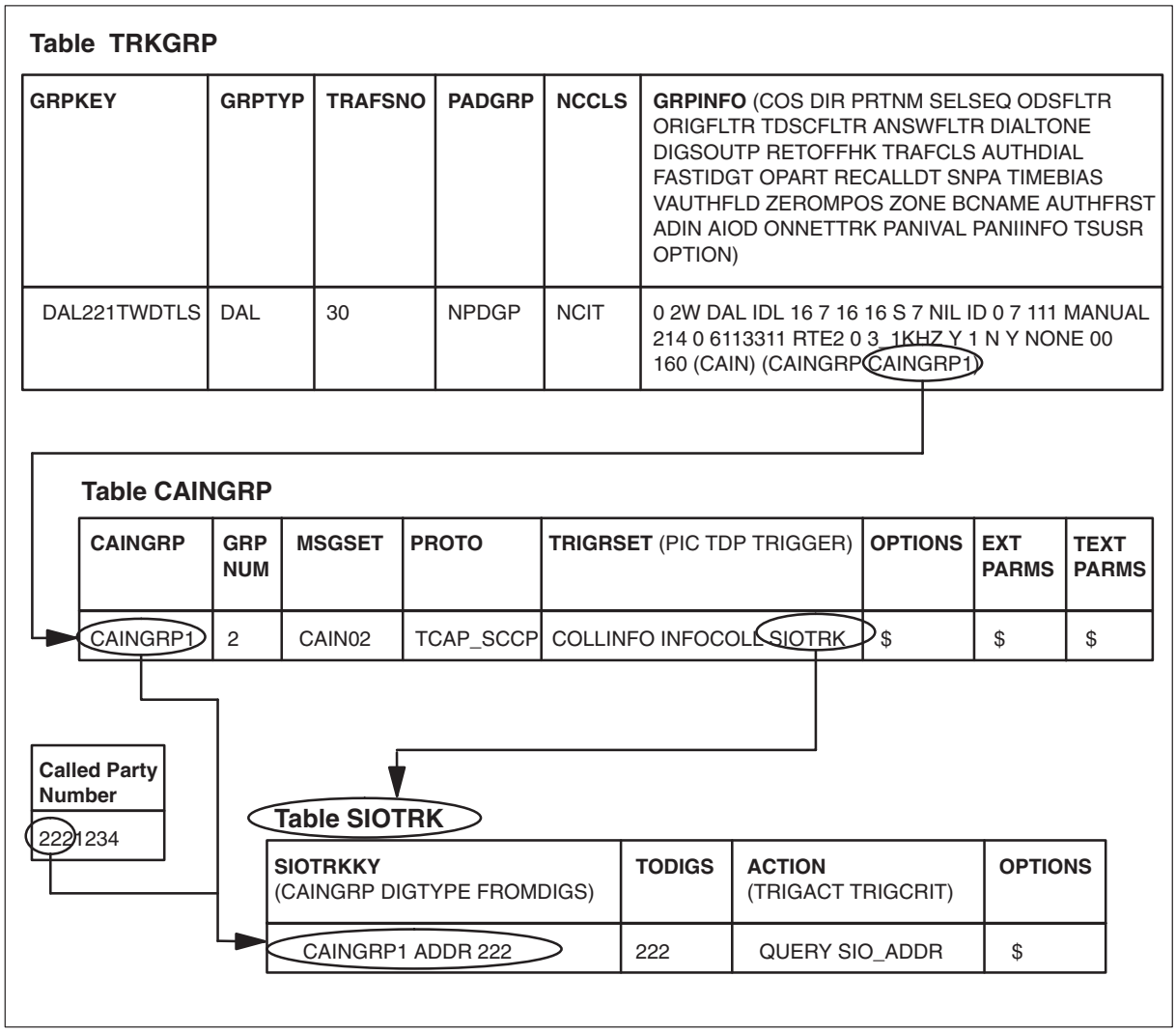
Refer to Volume 3, Chapter 10, "Incoming CAIN messages," for specific information regarding response messages; refer to Volume 3, Chapter 12, "Incoming CAIN message parameters," for detailed descriptions of the response parameters; refer to Volume 3, Chapter 3, "Event processing," for EDP specific message information.

Datafill

The following figure shows how a subscription table interacts with the *Shared_Interoffice_Trunk* trigger table (SIOTRK).

Shared_Interoffice_Trunk trigger (continued)

Subscription-SIOTRK table interaction



Provisioning the Shared_Interoffice_Trunk trigger

At the CI prompt

1 Provision the originating agent as CAIN-capable

2 Subscribe to a CAIN group (table STDPRTCT, authcode tables, ANI tables, TRKGRP, or CAINPARM)

Note: CAIN group subscription is handled differently for AXXESS agents. Refer to *UCS DMS-250 CAIN/FlexDial Interactions*.

3 Datafill CAIN group trigger subscription to the (COLLINFO INFOCOLL SIOTRK) trigger set (table CAINGRP).

4 Enter table SIOTRK.

5 Define the trigger criteria for a CAIN group by using the following format:

>ADD siotrkky todigs action options
where

siotrkky is comprised of three subfields: CAINGRP, DIGTYPE, FROMDIGS, where
- CAINGRP is the CAIN group that enables the trigger (from table CAINGRP).
- DIGTYPE is the digit type being analyzed (INFO, ADIN, ANI, ADDR,CIC).
- FROMDIGS is the first number used to define the range digits (0 to 9, *, #).

todigs is the second number used to define the range of digits (0 to 9, *, #).

action is comprised of two subfields: TRIGACT and TRIGCRIT, where
- TRIGACT is the trigger action taken when the specified digits are within the FROMDIGS-TODIGS range (BLOCK, IGNORE, QUERY, QUERYSCU, LEAVE_TDP, CONT_NOTRIG).
- TRIGCRIT is a refinement only available when ACTION is QUERY. Enter the trigger criteria sent to the SCP to identify the digits (STD, SIO_CIC, SIO_INFO, SIO_ANI, SIO_ADDR, SIO_ADIN, SIO_N00, or SIO_INTL)

Note: The TRIGCRIT refinement value is dependent on the DIGTYPE (with the exception of the STD trigger criteria). When DIGTYPE is INFO, TRIGCRIT must be SIO_INFO or STD; when DIGTYPE is ADDR, TRIGCRIT can be SIO_ADDR, SIO_INTL, SIO_N00, or STD; when DIGTYPE is ANI, TRIGCRIT must be SIO_ANI or STD; when DIGTYPE is CIC, TRIGCRIT must be SIO_CIC or STD; when DIGTYPE is ADIN, TRIGCRIT must be SIO_ADIN or STD.

6 Datafill the OPTIONS, where

options is only allowed when ACTION is QUERY. Enter up to 6 options: BUFFER, GT, T1OVFLGT, ACGOVFLGT, VERSION, or STREAM. When the ACTION is not QUERY enter $.

Sample entry: **>ADD caingrp1 addr 222 222 query sio_addr gt cain_addr_gt $**

Sample entry: **>ADD caingrp1 addr 222 222 query std gt cain_addr_gt $**

Shared_Interoffice_Trunk trigger (end)

Shared_Interoffice_Trunk trigger criteria is defined.

Associated OMs

CAINMSGS, CAINTRIG, CAINAGOM

PRI_B-Channel trigger

ATTENTION

The *PRI_B-Channel* trigger requires the CAIN0504 SOC option. Refer to Volume 5, Chapter 5, "NetworkBuilder SOC functionality," for more information.

Uses

When you provision NetworkBuilder services on the switch, the SCP can make major call processing decisions. You define the services offered on the SCP. Some examples of services that can be developed on the SCP for the *PRI_B-Channel* trigger are:

- Dial 1+ Services
  - — Speed Dial
  - — Hotline
  - — Account Code Screening
  - — Bill to Office
  - — Prepaid services
  - — CIC Routing and Branding
  - — Chat Lines
  - — Black Box Screening
- N00 Services
  - — Follow Me, Find Me, Do Not Disturb Me
  - — Call Screening
  - — Customized Call Branding
  - — Universal Access Authorization
- Subscriber Screening
  - — Authorization Code Screening
  - — Account Code Screening

Supported originating agencies

The *PRI_B-Channel* trigger supports the PRI originating agency.

Note: PRI does not support reorigination.

PRI_B-Channel trigger (continued)

Subscribing to the PRI_B-Channel trigger

Subscription to the *PRI_B-Channel* trigger is available on a

- SCP-returned basis
- address basis (table STDPRTCT, subtable STDPRT)
- authorization code basis (tables AUTHCODU, AUTHCDU2, AUTHCDU3, AUTHCDU4, or AUTHCDU5)
- ANI basis (table ANISCUSP, or tables ANIVAL and UNIPROF)
- agent basis (table CALLATTR)
- office basis (table CAINPARM)

Trigger evaluation

NetworkBuilder checks the *PRI_B-Channel* trigger table (PRIBCHNL) and evaluates the identified digits against the datafilled range associated with the appropriate CAIN group.

Note: The digit type (CLID, address, ADIN) is identified through datafill within the PRIBCHNL table.

Trigger actions

NetworkBuilder call processing supports the following actions for the *PRI_B-Channel* trigger:

- BLOCK – AINF treatment is applied.
- IGNORE – NetworkBuilder continues checking the remaining subscription methods for *PRI_B-Channel*. If datafill does not enable the trigger, call processing continues through the call model.
- QUERY – The switch builds an **`Info_Collected`** query and sends it to the SCP. The SCP analyzes the call and assists the switch with call processing.
- QUERYSCU – In-switch call processing is temporarily suspended. The call enters server mode and becomes a Programmable Service Node (PSN) call and a **`New_Call`** message is sent to the PSN SCU. Refer to *UCS DMS-250 Programmable Service Node (PSN) Application Guide* for more PSN information.

  Note: If an in-switch error occurs before all digits are present the call goes to treatment and does not become a PSN call as it would have under the existing NetworkBuilder functionality.

- LEAVE_TDP – NetworkBuilder call processing exits the ***Info_Collected*** TDP with no further evaluation.

- CONT_NOTRIG – NetworkBuilder call processing exits the ***Info_Collected*** TDP and prevents any further NetworkBuilder interaction for the call.

Trigger criteria

When a query message is built, the ***TriggerCriteriaType*** messaging parameter is populated by the datafill in field TRIGCRIT (table PRIBCHNL). Valid datafill values are: STD, CSP_INTL (channel setup PRI international), CSP_N00, CSP_CLID, CSP_ADDR, and CSP_ADIN.

Error actions

Error actions are not identified for the *PRI_B-Channel* trigger. The switch applies AINF treatment for fatal application errors.

Options

The *PRI_B-Channel* trigger supports the following options:

- BUFFER – to activate digit buffering while the SCP is queried.
- GT – to identify the global title used to identify the SCP handling the query.
- T1OVFLGT – to identify the specific SCP to query on T1 overflow.
- ACGOVFLGT – to identify the global title to use for requerying when a query is blocked by an ACG control
- VERSION – controls the CAIN protocol version for outgoing messages

  Note 1: The value provisioned in the VERSION option overrides the version provisioned in the CAIN_PROTOCOL_VERSION parameter of table CAINPARM on a transaction-by-transaction basis. Additionally, the VERSION option applies to both the outgoing and incoming messages within the same transaction. (Subsequent queries will use the setting provisioned in the CAIN_PROTOCOL_VERSION or provisioned setting of the VERSION option in the trigger table performing the subsequent query.)
- STREAM – to control protocol stream on a per-trigger tuple basis. The value of the STREAM option controls the set of parameters that are sent in NetworkBuilder messages.

Info_Collected TDP-Request

A **`Info_Collected`** TDP-Request (query) message is sent to the SCP in a query package, with a component type of Invoke_Last. The following table defines the parameters and usage requirements for the parameters the **`Info_Collected`** TDP-Request message may contain. Refer to Volume 3,

PRI_B-Channel trigger (continued)

Chapter 8, "Outgoing CAIN message parameters," for detailed descriptions of the request parameters.

Note: Parameters and extension parameters used for AXXESS agents are handled differently. Refer to *UCS DMS-250 CAIN/FlexDial Interactions* for more information.

Info_Collected TDP-Request message parameters for PRI_B-Channel

Parameter	Usage	Definition
UserID	Required	Contains the network identity of the originating agent
BearerCapability	Required	Contains the bearer capability of the call when the message is built
ChargeNumber (Note 6)	Optional	Contains the number that would be used to populate the CDR at this point in call processing. When field BILLTYP (table CALLATTR) is NONE, this parameter is not sent.
Lata (Note 7)	Optional	Contains the call's Local Access and Transport Area (LATA)
TriggerCriteriaType	Optional	Contains channelSetupPRI, CSP_INTL, CSP_N00, CSP_CLID, CSP_ADDR, or CSP_ADIN

Note 1: The `universalAccess` extension parameter is included in the query (a flat 10 digits) when parameter CAIN_PROTOCOL_VERSION is V0. When CAIN_PROTOCOL_VERSION is set to V1 or higher, the parameter contains up to 24 digits including a nature of address and numbering plan similarly to the ***CalledPartyID***.
Note 2: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS07 or higher.
Note 3: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS08 or higher.
Note 4: This parameter is not populated when the CAIN_PROTOCOL_VERSION is set to V3 or higher, and is therefore not sent.
Note 5: The parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS09 or higher.
Note 6: When the CAIN_PROTOCOL_VERSION is set to V4 or higher non-standard charge numbers (CARD, AUTH, ACCT, PIN, and N00) no longer populate the ***ChargeNumber*** parameter and all standard NOAs are supported.
Note 7: The parameter is supported only on DAL, FGD, and AXXESS agencies and the parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS11 or higher.
Note 8: Refer to Volume 3, Chapter 8, "Outgoing CAIN message parameters" for information on the population of this parameter when the CAIN_PROTOCOL_VERSION is set to V5 or higher.

—continued—

PRI_B-Channel trigger (continued)

Info_Collected TDP-Request message parameters for PRI_B-Channel (continued)

Parameter	Usage	Definition
CallingPartyID (Note 8)	Optional	Contains one of the following (listed in order of precedence): CLID when available Valid SNPA value in table TRKGRP Default SNPA value in table CAINPARM
ChargePartyStationType (Note 2)	Optional	Contains the information digits for the call
AccessCode (Note 4)	Optional	Contains the account code, when available
CollectedAddressInfo	Optional	Contains the Called Party Number from the SETUP message or the dialed address
CollectedDigits (Note 4)	Optional	Contains the collected PIN digits, when available
VerticalServiceCode	Optional	Contains feature codes dialed by the subscriber
ACGEncountered	Optional	Contains the ACG control encountered when the ACG control expires and a query is allowed to be sent
ExtensionParameter (Note 2)	Optional	Extension parameters require the CAIN0200 SOC option.

Note 1: The `universalAccess` extension parameter is included in the query (a flat 10 digits) when parameter CAIN_PROTOCOL_VERSION is V0. When CAIN_PROTOCOL_VERSION is set to V1 or higher, the parameter contains up to 24 digits including a nature of address and numbering plan similarly to the ***CalledPartyID***.
Note 2: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS07 or higher.
Note 3: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS08 or higher.
Note 4: This parameter is not populated when the CAIN_PROTOCOL_VERSION is set to V3 or higher, and is therefore not sent.
Note 5: The parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS09 or higher.
Note 6: When the CAIN_PROTOCOL_VERSION is set to V4 or higher non-standard charge numbers (CARD, AUTH, ACCT, PIN, and N00) no longer populate the ***ChargeNumber*** parameter and all standard NOAs are supported.
Note 7: The parameter is supported only on DAL, FGD, and AXXESS agencies and the parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS11 or higher.
Note 8: Refer to Volume 3, Chapter 8, “Outgoing CAIN message parameters” for information on the population of this parameter when the CAIN_PROTOCOL_VERSION is set to V5 or higher.

—continued—

PRI_B-Channel trigger (continued)

Info_Collected TDP-Request message parameters for PRI_B-Channel (continued)

Parameter	Usage	Definition
universalAccess (Note 1)	Optional	Contains the universal access number dialed by the caller to obtain switch dial tone. Since this number is not the actual address for the call, it is not transported in ***CollectedAddressInfo*** or ***CalledPartyID***.
cainGroup	Optional	Contains the group number (field GRPNUM, table CAINGRP) for the CAIN group associated with the trigger when field EXTPARM in table CAINGRP contains CAINGRP.
adin	Optional	Contains the authorization code database index (field ADIN in table CALLATTR for PRI agents) associated with the originating agency when field EXTPARM in table CAINGRP contains ADIN. ***Note:*** This does not apply to the authcode collected at *O_Feature_Requested*.
origTrunkInfo	Optional	Contains the originating trunk group number, trunk type, and trunk member number when field EXTPARM in table CAINGRP contains ORGTINFO.

Note 1: The universalAccess extension parameter is included in the query (a flat 10 digits) when parameter CAIN_PROTOCOL_VERSION is V0. When CAIN_PROTOCOL_VERSION is set to V1 or higher, the parameter contains up to 24 digits including a nature of address and numbering plan similarly to the ***CalledPartyID***.
Note 2: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS07 or higher.
Note 3: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS08 or higher.
Note 4: This parameter is not populated when the CAIN_PROTOCOL_VERSION is set to V3 or higher, and is therefore not sent.
Note 5: The parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS09 or higher.
Note 6: When the CAIN_PROTOCOL_VERSION is set to V4 or higher non-standard charge numbers (CARD, AUTH, ACCT, PIN, and N00) no longer populate the ***ChargeNumber*** parameter and all standard NOAs are supported.
Note 7: The parameter is supported only on DAL, FGD, and AXXESS agencies and the parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS11 or higher.
Note 8: Refer to Volume 3, Chapter 8, “Outgoing CAIN message parameters” for information on the population of this parameter when the CAIN_PROTOCOL_VERSION is set to V5 or higher.

—continued—

PRI_B-Channel trigger (continued)

Info_Collected TDP-Request message parameters for PRI_B-Channel (continued)

Parameter	Usage	Definition
netinfo (Note 2)	Optional	Contains external network ID, network customer group ID, and network class of service.
t1Overflow	Optional	Indicates to the SCP that an overflow occurred on the query to the initial SCP.
lnpReceived (Note 2)	Optional	Presence of this parameter indicates LNP information was received from a previous switch.
subscriptionInfo (Note 3)	Optional	Contains the digit type triggered on for the query and the method to which the triggering CAIN group subscribed.
jurisdictionInfo (Note 3)	Optional	Contains the originating switch's LRN.
switchID (Note 5)	Optional	Contains the switch ID
accountCode (Note 5)	Optional	Contains the account code when available
pinDigits (Note 5)	Optional	Contains the collected pin digits when available
billingNumber (Note 5)	Optional	Contains the non-standard charge number
acgRequery	Optional	When present, indicates to the SCP that the current query is a re-query message resulting from the original query being blocked by an ACG control

Note 1: The universalAccess extension parameter is included in the query (a flat 10 digits) when parameter CAIN_PROTOCOL_VERSION is V0. When CAIN_PROTOCOL_VERSION is set to V1 or higher, the parameter contains up to 24 digits including a nature of address and numbering plan similarly to the ***CalledPartyID***.
Note 2: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS07 or higher.
Note 3: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS08 or higher.
Note 4: This parameter is not populated when the CAIN_PROTOCOL_VERSION is set to V3 or higher, and is therefore not sent.
Note 5: The parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS09 or higher.
Note 6: When the CAIN_PROTOCOL_VERSION is set to V4 or higher non-standard charge numbers (CARD, AUTH, ACCT, PIN, and N00) no longer populate the ***ChargeNumber*** parameter and all standard NOAs are supported.
Note 7: The parameter is supported only on DAL, FGD, and AXXESS agencies and the parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS11 or higher.
Note 8: Refer to Volume 3, Chapter 8, "Outgoing CAIN message parameters" for information on the population of this parameter when the CAIN_PROTOCOL_VERSION is set to V5 or higher.

—end—

PRI_B-Channel trigger (continued)

Once the request message is built, NetworkBuilder formats the message and sends it to the message encoder. Once encoded, the message is sent to the SCP.

SCP response processing

The following response messages are supported:

- **`Analyze_Route`**
- **`Send_To_Resource`**
- **`Disconnect`**

  Note 1: EDPs may be armed through the **`Request_Report_BCM_Event`** component with the **`Analyze_Route`** message.

  Note 2: A **`Termination_Notification`** may be requested through the **`Send_Notification`** component with the SCP response message.

  Note 3: An **`ACG`** message may be sent with the SCP response message.

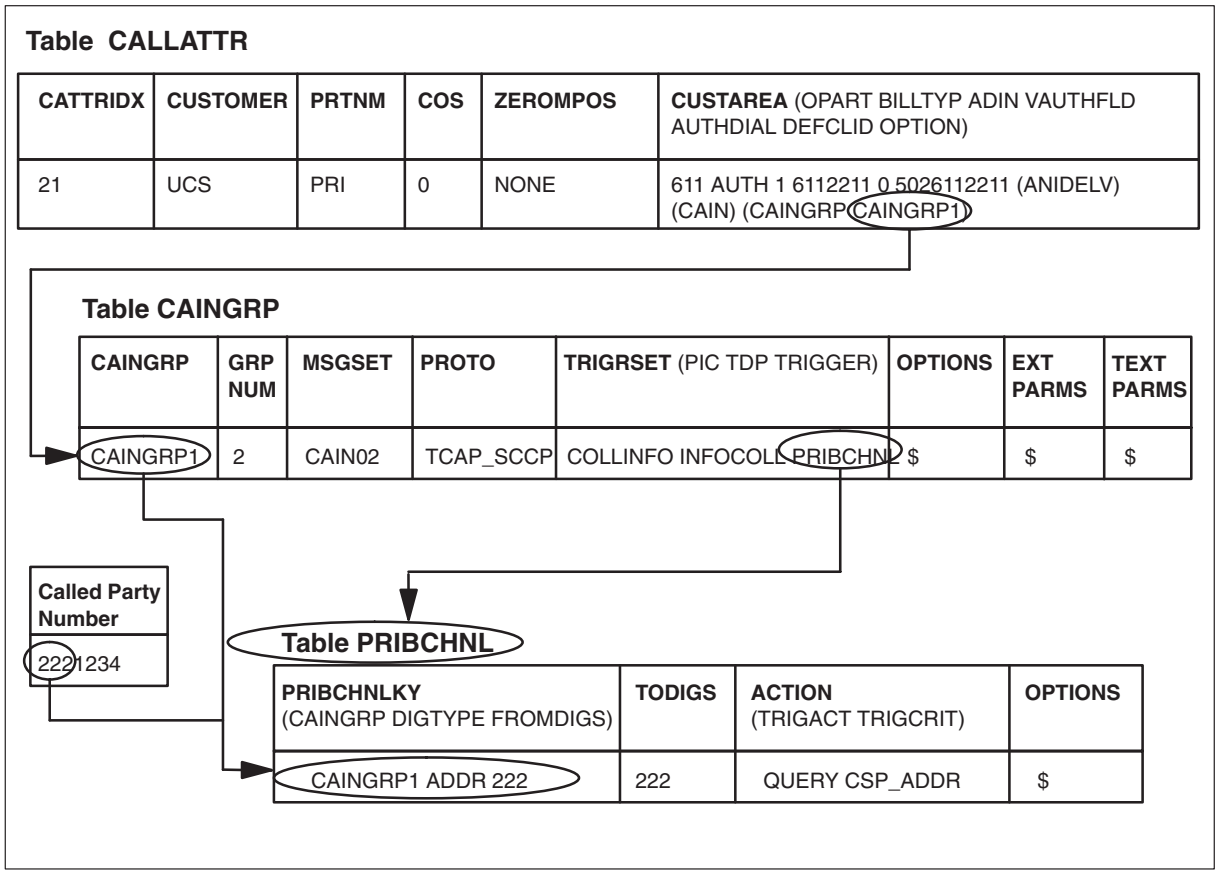
Refer to Volume 3, Chapter 10, “Incoming CAIN messages,” for specific information regarding response messages; refer to Volume 3, Chapter 12, “Incoming CAIN message parameters,” for detailed descriptions of the response parameters; refer to Volume 3, Chapter 3, “Event processing,” for EDP specific message information.

Datafill

The following figure shows how a subscription table interacts with the *PRI_B-Channel* trigger table (PRIBCHNL).

PRI_B-Channel trigger (continued)

Subscription-PRIBCHNL table interaction



Provisioning the PRI_B-Channel trigger

At the CI prompt

1 Provision the originating agent as CAIN-capable

2 Subscribe to a CAIN group (table STDPRTCT, authcode tables, ANI tables, CALLATTR, or CAINPARM)

3 Datafill CAIN group trigger subscription to the (COLLINFO INFOCOLL PRIBCHNL) trigger set (table CAINGRP).

4 Enter table PRIBCHNL.

5 Define the trigger criteria for a CAIN group by using the following format:

>ADD pribchnlky todigs action
where

pribchnlky is comprised of three subfields: CAINGRP, DIGTYPE, FROMDIGS, where

PRI_B-Channel trigger (end)

CAINGRP is the CAIN group that enables the trigger (from table CAINGRP).

DIGTYPE is the digit type being analyzed (CLID, ADDR, or ADIN).

FROMDIGS is the first number used to define the range of digits (0 to 9, *, #).

todigs is the second number used to define the range of digits (0 to 9, *, #).

action is comprised of two subfields: TRIGACT and TRIGCRIT, where

TRIGACT is the trigger action taken when the address is within the FROMDIGS-TODIGS range (BLOCK, IGNORE, QUERY, QUERYSCU, LEAVE_TDP, CONT_NOTRIG).

TRIGCRIT is a refinement only available when ACTION is QUERY. Enter the trigger criteria sent to the SCP to identify the digits (STD, CSP_INTL, CSP_N00, CSP_CLID, CSP_ADDR, or CSP_ADIN).

Note: The TRIGCRIT refinement value is dependent on the DIGTYPE (with the exception of the STD trigger criteria). When DIGTYPE is CLID, TRIGCRIT must be CSP_CLID or STD; when DIGTYPE is ADDR, TRIGCRIT can be CSP_ADDR, CSP_INTL, CSP_N00, or STD; when DIGTYPE is ADIN, TRIGCRIT must be CSP_ADIN or STD.

6 Datafill the OPTIONS, where

options is only allowed when ACTION is QUERY. Enter up to 6 options: BUFFER, GT, T1OVFLGT, ACGOVFLGT, VERSION, or STREAM. When the ACTION is not QUERY enter $.

Sample entry: **>ADD caingrp1 addr 222 222 query csp_addr t1ovflgt cain_addr $**

Sample entry: **>ADD caingrp1 addr 222 222 query std t1ovflgt cain_addr $**

PRI_B-Channel trigger criteria is defined.

Associated OMs

CAINMSGS, CAINTRIG, CAINAGOM

Analyze_Information PIC

Call processing enters **Analyze_Information** when either an **`Analyze_Route`** message is returned from the SCP or once the dialing plan is completed and the switch has:

- collected data from the initial address message (IAM) of the originating SS7 FGD, Inter-IMT, Global-IMT, or AXXESS trunks

  Note: Refer to *UCS DMS-250 CAIN/FlexDial Interactions* for more information on AXXESS agents.

- collected digits from a PTS DAL, FGD, AXXESS, or a PRI agency
- collected data from the SETUP message of the originating PRI agency
- performed IN/1 authorization code screening

ATTENTION

NetworkBuilder call processing always encounters ***Info_Analyzed*** whether or not a route index is identified through normal translations (where the called party number is provisioned in a translation table such as HNPACONT). By not identifying a route index, the switch allows the SCP to control routing. When the switch does not identify a route index, the SCP must provide a route index in an **`Analyze_Route`** message. If the SCP responds with a **`Continue`** message, treatment is set to indicate a translation failure.

O_Abandon EDP

Call processing encounters the ***O_Abandon*** EDP during the **Collect_Information**, **Analyze_Information**, **Select_Route**, **Send_Call**, and **O_Alerting** PICs when two requirements are met:

- EDPs are active
- the calling party disconnects before the called party answers

The *O_Abandon* event is detected. For information on the *O_Abandon* event, refer to Chapter 4, “Collect_Information PIC

Info_Analyzed TDP

Call processing encounters ***Info_Analyzed*** when

- called party identifier is available
- call type is available
- IN/1 screening of N00, account codes, authorization codes, and speed dial numbers (as required) is performed
- possible route selection are calculated from the collected information.

Instead of immediately seizing an outgoing agent, NetworkBuilder directs the switch to check for the following:

- call subscription to a CAIN group through table STDPRTCT, authcode tables, ANI tables, TRKGRP, CALLATTR, or CAINPARM

  Note: CAIN group subscription for AXXESS agents is handled differently. Refer to *UCS DMS-250 CAIN/FlexDial Interactions*.

- enabled triggers (Triggers are enabled through table CAINGRP.)

  Note: NetworkBuilder software supports the *Specific_Feature_Code*, *Customized_Dialing_Plan*, *Specific_Digit_String*, and *Office_Code* triggers at ***Info_Analyzed***.

When a partial dial or translation failure (not screening failure) situation occurs, NetworkBuilder checks the trigger criteria for the call.

- If trigger criteria is not met, or the action is LEAVE_TDP or CONT_NOTRIG, the switch applies PDIL or VACT treatment to the call.
- If trigger criteria is met with an action of QUERY, a query is sent to the SCP. The query identifies the PDIL or VACT treatment (if set by the switch) in the `treatment` extension parameter.

  Note: Extension parameters require the CAIN0200 SOC option.

  — If the SCP returns a **`Disconnect`** with the `treatment` extension parameter, the switch applies the SCP-supplied treatment and disconnects the call.

  Note: The SCP can echo the original treatment set by the switch or supply a new treatment. When the original treatment is used, returning a **`Continue`** works in the same way as a **`Disconnect`**.

  — If the SCP returns a **`Disconnect`** without the `treatment` extension parameter, the switch applies AIND treatment and disconnects the call.

- If trigger criteria is met with an action of BLOCK, the switch applies AINF treatment.

When the INFOANALYZED_FOR_RLT parameter in table CAINPARM is set to Y, ***Info_Analyzed*** triggers can be evaluated on the second leg of an SS7 Inter-IMT RLT call.

Subscription

The subscription method is determined in the following order:

1 SCP-returned CAIN groups, enabling the *Specific_Feature_Code* trigger

2 SCP-returned CAIN groups, enabling the *Customized_Dialing_Plan* trigger

3 SCP-returned CAIN groups, enabling the *Specific_Digit_String* trigger

4 SCP-returned CAIN groups, enabling the *Office_Code* trigger

5 Address subscription, enabling the *Specific_Feature_Code* trigger

6 Address subscription, enabling the *Customized_Dialing_Plan* trigger

7 Address subscription, enabling the *Specific_Digit_String* trigger

8 Address subscription, enabling the *Office_Code* trigger

9 Authorization code subscription, enabling the *Specific_Feature_Code* trigger

10 Authorization code subscription, enabling the *Customized_Dialing_Plan* trigger

11 Authorization code subscription, enabling the *Specific_Digit_String* trigger

12 Authorization code subscription, enabling the *Office_Code* trigger

13 ANI subscription, enabling the *Specific_Feature_Code* trigger

14 ANI subscription, enabling the *Customized_Dialing_Plan* trigger

15 ANI subscription, enabling the *Specific_Digit_String* trigger

16 ANI subscription, enabling the *Office_Code* trigger

17 Agent subscription, enabling the *Specific_Feature_Code* trigger

18 Agent subscription, enabling the *Customized_Dialing_Plan* trigger

19 Agent subscription, enabling the *Specific_Digit_String* trigger

20 Agent subscription, enabling the *Office_Code* trigger

21 Office subscription, enabling the *Specific_Feature_Code* trigger

22 Office subscription, enabling the *Customized_Dialing_Plan* trigger

23 Office subscription, enabling the *Specific_Digit_String* trigger

24 Office subscription, enabling the *Office_Code* trigger

Once a subscription method is identified and a *Specific_Feature_Code*, *Customized_Dialing_Plan*, *Specific_Digit_String*, or *Office_Code* trigger is enabled, NetworkBuilder performs the following steps:

1 Evaluates trigger criteria (datafilled in the trigger table) against gathered data.

2 Performs the datafilled trigger action.

   — If the trigger action is QUERY, an **`Info_Analyzed`** query message is sent to the SCP.

   — If the trigger action is IGNORE, subscription method determination continues at the point where the trigger was evaluated.

   — If the trigger action is BLOCK, subscription method determination stops, the call is blocked by call processing and AINF treatment is applied.

   — If the trigger action is LEAVE_TDP, call processing exits the current TDP with no further evaluation.

   — If the trigger action is CONT_NOTRIG, call processing exits the current TDP and prevents any further NetworkBuilder interaction for the current call.

3 If the SCP responds with a **`Continue`** message, subscription method determination continues at the point where the trigger was evaluated.

Effective with UCS17, the ***Info_Analyzed*** query message may include the AIN ExtensionParameter parameter, which was previously only allowed in the ***Info_Collected*** query message.

Specific_Feature_Code trigger

ATTENTION

The *Specific_Feature_Code* trigger requires the CAIN0500 SOC option. Refer to Volume 5, Chapter 5, "NetworkBuilder SOC functionality," for more information.

Uses

When you provision NetworkBuilder services on the switch, the SCP can make major call processing decisions. You define the services offered on the SCP. Some examples of services that can be developed on the SCP for the *Specific_Feature_Code* trigger are:

- VPN Services
  - — OFFNET Overflow
  - — Forced ONNET
  - — Alternate Billing Numbers
  - — Origination/Termination Screening
  - — Customized Announcements
  - — Black Box Screening
- CIC Routing and Branding

Supported originating agencies

The *Specific_Feature_Code* trigger supports the following originating agencies:

- DAL
- FGD
- PRI
- SS7 Inter-IMT
- SS7 Global-IMT
- AXXESS

  Note: Refer to *UCS DMS-250 CAIN/FlexDial Interactions* for more information on support for AXXESS agents.

Subscribing to the Specific_Feature_Code trigger

Subscription to the *Specific_Feature_Code* trigger is available on a

- SCP-returned basis

Specific_Feature_Code trigger (continued)

- address basis (table STDPRTCT, subtable STDPRT)
- authorization code basis (tables AUTHCODU, AUTHCDU2, AUTHCDU3, AUTHCDU4, or AUTHCDU5)
- ANI basis (table ANISCUSP, or tables ANIVAL and UNIPROF)
- agent basis (table TRKGRP or CALLATTR)
- office basis (table CAINPARM)

Note 1: An SCP-returned CAIN subscription group is received in **`Analyze_Route`** messages. When a reorigination occurs after an **`Analyze_Route`** is received, the SCP-returned CAIN group is available for any reoriginated call.

Note 2: CAIN group subscription for AXXESS agents is handled differently. Refer to *UCS DMS-250 CAIN/FlexDial Interactions*.

Trigger evaluation

NetworkBuilder checks the *Specific_Feature_Code* trigger table (SPECFEAT) and evaluates the call's service code (1 to 24 digit address) against the datafilled range associated with the appropriate CAIN group.

Note 1: The digit type (XLAADDR and ADDR) is identified through datafill within the SPECFEAT table.

Note 2: For evaluation due to an **`Analyze_Route`** message, only the XLAADDR digit type is considered.

Trigger actions

NetworkBuilder call processing supports the following actions for the *Specific_Feature_Code* trigger:

- BLOCK – Prevents the call from proceeding and applies AINF treatment.
- IGNORE – NetworkBuilder continues checking the remaining subscription methods for *Specific_Feature_Code*, *Customized_Dialing_Plan*, *Specific_Digit_String*, and *Office_Code*. If datafill does not enable any trigger, call processing continues through the call model.
- QUERY – The switch builds an **`Info_Analyzed`** query and sends it to the SCP. The SCP analyzes the call and assists the switch with call processing.
- LEAVE_TDP – NetworkBuilder call processing exits the ***Info_Analyzed*** TDP with no further evaluation.

- CONT_NOTRIG – NetworkBuilder call processing exits the ***Info_Analyzed*** TDP and prevents any further NetworkBuilder interaction for the call. This prevention is reset upon reorigination.

Error actions

When a fatal application occurs during an SCP query, one of the following error actions is performed (as datafilled):

- TREAT – AINF treatment is applied.
- ROUTE – The switch discards any SCP-provided data. NetworkBuilder continues checking the remaining subscription methods for *Specific_Feature_Code*, *Customized_Dialing_Plan*, *Specific_Digit_String*, and *Office_Code*. If datafill does not enable any trigger, call processing continues through the call model.

Options

The *Specific_Feature_Code* trigger supports the following options:

- BUFFER – to activate digit buffering while the SCP is queried.
- GT – to identify the global title used to identify the SCP handling the query.
- T1OVFLGT – to identify the specific SCP to query on T1 overflow.
- ACGOVFLGT – to identify the global title to use for requerying when a query is blocked by an ACG control
- VERSION – controls the CAIN protocol version for outgoing messages

  Note: The value provisioned in the VERSION option overrides the version provisioned in the CAIN_PROTOCOL_VERSION parameter of table CAINPARM on a transaction-by-transaction basis. Additionally, the VERSION option applies to both the outgoing and incoming messages within the same transaction. (Subsequent queries will use the setting provisioned in the CAIN_PROTOCOL_VERSION or provisioned setting of the VERSION option in the trigger table performing the subsequent query.)

- STREAM – to control protocol stream on a per-trigger tuple basis. The value of the STREAM option controls the set of parameters that are sent in NetworkBuilder messages.

Specific_Feature_Code trigger (continued)

The following figure shows how a call progresses through the call model, encounters ***Info_Analyzed*** and queries the SCP.

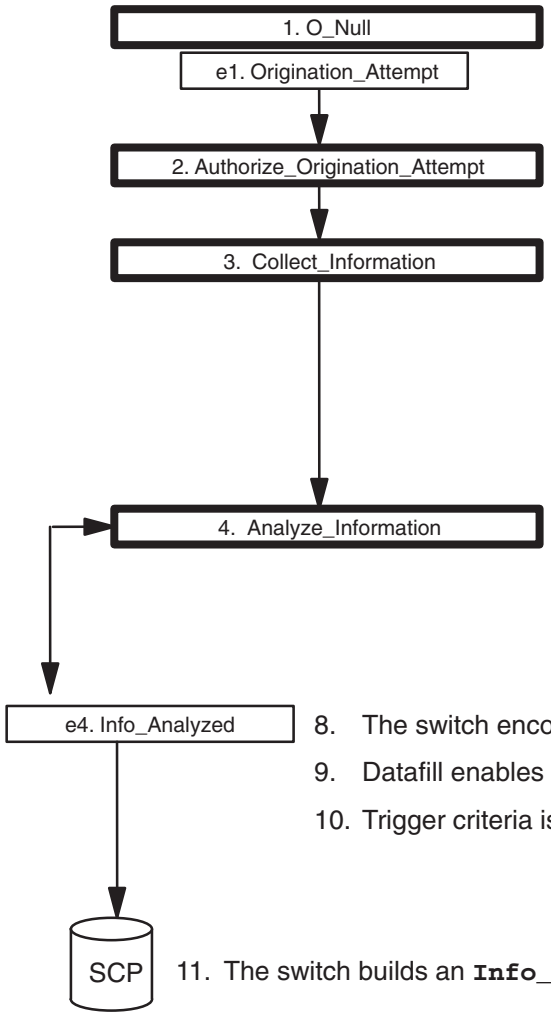
UA Auth call

FGD origination
Datafill:

Signaling: MF at origination
DTMF for digit collection
Dialing plan: KP + II + ANI + ST
KP + ADDR + ST
ADDR: (214) 611-3311

Table SPECFEAT

SFCKEY	TODIGS	ACTION	OPTION
CAINGRP1 ADDR 214	214	QUERY TREAT	$



1. The switch detects seizure and enters **O_Null**.
2. Call processing enters **Authorize_Origination_Attempt**.
3. Call processing enters **Collect_Information**.
4. The switch collects KP + II + ANI + ST.
5. The switch screens II and ANI digits.
6. The switch collects KP + ADDR + ST (ADDR is a UA number).
7. The switch performs digit screening (in-switch or off-board).
8. The switch encounters ***Info_Analyzed***.
9. Datafill enables *Specific_Feature_Code* trigger.
10. Trigger criteria is met.
11. The switch builds an **`Info_Analyzed`** query and sends it to the SCP.

Info_Analyzed TDP-Request

An **Info_Analyzed** TDP-Request (query) is sent to the SCP in a query package, with a component type of Invoke_Last. The following table defines the parameters and usage requirements for the parameters the **Info_Analyzed** TDP-Request may contain. Refer to Volume 3, Chapter 8, "Outgoing CAIN message parameters," for detailed descriptions of the request parameters.

Note: Parameters and extension parameters for AXXESS agents are handled differently. Refer to *UCS DMS-250 CAIN/FlexDial Interactions* for more information.

Info_Analyzed TDP-Request message parameters for Specific_Feature_Code

Parameter	Usage	Definition
UserID	Required	Contains the network identity of the originating agent
BearerCapability	Required	Contains the bearer capability of the call when the message is built
CalledPartyID	Optional	Contains the translated address
Lata (Note 8)	Optional	Contains the call's Local Access and Transport Area (LATA)
TriggerCriteriaType	Optional	Contains specificFeatureCode

Note 1: The `universalAccess` extension parameter is included in the query (a flat 10 digits) when parameter CAIN_PROTOCOL_VERSION is V0. When CAIN_PROTOCOL_VERSION is set to V1 or higher, the parameter contains up to 24 digits including a nature of address and numbering plan similarly to the ***CalledPartyID***.
Note 2: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS06 or higher.
Note 3: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS07 or higher.
Note 4: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS08 or higher.
Note 5: This parameter is not populated when the CAIN_PROTOCOL_VERSION is set to V3 or higher, and is therefore not sent.
Note 6: The parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS09 or higher.
Note 7: When the CAIN_PROTOCOL_VERSION is set to V4 or higher non-standard charge numbers (CARD, AUTH, ACCT, PIN, and N00) no longer populate the ***ChargeNumber*** parameter and all standard NOAs are supported.
Note 8: The parameter is supported only on DAL, FGD, and AXXESS agencies and the parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS11 or higher.
Note 9: Refer to Volume 3, Chapter 8, "Outgoing CAIN message parameters" for information on the population of this parameter when the CAIN_PROTOCOL_VERSION is set to V5 or higher.

—continued—

Specific_Feature_Code trigger (continued)

Info_Analyzed TDP-Request message parameters for Specific_Feature_Code (continued)

Parameter	Usage	Definition
ChargeNumber (Note 7)	Optional	Contains the billing number that would be used to populate the CDR at this point in call processing
CallingPartyID (Note 9)	Optional	Contains one of the following (listed in order of precedence): For SS7 FGD, SS7 Inter-IMT, SS7-Global IMT, and AXXESS calls: Calling_Party_Address from ISUP message, when available For FGD and AXXESS calls: valid ANI (information digits are not passed in this parameter) ***Note:*** Refer to *UCS DMS-250 CAIN/FlexDial Interactions* for more information on AXXESS agents. For PRI calls: CLID when available Valid SNPA value from table TRKGRP Default SNPA value from table CAINPARM
ChargePartyStationType (Note 3)	Optional	Contains the information digits for the call

Note 1: The `universalAccess` extension parameter is included in the query (a flat 10 digits) when parameter CAIN_PROTOCOL_VERSION is V0. When CAIN_PROTOCOL_VERSION is set to V1 or higher, the parameter contains up to 24 digits including a nature of address and numbering plan similarly to the ***CalledPartyID***.
Note 2: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS06 or higher.
Note 3: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS07 or higher.
Note 4: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS08 or higher.
Note 5: This parameter is not populated when the CAIN_PROTOCOL_VERSION is set to V3 or higher, and is therefore not sent.
Note 6: The parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS09 or higher.
Note 7: When the CAIN_PROTOCOL_VERSION is set to V4 or higher non-standard charge numbers (CARD, AUTH, ACCT, PIN, and N00) no longer populate the ***ChargeNumber*** parameter and all standard NOAs are supported.
Note 8: The parameter is supported only on DAL, FGD, and AXXESS agencies and the parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS11 or higher.
Note 9: Refer to Volume 3, Chapter 8, “Outgoing CAIN message parameters” for information on the population of this parameter when the CAIN_PROTOCOL_VERSION is set to V5 or higher.

—continued—

Specific_Feature_Code trigger (continued)

Info_Analyzed TDP-Request message parameters for Specific_Feature_Code (continued)

Parameter	Usage	Definition
Carrier	Optional	Contains the dialed CIC or the CIC value (with an indication of: Selected CIC presubscribed and not input by calling party) from table TRKGRP.
AccessCode(Note 5)	Optional	Contains the account code, when available
CollectedAddressInfo	Optional	Contains the address collected from the incoming agent (from the IAM, subscriber dialing, or datafilled hotline digits)
CollectedDigits(Note 5)	Optional	Contains the collected PIN digits, when available
VerticalServiceCode (Note 2)	Optional	Contains feature codes dialed by the subscriber
ACGEncountered	Optional	Contains the ACG control encountered when the ACG control expires and a query is allowed to be sent
ExtensionParameter (Note 3)	Optional	Extension parameters require the CAIN0200 SOC option.
universalAccess (Note 1)	Optional	Contains the universal access number dialed by the caller to obtain switch dial tone.

Note 1: The universalAccess extension parameter is included in the query (a flat 10 digits) when parameter CAIN_PROTOCOL_VERSION is V0. When CAIN_PROTOCOL_VERSION is set to V1 or higher, the parameter contains up to 24 digits including a nature of address and numbering plan similarly to the ***CalledPartyID***.
Note 2: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS06 or higher.
Note 3: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS07 or higher.
Note 4: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS08 or higher.
Note 5: This parameter is not populated when the CAIN_PROTOCOL_VERSION is set to V3 or higher, and is therefore not sent.
Note 6: The parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS09 or higher.
Note 7: When the CAIN_PROTOCOL_VERSION is set to V4 or higher non-standard charge numbers (CARD, AUTH, ACCT, PIN, and N00) no longer populate the ***ChargeNumber*** parameter and all standard NOAs are supported.
Note 8: The parameter is supported only on DAL, FGD, and AXXESS agencies and the parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS11 or higher.
Note 9: Refer to Volume 3, Chapter 8, “Outgoing CAIN message parameters” for information on the population of this parameter when the CAIN_PROTOCOL_VERSION is set to V5 or higher.

—continued—

Specific_Feature_Code trigger (continued)

Info_Analyzed TDP-Request message parameters for Specific_Feature_Code (continued)

Parameter	Usage	Definition
`cainGroup`	Optional	Contains the group number (field GRPNUM, table CAINGRP) for the CAIN group associated with the trigger when field EXTPARM in table CAINGRP contains CAINGRP.
`adin`	Optional	Contains the authorization code database index (field ADIN, table TRKGRP for DAL and FGD agencies; field ADIN, table CALLATTR for PRI agencies) associated with the originating agency when field EXTPARM in table CAINGRP contains ADIN. ***Note:*** The `adin` extension parameter is not supported for AXXESS agents.
`origTrunkInfo`	Optional	Contains the originating trunk group number, trunk type, and trunk member number when field EXTPARM in table CAINGRP contains ORGTINFO.
`treatment` (Note 2)	Optional	Contains the treatment set by regular call processing before the query was sent
`reorigCall` (Note 2)	Optional	Presence of this parameter indicates the call in progress is a result of reorigination

Note 1: The `universalAccess` extension parameter is included in the query (a flat 10 digits) when parameter CAIN_PROTOCOL_VERSION is V0. When CAIN_PROTOCOL_VERSION is set to V1 or higher, the parameter contains up to 24 digits including a nature of address and numbering plan similarly to the ***`CalledPartyID`***.
Note 2: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS06 or higher.
Note 3: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS07 or higher.
Note 4: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS08 or higher.
Note 5: This parameter is not populated when the CAIN_PROTOCOL_VERSION is set to V3 or higher, and is therefore not sent.
Note 6: The parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS09 or higher.
Note 7: When the CAIN_PROTOCOL_VERSION is set to V4 or higher non-standard charge numbers (CARD, AUTH, ACCT, PIN, and N00) no longer populate the ***`ChargeNumber`*** parameter and all standard NOAs are supported.
Note 8: The parameter is supported only on DAL, FGD, and AXXESS agencies and the parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS11 or higher.
Note 9: Refer to Volume 3, Chapter 8, “Outgoing CAIN message parameters” for information on the population of this parameter when the CAIN_PROTOCOL_VERSION is set to V5 or higher.

—continued—

Specific_Feature_Code trigger (continued)

Info_Analyzed TDP-Request message parameters for Specific_Feature_Code (continued)

Parameter	Usage	Definition
univIdx	Optional	Used by SS7 Global-IMT agents only. Contains the universal translations scheme to be used for the call.
netinfo (Note 3)	Optional	Contains external network ID, network customer group ID, and network class of service.
t1Overflow	Optional	Indicates to the SCP that an overflow occurred on the query to the initial SCP.
lnpReceived (Note 3)	Optional	Presence of this parameter indicates LNP information was received from a previous switch.
subscriptionInfo (Note 4)	Optional	Contains the digit type triggered on for the query and the method to which the triggering CAIN group subscribed.
switchID (Note 6)	Optional	Contains the switch ID
accountCode (Note 6)	Optional	Contains the account code when available
pinDigits (Note 6)	Optional	Contains the collected pin digits when available

Note 1: The universalAccess extension parameter is included in the query (a flat 10 digits) when parameter CAIN_PROTOCOL_VERSION is V0. When CAIN_PROTOCOL_VERSION is set to V1 or higher, the parameter contains up to 24 digits including a nature of address and numbering plan similarly to the ***CalledPartyID***.
Note 2: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS06 or higher.
Note 3: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS07 or higher.
Note 4: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS08 or higher.
Note 5: This parameter is not populated when the CAIN_PROTOCOL_VERSION is set to V3 or higher, and is therefore not sent.
Note 6: The parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS09 or higher.
Note 7: When the CAIN_PROTOCOL_VERSION is set to V4 or higher non-standard charge numbers (CARD, AUTH, ACCT, PIN, and N00) no longer populate the ***ChargeNumber*** parameter and all standard NOAs are supported.
Note 8: The parameter is supported only on DAL, FGD, and AXXESS agencies and the parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS11 or higher.
Note 9: Refer to Volume 3, Chapter 8, “Outgoing CAIN message parameters” for information on the population of this parameter when the CAIN_PROTOCOL_VERSION is set to V5 or higher.

—continued—

Specific_Feature_Code trigger (continued)

Info_Analyzed TDP-Request message parameters for Specific_Feature_Code (continued)

Parameter	Usage	Definition
`billingNumber` (Note 6)	Optional	Contains the non-standard charge number
`acgRequery`	Optional	When present, indicates to the SCP that the current query is a re-query message resulting from the original query being blocked by an ACG control
JurisdictionInfo (Note 4)	Optional	Contains the originating switch's location routing number (LRN).

Note 1: The `universalAccess` extension parameter is included in the query (a flat 10 digits) when parameter CAIN_PROTOCOL_VERSION is V0. When CAIN_PROTOCOL_VERSION is set to V1 or higher, the parameter contains up to 24 digits including a nature of address and numbering plan similarly to the ***CalledPartyID***.
Note 2: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS06 or higher.
Note 3: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS07 or higher.
Note 4: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS08 or higher.
Note 5: This parameter is not populated when the CAIN_PROTOCOL_VERSION is set to V3 or higher, and is therefore not sent.
Note 6: The parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS09 or higher.
Note 7: When the CAIN_PROTOCOL_VERSION is set to V4 or higher non-standard charge numbers (CARD, AUTH, ACCT, PIN, and N00) no longer populate the ***ChargeNumber*** parameter and all standard NOAs are supported.
Note 8: The parameter is supported only on DAL, FGD, and AXXESS agencies and the parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS11 or higher.
Note 9: Refer to Volume 3, Chapter 8, "Outgoing CAIN message parameters" for information on the population of this parameter when the CAIN_PROTOCOL_VERSION is set to V5 or higher.

—end—

Once the request message is built, NetworkBuilder formats the message and sends it to the message encoder. Once encoded, the message is sent to the SCP.

SCP response processing

The following response messages are supported:

- **`Analyze_Route`**
- **`Send_To_Resource`**
- **`Continue`** – NetworkBuilder continues checking the remaining subscription methods for *Specific_Feature_Code*, *Customized_Dialing_Plan*, *Specific_Digit_String*, and *Office_Code*. If datafill does not enable any trigger, call processing continues through the call model.

- **Disconnect**

Note 1: EDPs may be armed through the **Request_Report_BCM_Event** component with **Analyze_Route** and **Continue** messages.

Note 2: A **Termination_Notification** may be requested through the **Send_Notification** component with the SCP response message.

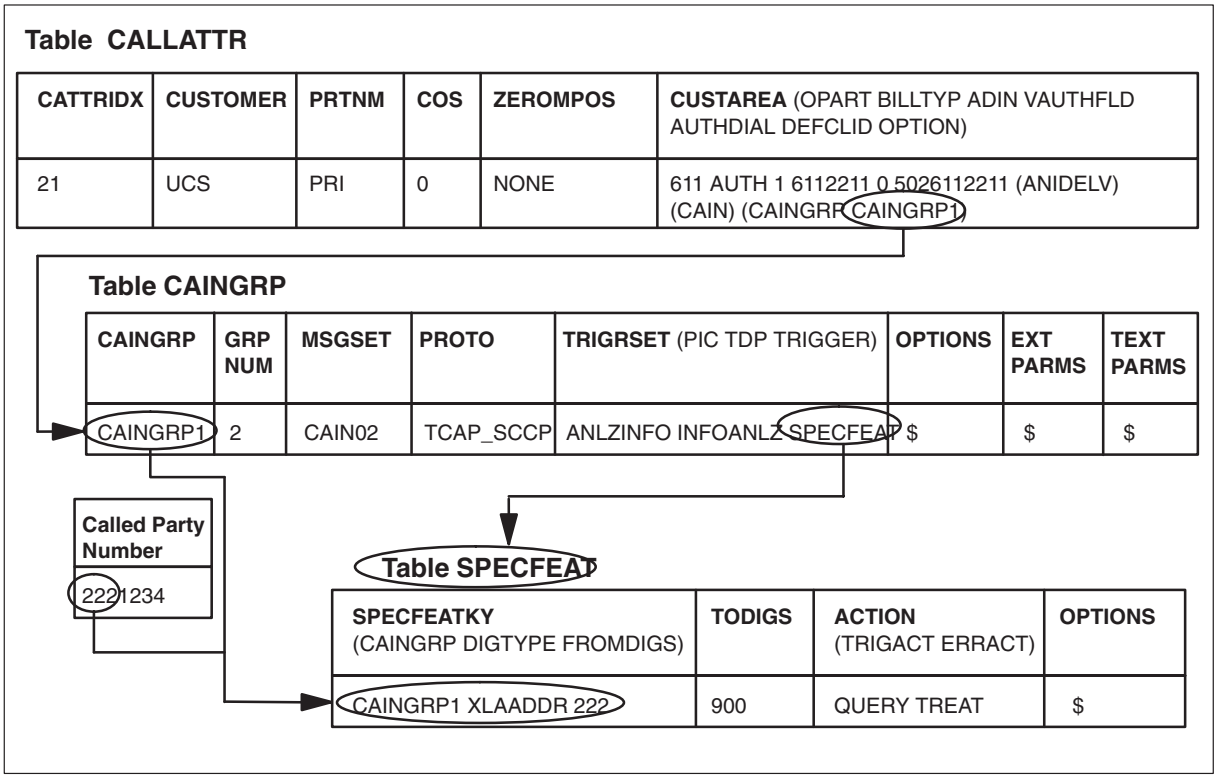
Note 3: An **ACG** message may be sent with the SCP response message.

Refer to Volume 3, Chapter 10, "Incoming CAIN messages," for specific information regarding response messages; refer to Volume 3, Chapter 12, "Incoming CAIN message parameters," for detailed descriptions of the response parameters; refer to Volume 3, Chapter 3, "Event processing," for EDP specific message information.

Datafill

The following figure shows how a subscription table interacts with the *Specific_Feature_Code* trigger table (SPECFEAT).

Subscription-SPECFEAT table interaction



Specific_Feature_Code trigger (continued)

Provisioning the Specific_Feature_Code trigger

At the CI prompt

1 Provision the originating agent as CAIN-capable.

2 Subscribe to a CAIN group (table STDPRTCT, authcode tables, ANI tables, TRKGRP, CALLATTR, or CAINPARM)

Note: CAIN group subscription is handled differently for AXXESS agents. Refer to *UCS DMS-250 CAIN/FlexDial Interactions*.

3 Datafill CAIN group trigger subscription to the (ANLZINFO INFOANLZ SPECFEAT) trigger set (table CAINGRP).

4 Enter table SPECFEAT.

5 Define the trigger criteria for a CAIN group by using the following format:

>ADD specfeatky todigs action options
where

specfeatky is comprised of three subfields: CAINGRP, DIGTYPE, and FROMDIGS, where
- CAINGRP is the CAIN group requiring *Specific_Feature_Code* trigger criteria (from table CAINGRP).
- DIGTYPE is the digit type being analyzed (XLAADDR, ADDR).
- FROMDIGS is the first number used to define the range of the collected address.

todigs is the second number used to define the range of the collected address.

action is comprised of one subfield: TRIGACT, where
- TRIGACT is the trigger action taken when the address is within the FROMDIGS-TODIGS range (BLOCK, IGNORE, QUERY, LEAVE_TDP, CONT_NOTRIG).

If you datafill TRIGACT as:	Go to:
BLOCK	step 7
IGNORE	step 7
QUERY	step 6
LEAVE_TDP	step 7
CONT_NOTRIG	step 7

6 Datafill the ERRACT refinement, where

ERRACT is the error action taken when a fatal application error occurs (TREAT, ROUTE).

7 Datafill the options, where

options is only allowed when ACTION is QUERY. Enter up to 6 options: BUFFER, GT, T1OVFLGT, ACGOVFLGT, VERSION, or STREAM. When the ACTION is not QUERY enter $.

Sample entry: **>ADD caingrp1 XLAADDR 223 900 query treat buffer $**

Specific_Feature_Code trigger criteria is defined.

Associated OMs

CAINMSGS, CAINTRIG, CAINAGOM

Customized_Dialing_Plan trigger

ATTENTION

The *Customized_Dialing_Plan* trigger requires the CAIN0500 SOC option. Refer to Volume 5, Chapter 5, "NetworkBuilder SOC functionality," for more information.

Uses

When you provision NetworkBuilder services on the switch, the SCP can make major call processing decisions. You define the services offered on the SCP. Some examples of services that can be developed on the SCP for the *Customized_Dialing_Plan* trigger are:

- VPN Services
  - — OFFNET Overflow
  - — Forced ONNET
  - — Alternate Billing Numbers
  - — Origination/Termination Screening
  - — Customized Announcements
  - — Black Box Screening
- CIC Routing and Branding

Supported originating agencies

The *Customized_Dialing_Plan* trigger supports the following originating agencies:

- DAL
- FGD
- PRI
- SS7 Inter-IMT
- SS7 Global-IMT
- AXXESS

  Note: Refer to *UCS DMS-250 CAIN/FlexDial Interactions* for more information on support for AXXESS agents.

Subscribing to the Customized_Dialing_Plan trigger

Subscription to the *Customized_Dialing_Plan* trigger is available on a

- SCP-returned basis

- address basis (table STDPRTCT, subtable STDPRT)
- authorization code basis (tables AUTHCODU, AUTHCDU2, AUTHCDU3, AUTHCDU4, or AUTHCDU5)
- ANI basis (table ANISCUSP, or tables ANIVAL and UNIPROF)
- agent basis (table TRKGRP or CALLATTR)
- office basis (table CAINPARM)

Note 1: An SCP-returned CAIN subscription group is received in **`Analyze_Route`** messages. When a reorigination occurs after an **`Analyze_Route`** is received, the SCP-returned CAIN group is available for any reoriginated call.

Note 2: CAIN group subscription for AXXESS agents is handled differently. Refer to *UCS DMS-250 CAIN/FlexDial Interactions*.

Trigger evaluation

NetworkBuilder checks the *Customized_Dialing_Plan* trigger table (CUSTDP) and evaluates the call's 1 to 24 digit address against the datafilled range associated with the appropriate CAIN group.

Note 1: The digit type (XLAADDR and ADDR) is identified through datafill within the CUSTDP table.

Note 2: For evaluation due to an **`Analyze_Route`** message, only the XLAADDR digit type is considered.

Trigger actions

NetworkBuilder call processing supports the following actions for the *Customized_Dialing_Plan* trigger:

- BLOCK – Prevents the call from proceeding and applies AINF treatment.
- IGNORE – NetworkBuilder continues checking the remaining subscription methods for *Specific_Feature_Code*, *Customized_Dialing_Plan*, *Specific_Digit_String*, and *Office_Code*. If datafill does not enable any trigger, call processing continues through the call model.
- QUERY – The switch builds an **`Info_Analyzed`** query and sends it to the SCP. The SCP analyzes the call and assists the switch with call processing.
- QUERYSCU – In-switch call processing is temporarily suspended. The call enters server mode and becomes a Programmable Service Node (PSN) call and a **`New_Call`** message is sent to the PSN SCU. Refer to the *UCS DMS-250 Programmable Service Node (PSN) Application Guide* for more PSN information.

Customized_Dialing_Plan trigger (continued)

Note: If an in-switch error occurs before all digits are present the call goes to treatment and does not become a PSN call as it would have under the existing NetworkBuilder functionality.

- LEAVE_TDP – NetworkBuilder call processing exits the ***Info_Analyzed*** TDP with no further evaluation.
- CONT_NOTRIG – NetworkBuilder call processing exits the ***Info_Analyzed*** TDP and prevents any further NetworkBuilder interaction for the call. This prevention is reset upon reorigination.

Error actions

When a fatal application occurs during an SCP query, one of the following error actions is performed (as datafilled):

- TREAT – AINF treatment is applied.
- ROUTE – The switch discards any SCP-provided data. NetworkBuilder continues checking the remaining subscription methods for *Specific_Feature_Code*, *Customized_Dialing_Plan*, *Specific_Digit_String*, and *Office_Code*. If datafill does not enable any trigger, call processing continues through the call model.

Options

The *Customized_Dialing_Plan* trigger supports the following options:

- BUFFER – to activate digit buffering while the SCP is queried.
- GT – to identify the global title used to identify the SCP handling the query.
- T1OVFLGT – to identify the specific SCP to query on T1 overflow.
- ACGOVFLGT – to identify the global title to use for requerying when a query is blocked by an ACG control
- VERSION – controls the CAIN protocol version for outgoing messages

  Note: The value provisioned in the VERSION option overrides the version provisioned in the CAIN_PROTOCOL_VERSION parameter of table CAINPARM on a transaction-by-transaction basis. Additionally, the VERSION option applies to both the outgoing and incoming messages within the same transaction. (Subsequent queries will use the setting provisioned in the CAIN_PROTOCOL_VERSION or provisioned setting of the VERSION option in the trigger table performing the subsequent query.)

Customized_Dialing_Plan trigger (continued)

- STREAM – to control protocol stream on a per-trigger tuple basis. The value of the STREAM option controls the set of parameters that are sent in NetworkBuilder messages.

Customized_Dialing_Plan trigger (continued)

The following figure shows how a call progresses through the call model, encounters ***Info_Analyzed*** and queries the SCP.

UA Auth call

FGD origination
Datafill:

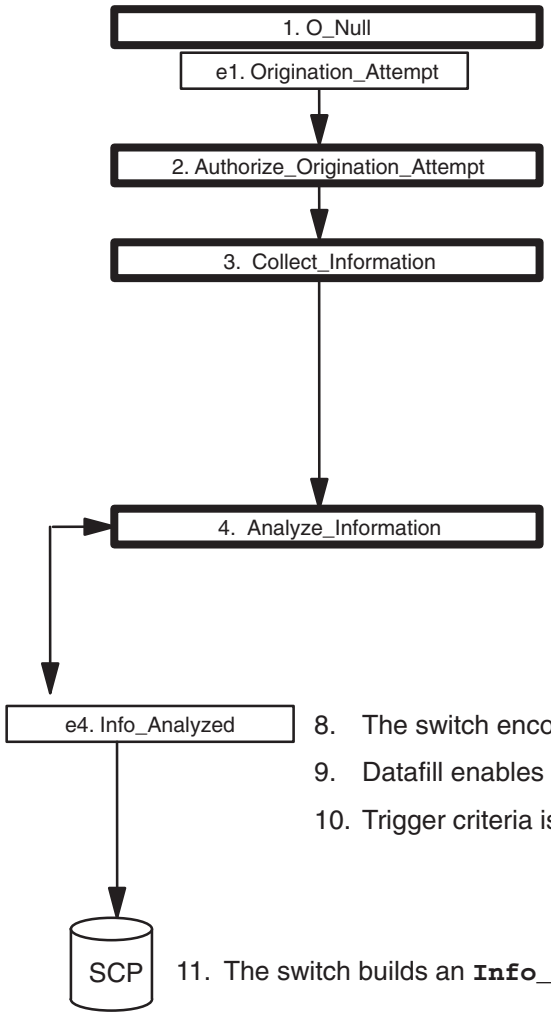
Signaling: MF at origination
DTMF for digit collection

Dialing plan: KP + II + ANI + ST
KP + ADDR + ST

ADDR: (214) 611-3311

Table CUSTDP

CUSTDPKY	TODIGS	ACTION	OPTION
CAINGRP1 ADDR 214	214	QUERY TREAT	$



1. The switch detects seizure and enters **O_Null**.
2. Call processing enters **Authorize_Origination_Attempt**.
3. Call processing enters **Collect_Information**.
4. The switch collects KP + II + ANI + ST.
5. The switch screens II and ANI digits.
6. The switch collects KP + ADDR + ST (ADDR is a UA number).
7. The switch performs digit screening (in-switch or off-board).
8. The switch encounters ***Info_Analyzed***.
9. Datafill enables *Customized_Dialing_Plan* trigger.
10. Trigger criteria is met.
11. The switch builds an **`Info_Analyzed`** query and sends it to the SCP.

Info_Analyzed TDP-Request

An **Info_Analyzed** TDP-Request (query) is sent to the SCP in a query package, with a component type of Invoke_Last. The following table defines the parameters and usage requirements for the parameters the **Info_Analyzed** TDP-Request may contain. Refer to Volume 3, Chapter 8, "Outgoing CAIN message parameters," for detailed descriptions of the request parameters.

Note: Parameters and extension parameters for AXXESS agents are handled differently. Refer to *UCS DMS-250 CAIN/FlexDial Interactions* for more information.

Info_Analyzed TDP-Request message parameters for Customized_Dialing_Plan

Parameter	Usage	Definition
UserID	Required	Contains the network identity of the originating agent
BearerCapability	Required	Contains the bearer capability of the call when the message is built
CalledPartyID	Optional	Contains the translated address
Lata (Note 8)	Optional	Contains the call's Local Access and Transport Area (LATA)
TriggerCriteriaType	Optional	Contains customIntercom

Note 1: The `universalAccess` extension parameter is included in the query (a flat 10 digits) when parameter CAIN_PROTOCOL_VERSION is V0. When CAIN_PROTOCOL_VERSION is set to V1 or higher, the parameter contains up to 24 digits including a nature of address and numbering plan similarly to the ***CalledPartyID***.
Note 2: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS06 or higher.
Note 3: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS07 or higher.
Note 4: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS08 or higher.
Note 5: This parameter is not populated when the CAIN_PROTOCOL_VERSION is set to V3 or higher, and is therefore not sent.
Note 6: The parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS09 or higher.
Note 7: When the CAIN_PROTOCOL_VERSION is set to V4 or higher non-standard charge numbers (CARD, AUTH, ACCT, PIN, and N00) no longer populate the ***ChargeNumber*** parameter and all standard NOAs are supported.
Note 8: The parameter is supported only on DAL, FGD, and AXXESS agencies and the parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS11 or higher.
Note 9: Refer to Volume 3, Chapter 8, "Outgoing CAIN message parameters" for information on the population of this parameter when the CAIN_PROTOCOL_VERSION is set to V5 or higher.

—continued—

Customized_Dialing_Plan trigger (continued)

Info_Analyzed TDP-Request message parameters for Customized_Dialing_Plan (continued)

Parameter	Usage	Definition
ChargeNumber(Note 7)	Optional	Contains the billing number that would be used to populate the CDR at this point in call processing
CallingPartyID (Note 9)	Optional	Contains one of the following (listed in order of precedence): For SS7 FGD, SS7 Inter-IMT, SS7-Global IMT, and AXXESS calls: Calling_Party_Address from ISUP message, when available For FGD and AXXESS calls: valid ANI (information digits are not passed in this parameter) ***Note:*** Refer to *UCS DMS-250 CAIN/FlexDial Interactions* for more information on AXXESS agents. For PRI calls: CLID when available Valid SNPA value from table TRKGRP Default SNPA value from table CAINPARM
ChargePartyStationType (Note 3)	Optional	Contains the information digits for the call

Note 1: The `universalAccess` extension parameter is included in the query (a flat 10 digits) when parameter CAIN_PROTOCOL_VERSION is V0. When CAIN_PROTOCOL_VERSION is set to V1 or higher, the parameter contains up to 24 digits including a nature of address and numbering plan similarly to the ***CalledPartyID***.
Note 2: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS06 or higher.
Note 3: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS07 or higher.
Note 4: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS08 or higher.
Note 5: This parameter is not populated when the CAIN_PROTOCOL_VERSION is set to V3 or higher, and is therefore not sent.
Note 6: The parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS09 or higher.
Note 7: When the CAIN_PROTOCOL_VERSION is set to V4 or higher non-standard charge numbers (CARD, AUTH, ACCT, PIN, and N00) no longer populate the ***ChargeNumber*** parameter and all standard NOAs are supported.
Note 8: The parameter is supported only on DAL, FGD, and AXXESS agencies and the parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS11 or higher.
Note 9: Refer to Volume 3, Chapter 8, “Outgoing CAIN message parameters” for information on the population of this parameter when the CAIN_PROTOCOL_VERSION is set to V5 or higher.

—continued—

Customized_Dialing_Plan trigger (continued)

Info_Analyzed TDP-Request message parameters for Customized_Dialing_Plan (continued)

Parameter	Usage	Definition
Carrier	Optional	Contains the dialed CIC or the CIC value (with an indication of: Selected CIC presubscribed and not input by calling party) from table TRKGRP.
AccessCode (Note 5)	Optional	Contains the account code, when available
CollectedAddressInfo	Optional	Contains the address collected from the incoming agent (from the IAM, subscriber dialing, or datafilled hotline digits)
CollectedDigits (Note 5)	Optional	Contains the collected PIN digits, when available
VerticalServiceCode (Note 2)	Optional	Contains feature codes dialed by the subscriber
ACGEncountered	Optional	Contains the ACG control encountered when the ACG control expires and a query is allowed to be sent
ExtensionParameter (Note 3)	Optional	Extension parameters require the CAIN0200 SOC option.
universalAccess (Note 1)	Optional	Contains the universal access number dialed by the caller to obtain switch dial tone.

Note 1: The universalAccess extension parameter is included in the query (a flat 10 digits) when parameter CAIN_PROTOCOL_VERSION is V0. When CAIN_PROTOCOL_VERSION is set to V1 or higher, the parameter contains up to 24 digits including a nature of address and numbering plan similarly to the ***CalledPartyID***.
Note 2: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS06 or higher.
Note 3: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS07 or higher.
Note 4: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS08 or higher.
Note 5: This parameter is not populated when the CAIN_PROTOCOL_VERSION is set to V3 or higher, and is therefore not sent.
Note 6: The parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS09 or higher.
Note 7: When the CAIN_PROTOCOL_VERSION is set to V4 or higher non-standard charge numbers (CARD, AUTH, ACCT, PIN, and N00) no longer populate the ***ChargeNumber*** parameter and all standard NOAs are supported.
Note 8: The parameter is supported only on DAL, FGD, and AXXESS agencies and the parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS11 or higher.
Note 9: Refer to Volume 3, Chapter 8, “Outgoing CAIN message parameters” for information on the population of this parameter when the CAIN_PROTOCOL_VERSION is set to V5 or higher.

—continued—

Customized_Dialing_Plan trigger (continued)

Info_Analyzed TDP-Request message parameters for Customized_Dialing_Plan (continued)

Parameter	Usage	Definition
`cainGroup`	Optional	Contains the group number (field GRPNUM, table CAINGRP) for the CAIN group associated with the trigger when field EXTPARM in table CAINGRP contains CAINGRP.
`adin`	Optional	Contains the authorization code database index (field ADIN, table TRKGRP for DAL and FGD agencies; field ADIN, table CALLATTR for PRI agencies) associated with the originating agency when field EXTPARM in table CAINGRP contains ADIN. ***Note:*** The `adin` extension parameter is not supported for AXXESS agents.
`origTrunkInfo`	Optional	Contains the originating trunk group number, trunk type, and trunk member number when field EXTPARM in table CAINGRP contains ORGTINFO.
`treatment` (Note 2)	Optional	Contains the treatment set by regular call processing before the query was sent
`reorigCall` (Note 2)	Optional	Presence of this parameter indicates the call in progress is a result of reorigination

Note 1: The `universalAccess` extension parameter is included in the query (a flat 10 digits) when parameter CAIN_PROTOCOL_VERSION is V0. When CAIN_PROTOCOL_VERSION is set to V1 or higher, the parameter contains up to 24 digits including a nature of address and numbering plan similarly to the ***`CalledPartyID`***.
Note 2: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS06 or higher.
Note 3: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS07 or higher.
Note 4: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS08 or higher.
Note 5: This parameter is not populated when the CAIN_PROTOCOL_VERSION is set to V3 or higher, and is therefore not sent.
Note 6: The parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS09 or higher.
Note 7: When the CAIN_PROTOCOL_VERSION is set to V4 or higher non-standard charge numbers (CARD, AUTH, ACCT, PIN, and N00) no longer populate the ***`ChargeNumber`*** parameter and all standard NOAs are supported.
Note 8: The parameter is supported only on DAL, FGD, and AXXESS agencies and the parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS11 or higher.
Note 9: Refer to Volume 3, Chapter 8, “Outgoing CAIN message parameters” for information on the population of this parameter when the CAIN_PROTOCOL_VERSION is set to V5 or higher.

—continued—

Customized_Dialing_Plan trigger (continued)

Info_Analyzed TDP-Request message parameters for Customized_Dialing_Plan (continued)

Parameter	Usage	Definition
univIdx	Optional	Used by SS7 Global-IMT agents only. Contains the universal translations scheme to be used for the call.
netinfo (Note 3)	Optional	Contains external network ID, network customer group ID, and network class of service.
t1Overflow	Optional	Indicates to the SCP that an overflow occurred on the query to the initial SCP.
lnpReceived (Note 3)	Optional	Presence of this parameter indicates LNP information was received from a previous switch.
subscriptionInfo (Note 4)	Optional	Contains the digit type triggered on for the query and the method to which the triggering CAIN group subscribed.
switchID (Note 6)	Optional	Contains the switch ID
accountCode (Note 6)	Optional	Contains the account code when available
pinDigits (Note 6)	Optional	Contains the collected pin digits when available
billingNumber (Note 6)	Optional	Contains the non-standard charge number

Note 1: The universalAccess extension parameter is included in the query (a flat 10 digits) when parameter CAIN_PROTOCOL_VERSION is V0. When CAIN_PROTOCOL_VERSION is set to V1 or higher, the parameter contains up to 24 digits including a nature of address and numbering plan similarly to the ***CalledPartyID***.
Note 2: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS06 or higher.
Note 3: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS07 or higher.
Note 4: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS08 or higher.
Note 5: This parameter is not populated when the CAIN_PROTOCOL_VERSION is set to V3 or higher, and is therefore not sent.
Note 6: The parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS09 or higher.
Note 7: When the CAIN_PROTOCOL_VERSION is set to V4 or higher non-standard charge numbers (CARD, AUTH, ACCT, PIN, and N00) no longer populate the ***ChargeNumber*** parameter and all standard NOAs are supported.
Note 8: The parameter is supported only on DAL, FGD, and AXXESS agencies and the parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS11 or higher.
Note 9: Refer to Volume 3, Chapter 8, “Outgoing CAIN message parameters” for information on the population of this parameter when the CAIN_PROTOCOL_VERSION is set to V5 or higher.

—continued—

Customized_Dialing_Plan trigger (continued)

Info_Analyzed TDP-Request message parameters for Customized_Dialing_Plan (continued)

Parameter	Usage	Definition
acgRequery	Optional	When present, indicates to the SCP that the current query is a re-query message resulting from the original query being blocked by an ACG control
JurisdictionInfo (Note 4)	Optional	Contains the originating switch's location routing number (LRN).

Note 1: The universalAccess extension parameter is included in the query (a flat 10 digits) when parameter CAIN_PROTOCOL_VERSION is V0. When CAIN_PROTOCOL_VERSION is set to V1 or higher, the parameter contains up to 24 digits including a nature of address and numbering plan similarly to the ***CalledPartyID***.
Note 2: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS06 or higher.
Note 3: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS07 or higher.
Note 4: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS08 or higher.
Note 5: This parameter is not populated when the CAIN_PROTOCOL_VERSION is set to V3 or higher, and is therefore not sent.
Note 6: The parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS09 or higher.
Note 7: When the CAIN_PROTOCOL_VERSION is set to V4 or higher non-standard charge numbers (CARD, AUTH, ACCT, PIN, and N00) no longer populate the ***ChargeNumber*** parameter and all standard NOAs are supported.
Note 8: The parameter is supported only on DAL, FGD, and AXXESS agencies and the parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS11 or higher.
Note 9: Refer to Volume 3, Chapter 8, "Outgoing CAIN message parameters" for information on the population of this parameter when the CAIN_PROTOCOL_VERSION is set to V5 or higher.

—end—

Once the request message is built, NetworkBuilder formats the message and sends it to the message encoder. Once encoded, the message is sent to the SCP.

SCP response processing

The following response messages are supported:

- **Analyze_Route**
- **Send_To_Resource**
- **Continue** – NetworkBuilder continues checking the remaining subscription methods for *Specific_Feature_Code*, *Customized_Dialing_Plan*, *Specific_Digit_String*, and *Office_Code*. If datafill does not enable any trigger, call processing continues through the call model.
- **Disconnect**

Note 1: EDPs may be armed through the **`Request_Report_BCM_Event`** component with **`Analyze_Route`** and **`Continue`** messages.

Note 2: A **`Termination_Notification`** may be requested through the **`Send_Notification`** component with the SCP response message.

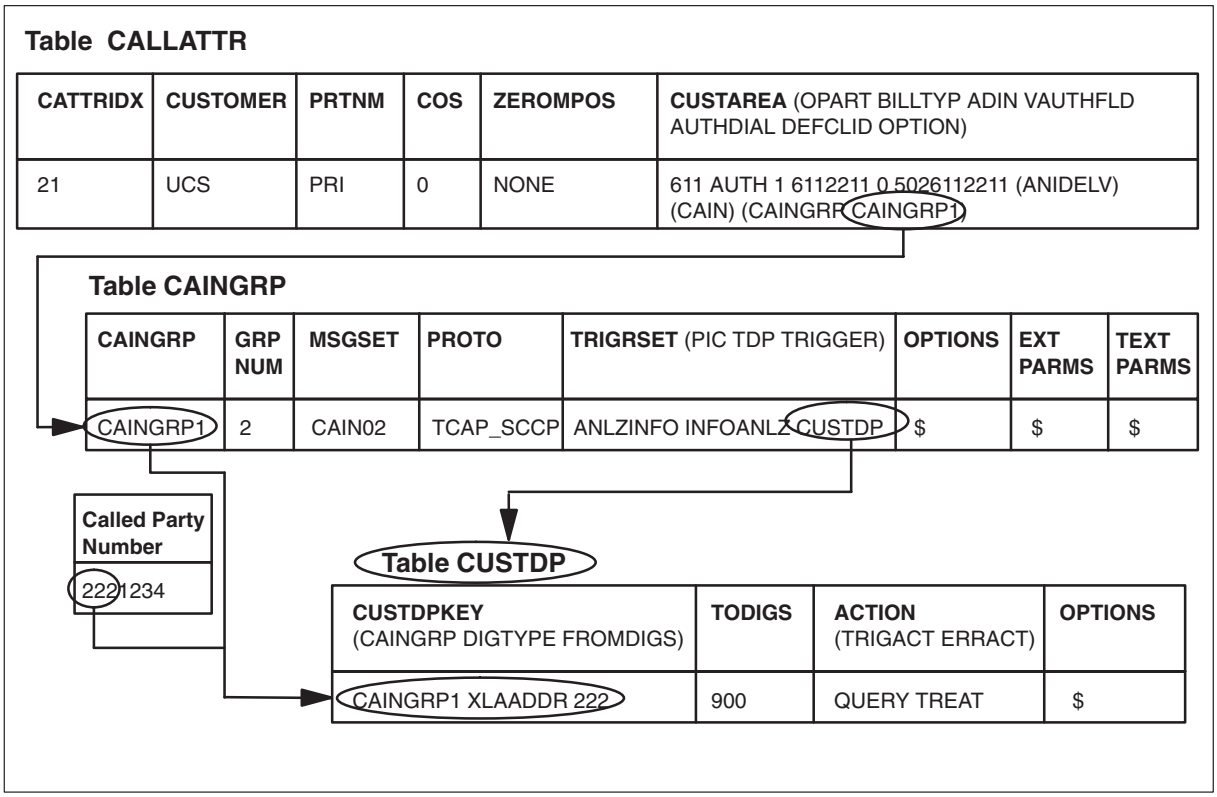
Note 3: An **`ACG`** message may be sent with the SCP response message.

Refer to Volume 3, Chapter 10, "Incoming CAIN messages," for specific information regarding response messages; refer to Volume 3, Chapter 12, "Incoming CAIN message parameters," for detailed descriptions of the response parameters; refer to Volume 3, Chapter 3, "Event processing," for EDP specific message information.

Datafill

The following figure shows how a subscription table interacts with the *Customized_Dialing_Plan* trigger table (CUSTDP).

Subscription-CUSTDP table interaction



Customized_Dialing_Plan trigger (continued)

Provisioning the Customized_Dialing_Plan trigger

At the CI prompt

1 Provision the originating agent as CAIN-capable.

2 Subscribe to a CAIN group (table STDPRTCT, authcode tables, ANI tables, TRKGRP, CALLATTR, or CAINPARM).

Note: CAIN group subscription is handled differently for AXXESS agents. Refer to *UCS DMS-250 CAIN/FlexDial Interactions*.

3 Datafill CAIN group trigger subscription to the (ANLZINFO INFOANLZ CUSTDP) trigger set (table CAINGRP).

4 Enter table CUSTDP.

5 Define the trigger criteria for a CAIN group by using the following format:

>ADD custdpkey todigs action options
where

custdpkey is comprised of three subfields: CAINGRP, DIGTYPE, and FROMDIGS, where
- CAINGRP is the CAIN group requiring Customized_Dialing_Plan trigger criteria (from table CAINGRP).
- DIGTYPE is the digit type being analyzed (XLAADDR, ADDR).
- FROMDIGS is the first number used to define the range of the collected address.

todigs is the second number used to define the range of the collected address.

action is comprised of one subfield: TRIGACT, where
- TRIGACT is the trigger action taken when the address is within the FROMDIGS-TODIGS range (BLOCK, IGNORE, QUERY, QUERYSCU, LEAVE_TDP, CONT_NOTRIG).

If you datafill TRIGACT as:	Go to:
BLOCK	step 7
IGNORE	step 7
QUERY	step 6
QUERYSCU	step 7

Customized_Dialing_Plan trigger (end)

If you datafill TRIGACT as:	Go to:
LEAVE_TDP	step 7
CONT_NOTRIG	step 7

6 Datafill the ERRACT refinement, where

ERRACT is the error action taken when a fatal application error occurs (TREAT, ROUTE).

7 Datafill the options, where

options is only allowed when the ACTION is QUERY. Enter up to 6 options: BUFFER, GT, T1OVFLGT, ACGOVFLGT, VERSION, or STREAM. When the ACTION is not QUERY enter $.

Sample entry: **>ADD caingrp1 XLAADDR 222 222 block $**

Sample entry: **>ADD caingrp1 XLAADDR 223 900 query treat buffer $**

Customized_Dialing_Plan trigger criteria is defined.

Associated OMs

CAINMSGS, CAINTRIG, CAINAGOM

Specific_Digit_String trigger

Trigger definition

ATTENTION

The *Specific_Digit_String* trigger requires the CAIN0501 SOC option. Refer to Volume 5, Chapter 5, "NetworkBuilder SOC functionality," for more information.

Uses

When you provision NetworkBuilder services on the switch, the SCP can make major call processing decisions. You define the services offered on the SCP. Some examples of services that can be developed on the SCP for the *Specific_Digit_String* trigger are:

- Dial 1+ Services
  - — Speed Dial
  - — Hotline
  - — Account Code Screening
  - — Bill to Office
  - — CIC Routing and Branding
  - — Black Box Screening
- N00 Services
  - — Follow Me, Find Me, Do Not Disturb Me
  - — Call Screening
  - — Customized Call Branding
  - — Universal Access Authorization
- Subscriber Screening
  - — Authorization Code Screening
  - — Account Code Screening

Supported originating agencies

The *Specific_Digit_String* trigger supports the following originating agencies:

- DAL (loop and ground starts)
- FGD
- PRI
- SS7 Inter-IMT

- SS7 Global-IMT
- AXXESS

  Note: Refer to *UCS DMS-250 CAIN/FlexDial Interactions* for more information on support for AXXESS agents.

Subscribing to the Specific_Digit_String trigger

Subscription to the *Specific_Digit_String* trigger is available on a

- SCP-returned basis
- address basis (table STDPRTCT, subtable STDPRT)
- authorization code basis (tables AUTHCODU, AUTHCDU2, AUTHCDU3, AUTHCDU4, or AUTHCDU5)
- ANI basis (table ANISCUSP, or tables ANIVAL and UNIPROF)
- agent basis (table TRKGRP or CALLATTR)
- office basis (table CAINPARM)

Note: An SCP-returned CAIN subscription group is received in **`Analyze_Route`** messages. When a reorigination occurs after an **`Analyze_Route`** is received, the SCP-returned CAIN group is available for any reoriginated call.

Note: CAIN group subscription for AXXESS agents is handled differently. Refer to *UCS DMS-250 CAIN/FlexDial Interactions*.

Trigger evaluation

NetworkBuilder checks the *Specific_Digit_String* trigger table (SPECDIG) and evaluates the identified digits against the datafilled range associated with the appropriate CAIN group. Digits are checked in the following order:

1 Information digits – provide additional information about the originating party

2 ADIN digits – authcode database index

3 ANI digits – automatic number identification digits

4 XLAADDR digits – translated address

5 Address digits – originally dialed address

6 CIC digits – carrier identification code

Note 1: The evaluated address is the initially collected address. Refer to Volume 3, Chapter 8, “Outgoing CAIN message parameters,” for parameter population details.

Specific_Digit_String trigger (continued)

Note 2: For evaluation due to an **`Analyze_Route`** message, only the XLAADDR digit type is considered.

Trigger actions

NetworkBuilder call processing supports the following actions for the *Specific_Digit_String* trigger:

- BLOCK – Prevents the call from proceeding and applies AINF treatment.
- IGNORE – NetworkBuilder continues checking the remaining subscription methods for *Specific_Feature_Code*, *Customized_Dialing_Plan*, *Specific_Digit_String*, and *Office_Code* triggers. If datafill does not enable any trigger, call processing continues through the call model.
- QUERY – The switch builds an **`Info_Analyzed`** query and sends it to the SCP. The SCP analyzes the call and assists the switch with call processing.
- QUERYSCU – In-switch call processing is temporarily suspended. The call enters server mode and becomes a Programmable Service Node (PSN) call and a **`New_Call`** message is sent to the PSN SCU. Refer to *UCS DMS-250 Programmable Service Node (PSN) Application Guide* for more PSN information.

  Note: If an in-switch error occurs before all digits are present the call goes to treatment and does not become a PSN call as it would have under the existing NetworkBuilder functionality.

- LEAVE_TDP – NetworkBuilder call processing exits the ***Info_Analyzed*** TDP with no further evaluation.
- CONT_NOTRIG – NetworkBuilder call processing exits the ***Info_Analyzed*** TDP and prevents any further NetworkBuilder interaction for the call. This prevention is reset upon reorigination.

Error actions

When a fatal application occurs during an SCP query, one of the following error actions is performed (as datafilled):

- TREAT – AINF treatment is applied.
- ROUTE – The switch discards any SCP-provided data. NetworkBuilder continues checking the remaining subscription methods for *Specific_Feature_Code*, *Customized_Dialing_Plan*, *Specific_Digit_String*, and *Office_Code* triggers. If datafill does not enable any trigger, call processing continues through the call model.

Options

The *Specific_Digit_String* trigger supports the following options:

- BUFFER – to activate digit buffering while the SCP is queried.
- GT – to identify the global title used to identify the SCP handling the query.
- T1OVFLGT – to identify the specific SCP to query on T1 overflow.
- ACGOVFLGT – to identify the global title to use for requerying when a query is blocked by an ACG control
- VERSION – controls the CAIN protocol version for outgoing messages

  Note: The value provisioned in the VERSION option overrides the version provisioned in the CAIN_PROTOCOL_VERSION parameter of table CAINPARM on a transaction-by-transaction basis. Additionally, the VERSION option applies to both the outgoing and incoming messages within the same transaction. (Subsequent queries will use the setting provisioned in the CAIN_PROTOCOL_VERSION or provisioned setting of the VERSION option in the trigger table performing the subsequent query.)

- STREAM – to control protocol stream on a per-trigger tuple basis. The value of the STREAM option controls the set of parameters that are sent in NetworkBuilder messages.

Info_Analyzed TDP-Request

An **`Info_Analyzed`** TDP-Request (query) is sent to the SCP in a query package, with a component type of Invoke_Last. The following table defines the parameters and usage requirements for the parameters the **`Info_Analyzed`** TDP-Request may contain. Refer to Volume 3, Chapter 8, "Outgoing CAIN message parameters," for detailed descriptions of the request parameters.

Note: Parameters and extension parameters for AXXESS agents are handled differently. Refer to *UCS DMS-250 CAIN/FlexDial Interactions* for more information.

Specific_Digit_String trigger (continued)

Info_Analyzed TDP-Request message parameters for Specific_Digit_String

Parameter	Usage	Definition
UserID	Required	Contains the network identity of the originating agent
BearerCapability	Required	Contains the bearer capability of the call when the message is built
CalledPartyID	Optional	Contains the translated address
LATA (Note 8)	Optional	Contains the call's Local Access and Transport Area (LATA)
TriggerCriteriaType	Optional	Contains SDS_INFO, SDS_ADIN, SDS_ADDR, SDS_INTL, SDS_N00, SDS_ANI, SDS_CIC, NPA, NPA_N, NPA_NX, NPA_NXX, NPA_NXXX, NPA_NXXXX, NPA_XXXXX, or NPA_NXXXXXX
ChargeNumber (Note 7)	Optional	Contains the billing number that would be used to populate the CDR at this point in call processing

Note 1: The `universalAccess` extension parameter is included in the query (a flat 10 digits) when parameter CAIN_PROTOCOL_VERSION is V0. When CAIN_PROTOCOL_VERSION is set to V1 or higher, the parameter contains up to 24 digits including a nature of address and numbering plan similarly to the ***CalledPartyID***.
Note 2: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS06 or higher.
Note 3: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS07 or higher.
Note 4: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS08 or higher.
Note 5: This parameter is not populated when the CAIN_PROTOCOL_VERSION is set to V3 or higher, and is therefore not sent.
Note 6: The parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS09 or higher.
Note 7: When the CAIN_PROTOCOL_VERSION is set to V4 or higher non-standard charge numbers (CARD, AUTH, ACCT, PIN, and N00) no longer populate the ***ChargeNumber*** parameter and all standard NOAs are supported.
Note 8: The parameter is supported only on DAL, FGD, and AXXESS agencies and the parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS11 or higher.
Note 9: Refer to Volume 3, Chapter 8, "Outgoing CAIN message parameters" for information on the population of this parameter when the CAIN_PROTOCOL_VERSION is set to V5 or higher.

—continued—

Specific_Digit_String trigger (continued)

Info_Analyzed TDP-Request message parameters for Specific_Digit_String (continued)

Parameter	Usage	Definition
CallingPartyID (Note 9)	Optional	Contains the one of the following (listed in order of precedence): For SS7 FGD calls: Calling_Party_Address from ISUP message, when available For FGD and AXXESS calls: valid ANI (information digits are not passed in this parameter) ***Note:*** Refer to *UCS DMS-250 CAIN/FlexDial Interactions* for more information on AXXESS agents. For PRI calls: CLID when available Valid SNPA value from table TRKGRP Default SNPA value from table CAINPARM
ChargePartyStationType (Note 3)	Optional	Contains the information digits for the call
Carrier	Optional	Contains the dialed CIC or the CIC value (with an indication of: Selected CIC presubscribed and not input by calling party) from table TRKGRP.
AccessCode(Note 5)	Optional	Contains the account code, when available

Note 1: The `universalAccess` extension parameter is included in the query (a flat 10 digits) when parameter CAIN_PROTOCOL_VERSION is V0. When CAIN_PROTOCOL_VERSION is set to V1 or higher, the parameter contains up to 24 digits including a nature of address and numbering plan similarly to the ***CalledPartyID***.
Note 2: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS06 or higher.
Note 3: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS07 or higher.
Note 4: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS08 or higher.
Note 5: This parameter is not populated when the CAIN_PROTOCOL_VERSION is set to V3 or higher, and is therefore not sent.
Note 6: The parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS09 or higher.
Note 7: When the CAIN_PROTOCOL_VERSION is set to V4 or higher non-standard charge numbers (CARD, AUTH, ACCT, PIN, and N00) no longer populate the ***ChargeNumber*** parameter and all standard NOAs are supported.
Note 8: The parameter is supported only on DAL, FGD, and AXXESS agencies and the parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS11 or higher.
Note 9: Refer to Volume 3, Chapter 8, “Outgoing CAIN message parameters” for information on the population of this parameter when the CAIN_PROTOCOL_VERSION is set to V5 or higher.

—continued—

Specific_Digit_String trigger (continued)

Info_Analyzed TDP-Request message parameters for Specific_Digit_String (continued)

Parameter	Usage	Definition
CollectedAddressInfo	Optional	Contains the address collected from the incoming agent (from the IAM, subscriber dialing, or datafilled hotline digits)
CollectedDigits(Note 5)	Optional	Contains the collected PIN digits, when available
VerticalServiceCode (Note 2)	Optional	Contains feature codes dialed by the subscriber
ACGEncountered	Optional	Contains the ACG control encountered when the ACG control expires and a query is allowed to be sent
ExtensionParameter (Note 3)	Optional	Extension parameters require the CAIN0200 SOC option.
universalAccess (Note1)	Optional	Contains the universal access number dialed by the caller to obtain switch dial tone.
cainGroup	Optional	Contains the group number (field GRPNUM, table CAINGRP) for the CAIN group associated with the trigger when field EXTPARM in table CAINGRP contains CAINGRP.

Note 1: The universalAccess extension parameter is included in the query (a flat 10 digits) when parameter CAIN_PROTOCOL_VERSION is V0. When CAIN_PROTOCOL_VERSION is set to V1 or higher, the parameter contains up to 24 digits including a nature of address and numbering plan similarly to the ***CalledPartyID***.
Note 2: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS06 or higher.
Note 3: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS07 or higher.
Note 4: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS08 or higher.
Note 5: This parameter is not populated when the CAIN_PROTOCOL_VERSION is set to V3 or higher, and is therefore not sent.
Note 6: The parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS09 or higher.
Note 7: When the CAIN_PROTOCOL_VERSION is set to V4 or higher non-standard charge numbers (CARD, AUTH, ACCT, PIN, and N00) no longer populate the ***ChargeNumber*** parameter and all standard NOAs are supported.
Note 8: The parameter is supported only on DAL, FGD, and AXXESS agencies and the parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS11 or higher.
Note 9: Refer to Volume 3, Chapter 8, “Outgoing CAIN message parameters” for information on the population of this parameter when the CAIN_PROTOCOL_VERSION is set to V5 or higher.

—continued—

Specific_Digit_String trigger (continued)

Info_Analyzed TDP-Request message parameters for Specific_Digit_String (continued)

Parameter	Usage	Definition
adin	Optional	Contains the authorization code database index (field ADIN, table TRKGRP for DAL and FGD agencies; field ADIN, table CALLATTR for PRI agencies) associated with the originating agency when field EXTPARM in table CAINGRP contains ADIN. ***Note:*** The adin extension parameter is not supported for AXXESS agents.
origTrunkInfo	Optional	Contains the originating trunk group number, trunk type, and trunk member number when field EXTPARM in table CAINGRP contains ORGTINFO.
treatment (Note 2)	Optional	Contains the treatment set by regular call processing before the query was sent
reorigCall (Note 2)	Optional	Presence of this parameter indicates the call in progress is a result of reorigination
univIdx	Optional	Used by SS7 Global-IMT agents only. Contains the universal translations scheme to be used for the call.
netinfo (Note 3)	Optional	Contains external network ID, network customer group ID, and network class of service.

Note 1: The universalAccess extension parameter is included in the query (a flat 10 digits) when parameter CAIN_PROTOCOL_VERSION is V0. When CAIN_PROTOCOL_VERSION is set to V1 or higher, the parameter contains up to 24 digits including a nature of address and numbering plan similarly to the ***CalledPartyID***.
Note 2: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS06 or higher.
Note 3: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS07 or higher.
Note 4: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS08 or higher.
Note 5: This parameter is not populated when the CAIN_PROTOCOL_VERSION is set to V3 or higher, and is therefore not sent.
Note 6: The parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS09 or higher.
Note 7: When the CAIN_PROTOCOL_VERSION is set to V4 or higher non-standard charge numbers (CARD, AUTH, ACCT, PIN, and N00) no longer populate the ***ChargeNumber*** parameter and all standard NOAs are supported.
Note 8: The parameter is supported only on DAL, FGD, and AXXESS agencies and the parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS11 or higher.
Note 9: Refer to Volume 3, Chapter 8, “Outgoing CAIN message parameters” for information on the population of this parameter when the CAIN_PROTOCOL_VERSION is set to V5 or higher.

—continued—

Specific_Digit_String trigger (continued)

Info_Analyzed TDP-Request message parameters for Specific_Digit_String (continued)

Parameter	Usage	Definition
`t1Overflow`	Optional	Indicates to the SCP that an overflow occurred on the query to the initial SCP.
`lnpReceived` (Note 3)	Optional	Presence of this parameter indicates LNP information was received from a previous switch.
`subscriptionInfo` (Note 4)	Optional	Contains the digit type triggered on for the query and the method to which the triggering CAIN group subscribed.
`switchID` (Note 6)	Optional	Contains the switch ID
`accountCode` (Note 6)	Optional	Contains the account code when available
`pinDigits` (Note 6)	Optional	Contains the collected pin digits when available
`billingNumber` (Note 6)	Optional	Contains the non-standard charge number
`acgRequery`	Optional	When present, indicates to the SCP that the current query is a re-query message resulting from the original query being blocked by an ACG control
JurisdictionInfo (Note 4)	Optional	Contains the originating switch's location routing number (LRN).

Note 1: The `universalAccess` extension parameter is included in the query (a flat 10 digits) when parameter CAIN_PROTOCOL_VERSION is V0. When CAIN_PROTOCOL_VERSION is set to V1 or higher, the parameter contains up to 24 digits including a nature of address and numbering plan similarly to the ***CalledPartyID***.
Note 2: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS06 or higher.
Note 3: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS07 or higher.
Note 4: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS08 or higher.
Note 5: This parameter is not populated when the CAIN_PROTOCOL_VERSION is set to V3 or higher, and is therefore not sent.
Note 6: The parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS09 or higher.
Note 7: When the CAIN_PROTOCOL_VERSION is set to V4 or higher non-standard charge numbers (CARD, AUTH, ACCT, PIN, and N00) no longer populate the ***ChargeNumber*** parameter and all standard NOAs are supported.
Note 8: The parameter is supported only on DAL, FGD, and AXXESS agencies and the parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS11 or higher.
Note 9: Refer to Volume 3, Chapter 8, "Outgoing CAIN message parameters" for information on the population of this parameter when the CAIN_PROTOCOL_VERSION is set to V5 or higher.

—end—

Once the request message is built, NetworkBuilder formats the message and sends it to the message encoder. Once encoded, the message is sent to the SCP.

SCP response processing

The following response messages are supported:

- **Analyze_Route**
- **Send_To_Resource**
- **Continue** – NetworkBuilder continues checking the remaining subscription methods for *Specific_Feature_Code*, *Customized_Dialing_Plan*, *Specific_Digit_String*, and *Office_Code*. If datafill does not enable any trigger, call processing continues through the call model.
- **Disconnect**

  Note 1: EDPs may be armed through the **Request_Report_BCM_Event** component with **Analyze_Route** and **Continue** messages.

  Note 2: A **Termination_Notification** may be requested through the **Send_Notification** component with the SCP response message.

  Note 3: An **ACG** message may be sent with the SCP response message.

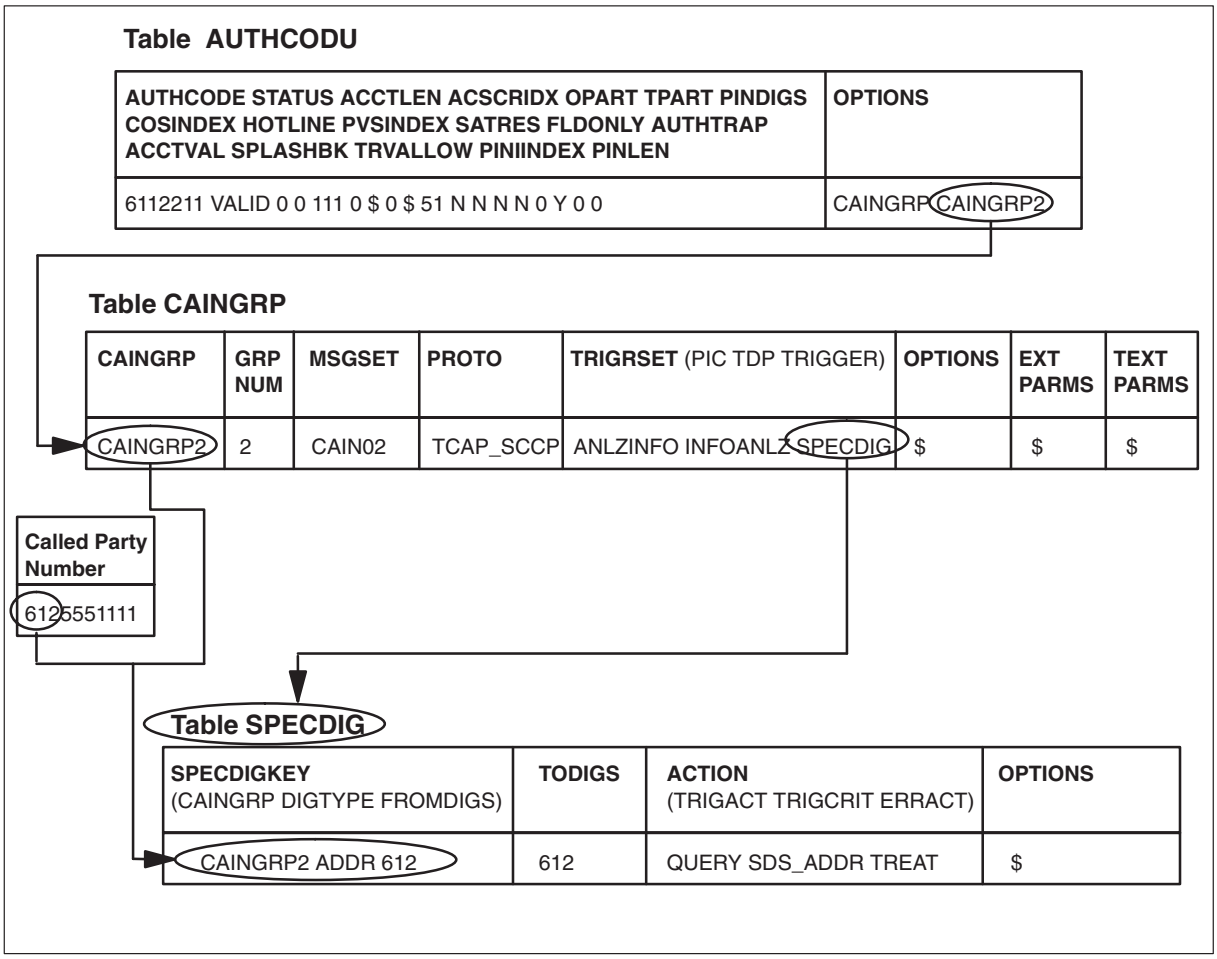
Refer to Volume 3, Chapter 10, "Incoming CAIN messages," for specific information regarding response messages; refer to Volume 3, chapter 12, "Incoming CAIN message parameters," for detailed descriptions of the response parameters; refer to Volume 3, Chapter 3, "Event processing," for EDP specific message information.

Datafill

The following figure shows how the subscription tables interact with the *Specific_Digit_String* trigger table (SPECDIG).

Specific_Digit_String trigger (continued)

Subscription-SPECDIG table interaction



Provisioning the Specific_Digit_String trigger

At the CI prompt

1 Provision the originating agent as CAIN-capable.

2 Subscribe to a CAIN group (table STDPRTCT, authcode tables, ANI tables, TRKGRP, CALLATTR, or CAINPARM).

Note: CAIN group subscription is handled differently for AXXESS agents. Refer to *UCS DMS-250 CAIN/FlexDial Interactions*.

3 Datafill CAIN group trigger subscription to the (ANLZINFO INFOANLZ SPECDIG) trigger set (table CAINGRP).

4 Enter table SPECDIG.

Specific_Digit_String trigger (continued)

5 Define the trigger criteria for a CAIN group using the following format:

>ADD specdigky todigs action options
where

specdigky is comprised of three subfields (CAINGRP, DIGTYP, FROMDIGS).
- CAINGRP is the CAIN group requiring Specific_Digit_String trigger criteria (from table CAINGRP).
- DIGTYPE is the digit type being analyzed (INFO, ADIN, ANI, XLAADDR, ADDR, CIC).
- FROMDIGS is the first number used to define the range of criteria digits.

todigs is the second number used to define the range of criteria digits.

action is a field comprised of two subfields: TRIGACT and TRIGCRIT, where
- TRIGACT is the trigger action taken when the digits are within the FROMDIGS-TODIGS range (BLOCK, IGNORE, QUERY, QUERYSCU, LEAVE_TDP, CONT_NOTRIG).

If you datafill TRIGACT as:	Go to:
BLOCK	step 8
IGNORE	step 8
QUERY	step 6
QUERYSCU	step 8
LEAVE_TDP	step 8
CONT_NOTRIG	step 8

6 Datafill the trigger criteria refinement when ACTION is QUERY. Enter the trigger criteria sent to the SCP to identify the digits (STD, SDS_INFO, SDS_ADIN, SDS_ADDR, SDS_INTL, SDS_N00, SDS_ANI, SDS_CIC).

The STD trigger criteria value indicates the corresponding *GR-1298-CORE* trigger criteria type value is sent to the SCP. A mapping of these values is listed in the table below.

Specific_Digit_String trigger (continued)

Table 5-1
Standard trigger criteria values

Criteria	Standard value
3 digit address match or less, or match on INFO, ANI, or CIC criteria	npa
4 digit address match	npaN
5 digit address match	npaNX
6 digit address match	npaNXX
7 digit address match	npaNXXX
8 digit address match	npaNXXXX
9 digit address match	npaNXXXXX
10 or more digit address match	npaNXXXXXX
Note: For international calls at the *Specific_Digit_String* trigger, it is recommended you use the SDS_INTL value in the trigger table for trigger criteria. This value maps to a trigger criteria of countryCodeNPANXXXXX to be sent to the SCP.	
—end—	

Note: The TRIGCRIT refinement value is dependent on the DIGTYPE (with the exception of the STD trigger criteria). When DIGTYPE is INFO, TRIGCRIT must be SDS_INFO or STD; when DIGTYPE is ADDR or XLAADDR, TRIGCRIT can be SDS_ADDR, SDS_INTL, SDS_N00, or STD; when DIGTYPE is ANI, TRIGCRIT must be SDS_ANI or STD; when DIGTYPE is CIC, TRIGCRIT must be SDS_CIC or STD; when DIGTYPE is ADIN, TRIGCRIT must be SDS_ADIN or STD.

7 Datafill the ERRACT refinement, where

ERRACT is the error trigger action that is taken when a fatal application error occurs (TREAT, ROUTE).

8 Datafill the options, where

options is only allowed when ACTION is QUERY. Enter up to 6 options: BUFFER, GT, T1OVFLGT, ACGOVFLGT, VERSION, or STREAM. When the ACTION is not QUERY enter $.

Sample entry: **>ADD caingrp2 addr 212 212 block $**

Sample entry: **>ADD caingrp2 addr 214 214 query sds_addr treat gt cain_addr_gt**

Specified_Digit_String trigger criteria is defined.

Associated OMs

CAINMSGS, CAINTRIG, CAINAGOM

Office_Code trigger

ATTENTION

The *Office_Code* trigger requires the CAIN0700 SOC option. Refer to Volume 5, Chapter 5, "NetworkBuilder SOC functionality," for more information.

Uses

When you provision NetworkBuilder services on the switch, the SCP can make major call processing decisions. You define the services offered on the SCP. The *Office_Code* trigger is used to implement the Local Number Portability (LNP) service.

Note: *Office_Code* triggering can be blocked based on the STS of a call by datafilling the NO_LNP option against the STS in table CAINSTS. Refer to the *UCS DMS-250 Local Number Portability Application Guide* for more information on LNP.

Supported originating agencies

The *Office_Code* trigger supports the following originating agencies:

- DAL
- FGB
- FGD
- PRI
- SS7 Inter-IMT
- AXXESS

  Note: Refer to *UCS DMS-250 CAIN/FlexDial Interactions* for more information on support for AXXESS agents.

Subscribing to the Office_Code trigger

Subscription to the *Office_Code* trigger is available on a

- SCP-returned basis
- address basis (table STDPRTCT, subtable STDPRT)
- authorization code basis (tables AUTHCODU, AUTHCDU2, AUTHCDU3, AUTHCDU4, or AUTHCDU5)
- ANI basis (table ANISCUSP, or tables ANIVAL or UNIPROF)
- agent basis (table TRKGRP or CALLATTR)

- office basis (table CAINPARM)

Note: An SCP-returned CAIN subscription group is received in **`Analyze_Route`** messages. When a reorigination occurs after an **`Analyze_Route`** is received, the SCP-returned CAIN group is available for any reoriginated call.

Note: CAIN group subscription for AXXESS agents is handled differently. Refer to *UCS DMS-250 CAIN/FlexDial Interactions*.

Trigger evaluation

NetworkBuilder checks the *Office_Code* trigger table (OFFCCODE) and evaluates the call's 10 digit translated national address [XLAADDR] against the datafilled range associated with the appropriate CAIN group.

Trigger actions

NetworkBuilder call processing supports the following actions for the *Office_Code* trigger:

- BLOCK – Prevents the call from proceeding and applies AINF treatment.
- IGNORE – NetworkBuilder continues checking the remaining subscription methods for *Specific_Feature_Code*, *Customized_Dialing_Plan*, *Specific_Digit_String*, and *Office_Code* triggers. If datafill does not enable any trigger, call processing continues through the call model.
- QUERY – The switch builds an **`Info_Analyzed`** query and sends it to the SCP. The SCP analyzes the call and assists the switch with call processing.
- LEAVE_TDP – NetworkBuilder call processing exits the ***Info_Analyzed*** TDP with no further evaluation.
- CONT_NOTRIG – NetworkBuilder call processing exits the ***Info_Analyzed*** TDP and prevents any further NetworkBuilder interaction for the call. This prevention is reset upon reorigination.

Error actions

When a fatal application occurs during an SCP query, the default error action, ROUTE, is performed.

- ROUTE – The switch discards any SCP-provided data. NetworkBuilder continues checking the remaining subscription methods for *Specific_Feature_Code*, *Customized_Dialing_Plan*, *Specific_Digit_String*, and *Office_Code*. If datafill does not enable any trigger, call processing continues through the call model.

Office_Code trigger (continued)

Options

The *Office_Code* trigger does not support any options, however the NIL value may be datafilled.

Office_Code trigger (continued)

The following figure shows how a call progresses through the call model, encounters ***Info_Analyzed*** and queries the SCP.

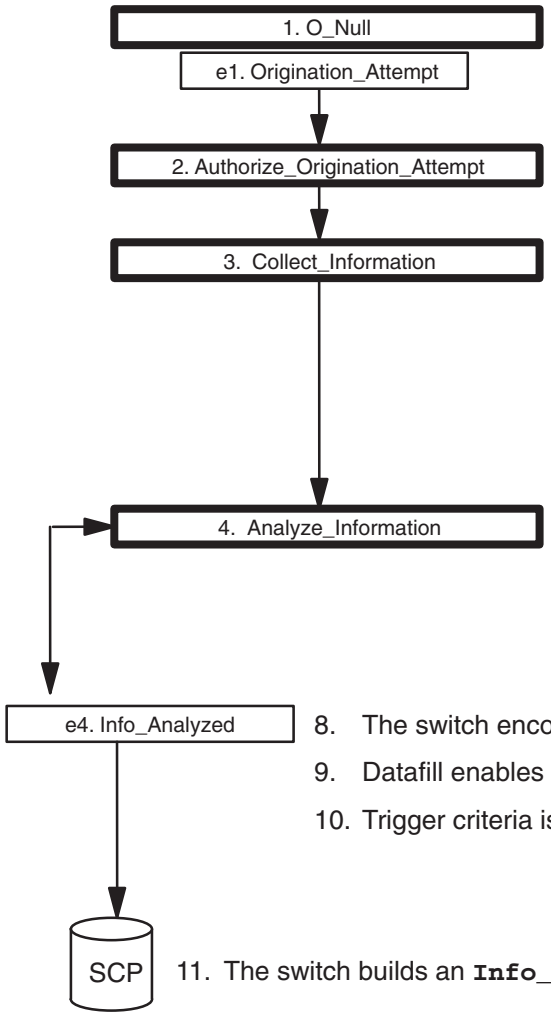
FGD 1+ call

FGD origination
Datafill:

Signaling: MF at origination
DTMF for digit collection
Dialing plan: KP + II + ANI + ST
KP + ADDR + ST
ADDR: (214)684-1111

Table OFFCCODE

OFFCCODEKY	TODIGS	TRIGCRIT	ACTION	OPTION
CAINGRP1 214	2146841111	NPA	QUERY	NIL



1. The switch detects seizure and enters **O_Null**.
2. Call processing enters **Authorize_Origination_Attempt**.
3. Call processing enters **Collect_Information**.
4. The switch collects KP + II + ANI + ST.
5. The switch screens II and ANI digits.
6. The switch collects KP + ADDR + ST.
7. The switch performs digit screening (in-switch or off-board).
8. The switch encounters ***Info_Analyzed***.
9. Datafill enables *Office_Code* trigger.
10. Trigger criteria is met.
11. The switch builds an **`Info_Analyzed`** query and sends it to the SCP.

Office_Code trigger (continued)

Info_Analyzed TDP-Request

An **Info_Analyzed** TDP-Request (query) is sent to the SCP in a query package, with a component type of Invoke_Last. The following table defines the parameters and usage requirements for the parameters the **Info_Analyzed** TDP-Request may contain. Refer to Volume 3, Chapter 8, "Outgoing CAIN message parameters," for detailed descriptions of the request parameters.

The *Office_Code* trigger provides an AIN 0.1 compliant version of the **Info_Analyzed** message which results in a limited set of supported parameters and parameter values. Refer to the *UCS DMS-250 Local Number Portability Application Guide* for more information on the AIN 0.1 compliant version of the **Info_Analyzed** message.

A limited set of parameters and values are supported in the AIN 0.1 compliant **Analyze_Route** message.

Note: Parameters for AXXESS agents are handled differently. Refer to *UCS DMS-250 CAIN/FlexDial Interactions* for more information.

Office_Code trigger (continued)

Info_Analyzed TDP-Request message parameters for Office_Code

Parameter	Usage	Definition
UserID	Required	Contains the network identity of the originating agent
BearerCapability	Required	Contains the bearer capability of the call when the message is built
CalledPartyID	Optional	Contains the translated address
LATA (Note 4)	Optional	Contains the call's Local Access and Transport Area (LATA)
TriggerCriteriaType	Optional	Contains npa, npaNXX, npaNXXX, npaNXXXX, npaNXXXXX, npaNXXXXXX, or Bellcore's lnpOfficeCode
ChargeNumber (Note 3)	Optional	Contains the billing number that would be used to populate the CDR at this point in call processing

Note 1: The LNP_PARAMETER_SET parameter in table CAINPARM is used to enhance realtime processing of LNP calls by controlling whether a complete set or minimum set of parameters are sent in an **Info_Analyzed** message. When the LNP_PARAMETER_SET parameter is set to COMPLETE_SET, the ***UserID***, ***BearerCapability***, ***CalledPartyID***, ***TriggerCriteriaType***, ***CallingPartyID***, ***Carrier***, ***ChargeNumber***, and ***ChargePartyStationType*** parameters are sent in the **Info_Analyzed** message. When the parameter is set to MINIMUM_SET, the parameters sent in **Info_Analyzed** are ***UserID***, ***BearerCapability***, ***CalledPartyID***, and ***TriggerCriteriaType***.
Note 2: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS08 or higher.
Note 3: When the CAIN_PROTOCOL_VERSION is set to V4 or higher non-standard charge numbers (CARD, AUTH, ACCT, PIN, and N00) no longer populate the ***ChargeNumber*** parameter and all standard NOAs are supported.
Note 4: The parameter is supported only on DAL, FGD, and AXXESS agencies and the parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS11 or higher.
Note 5: Refer to Volume 3, Chapter 8, "Outgoing CAIN message parameters" for information on the population of this parameter when the CAIN_PROTOCOL_VERSION is set to V5 or higher.

—continued—

Office_Code trigger (continued)

Info_Analyzed TDP-Request message parameters for Office_Code

Parameter	Usage	Definition
CallingPartyID (Note 5)	Optional	Contains one of the following (listed in order of precedence): For SS7 FGD, SS7 Inter-IMT, and AXXESS calls: Calling_Party_Address from ISUP message, when available For FGD and AXXESS calls: valid ANI (information digits are not passed in this parameter) ***Note:*** Refer to *UCS DMS-250 CAIN/FlexDial Interactions* for more information on AXXESS agents. For PRI calls: CLID when available Valid SNPA value from table TRKGRP Default SNPA value from table CAINPARM
ChargePartyStationType	Optional	Contains the information digits for the call
Carrier	Optional	Contains the dialed CIC or the CIC value (with an indication of: Selected CIC presubscribed and not input by calling party) from table TRKGRP.
JurisdictionInfo (Note 2)	Optional	Contains the originating switch's location routing number (LRN).

Note 1: The LNP_PARAMETER_SET parameter in table CAINPARM is used to enhance realtime processing of LNP calls by controlling whether a complete set or minimum set of parameters are sent in an **Info_Analyzed** message. When the LNP_PARAMETER_SET parameter is set to COMPLETE_SET, the ***UserID***, ***BearerCapability***, ***CalledPartyID***, ***TriggerCriteriaType***, ***CallingPartyID***, ***Carrier***, ***ChargeNumber***, and ***ChargePartyStationType*** parameters are sent in the **Info_Analyzed** message. When the parameter is set to MINIMUM_SET, the parameters sent in **Info_Analyzed** are ***UserID***, ***BearerCapability***, ***CalledPartyID***, and ***TriggerCriteriaType***.
Note 2: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS08 or higher.
Note 3: When the CAIN_PROTOCOL_VERSION is set to V4 or higher non-standard charge numbers (CARD, AUTH, ACCT, PIN, and N00) no longer populate the ***ChargeNumber*** parameter and all standard NOAs are supported.
Note 4: The parameter is supported only on DAL, FGD, and AXXESS agencies and the parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS11 or higher.
Note 5: Refer to Volume 3, Chapter 8, "Outgoing CAIN message parameters" for information on the population of this parameter when the CAIN_PROTOCOL_VERSION is set to V5 or higher.

—end—

Once the request message is built, NetworkBuilder formats the message and sends it to the message encoder. Once encoded, the message is sent to the SCP.

SCP response processing

The following response messages are supported:

- **`Analyze_Route`**

  Note: When a CAIN query to the SCP returns an **`Analyze_Route`** prior to the *Office_Code* trigger being reached, several call-related parameters are preserved from the original **`Analyze_Route`** message before an LNP query is performed. Refer to the *UCS DMS-250 Local Number Portability Application Guide* for more information on preserved parameters and SCP responses for LNP calls.

- **`Send_To_Resource`**
- **`Continue`** – NetworkBuilder continues checking the remaining subscription methods for *Specific_Feature_Code*, *Customized_Dialing_Plan*, *Specific_Digit_String*, and *Office_Code*. If datafill does not enable any trigger, call processing continues through the call model.
- **`Disconnect`**

Note 1: Although ***Info_Analyzed*** may be encountered for the *Office_Code* trigger when EDPs are active, EDPs may not be armed in response to an *Office_Code* query. Refer to Volume 3, Chapter 3, "Event processing," for more information on EDPs.

Note 2: An **`ACG`** message may be sent with the SCP response message.

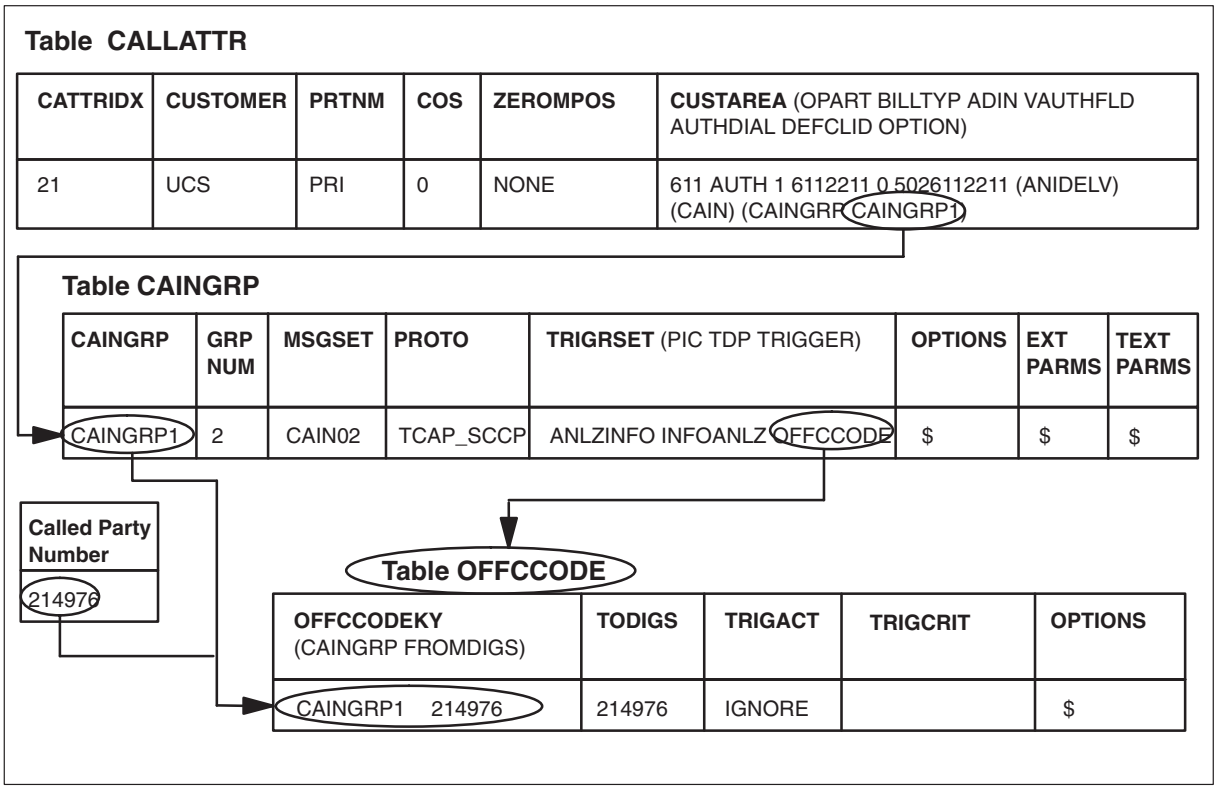
Note 3: The **`Send_Notification`** component is not allowed for *Office_Code*. A non-fatal application error occurs when the **`Send_Notification`** component is received with the SCP response message.

Datafill

The following figure shows how a subscription table interacts with the *Office_Code* trigger table (OFFCCODE).

Office_Code trigger (continued)

Subscription-OFFCCODE table interaction



Provisioning the Office_Code trigger

At the CI prompt

1 Provision the originating agent as CAIN-capapble.

2 Subscribe to a CAIN group (table STDPRTCT, authcode tables, ANI tables, TRKGRP, CALLATTR, or CAINPARM).

Note: CAIN group subscription is handled differently for AXXESS agents. Refer to *UCS DMS-250 CAIN/FlexDial Interactions*.

3 Datafill CAIN group trigger subscription to the (ANLZINFO INFOANLZ OFFCCODE) trigger set (table CAINGRP).

4 Enter table OFFCCODE.

5 Define the trigger criteria for a CAIN group by using the following format:

>ADD offccodeky todigs action options
where

offccodeky is comprised of two subfields: CAINGRP and FROMDIGS, where

CAINGRP is the CAIN group requiring Office_Code trigger criteria (from table CAINGRP).

FROMDIGS is the first number used to define the range of the collected address.

todigs is the second number used to define the range of the collected address.

action is comprised of 2 subfields: TRIGACT and TRIGCRIT, where

TRIGACT is the trigger action taken when the address is within the FROMDIGS-TODIGS range (BLOCK, IGNORE, QUERY, LEAVE_TDP, CONT_NOTRIG).

TRIGCRIT is a refinement only available when the ACTION is QUERY. Enter the trigger criteria sent to the SCP to identify the digits (NPA, NPA_NXX, NPA_NXXX, NPA_NXXXX, NPA_NXXXXX, NPA_NXXXXXX, LNP_OFCD).

6 Datafill the OPTIONS, where

options NIL is the only option available. Enter $.

Sample entry: **>ADD caingrp1 222 222 query npa $**

Sample entry: **>ADD caingrp1 222 222 block $**

Office_Code trigger criteria is defined.

Associated OMs

CAINLNP, CAINMSGS, CAINTRIG, CAINAGOM

Select_Route PIC

Call processing enters **Select_Route** once a route index is identified.

O_Abandon EDP

Call processing encounters the ***O_Abandon*** EDP during the **Collect_Information**, **Analyze_Information**, **Select_Route**, **Send_Call**, and **O_Alerting** PICs when two requirements are met:

- EDPs are active
- the calling party disconnects before the called party answers

The *O_Abandon* event is detected. For information on the *O_Abandon* event, refer to Chapter 4, "Collect_Information PIC."

Network_Busy DP

Call processing encounters the ***Network_Busy*** detection point (DP) when one of the following occurs:

- all route lists are exhausted and no additional CAIN routing parameters are available at the local switch
- the switch attempts to route the call to a DAL or AXXESS agent that has field ONNETTRK (table TRKGRP or TRKFEAT) set to N and the trunk is busy

  Note: For more information on AXXESS agents refer to *UCS DMS-250 CAIN/FlexDial Interactions*.

- the switch receives one of the following, (indicating the route is busy at the tandem switch):
  - — an ISUP Release message from a terminating SS7 agency, containing a Cause Indicator parameter (Table 6-1) indicating a network busy condition
  - — a Release message from a terminating PRI agency, containing a Cause information element (Table 6-2) indicating a network busy condition

Table 6-1
Cause values for an SS7 network busy condition

Cause name	Cause value (in hexidecimal)
No circuit available	#22
Termination resource unavailable	#23
Temporary failure	#29
Switching equipment congestion	#2A
Resource unavailable – unspecified	#2F
Channel unavailable	#2C
Network out of order	#26
User information discarded	#2B

Table 6-2
Cause values for a PRI network busy condition

Cause name	Cause value (in hexidecimal)
No channel or circuit available	#22
Network out of order	#26
Temporary failure	#29
Switching equipment congestion	#2A
User information discarded	#2B
Requested circuit/channel not available	#2C
Resource unavailable – unspecified	#2F

If EDPs are active when the ***Network_Busy*** detection point (DP) is reached, ***Network_Busy*** is encountered as an EDP. If EDPs are not active when the ***Network_Busy*** detection point (DP) is reached, ***Network_Busy*** is encountered as a TDP.

When ***Network_Busy*** is encountered as a TDP, NetworkBuilder directs the switch to check for the following:

- call subscription to a CAIN group through an SCP-determined group or through table STDPRTCT, authcode tables, ANI tables, TRKGRP, CALLATTR, or CAINPARM
- enabled triggers (triggers are enabled through table CAINGRP)

Note 1: NetworkBuilder software supports the *Network_Busy* trigger at the ***Network_Busy*** TDP.

Note 2: CAIN group subscription for AXXESS agents is handled differently. Refer to *UCS DMS-250 CAIN/FlexDial Interactions*.

When ***Network_Busy*** is encountered as an EDP and the ***Network_Busy*** EDP is active, the switch may take one of the following actions:

- The **`Request_Report_BCM_Event`** component is checked for the `networkBusyActions` extension parameter to determine the action to take.
- If the **`Request_Report_BCM_Event`** component was received without a `networkBusyActions` extension parameter, datafill for the active CAIN group in table CAINXDFT is checked for the `networkBusyActions` extension parameter to determine the action to take.
- If the **`Request_Report_BCM_Event`** component was received without a `networkBusyActions` extension parameter and the `networkBusyActions` extension parameter is not datafilled for the active CAIN group in table CAINXDFT, a **`Network_Busy`** EDP-Request message is sent to the SCP. The next event list (NEL) is cleared and the conversation transaction is maintained. From this point forward, call processing proceeds as if a **`Network_Busy`** TDP-Request message were being sent.

Note: If EDPs are active when the ***Network_Busy*** detection point (DP) is reached, but the ***Network_Busy*** EDP is not active, the call is treated as if an IGNORE action had occurred, and the call continues through the call model.

Terminology

The switch begins attempting to route the call at the **Select_Route** PIC (refer to Volume 3, Chapter 10, “Incoming CAIN messages,” for more information). The following terms are used to define CAIN routing:

- **Route indexes** are determined by the data provided in an **`Analyze_Route`** response or through normal translations. The index points to the appropriate table that identifies a routing list.

- **Route lists** contain the routes to be attempted by the switch when establishing the call. Route lists are provisioned in tables, such as: TANDMRTE, TERMRTE, HNPACONT, FNPACONT, CTRTE, and OFRT. Each route list can contain multiple routes.
- **Routes** typically consists of a route selector, connection type, and common language location identifier (CLLI).
- **CAIN routing parameters** are provided in an **`Analyze_Route`** response. The parameters are: ***`PrimaryTrunkGroup`***, ***`AlternateTrunkGroup`***, ***`SecondAlternateTrunkGroup`***, ***`Carrier`***, ***`AlternateCarrier`***, ***`SecondAlternateCarrier`***,***`CalledPartyID`***, and the ***`GenericAddressList`*** parameter's `OverflowRoutingNo`.

  Note 1: Parameters ***`PrimaryTrunkGroup`***, ***`AlternateTrunkGroup`***, and ***`SecondAlternateTrunkGroup`*** are used to identify direct termination route indexes into tables TANDMRTE and TERMRTE.

  Note 2: Standard routing parameters, ***`Carrier`***, ***`AlternateCarrier`***, and ***`SecondAlternateCarrier`*** are used to identify route indexes.

  Note 3: Standard routing parameters, ***`CalledPartyID`*** and/or the `servTranslationScheme` (or `univIdx` for SS7 Global-IMTs), and ***`GenericAddressList`*** parameter's `OverflowRoutingNo` require in-switch translations to derive the route index. One route index is calculated for each parameter.
- **Serving translation scheme (STS)** extension parameters are provided in an **`Analyze_Route`** response along with CAIN routing parameters. Each CAIN routing parameter has an associated STS extension parameter. Default STS extension parameters are also datafilled in table CAINXDFT to be used when the SCP fails to return the associated STS extension parameter with the CAIN routing parameter. Additionally, the original STS derived prior to the query message may be used when the STS extension parameter is not returned from the SCP and is not datafilled in table CAINXDFT. Figure 2-19 illustrates the order of precedence used to obtain the STS for each CAIN routing parameter:

Figure 6-1
Precedence order for STS extension parameters

Route Parameters Returned by the SCP \ STS Extension Parameters	PRISTS	ALTSTS	SALTSTS	STS	OVLFSTS	table CAINXDFT PRISTS	table CAINXDFT ALTSTS	table CAINXDFT SALTSTS	table CAINXDFT STS	table CAINXDFT OVLFSTS	Pre Query STS
PrimaryTrunkGroup	1			3		2			4		5
AlternateTrunkGroup		1		3			2		4		5
SecondAlternateTrunkGroup			1	3				2	4		5
CalledPartyID				1					2		3
OverflowRoutingNo				3	1				4	2	5

LEGEND

PRISTS (primaryTrunkGroupSTS)	– STS to be used with the ***PrimaryTrunkGroup*** parameter
ALTSTS (alternateTrunkGroupSTS)	– STS to be used with the ***AlternateTrunkGroup*** parameter
SALTSTS (secondAlternateTrunkGroupSTS)	– STS to be used with the ***SecondAlternateTrunkGroup*** parameter
STS (servTranslationScheme)	– STS to be used with the ***CalledPartyID*** parameter
OVFLSTS (overflowRoutingNo)	– STS to be used with the OverflowRoutingNo parameter

Note: The univIdx extension parameter is used to specify the translations scheme for SS7 Global-IMT agents.

- **Standard route advance** – The switch attempts the first route in route list, and when unable to terminate using the route, attempts to establish the call using the next route in the route list.
- **CAIN routing parameter advance** – The switch attempts to route according to data provided in the next untried CAIN routing parameter. When unable to establish the connection using the data, call processing advances to the next NetworkBuilder parameter to derive a new route list and attempts to route the call.

Network_Busy trigger/event

ATTENTION

The *Network_Busy* trigger requires the CAIN0506 SOC option. The *Network_Busy* event requires the CAIN0602 SOC option. Refer to Volume 5, Chapter 5, "NetworkBuilder SOC functionality," for more information.

Uses

When you provision NetworkBuilder services on the switch, the SCP can make major call processing decisions. You define the services offered on the SCP. Some examples of services that can be developed on the SCP for the *Network_Busy* trigger/event are:

- Network Forwarding
- Network Queuing
- Rerouting due to network congestion

Supported originating agencies

The *Network_Busy* trigger/event supports the following originating agencies:

- DAL
- FGD
- PRI
- SS7 Inter-IMT
- SS7 Global-IMT
- AXXESS

  Note: Refer to *UCS DMS-250 CAIN/FlexDial Interactions* for more information on support for AXXESS agents.

Subscribing to the Network_Busy trigger

Subscription to the *Network_Busy* trigger is available on a

- SCP-determined basis
- address basis (table STDPRTCT)
- authorization code basis (tables AUTHCODU, AUTHCDU2, AUTHCDU3, AUTHCDU4, or AUTHCDU5)
- ANI basis (table ANISCUSP, or tables ANIVAL and UNIPROF)
- agent basis (table TRKGRP or CALLATTR)

- office basis (table CAINPARM)

Note: CAIN group subscription for AXXESS agents is handled differently, refer to *UCS DMS-250 CAIN/FlexDial Interactions*.

Trigger/Event evaluation

NetworkBuilder checks for CAIN routing parameters and standard routing parameters to determine the routing status. Following are routing status possibilities for a *Network_Busy* trigger/event:

— RTEAVAIL – indicates that additional routing information is available to the switch for call completion. When the switch fails to seize an outgoing trunk it automatically attempts to route using the next route available without trigger or event evaluation.

— RTESDONE – indicates that all routes have been attempted and no additional routes are available to the switch for call completion. RTESDONE is also encountered when the switch fails to seize an outgoing trunk at the querying switch.

Note: The RTESDONE routing criteria is evaluated when either of the following occur:

– There is no matching TERMRTE_GNCT tuple/extension parameter for which the action can be performed (for example, when the provisioned action is NEXTCNRTE but there are only standard route list choices remaining).

– There are no TERMRTE_GNCT tuples provisioned for any of the CAIN groups available for this trigger, or the TERMRTE_GNCT action is IGNORE. This type of RTESDONE is evaluated when there are no more CAIN routing choices to evaluate.

— TERMRTE_GNCT – indicates that a release cause value of TERM_RESOURCE_UNAVAILABLE was received on a terminating SS7 agency and there is additional route information available. This release cause is set when attempting to terminate to a TERMRTE route choice when all trunks are busy.

Note: Provisioning TERMRTE_GNCT routing criteria with an action of NEXTCNRTE implies that RTESDONE will be evaluated when all CAIN routing parameters have been attempted.

Network_Busy trigger/event (continued)

Note: Refer to Volume 3, Chapter 10, "Incoming CAIN messages," for more information on routing.

ATTENTION

When evaluating multiple CAIN groups for this trigger, if some are provisioned with an action of NEXTRTE and some are provisioned with an action of NEXTCNRTE, the evaluation of RTESDONE will not occur until all routes (standard and NetworkBuilder) have been attempted.

Trigger evaluation

1 The current routing status is compared to the datafilled trigger criteria in table NETBUSY.

2 Identified digits are checked against the datafilled range associated with the appropriate CAIN group.

Note 1: The digit type (information, address, ANI, XLAADDR, CIC) are identified through datafill within the NETBUSY table.

Note 2: Call processing cannot meet digit criteria for a digit type of ADDR if the call queried at *Off_Hook_Immediate* and the SCP responded with an **`Analyze_Route`**.

ATTENTION

When you provision the digit type as ADDR, the actual digits sent in ***`CalledPartyID`***may not be the same as the matching criteria used by the switch to evaluate the trigger. This situation is most likely to occur when the SCP returns an **`Analyze_Route`** message. Triggering occurs based on the originally translated number, but the query message contains the SCP-returned value. In this case, a triggering loop may occur until the MAX_NUM_SERIAL_TRIGGERS is met.

Trigger actions

NetworkBuilder call processing supports the following actions for the Network_Busy trigger:

- IGNORE – NetworkBuilder continues checking the remaining subscription methods for *Network_Busy*. If datafill does not enable any trigger, call processing continues through the call model.

Network_Busy trigger/event (continued)

- QUERY – The switch builds a query message and sends the message to the SCP. You can provision QUERY whether or not all routing options were exhausted. The SCP analyzes the call and assists the switch with call processing.
- NEXTRTE – Call processing routes using the next routing option according to the following precedences:
  - — 1. The switch route advances and attempts the next route available in the route list.
  - — 2. The switch performs a CAIN routing parameter advance and attempts to route accordingly.

  Note: Provision NEXTRTE only when the criteria is datafilled as RTEAVAIL or TERMRTE_GNCT.

- NEXTCNRTE – Routing is performed using the next CAIN route choice.

  Note 1: Provision NEXTCNRTE only when the criteria is datafilled as TERMRTE_GNCT.

  Note 2: This action is only considered valid when there are CAIN routing parameters remaining.
- LEAVE_TDP – NetworkBuilder call processing exits the ***Network_Busy*** TDP with no further evaluation.
- CONT_NOTRIG – NetworkBuilder call processing exits the ***Network_Busy*** TDP and prevents any further NetworkBuilder interaction for the call. This prevention is reset upon reorigination.

Event evaluation

When the ***Network_Busy*** EDP is encountered and active there are several actions the switch may take:

- If a `networkBusyActions` extension parameter was received with the **`Request_Report_BCM_Event`** component, the action taken is based on the extension parameter field corresponding to the current routing status.
- If a `networkBusyActions` extension parameter was not received with the **`Request_Report_BCM_Event`** component, but the active CAIN group has the `networkBusyActions` extension parameter datafilled in table CAINXDFT, the action taken is based on the current routing status.

Network_Busy trigger/event (continued)

- If the **`Request_Report_BCM_Event`** component was received without a `networkBusyActions` extension parameter and `networkBusyActions` extension parameter is not datafilled for the active CAIN group in table CAINXDFT, a **`Network_Busy`** EDP-Request message is sent to the SCP. The next event list (NEL) is cleared and the conversation transaction is maintained. From this point forward, call processing proceeds as if a **`Network_Busy`** TDP-Request message were being sent.

Event actions

NetworkBuilder call processing supports the following actions for the Network_Busy event:

- IGNORE – NetworkBuilder call processing exits the ***Network_Busy*** DP with no further evaluation. All active EDPs remain active.
- REQUEST – The switch discards the next event list (NEL), builds a **`Network_Busy`** EDP-Request message, and sends the message to the SCP. The SCP analyzes the call and assists the switch with call processing.
- NEXTRTE – Call processing routes using the next routing option according to the following precedences:
  - — 1. The switch route advances and attempts the next route available in the route list.
  - — 2. The switch performs a CAIN routing parameter advance and attempts to route accordingly.

  All active EDPs remain active.

  Note: NEXTRTE is supported only when the routing status is RTEAVAIL or TERMRTE_GNCT.

- NEXTCNRTE – Routing is performed using the next CAIN route choice. All active EDPs remain active.

  Note 1: NEXTCNRTE is supported only when the routing status is TERMRTE_GNCT.

  Note 2: This action is only considered valid when there are CAIN routing parameters remaining.

Error actions

Error actions are not identified for the *Network_Busy* trigger or event. The switch applies AINF treatment for fatal application errors.

Network_Busy trigger/event (continued)

Options

The *Network_Busy* trigger supports the following options:

- BUFFER – to activate digit buffering while the SCP is queried.
- GT – to identify the global title used to identify the SCP handling the query.
- T1OVFLGT – to identify the specific SCP to query on T1 overflow.
- ACGOVFLGT – to identify the global title to use for requerying when a query is blocked by an ACG control
- VERSION – controls the CAIN protocol version for outgoing messages

  Note: The value provisioned in the VERSION option overrides the version provisioned in the CAIN_PROTOCOL_VERSION parameter of table CAINPARM on a transaction-by-transaction basis. Additionally, the VERSION option applies to both the outgoing and incoming messages within the same transaction. (Subsequent queries will use the setting provisioned in the CAIN_PROTOCOL_VERSION or provisioned setting of the VERSION option in the trigger table performing the subsequent query.)

- STREAM – to control protocol stream on a per-trigger tuple basis. The value of the STREAM option controls the set of parameters that are sent in NetworkBuilder messages.

The *Network_Busy* event supports buffering through the `edpBuffer` extension parameter. Refer to Volume 3, Chapter 3, “Event processing” for more information.

Route advancing

The following figure describes how NetworkBuilder route advances using the *Network_Busy* trigger table (NETBUSY). Route advancing with the *Network_Busy* event is handled the same as for the *Network_Busy* trigger, except that the next event list (NEL) is checked rather than NETBUSY trigger table. See Volume 3, Chapter 10, “Incoming CAIN messages,” for more information on direct termination routing.

Network_Busy trigger/event (continued)

Route advance with the Network_Busy trigger

Signaling: DTMF
Dialing plan: 0 + ADDRESS
Address dialed: (214) 555-1111

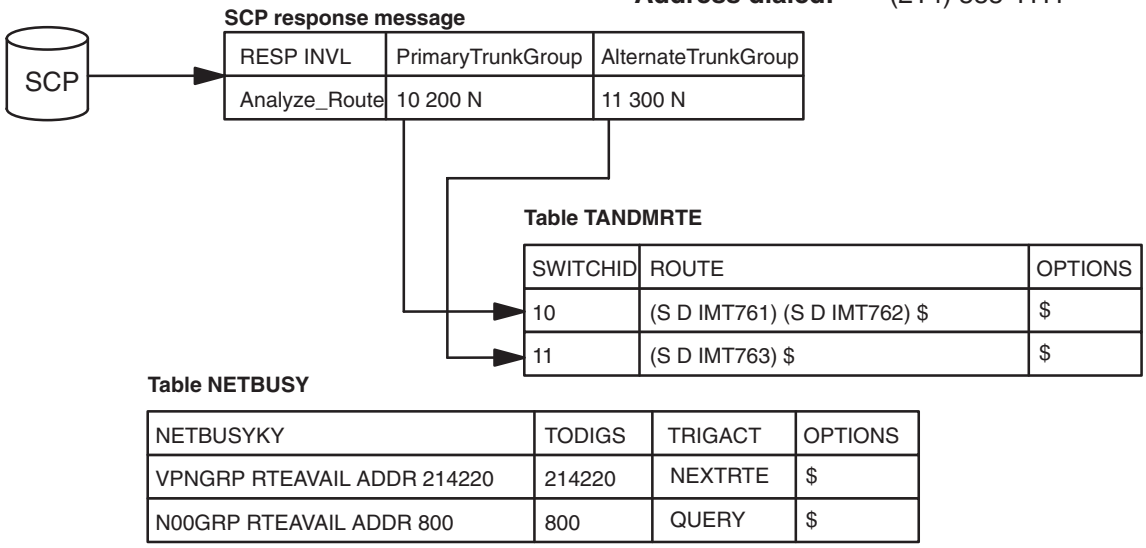


Table NETBUSY

NETBUSYKY	TODIGS	TRIGACT	OPTIONS
VPNGRP RTEAVAIL ADDR 214220	214220	NEXTRTE	$
N00GRP RTEAVAIL ADDR 800	800	QUERY	$

1 The switch receives an **`Analyze_Route`** message containing the ***`PrimaryTrunkGroup`*** and ***`AlternateTrunkGroup`*** parameters.

2 The switch attempts to complete the call using the route index identified by the ***`PrimaryTrunkGroup`*** parameter.

3 The switch seizes the trunk (IMT761) and sends an IAM message.

4 The switch receives a release message with a cause value indicating a Network Busy condition.

5 The switch checks to see if the call subscribes to NetworkBuilder services and if the *Network_Busy* trigger is enabled. If subscribed and enabled, the switch accesses the NETBUSY table.

- If CAINGRP=VPNGRP and digits match, the switch route advances and attempts routing using the next trunk (IMT762). If routing over IMT762 is attempted and a Network Busy condition is indicated, the next CAIN routing parameter is tried.
- If CAINGRP=N00GRP and digits match, the switch suspends call processing and sends a **`Network_Busy`** TDP-Request message to the SCP.

6 If the *Network_Busy* trigger is not enabled, the call is processed normally as it would without Networkbuilder service interaction.

Network_Busy trigger/event (continued)

The following figure and steps provide an additional example of route advancing with *Network_Busy*.

1 A NetworkBuilder-provisioned call triggers at Switch 11 and queries the SCP.

2 The SCP returns an **Analyze_Route** response containing ***PrimaryTrunkGroup*** and ***CalledPartyID***.

3 Switch 11 decodes the ***PrimaryTrunkGroup*** in order to perform direct termination routing. The switch compares the SWID returned in the **Analyze_Route** response (77) to the current SWID (11). Since the numbers don't match, call processing accesses tuple 77 of table TANDMRTE. Call processing routes to the first route choice IMT55A and delivers the required SWID (77) in an IAM.

4 Switch 55 receives the IAM and compares the current SWID (55) to the received SWID (77). Since the SWIDs don't match, call processing accesses tuple 77 of table TANDMRTE. Call processing routes to 77A and delivers the required SWID (77) in an IAM.

5 Switch 77 receives the IAM and compares the current SWID (77) to the received SWID (77). Now that the SWIDs match, call processing accesses tuple 24 (the trunk number received in the **Analyze_Route** message) and routes to DAL77CUST.

However, DAL77CUST, currently, has no idle trunk members; also field ONNETTRK in table TRKGRP is set to N, indicating that DAL77CUST terminates to a network instead of a user. Switch 77 sets GNCT treatment and sends a REL message with a NO_CIRCUIT_AVAILABLE cause value to the originating switch (Switch 11 through Switch 55).

If table TERMRTE had contained more than one route choice, call processing would have attempted to route the call until all route choices were attempted.

6 Switch 11 receives the REL message and tuple GRP1 of table NETBUSY is accessed and the RTEAVAIL criteria is met (since call processing has not attempted every route in the route list obtained from table TANDMRTE, plus the additional CAIN routing parameter). Call processing performs the NEXTRTE action and routes to the next route list choice IMT22 and delivers the required SWID (77) in an IAM.

7 Switch 22 receives the IAM and compares the current SWID (22) to the received SWID (77). Since the SWIDs don't match, call processing accesses tuple 77 of table TANDMRTE. Call processing routes to IMT33 and delivers the required SWID (77) in the Generic Digits Parameter of the outgoing IAM.

Network_Busy trigger/event (continued)

8 Switch 33 receives the IAM and compares the current SWID (33) to the received SWID (77). Since the SWIDs don't match, call processing accesses tuple 77 of table TANDMRTE. Call processing routes to IMT77B and delivers the required SWID (77) in an IAM.

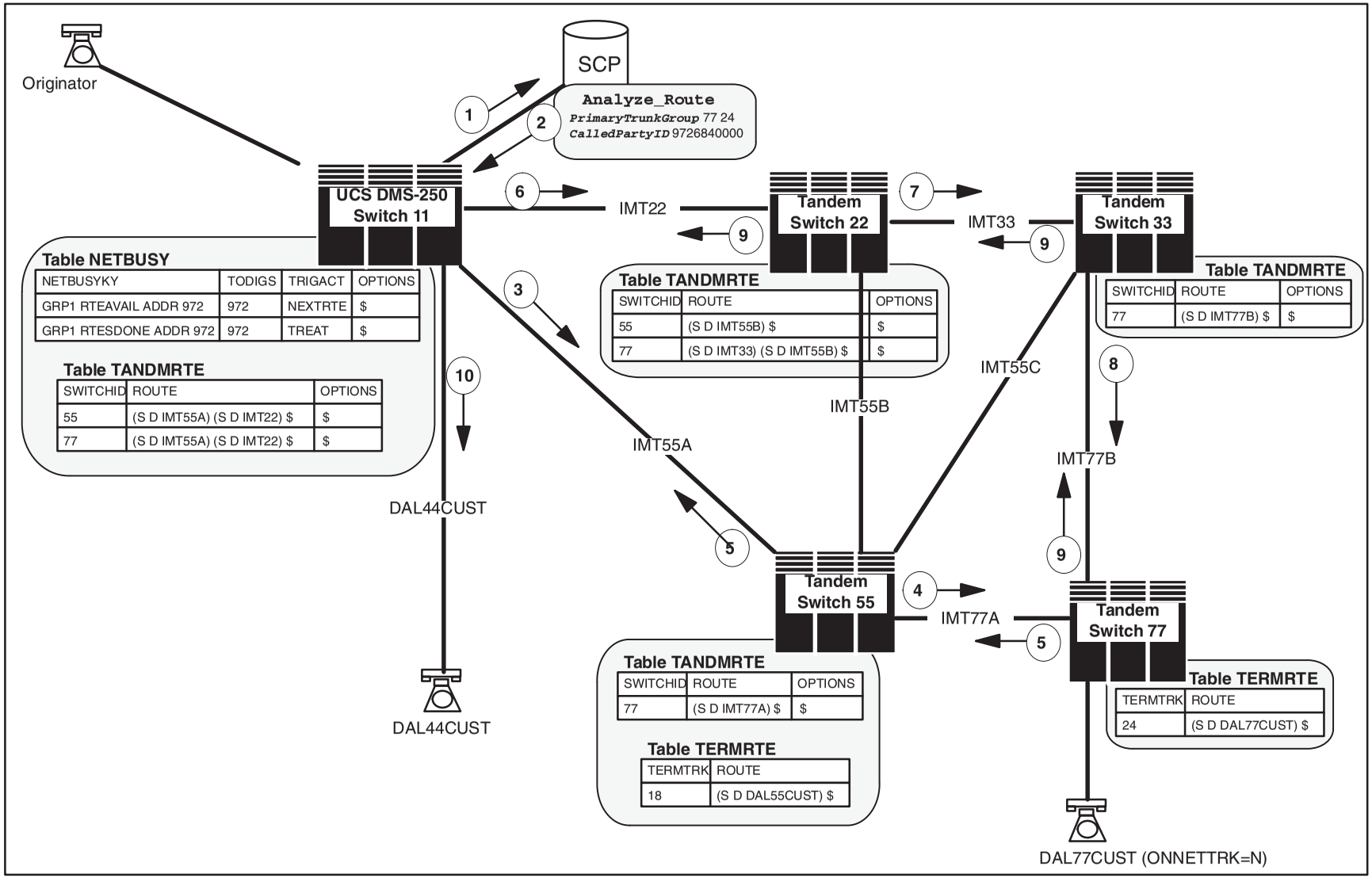
9 Switch 77 receives the IAM and compares the current SWID (77) to the received SWID (77). Now that the SWIDs match, call processing accesses tuple 24 (the trunk number received in the **`Analyze_Route`** message) and routes to DAL77CUST.

However, there are still no idle trunk members. Switch 77 sets GNCT treatment and sends a REL message with a NO_CIRCUIT_AVAILABLE cause value to the originating switch (Switch 11 through Switch 33 and Switch 22).

Note: If table TERMRTE had contained more than one route choice, call processing would have attempted to route the call until all route choices were attempted.

10 Switch 11 receives the REL message and the RTEAVAIL criteria is met, tuple GRP1 of table NETBUSY is accessed and the RTEAVAIL criteria is met (since all CAIN routing parameters have not been attempted). Call processing performs the NEXTRTE action and routes the call based on the ***`CalledPartyID`***. Normal call processing routes the call to trunk DAL44CUST off Switch 11.

Network_Busy trigger/event (continued)



Network_Busy trigger/event (continued)

Network_Busy TDP-Request

A **Network_Busy** TDP-Request (query) is sent to the SCP in a query package, with a component type of Invoke_Last. The following table defines the parameters and usage requirements for the parameters the **Network_Busy** TDP-Request may contain. Refer to Volume 3, Chapter 8, "Outgoing CAIN message parameters," for detailed descriptions of the request parameters.

Note: Parameters and extension parameters for AXXESS agents are handled differently. Refer to *UCS DMS-250 CAIN/FlexDial Interactions* for more information.

Network_Busy TDP-Request message parameters

Parameter	Usage	Definition
UserID	Required	Contains the network identity of the originating agent
BearerCapability	Required	Contains the bearer capability of the call when the message is built
ChargeNumber (Note 5)	Optional	Contains the billing number that would be used to populate the CDR at this point in call processing
Lata (Note 6)	Optional	Contains the call's Local Access and Transport Area (LATA)
TriggerCriteriaType	Optional	Contains networkBusy

Note 1: The `universalAccess` extension parameter is included in the query (a flat 10 digits) when parameter CAIN_PROTOCOL_VERSION is V0. When CAIN_PROTOCOL_VERSION is set to V1 or higher, the parameter contains up to 24 digits including a nature of address and numbering plan similarly to the ***CalledPartyID***.
Note 2: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS07 or higher.
Note 3: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS08 or higher.
Note 4: The parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS09 or higher.
Note 5: When the CAIN_PROTOCOL_VERSION is set to V4 or higher non-standard charge numbers (CARD, AUTH, ACCT, PIN, and N00) no longer populate the ***ChargeNumber*** parameter and all standard NOAs are supported.
Note 6: The parameter is supported only on DAL, FGD, and AXXESS agencies and the parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS11 or higher.
Note 7: Refer to Volume 3, Chapter 8, "Outgoing CAIN message parameters" for information on the population of this parameter when the CAIN_PROTOCOL_VERSION is set to V5 or higher.

—continued—

Network_Busy trigger/event (continued)

Network_Busy TDP-Request message parameters (continued)

Parameter	Usage	Definition
CallingPartyID (Note 7)	Optional	Contains the one of the following (listed in order of precedence): For SS7 FGD, SS7 Inter-IMT, and SS7 Global-IMT calls: Calling_Party_Address from ISUP message, when available For PRI calls: CLID when available For FGD and AXXESS calls: valid ANI (information digits are not passed in this parameter) ***Note:*** Refer to *UCS DMS-250 CAIN/FlexDial Interactions* for more information on AXXESS agents. Valid SNPA value from table TRKGRP Default SNPA value from table CAINPARM
ChargePartyStationType (Note 2)	Optional	Contains the information digits for the call
CalledPartyID	Optional	Contains the translated address
Carrier	Optional	Contains the dialed CIC or the CIC value (with an indication of: Selected CIC presubscribed and not input by calling party) from table TRKGRP1

Note 1: The `universalAccess` extension parameter is included in the query (a flat 10 digits) when parameter CAIN_PROTOCOL_VERSION is V0. When CAIN_PROTOCOL_VERSION is set to V1 or higher, the parameter contains up to 24 digits including a nature of address and numbering plan similarly to the ***CalledPartyID***.
Note 2: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS07 or higher.
Note 3: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS08 or higher.
Note 4: The parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS09 or higher.
Note 5: When the CAIN_PROTOCOL_VERSION is set to V4 or higher non-standard charge numbers (CARD, AUTH, ACCT, PIN, and N00) no longer populate the ***ChargeNumber*** parameter and all standard NOAs are supported.
Note 6: The parameter is supported only on DAL, FGD, and AXXESS agencies and the parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS11 or higher.
Note 7: Refer to Volume 3, Chapter 8, “Outgoing CAIN message parameters” for information on the population of this parameter when the CAIN_PROTOCOL_VERSION is set to V5 or higher.

—continued—

Network_Busy trigger/event (continued)

Network_Busy TDP-Request message parameters (continued)

Parameter	Usage	Definition
ACGEncountered	Optional	Contains the ACG control encountered when the ACG control expires and a query is allowed to be sent
ExtensionParameter (Note 2)	Optional	Extension parameters require the CAIN0200 SOC option
busyRoute	Optional	Contains the number of routing attempts for a call at the *Network_Busy* trigger. When an outpulse number is available, it is included as the routing number. If the call was routed using a trunk group and SWID from the ***PrimaryTrunkGroup***, ***AlternateTrunkGroup***, or ***SecondAlternateTrunkGroup*** parameter, the route index is included.
universalAccess (Note 1)	Optional	Contains the universal access number dialed by the caller to obtain switch dial tone.
cainGroup	Optional	Contains the group number (field GRPNUM, table CAINGRP) for the CAIN group associated with the trigger when field EXTPARM in table CAINGRP contains CAINGRP
origTrunkInfo	Optional	Contains the originating trunk group number, trunk type, and trunk member number when field EXTPARM in table CAINGRP contains ORGTINFO
termTrunkInfo	Optional	Contains the terminating trunk group number, trunk type, and trunk member number

Note 1: The universalAccess extension parameter is included in the query (a flat 10 digits) when parameter CAIN_PROTOCOL_VERSION is V0. When CAIN_PROTOCOL_VERSION is set to V1 or higher, the parameter contains up to 24 digits including a nature of address and numbering plan similarly to the ***CalledPartyID***.
Note 2: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS07 or higher.
Note 3: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS08 or higher.
Note 4: The parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS09 or higher.
Note 5: When the CAIN_PROTOCOL_VERSION is set to V4 or higher non-standard charge numbers (CARD, AUTH, ACCT, PIN, and N00) no longer populate the ***ChargeNumber*** parameter and all standard NOAs are supported.
Note 6: The parameter is supported only on DAL, FGD, and AXXESS agencies and the parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS11 or higher.
Note 7: Refer to Volume 3, Chapter 8, “Outgoing CAIN message parameters” for information on the population of this parameter when the CAIN_PROTOCOL_VERSION is set to V5 or higher.

—continued—

Network_Busy trigger/event (continued)

Network_Busy TDP-Request message parameters (continued)

Parameter	Usage	Definition
`netinfo` (Note 2)	Optional	Contains external network ID, network customer group ID, and network class of service.
`t1Overflow`	Optional	Indicates to the SCP that an overflow occurred on the query to the initial SCP.
`subscriptionInfo` (Note 3)	Optional	Contains the digit type the switch triggered on and which subscription method was in use when the query occurred
`jurisdictionInfo` (Note 3)	Optional	Contains the originating switch's LRN.
`collectedAddress` (Note 3)	Optional	Contains the address collected from the incoming agent message, through subscriber dialing, through datafilled hotline digits, or through the *O_Feature_Requested* trigger.
`switchID` (Note 4)	Optional	Contains the switch ID
`billingNumber` (Note 4)	Optional	Contains the non-standard charge number
`acgRequery`	Optional	When present, indicates to the SCP that the current query is a re-query message resulting from the original query being blocked by an ACG control
BusyCause	Optional	Contains the cause value received from the SS7 or PRI agent or if the route list is exhausted, contains "noCircuitAvailable"

Note 1: The `universalAccess` extension parameter is included in the query (a flat 10 digits) when parameter CAIN_PROTOCOL_VERSION is V0. When CAIN_PROTOCOL_VERSION is set to V1 or higher, the parameter contains up to 24 digits including a nature of address and numbering plan similarly to the ***CalledPartyID***.
Note 2: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS07 or higher.
Note 3: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS08 or higher.
Note 4: The parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS09 or higher.
Note 5: When the CAIN_PROTOCOL_VERSION is set to V4 or higher non-standard charge numbers (CARD, AUTH, ACCT, PIN, and N00) no longer populate the ***ChargeNumber*** parameter and all standard NOAs are supported.
Note 6: The parameter is supported only on DAL, FGD, and AXXESS agencies and the parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS11 or higher.
Note 7: Refer to Volume 3, Chapter 8, "Outgoing CAIN message parameters" for information on the population of this parameter when the CAIN_PROTOCOL_VERSION is set to V5 or higher.

—end—

Network_Busy trigger/event (continued)

Once the request message is built, NetworkBuilder formats the message and sends it to the message encoder. Once encoded, the message is sent to the SCP.

Network_Busy EDP-Request

A **Network_Busy** EDP-Request is sent to the SCP in a conversation package with a component type of Invoke_Last. The following table defines the parameters and usage requirements for the parameters the **Network_Busy** EDP-Request may contain. Refer to Volume 3, Chapter 8, "Outgoing CAIN message parameters," for detailed descriptions of the parameters. Refer to Volume 3, Chapter 3, "Event processing," for detailed descriptions of EDPs.

Network_Busy EDP-Request message parameters

Parameter	Usage	Definition
UserID	Required	Contains the network identity of the originating agent
BearerCapability	Required	Contains the bearer capability of the call when the message is built
ExtensionParameter	Optional	***ExtensionParameter***s require the CAIN0200 SOC option
busyRoute	Optional	Contains the number of routing attempts for a call at the *Network_Busy* event. When an outpulse number is available, it is included as the routing number. If the call was routed using a trunk group and SWID from the ***PrimaryTrunkGroup***, ***AlternateTrunkGroup***, or ***SecondAlternateTrunkGroup*** parameter, the route index is included.
termTrunkInfo	Optional	Contains the terminating trunk group number, trunk type, and trunk member number
NotificationIndicator	Optional	Contains the value of FALSE to identify the message type as Request
BusyCause	Optional	Contains the cause value received from the SS7 or PRI agent or if the route list is exhausted, contains noCircuitAvailable.The CAIN office parameter, RESTRICT_NETBUSY_BUSYCAUSE, in table CAINPARM controls the sending of this parameter. When set to N, the parameter is included in the message. When set to Y, the parameter is not included in the message.
		—end—

Once the request message is built, NetworkBuilder formats the message and sends it to the message encoder. Once encoded, the message is sent to the SCP.

SCP response processing

The following response messages are supported:

- **`Analyze_Route`**
- **`Send_To_Resource`**
- **`Disconnect`**

Note 1: In-switch cause mapping is not performed when the switch receives a **`Disconnect`** message.

Note 2: EDPs may be armed through the **`Request_Report_BCM_Event`** component with the **`Analyze_Route`** message in response to the **`Network_Busy`** TDP- or EDP-Request message.

Note 3: A **`Termination_Notification`** may be requested through the **`Send_Notification`** component with the SCP response message.

Note 4: An **`ACG`** message may be sent with the SCP response message.

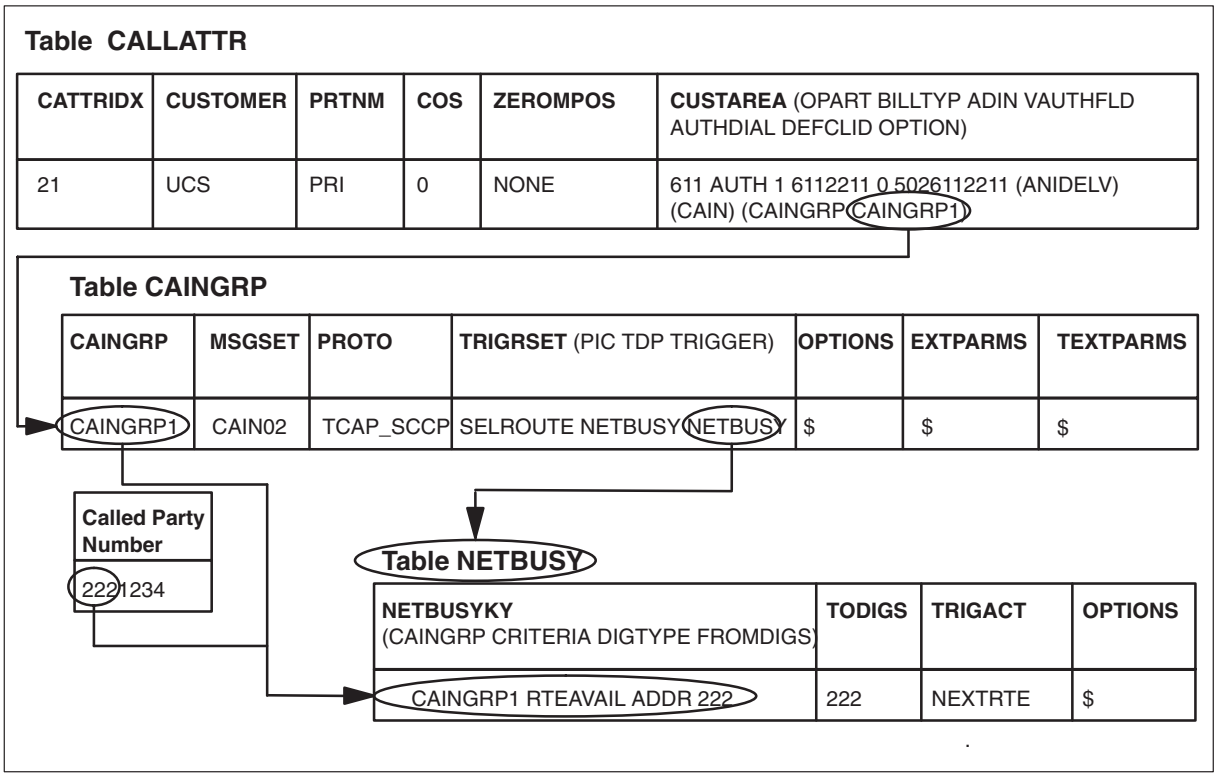
Refer to Volume 3, Chapter 10, "Incoming CAIN messages," for specific information regarding response messages; refer to Volume 3, Chapter 12, "Incoming CAIN message parameters," for detailed descriptions of the response parameters; refer to Volume 3, Chapter 3, "Event processing," for EDP specific message information.

Datafill

The following figure shows how a subscription table interacts with the *Network_Busy* trigger table (NETBUSY).

Network_Busy trigger/event (continued)

Subscription-NETBUSY table interaction



Provisioning the Network_Busy trigger

At the CI prompt

1 Provision the originating agent as CAIN-capable.

2 Subscribe to a CAIN group (table STDPRTCT, authcode tables, ANI tables, TRKGRP, CALLATTR, or CAINPARM).

Note: CAIN group subscription is handled differently for AXXESS agents. Refer to *UCS DMS-250 CAIN/FlexDial Interactions*.

3 Datafill CAIN group trigger subscription to the (SELROUTE NETBUSY NETBUSY) trigger set (table CAINGRP).

4 Enter table NETBUSY.

5 Define the trigger criteria for a CAIN group by using the following format:

>ADD netbusyky todigs trigact options
where

netbusyky is comprised of four subfields: CAINGRP, CRITERIA, DIGTYPE, FROMDIGS, where

Network_Busy trigger/event (end)

	CAINGRP	is the CAIN group that enables the trigger (from table CAINGRP).
	CRITERIA	is the routing status (RTEAVAIL, RTESDONE, TERMRTE_GNCT).
	DIGTYPE	is the digit type being analyzed (INFO, ANI, XLAADDR, ADDR, CIC).
	FROMDIGS	is the first number used to define the digit range (0 to 9, *, #).
todigs		is the second number used to define the digit range (0 to 9, *, #).
trigact		is is the trigger action taken when the address is within the FROMDIGS-TODIGS range (IGNORE, QUERY, NEXTRTE, NEXTCNRTE, LEAVE_TDP, CONT_NOTRIG).
options		is only allowed when ACTION is QUERY. Enter up to 6 options: BUFFER, GT, T1OVFLGT, ACGOVFLGT, VERSION, or STREAM. When ACTION is not QUERY enter $.

Sample entry: **>ADD caingrp1 rteavail addr 222 222 nextrte $**

Network_Busy trigger criteria is defined.

Associated OMs

CAINMSGS, CAINTRIG, CAINAGOM, CAINOM

Send_Call PIC

Call processing enters **Send_Call** once the switch has

- identified a route list
- selected a route choice from the route list

During **Send_Call** call processing

- locates an idle trunk member in the terminating trunk group
- outpulses the appropriate information on the terminating trunk

The ***O_Mid_Call*** TDP can be encountered at the **Send_Call** PIC when reorigination indication is received at the UCS DMS-250 switch. Refer to Chapter 9, "O_Active PIC," for more information on the ***O_Mid_Call*** TDP and *O_IEC_Reorigination* trigger.

O_Abandon EDP

Call processing encounters the ***O_Abandon*** EDP during the **Collect_Information**, **Analyze_Information**, **Select_Route**, **Send_Call**, and **O_Alerting** PICs when two requirements are met:

- EDPs are active
- the calling party disconnects before the called party answers

The *O_Abandon* event is detected. For information on the *O_Abandon* event, refer to Chapter 4, "Collect_Information PIC."

O_Mid_Call EDP

The switch encounters the ***O_Mid_Call*** EDP at the **O_Alerting** and **O_Active** PICs during four call configurations:

- CC2 (stable two-party call)
- CC4 (three-party call in the setup phase)
- CC6 (party on hold)
- CC10 (stable multi-party call)

During CC2, the switch encounters the ***O_Mid_Call*** EDP when the passive leg (***LegID*** 1) presses the asterisk "*" key for 40 milliseconds.

During CC4, CC6, and CC10, the switch encounters the ***O_Mid_Call*** EDP when the controlling leg (***LegID*** 0) presses the asterisk "*" key for 40 milliseconds.

The switch detects the *Switch_Hook_Flash* event. Refer to Chapter 9, "O_Active and O_Suspended PICs," for more information on the ***O_Mid_Call*** EDP and the *Switch_Hook_Flash* event.

O_Mid_Call TDP

The ***O_Mid_Call*** TDP can be encountered at the **O_Alerting** PIC when reorigination indication is received at the UCS DMS-250 switch. Refer to Chapter 9, "O_Active and O_Suspended PICs," for more information on the ***O_Mid_Call*** TDP and *O_IEC_Reorigination* trigger.

O_Term_Seized EDP

The ***O_Term_Seized*** EDP occurs at the **Send_Call** PIC. This detection point may only be encountered as an EDP. It is encountered when a terminating trunk is seized on the UCS DMS-250 switch. The terminating trunk is considered seized when the UCS DMS-250 switch has:

- located an idle trunk member in the terminating trunk group
- outpulsed the appropriate information on the terminating trunk
- made a two-way network connection
- supervised the the originating and terminating trunks

Note: Call processing enters call configuration 2 (stable two-party call) when the switch selects a route and is about to seize the terminating trunk. Refer to Volume 3, Chapter 4, "Call Configuration Model," for more information on the stable two-party call and call configurations.

According to Bellcore specifications, ***O_Term_Seized*** is detected when the terminating party is alerted of the call and audible ringing is provided to the calling party. Due to technical complexity, NetworkBuilder treats all terminators as conventional trunks in regards to the *O_Term_Seized* event. This means that no detection of alerting is performed before this event is encountered.

If the ***O_No_Answer*** EDP is armed with an action other than IGNORE, or the *O_No_Answer* trigger is enabled with an action other than IGNORE or LEAVE_TDP when this detection point is reached, the O_No_Answer timer is started. The following list indicates the precedence used to determine the time limit to set:

1 value of the ***ONoAnswerTimer*** parameter received in the **Request_Report_BCM_Event** message.

2 value of the OANSTIME option in table CAINGRP.

3 value of the O_NO_ANSWER_TIMER field in table CAINPARM.

Note: In order to conserve switch resources the O_No_Answer timer is not started under the following conditions: 1) the *O_No_Answer* trigger action is provisioned as IGNORE or LEAVE_TDP or 2) an ***O_No_Answer*** EDP is armed with an action of IGNORE.

If the ***O_Term_Seized*** EDP is armed, an **O_Term_Seized** EDP-Notification message is sent.

O_Term_Seized event

Uses

This event is used to identify that a terminating trunk has been seized on the UCS DMS-250 switch. The O_No_Answer_Timer may be started.

Supported originating agencies

The *O_Term_Seized* event supports the following originating agencies:

- DAL
- FGD
- PRI
- SS7 Inter-IMT
- SS7 Global-IMT
- AXXESS

  Note: Refer to *UCS DMS-250 CAIN/FlexDial Interactions* for more information on support for AXXESS agents.

Event evaluation and actions

When the ***O_Term_Seized*** EDP is encountered and found to be active, NetworkBuilder directs the switch to send the **`O_Term_Seized`** EDP-Notification message to the SCP and call processing continues normally. All active EDPs remain active.

O_Term_Seized EDP-Notification

The following table defines the parameters and usage requirements for the parameters the **`O_Term_Seized`** EDP-Notification may contain. Refer to Volume 3, Chapter 8, "Outgoing CAIN message parameters," for detailed descriptions of parameters.

O_Term_Seized EDP-Notification message parameters

Parameter	Usage	Definition
`UserID`	Required	Contains the network identity of the originating agent
`BearerCapability`	Required	Contains the bearer capability of the call when the message is built

—continued—

O_Term_Seized event (continued)

O_Term_Seized EDP-Notification message parameters (continued)

Parameter	Usage	Definition
NotificationIndicator	Optional	Contains the value TRUE identifying the message type as Notification.
ExtensionParameter	Optional	***ExtensionParameter***s require the CAIN0200 SOC option.
termTrunkInfo	Optional	Contains the terminating trunk group number, trunk type, and trunk member number.
	—end—	

Once the EDP-Notification message is built, NetworkBuilder formats the message and sends it to the message encoder. Once encoded, the message is sent to the SCP. No response is expected from the SCP.

O_Called_Party_Busy trigger/event

O_Called_Party_Busy DP

Call processing encounters the ***O_Called_Party_Busy*** detection point (DP) when

- the switch attempts to route to a busy DAL or AXXESS agent (with ONNETTRK=Y)
- the switch receives an indication from the terminating agent that this end-user is busy:
  - — an ISUP Release message from a terminating SS7 agency, containing a Cause Indicator parameter (Table 7-1) indicating a called party busy condition
  - — a Release message from a terminating PRI agency, containing a Cause information element (Table 7-2) indicating a called party busy condition

Table 7-1
Cause values for an SS7 called party busy condition

Cause name	Cause value (in hexidecimal)
User busy	#11
No user responding	#12
No answer from user	#13
Call rejected	#15

Table 7-2
Cause values for a PRI called party busy condition

Cause name	Cause value (in hexidecimal)
User busy	#11
No user responding	#12
User alerting, no answer	#13
Call rejected	#15

O_Called_Party_Busy trigger/event (continued)

If EDPs are active when the ***O_Called_Party_Busy*** detection point (DP) is reached, ***O_Called_Party_Busy*** is encountered as an EDP. If EDPs are not active when the ***O_Called_Party_Busy*** detection point (DP) is reached, ***O_Called_Party_Busy*** is encountered as a TDP.

When ***O_Called_Party_Busy*** is encountered as a TDP, NetworkBuilder directs the switch to check for the following:

- Call subscription to a CAIN group through an SCP-determined group or through table STDPRTCT, authcode tables, ANI tables, TRKGRP, CALLATTR, or CAINPARM

Note: CAIN group subscription is handled differently for AXXESS agents. Refer to *UCS DMS-250 CAIN/FlexDial Interactions*.

- Enabled triggers (Triggers are enabled through table CAINGRP.)

Note: NetworkBuilder software supports the *O_Called_Party_Busy* trigger at the ***O_Called_Party_Busy*** TDP.

When ***O_Called_Party_Busy*** is encountered as an EDP and the ***O_Called_Party_Busy*** EDP is active the switch may take one of the following actions:

- The **`Request_Report_BCM_Event`** component is checked for the `oCalledPartyBusyActions` extension parameter to determine the action to take.
- If the **`Request_Report_BCM_Event`** component was received without an `oCalledPartyBusyActions` extension parameter, datafill for the active CAIN group in table CAINXDFT is checked for the `oCalledPartyBusyActions` extension parameter to determine the action to take.
- If the **`Request_Report_BCM_Event`** component was received without an `oCalledPartyBusyActions` extension parameter and the `oCalledPartyBusyActions` extension parameter is not datafilled for the active CAIN group in table CAINXDFT, an **`O_Called_Party_Busy`** EDP-Request message is sent to the SCP. The next event list (NEL) is cleared and the conversation transaction is maintained. From this point forward, call processing proceeds as if an **`O_Called_Party_Busy`** TDP-Request message were being sent.

Note: If EDPs are active when the ***O_Called_Party_Busy*** detection point (DP) is reached, but the ***O_Called_Party_Busy*** EDP is not active the call is treated as if an IGNORE action had occurred and the call continues through the call model.

O_Called_Party_Busy trigger/event (continued)

ATTENTION

The *O_Called_Party_Busy* trigger requires the CAIN0507 SOC option. The *O_Called_Party_Busy* event requires the CAIN0602 SOC option. Refer to Volume 5, Chapter 5, "NetworkBuilder SOC functionality," for more information.

Uses

The trigger/event can be used to implement the following services:

When you provision NetworkBuilder services on the switch, major call processing decisions can be made by the SCP. You define the services offered on the SCP. Some examples of services that can be developed on the SCP for the *O_Called_Party_Busy* trigger/event are:

- Call Forwarding
- Rerouting on busy signal

Supported originating agencies

The *O_Called_Party_Busy* trigger/event supports the following originating agencies:

- DAL
- FGD
- PRI
- SS7 Inter-IMT
- SS7 Global-IMT
- AXXESS

  Note: Refer to *UCS DMS-250 CAIN/FlexDial Interactions* for more information on support for AXXESS agents.

Subscribing to the O_Called_Party_Busy trigger

Subscription to the *O_Called_Party_Busy* trigger is available on a

- SCP-determined basis
- address basis (table STDPRTCT)
- authorization code basis (tables AUTHCODU, AUTHCDU2, AUTHCDU3, AUTHCDU4, or AUTHCDU5)
- ANI basis (table ANISCUSP, or tables ANIVAL and UNIPROF)

- agent basis (table TRKGRP or CALLATTR)
- office basis (table CAINPARM)

Note: CAIN group subscription for AXXESS agents is handled differently. Refer to *UCS DMS-250 CAIN/FlexDial Interactions*.

Trigger/Event evaluation

NetworkBuilder checks for CAIN routing parameters and standard routing parameters to determine the routing status. Following are routing status possibilities for an *O_Called_Party_Busy* trigger/event:

— RTEAVAIL – indicates that additional routing information is available to the switch for call completion. When the switch fails to seize an outgoing trunk it automatically attempts to route using the next route available without trigger or event evaluation.

— RTESDONE – indicates that all routes have been attempted and no additional routes are available to the switch for call completion.

Note: The RTESDONE routing criteria is evaluated when either of the following occur:

– There is no matching RTEAVAIL tuple/extension parameter for which the action can be performed (for example, when the provisioned action is NEXTCNRTE but there are only standard route list choices remaining).

– There are no RTEAVAIL tuples provisioned for any of the CAIN groups available for this trigger, or the RTEAVAIL action is IGNORE. This type of RTESDONE is evaluated when there are no more CAIN routing choices to evaluate.

Note: Refer to Volume 3, Chapter 10, “Incoming CAIN messages,” for more information on routing.

ATTENTION

When evaluating multiple CAIN groups for this trigger, if some are provisioned with an action of NEXTRTE and some are provisioned with an action of NEXTCNRTE, the evaluation of RTESDONE will not occur until all routes (standard and NetworkBuilder) have been attempted.

O_Called_Party_Busy trigger/event (continued)

Trigger evaluation

1 The current routing status is compared to the datafilled trigger criteria in table OCLDBUSY.

2 Identified digits are checked against the datafilled range associated with the appropriate CAIN group.

Note 1: The digit type (information, ANI, XLAADDR, address, CIC) are identified through datafill within the OCLDBUSY table.

Note 2: Call processing cannot meet digit criteria for a digit type of ADDR if the call queried at *Off_Hook_Immediate* and the SCP responded with an **`Analyze_Route`**.

ATTENTION

When you provision the digit type as ADDR, the actual digits sent in ***`CalledPartyID`***may not be the same as the matching criteria used by the switch to evaluate the trigger. This situation is most likely to occur when the SCP returns an **`Analyze_Route`** message. Triggering occurs based on the originally translated number, but the query message contains the SCP-returned value. In this case, a triggering loop may occur until the MAX_NUM_SERIAL_TRIGGERS is met.

Trigger actions

NetworkBuilder call processing supports the following actions for the *O_Called_Party_Busy* trigger:

- IGNORE – NetworkBuilder continues checking the remaining subscription methods for *O_Called_Party_Busy*. If datafill does not enable any trigger, call processing continues through the call model.
- QUERY – A query message is built and sent to the SCP. QUERY can be provisioned whether or not all routing options have been exhausted. The SCP analyzes the call and assists the switch with call processing.
- NEXTRTE – Call processing routes using the next routing option according to the following precedences:
  - — 1. The switch route advances and attempts the next route available in the route list.
  - — 2. The switch performs a CAIN routing parameter advance and attempts to route accordingly.

  Note 1: Provision NEXTRTE only when the criteria is datafilled as RTEAVAIL.

Note 2: Use of the NEXTRTE action allows for call control based on trigger table provisioning without SCP interaction.

- NEXTCNRTE – Call processing attempts to route using the next CAIN routing parameter.

  Note 1: Provision NEXTCNRTE only when the criteria is datafilled as RTEAVAIL.

  Note 2: This action is only considered valid when there are CAIN routing parameters remaining.

- LEAVE_TDP – NetworkBuilder call processing exits the ***O_Called_Party_Busy*** TDP with no further evaluation.
- CONT_NOTRIG – NetworkBuilder call processing exits the ***O_Called_Party_Busy*** TDP and prevents any further NetworkBuilder interaction for the call. This prevention is reset upon reorigination.

Event evaluation

When the ***O_Called_Party_Busy*** EDP is encountered and active there are several actions the switch may take:

- If an `oCalledPartyBusyActions` extension parameter was received with the **`Request_Report_BCM_Event`** component, the action taken is based on the current routing status.
- If an `oCalledPartyBusyActions` extension parameter was not received with the **`Request_Report_BCM_Event`** component, but the active CAIN group has the `oCalledPartyBusyActions` extension parameter datafilled in table CAINXDFT, the action taken is based on the current routing status.
- If the **`Request_Report_BCM_Event`** component was received without an `oCalledPartyBusyActions` extension parameter and the `oCalledPartyBusyActions` extension parameter is not datafilled for the active CAIN group in table CAINXDFT, an **`O_Called_Party_Busy`** EDP-Request message is sent to the SCP. The next event list (NEL) is cleared and the conversation transaction is maintained. From this point forward, call processing proceeds as if an **`O_Called_Party_Busy`** TDP-Request message were being sent.

Event actions

NetworkBuilder call processing supports the following actions for the *O_Called_Party_Busy* event:

- IGNORE – NetworkBuilder call processing exits the ***O_Called_Party_Busy*** DP with no further evaluation. All active EDPs remain active.

O_Called_Party_Busy trigger/event (continued)

- REQUEST – A request message is built and sent to the SCP. The SCP analyzes the call and assists the switch with call processing. The next event list (NEL) is discarded.
- NEXTRTE – Call processing routes using the next routing option according to the following precedences:
  - — 1. The switch route advances and attempts the next route available in the route list.
  - — 2. The switch performs a CAIN routing parameter advance and attempts to route accordingly.

  All active EDPs remain active.

  Note: NEXTRTE is supported only when the routing status is RTEAVAIL.

- NEXTCNRTE – Call processing attempts to route using the next CAIN routing parameter. All active EDPs remain active.

  Note 1: NEXTCNRTE is supported only when the routing status is RTEAVAIL.

  Note 2: This action is only considered valid when there are CAIN routing parameters remaining.

Error actions

Error actions are not identified for the *O_Called_Party_Busy* trigger or event. The switch applies AINF treatment for fatal application errors.

Options

The *O_Called_Party_Busy* trigger supports the following options.

- BUFFER – to activate digit buffering while the SCP is queried.
- GT – to identify the global title used to identify the SCP handling the query.
- T1OVFLGT – to identify the specific SCP to query on T1 overflow.
- ACGOVFLGT – to identify the global title to use for requerying when a query is blocked by an ACG control
- VERSION – controls the CAIN protocol version for outgoing messages

Note: The value provisioned in the VERSION option overrides the version provisioned in the CAIN_PROTOCOL_VERSION parameter of table CAINPARM on a transaction-by-transaction basis. Additionally, the VERSION option applies to both the outgoing and incoming messages within the same transaction. (Subsequent queries will use the setting provisioned in the CAIN_PROTOCOL_VERSION or provisioned setting of the VERSION option in the trigger table performing the subsequent query.)

- STREAM – to control protocol stream on a per-trigger tuple basis. The value of the STREAM option controls the set of parameters that are sent in NetworkBuilder messages.

The *O_Called_Party_Busy* event supports buffering through the `edpBuffer` extension parameter. Refer to Volume 3, Chapter 3, "Event processing," for more information.

Route advancing

The following figure describes how NetworkBuilder route advances using the *O_Called_Party_Busy* trigger table (OCLDBUSY). Route advancing with the *O_Called_Party_Busy* event is handled the same as for the *O_Called_Party_Busy* trigger, except that the next event list (NEL) is checked rather than OCLDBUSY trigger table. See Volume 3, Chapter 10, "Incoming CAIN messages," for more information on direct termination routing.

O_Called_Party_Busy trigger/event (continued)

Route advance with the O_Called_Party_Busy trigger

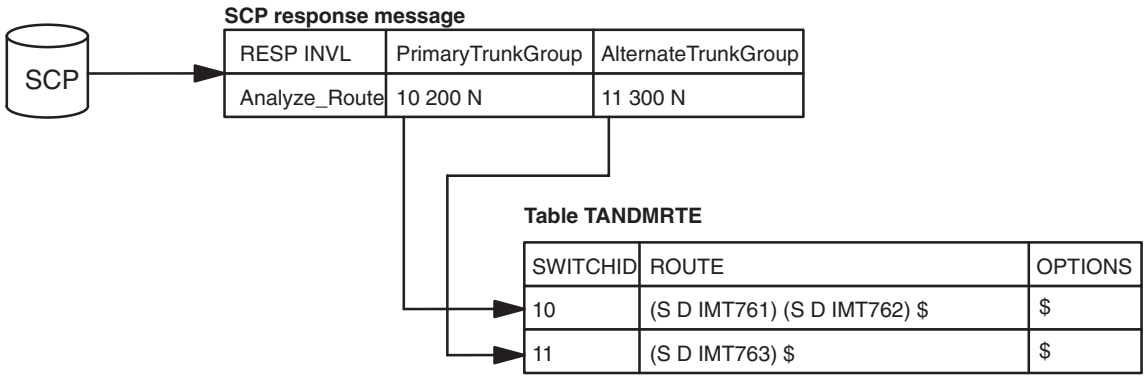


Table TANDMRTE

SWITCHID	ROUTE	OPTIONS
10	(S D IMT761) (S D IMT762) $	$
11	(S D IMT763) $	$

Table OCLDBUSY

OCLDBUSYKY	TODIGS	TRIGACT	OPTIONS
VPNGRP RTEAVAIL ADDR 214220	214220	NEXTCNRTE	$
N00GRP RTEAVAIL ADDR 800	800	QUERY	$

1 The switch receives an **`Analyze_Route`** message containing the ***`PrimaryTrunkGroup`*** and ***`AlternateTrunkGroup`*** parameters.

2 The switch attempts to complete the call using the route index identified by the ***`PrimaryTrunkGroup`*** parameter.

3 The switch seizes the trunk (IMT761) and sends an IAM message.

4 The switch receives a release message with a cause value indicating a User Busy condition.

5 The switch checks to see if the call subscribes to NetworkBuilder services and if the *O_Called_Party_Busy* trigger is enabled. If subscribed and enabled, the switch accesses the OCLDBUSY table.

   a If CAINGRP=VPNGRP and other criteria is met, the switch route advances and attempts routing using the next CAIN route choice (IMT763).

   b If CAINGRP=N00GRP and other criteria is met, the switch suspends call processing and sends an **`O_Called_Party_Busy`** query message to the SCP.

6 If the O_Called_Party_Busy trigger is not enabled, the call is processed normally as it would without Networkbuilder service interaction.

O_Called_Party_Busy trigger/event (continued)

The following figure and steps provide an additional example of routing advancing with *O_Called_Party_Busy*.

1 A NetworkBuilder-provisioned call triggers at Switch 11 and queries the SCP.

2 The SCP returns an **Analyze_Route** response containing ***PrimaryTrunkGroup***, ***AlternateTrunkGroup***, and ***CalledPartyID***.

3 Switch 11 decodes the ***PrimaryTrunkGroup*** in order to perform direct termination routing. The switch compares the SWID returned in the **Analyze_Route** response (77) to the current SWID (11). Since the numbers don't match, call processing accesses tuple 77 of table TANDMRTE. Call processing routes to the first index (IMT55A) and delivers the required SWID (77) in an IAM.

4 Switch 55 receives the IAM and compares the current SWID (55) to the received SWID (77). Since the SWIDs don't match, call processing accesses tuple 77 of table TANDMRTE. Call processing routes to the index (77A) and delivers the required SWID (77) in an IAM.

5 Switch 77 receives the IAM and compares the current SWID (77) to the received SWID (77). Now that the SWIDs match, call processing accesses tuple 24 (the trunk number received in the **Analyze_Route** message) and routes to DAL77CUST.

However, DAL77CUST, currently, has no idle trunk members; also, field ONNETTRK in table TRKGRP is set to Y, indicating that DAL77CUST terminates to a user instead of a network. Switch 77 sets BUSY treatment and sends a REL message with a USER_BUSY cause value to the originating switch (Switch 11 through Switch 55).

Note: If table TERMRTE had contained more than one route choice, call processing would have attempted to route the call until all route choices were attempted.

6 Switch 11 receives the REL message and tuple GRP1 of table OCLDBUSY is accessed and the RTEAVAIL criteria is met (since call processing has not attempted every CAIN routing parameter). Call processing performs the NEXTCNRTE action and routes to the next CAIN routing choice (***AlternateTrunkGroup***) and delivers the required SWID (33) in an IAM.

7 Switch 22 receives the IAM and compares the current SWID (22) to the received SWID (33). Since the SWIDs don't match, call processing accesses tuple 33 of table TANDMRTE. Call processing routes to the index (33) and delivers the required SWID (33) in the Generic Digits parameter of the outgoing IAM.

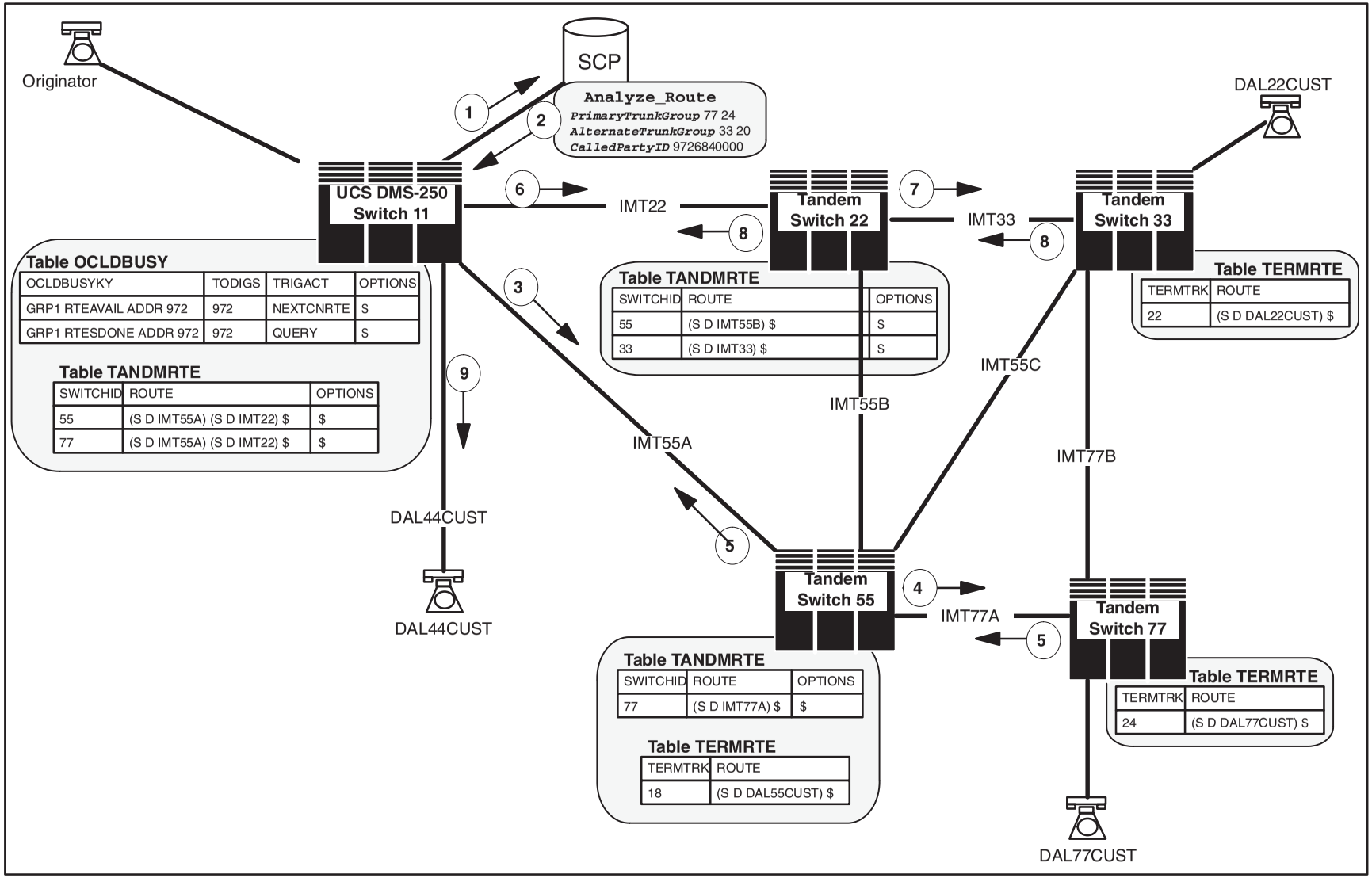
O_Called_Party_Busy trigger/event (continued)

8 Switch 33 receives the IAM and compares the current SWID (33) to the received SWID (33). Since the SWIDs match, call processing accesses tuple 22 (the trunk number received in the **`Analyze_Route`** message) and routes to DAL22CUST.

However, there are no idle trunk members. Switch 33 sets BUSY treatment and sends a REL message with a USER_BUSY cause value to the originating switch (Switch 11 through Switch 22).

Note: If table TERMRTE had contained more than one route choice, call processing would have attempted to route the call until all route choices were attempted.

9 Switch 11 receives the REL message, tuple GRP1 of table OCLDBUSY is accessed and the RTEAVAIL criteria is met (since all CAIN routing parameters have not been attempted). Call processing performs the NEXTCNRTE action and routes the call based on the ***`CalledPartyID`***. Normal call processing routes the call to trunk DAL44CUST on Switch 11.


Originator
SCP
Analyze_Route
PrimaryTrunkGroup 77 24
AlternateTrunkGroup 33 20
CalledPartyID 9726840000
1
2
UCS DMS-250
Switch 11
6
IMT22
Tandem
Switch 22
7
IMT33
Tandem
Switch 33
8
8
DAL22CUST
Table OCLDBUSY
OCLDBUSYKY | TODIGS | TRIGACT | OPTIONS
GRP1 RTEAVAIL ADDR 972 | 972 | NEXTCNRTE | $
GRP1 RTESDONE ADDR 972 | 972 | QUERY | $
Table TANDMRTE
SWITCHID | ROUTE | OPTIONS
55 | (S D IMT55A) (S D IMT22) $ | $
77 | (S D IMT55A) (S D IMT22) $ | $
Table TANDMRTE
SWITCHID | ROUTE | OPTIONS
55 | (S D IMT55B) $ | $
33 | (S D IMT33) $ | $
Table TERMRTE
TERMTRK | ROUTE
22 | (S D DAL22CUST) $
3
9
IMT55C
IMT55B
IMT55A
IMT77B
DAL44CUST
DAL44CUST
5
Tandem
Switch 55
4
IMT77A
Tandem
Switch 77
5
Table TANDMRTE
SWITCHID | ROUTE | OPTIONS
77 | (S D IMT77A) $ | $
Table TERMRTE
TERMTRK | ROUTE
18 | (S D DAL55CUST) $
Table TERMRTE
TERMTRK | ROUTE
24 | (S D DAL77CUST) $
DAL77CUST

O_Called_Party_Busy trigger/event (continued)

O_Called_Party_Busy TDP-Request

An **O_Called_Party_Busy** TDP-Request (query) is sent to the SCP in a query package, with a component type of Invoke_Last. The following table defines the parameters and usage requirements for the parameters the **O_Called_Party_Busy** TDP-Request may contain. Refer to Volume 3, Chapter 8, "Outgoing CAIN message parameters," for detailed descriptions of the request parameters.

Note: Parameters and extension parameters for AXXESS agents are handled differently. Refer to *UCS DMS-250 CAIN/FlexDial Interactions* for more information.

O_Called_Party_Busy TDP-Request message parameters

Parameter	Usage	Definition
UserID	Required	Contains the network identity of the originating agent
BearerCapability	Required	Contains the bearer capability of the call when the message is built
CalledPartyID	Optional	Contains the translated address
Lata (Note 6)	Optional	Contains the call's Local Access and Transport Area (LATA)
TriggerCriteriaType	Optional	Contains oCalledPartyBusy
ChargeNumber (Note 5)	Optional	Contains the billing number that would be used to populate the CDR at this point in call processing

Note 1: The `universalAccess` extension parameter is included in the query (a flat 10 digits) when parameter CAIN_PROTOCOL_VERSION is V0. When the CAIN_PROTOCOL_VERSION is set to V1 or higher, the parameter contains up to 24 digits including a nature of address and numbering plan similarly to the ***CalledPartyID***.
Note 2: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS07 or higher.
Note 3: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS08 or higher.
Note 4: The parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS09 or higher.
Note 5: When the CAIN_PROTOCOL_VERSION is set to V4 or higher non-standard charge numbers (CARD, AUTH, ACCT, PIN, and N00) no longer populate the ***ChargeNumber*** parameter and all standard NOAs are supported.
Note 6: The parameter is supported only on DAL, FGD, and AXXESS agencies and the parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS11 or higher.
Note 7: Refer to Volume 3, Chapter 8, "Outgoing CAIN message parameters" for information on the population of this parameter when the CAIN_PROTOCOL_VERSION is set to V5 or higher.

—continued—

O_Called_Party_Busy trigger/event (continued)

O_Called_Party_Busy TDP-Request message parameters (continued)

Parameter	Usage	Definition
CallingPartyID (Note 7)	Optional	Contains the one of the following (listed in order of precedence): For SS7 FGD, SS7 Inter-IMT, and SS7 Global-IMT calls: Calling_Party_Address from ISUP message, when available For PRI calls: CLID when available For FGD and AXXESS calls: valid ANI (information digits are not passed in this parameter) ***Note:*** Refer to *UCS DMS-250 CAIN/FlexDial Interactions* for more information on AXXESS agents. Valid SNPA value from table TRKGRP Default SNPA value from table CAINPARM
ChargePartyStationType (Note 2)	Optional	Contains the information digits for the call
Carrier	Optional	Contains the dialed CIC or the CIC value (with an indication of: Selected CIC presubscribed and not input by calling party) from table TRKGRP1.

Note 1: The `universalAccess` extension parameter is included in the query (a flat 10 digits) when parameter CAIN_PROTOCOL_VERSION is V0. When the CAIN_PROTOCOL_VERSION is set to V1 or higher, the parameter contains up to 24 digits including a nature of address and numbering plan similarly to the ***CalledPartyID***.
Note 2: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS07 or higher.
Note 3: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS08 or higher.
Note 4: The parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS09 or higher.
Note 5: When the CAIN_PROTOCOL_VERSION is set to V4 or higher non-standard charge numbers (CARD, AUTH, ACCT, PIN, and N00) no longer populate the ***ChargeNumber*** parameter and all standard NOAs are supported.
Note 6: The parameter is supported only on DAL, FGD, and AXXESS agencies and the parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS11 or higher.
Note 7: Refer to Volume 3, Chapter 8, “Outgoing CAIN message parameters” for information on the population of this parameter when the CAIN_PROTOCOL_VERSION is set to V5 or higher.

—continued—

O_Called_Party_Busy trigger/event (continued)

O_Called_Party_Busy TDP-Request message parameters (continued)

Parameter	Usage	Definition
BusyCause	Optional	Contains the cause value received from the SS7 or PRI agent; or if the terminating agent is a DAL or AXXESS with ONNETTRK=Y, the cause is set to user busy.
ACGEncountered	Optional	Contains the ACG control encountered when the ACG control expires and a query is allowed to be sent
ExtensionParameter (Note 2)	Optional	Extension parameters require the CAIN0200 SOC option.
busyRoute	Optional	Contains the number of routing attempts for a call at the *O_Called_Party_Busy* trigger. When an outpulse number is available, it is included as the routing number. If the call was routed using a trunk group and SWID from the ***PrimaryTrunkGroup***, ***AlternateTrunkGroup***, or ***SecondAlternateTrunkGroup*** parameter, the route index is included.
universalAccess (Note1)	Optional	Contains the universal access number dialed by the caller to obtain switch dial tone.
cainGroup	Optional	Contains the group number (field GRPNUM, table CAINGRP) for the CAIN group associated with the trigger when field EXTPARM in table CAINGRP contains CAINGRP.

Note 1: The universalAccess extension parameter is included in the query (a flat 10 digits) when parameter CAIN_PROTOCOL_VERSION is V0. When the CAIN_PROTOCOL_VERSION is set to V1 or higher, the parameter contains up to 24 digits including a nature of address and numbering plan similarly to the ***CalledPartyID***.
Note 2: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS07 or higher.
Note 3: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS08 or higher.
Note 4: The parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS09 or higher.
Note 5: When the CAIN_PROTOCOL_VERSION is set to V4 or higher non-standard charge numbers (CARD, AUTH, ACCT, PIN, and N00) no longer populate the ***ChargeNumber*** parameter and all standard NOAs are supported.
Note 6: The parameter is supported only on DAL, FGD, and AXXESS agencies and the parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS11 or higher.
Note 7: Refer to Volume 3, Chapter 8, “Outgoing CAIN message parameters” for information on the population of this parameter when the CAIN_PROTOCOL_VERSION is set to V5 or higher.

—continued—

O_Called_Party_Busy trigger/event (continued)

O_Called_Party_Busy TDP-Request message parameters (continued)

Parameter	Usage	Definition
origTrunkInfo	Optional	Contains the originating trunk group number, trunk type, and trunk member number when field EXTPARM in table CAINGRP contains ORGTINFO.
termTrunkInfo	Optional	Contains the terminating trunk group number, trunk type, and trunk member number.
netinfo (Note 2)	Optional	Contains external network ID, network customer group ID, and network class of service.
t1Overflow	Optional	Indicates to the SCP that an overflow occurred on the query to the initial SCP.
subscriptionInfo (Note 3)	Optional	Contains the digit type the switch triggered on and which subscription method was in use when the query occurred
jurisdictionInfo (Note 3)	Optional	Contains the originating switch's LRN.
collectedAddress (Note 3)	Optional	Contains the address collected from the incoming agent message, through subscriber dialing, through datafilled hotline digits, or through the *O_Feature_Requested* trigger.
switchID (Note 4)	Optional	Contains the switch ID

Note 1: The universalAccess extension parameter is included in the query (a flat 10 digits) when parameter CAIN_PROTOCOL_VERSION is V0. When the CAIN_PROTOCOL_VERSION is set to V1 or higher, the parameter contains up to 24 digits including a nature of address and numbering plan similarly to the ***CalledPartyID***.
Note 2: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS07 or higher.
Note 3: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS08 or higher.
Note 4: The parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS09 or higher.
Note 5: When the CAIN_PROTOCOL_VERSION is set to V4 or higher non-standard charge numbers (CARD, AUTH, ACCT, PIN, and N00) no longer populate the ***ChargeNumber*** parameter and all standard NOAs are supported.
Note 6: The parameter is supported only on DAL, FGD, and AXXESS agencies and the parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS11 or higher.
Note 7: Refer to Volume 3, Chapter 8, "Outgoing CAIN message parameters" for information on the population of this parameter when the CAIN_PROTOCOL_VERSION is set to V5 or higher.

—continued—

O_Called_Party_Busy trigger/event (continued)

O_Called_Party_Busy TDP-Request message parameters (continued)

Parameter	Usage	Definition
`billingNumber` (Note 4)	Optional	Contains the non-standard charge number
`acgRequery`	Optional	When present, indicates to the SCP that the current query is a re-query message resulting from the original query being blocked by an ACG control

Note 1: The `universalAccess` extension parameter is included in the query (a flat 10 digits) when parameter CAIN_PROTOCOL_VERSION is V0. When the CAIN_PROTOCOL_VERSION is set to V1 or higher, the parameter contains up to 24 digits including a nature of address and numbering plan similarly to the ***`CalledPartyID`***.
Note 2: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS07 or higher.
Note 3: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS08 or higher.
Note 4: The parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS09 or higher.
Note 5: When the CAIN_PROTOCOL_VERSION is set to V4 or higher non-standard charge numbers (CARD, AUTH, ACCT, PIN, and N00) no longer populate the ***`ChargeNumber`*** parameter and all standard NOAs are supported.
Note 6: The parameter is supported only on DAL, FGD, and AXXESS agencies and the parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS11 or higher.
Note 7: Refer to Volume 3, Chapter 8, "Outgoing CAIN message parameters" for information on the population of this parameter when the CAIN_PROTOCOL_VERSION is set to V5 or higher.

—end—

Once the request message is built, NetworkBuilder formats the message and sends it to the message encoder. Once encoded, the message is sent to the SCP.

O_Called_Party_Busy EDP-Request

An **`O_Called_Party_Busy`** EDP-Request is sent to the SCP in a conversation package with a component type of Invoke_Last. The following table defines the parameters and usage requirements for the parameters the **`O_Called_Party_Busy`** EDP-Request may contain. Refer to Volume 3, Chapter 8, "Outgoing CAIN message parameters," for detailed descriptions of the request parameters. Refer to Volume 3, Chapter 3, "Event processing," for detailed descriptions of EDPs.

O_Called_Party_Busy trigger/event (continued)

O_Called_Party_Busy EDP-Request message parameters

Parameter	Usage	Definition
UserID	Required	Contains the network identity of the originating agent
BearerCapability	Required	Contains the bearer capability of the call when the message is built
BusyCause	Optional	Contains the cause value received from the SS7 or PRI agent; or if the terminating agent is a DAL or AXXESS with ONNETTRK=Y, the cause is set to `userbusy`
NotificationIndicator	Optional	Contains the value FALSE to identify the message type as Request
ExtensionParameter	Optional	Extension parameters require the CAIN0200 SOC option
busyRoute	Optional	Contains the number of routing attempts for a call at the *O_Called_Party_Busy* event. When an outpulse number is available, it is included as the routing number. If the call was routed using a trunk group and SWID from the ***PrimaryTrunkGroup***, ***AlternateTrunkGroup***, or ***SecondAlternateTrunkGroup*** parameter, the route index is included.
termTrunkInfo	Optional	Contains the terminating trunk group number, trunk type, and trunk member number
—end—		

Once the request message is built, NetworkBuilder formats the message and sends it to the message encoder. Once encoded, the message is sent to the SCP.

SCP response processing

The following response messages are supported in response to an **O_Called_Party_Busy** TDP- or EDP-Request message:

- **Analyze_Route**
- **Send_To_Resource**
- **Continue**

O_Called_Party_Busy trigger/event (continued)

Note 1: When the switch receives a **`Continue`** message, the call is routed to BUSY treatment and in-switch cause mapping functionality is not performed.

Note 2: EDPs may be armed through the **`Request_Report_BCM_Event`** component with the **`Analyze_Route`** message in response to the **`O_Called_Party_Busy`** TDP- or EDP-Request message.

Note 3: EDPs may NOT be armed when the switch receives the **`Request_Report_BCM_Event`** component with the **`Continue`** message in response to the **`O_Called_Party_Busy`** TDP- or EDP-Request message.

Note 4: A **`Termination_Notification`** may be requested through the **`Send_Notification`** component with the SCP response message.

Note 5: An **`ACG`** message may be sent with the SCP response message.

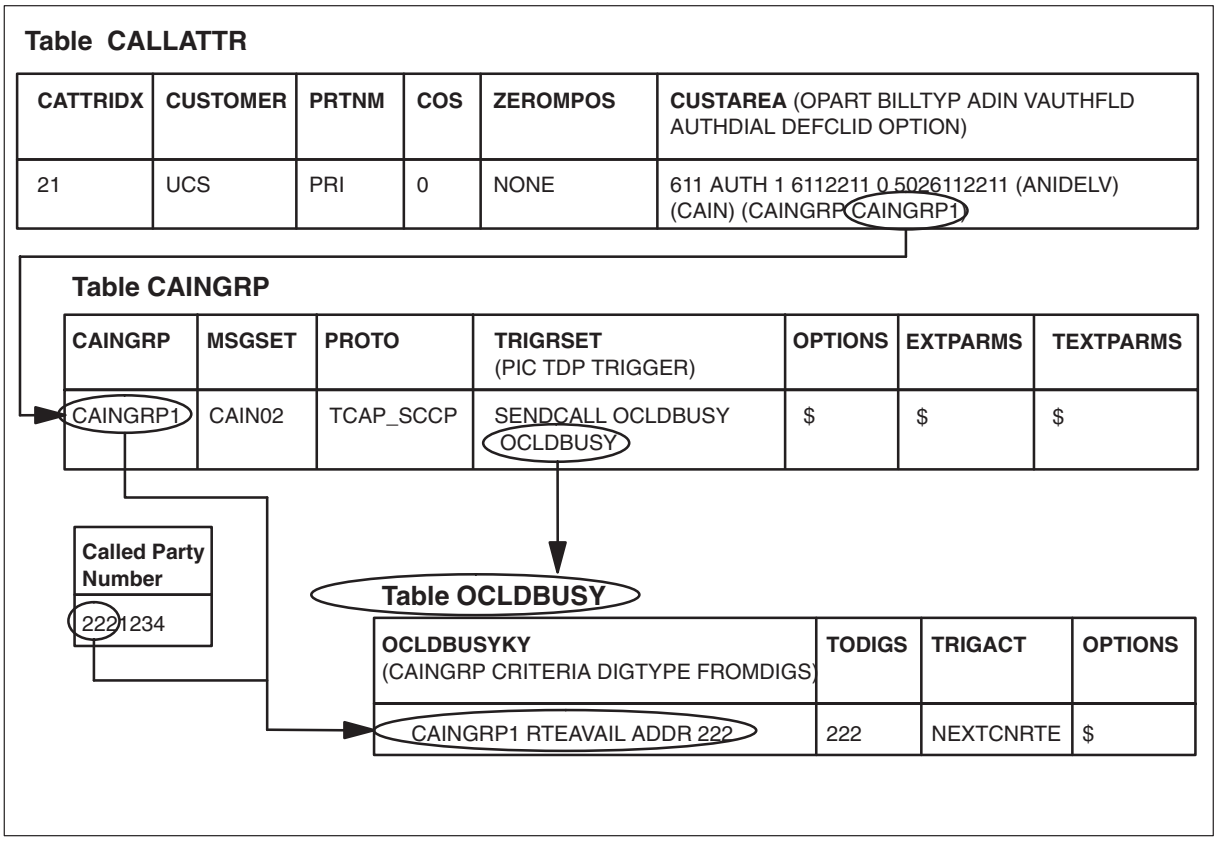
Refer to Volume 3, Chapter 10, "Incoming CAIN messages," for specific information regarding response messages; refer to Volume 3, Chapter 12, "Incoming CAIN message parameters," for detailed descriptions of the response parameters; refer to Volume 3, Chapter 3, "Event processing," for EDP specific message information.

Datafill

The following figure shows how a subscription table interacts with the *O_Called_Party_Busy* trigger table (OCLDBUSY).

O_Called_Party_Busy trigger/event (continued)

Subscription-OCLDBUSY table interaction



Provisioning the O_Called_Party_Busy trigger

At the CI prompt

1 Provision the originating agent as CAIN-capable.

2 Subscribe to a CAIN group (table STDPRTCT, authcode tables, ANI tables, TRKGRP, CALLATTR, or CAINPARM).

Note: CAIN group subscription is handled differently for AXXESS agents. Refer to *UCS DMS-250 CAIN/FlexDial Interactions*.

3 Datafill CAIN group trigger subscription to the (SENDCALL OCLDBUSY OCLDBUSY) trigger set (table CAINGRP).

4 Enter table OCLDBUSY.

5 Define the trigger criteria for a CAIN group by using the following format:

>ADD ocldbusyky todigs trigact options
where

O_Called_Party_Busy trigger/event (end)

ocldbusyky is comprised of three subfields: CAINGRP, CRITERIA, DIGTYPE, FROMDIGS, where

- CAINGRP is the CAIN group that enables the trigger (from table CAINGRP).
- CRITERIA is the routing status (RTEAVAIL or RTESDONE).
- DIGTYPE is the digit type being analyzed (INFO, ANI, XLAADDR, ADDR, CIC).
- FROMDIGS is the first number used to define the digit range (0 to 9, *, #).

todigs is the second number used to define the digit range (0 to 9, *, #).

trigact is is the trigger action taken when the address is within the FROMDIGS-TODIGS range (IGNORE, QUERY, NEXTCNRTE).

options is only allowed when ACTION is QUERY. Enter up to 6 options: BUFFER, GT, T1OVFLGT, ACGOVFLGT, VERSION, or STREAM. When ACTION is not QUERY enter $

Sample entry: **>ADD caingrp1 rteavail addr 222 222 nextcnrte $**

O_Called_Party_Busy trigger criteria is defined.

Associated OMs

CAINMSGS, CAINTRIG, CAINAGOM

O_Alerting PIC

Call processing enters **O_Alerting** when the switch

- locates an idle trunk member in the terminating trunk group
- outpulses the appropriate information on the terminating trunk

During **O_Alerting** call processing alerts the terminating party of the call.

O_Abandon EDP

Call processing encounters the ***O_Abandon*** EDP during the **Collect_Information**, **Analyze_Information**, **Select_Route**, **Send_Call**, and **O_Alerting** PICs when two requirements are met:

- EDPs are active
- the calling party disconnects before the called party answers

The *O_Abandon* event is detected. For information on the *O_Abandon* event, refer to Chapter 4, "Collect_Information PIC."

O_Mid_Call EDP

The switch encounters the ***O_Mid_Call*** EDP at the **O_Alerting** and **O_Active** PICs during four call configurations:

- CC2 (stable two-party call)
- CC4 (three-party call in the setup phase)
- CC6 (party on hold)
- CC10 (stable multi-party call)

During CC2, the switch encounters the ***O_Mid_Call*** EDP when the passive leg (***LegID*** 1) presses the asterisk "*" key for 40 milliseconds.

During CC4, CC6, and CC10, the switch encounters the ***O_Mid_Call*** EDP when the controlling leg (***LegID*** 0) presses the asterisk "*" key for more than 40 milliseconds.

The switch detects the *Switch_Hook_Flash* event. Refer to Chapter 9, "O_Active and O_Suspended PICs," for more information on the ***O_Mid_Call*** EDP and the *Switch_Hook_Flash* event.

O_Mid_Call TDP

The ***O_Mid_Call*** TDP can be encountered at the **O_Alerting** PIC when reorigination indication is received at the UCS DMS-250 switch. Refer to Chapter 9, "O_Active and O_Suspended PICs," for more information on the ***O_Mid_Call*** TDP and *O_IEC_Reorigination* trigger.

O_Answer EDP

Call processing encounters the ***O_Answer*** EDP when the terminating party answers the call. The O_No_Answer timer, if running is canceled, and the *O_Answer* event is evaluated.

O_Answer event

Uses

This event is used to identify that the terminating party has answered the call.

Supported originating agencies

The *O_Answer* event supports the following originating agencies:

- DAL
- FGD
- PRI
- SS7 Inter-IMT
- SS7 Global-IMT
- AXXESS

  Note: Refer to *UCS DMS-250 CAIN/FlexDial Interactions* for more information on support for AXXESS agents.

Event evaluation and actions

When the ***O_Answer*** EDP is encountered and found to be active, NetworkBuilder directs the switch to send the **`O_Answer`** EDP-Notification message to the SCP. When no more active EDPs are reachable after the ***O_Answer*** EDP is encountered, a **`Close`** message with a ***`CloseCause`*** parameter value `eDPsCompleted` is sent to the SCP to close the EDP transaction. When ***O_Answer*** EDP is encountered, the O_No_Answer timer, if running, is canceled. The Timeout timer is started when the *Timeout* event has been requested. Refer to Volume 3, Chapter 3, "Event processing," for more information on population of the **`Close`** message.

Note: The O_No_Answer timer, if running, is canceled here, regardless of whether it was started by the *O_No_Answer* trigger or event.

O_Answer EDP-Notification

The following table defines the parameters and usage requirements for the parameters the **`O_Answer`** EDP-Notification may contain. Refer to Volume 3, Chapter 8, "Outgoing CAIN message parameters," for detailed descriptions of parameters.

O_Answer event (end)

O_Answer EDP-Notification message parameters

Parameter	Usage	Definition
UserID	Required	Contains the network identity of the originating agent
BearerCapability	Required	Contains the bearer capability of the call when the message is built
NotificationIndicator	Optional	Contains the value TRUE to identify the message type as Notification.
ExtensionParameter	Optional	Extension parameters require the CAIN0200 SOC option
termTrunkInfo	Optional	Contains the terminating trunk group number, trunk type, and trunk member number
—end—		

Once the notification message is built, NetworkBuilder formats the message and sends it to the message encoder. Once encoded, the message is sent to the SCP. No response is expected from the SCP.

O_No_Answer DP

Call processing encounters ***O_No_Answer*** detection point (DP) when the O_No_Answer timer exceeds the time limit set for the called party to answer. The timer is defined by the ***ONoAnswerTimer*** parameter in the **Request_Report_BCM_Event** component, the OANSTIME option in table CAINGRP, or by the default O_NO_ANSWER_TIMER parameter in table CAINPARM.

If EDPs are active when the ***O_No_Answer*** detection point (DP) is reached, ***O_No_Answer*** is encountered as an EDP. If EDPs are not active when the ***O_No_Answer*** detection point (DP) is reached, ***O_No_Answer*** is encountered as a TDP.

When ***O_No_Answer*** is encountered as a TDP, NetworkBuilder directs the switch to check for the following:

- Call subscription to a CAIN group through an SCP-determined group or through table STDPRTCT, authcode tables, ANI tables, TRKGRP, CALLATTR, or CAINPARM

Note: CAIN group subscription is handled differently for AXXESS agents. Refer to *UCS DMS-250 CAIN/FlexDial Interactions*.

- Enabled triggers (Triggers are enabled through table CAINGRP.)

Note: NetworkBuilder software supports the *O_No_Answer* trigger at the ***O_No_Answer*** TDP.

When ***O_No_Answer*** is encountered as an EDP and the ***O_No_Answer*** EDP is active the switch may take one of the following actions:

- The **Request_Report_BCM_Event** component is checked for the `oNoAnswerActions` extension parameter to determine the action to take.
- If the **Request_Report_BCM_Event** component was received without the `oNoAnswerActions` extension parameter, datafill for the active CAIN group in table CAINXDFT is checked for the `oNoAnswerActions` extension parameter to determine the action to take.
- If the **Request_Report_BCM_Event** component was received without the `oNoAnswerActions` extension parameter and the `oNoAnswerActions` extension parameter is not datafilled for the active CAIN group in table CAINXDFT, an **O_No_Answer** EDP-Request message is sent to the SCP. The next event list (NEL) is cleared and the conversation transaction is maintained. From this point forward, call processing proceeds as if an **O_No_Answer** TDP-Request message were being sent.

O_No_Answer trigger/event (continued)

Note: If EDPs are active when the ***O_No_Answer*** detection point (DP) is reached, but the ***O_No_Answer*** EDP is not active, the call is treated as if an IGNORE action had occurred, and the call continues through the call model.

O_No_Answer timer

When call processing reaches the ***O_Term_Seized*** EDP, NetworkBuilder checks for an enabled *O_No_Answer* trigger or event. If the trigger or event is enabled, a timer starts.

When the ***O_Term_Seized*** EDP is reached, if the ***O_No_Answer*** EDP is armed with an action other than IGNORE, or the *O_No_Answer* trigger is enabled with an action other than IGNORE or LEAVE_TDP, the O_No_Answer timer is started. The following list indicates the precedence used to determine the time limit to set:

1 value of the ***ONoAnswerTimer*** parameter received in the **Request_Report_BCM_Event** message.

2 value of the OANSTIME option in table CAINGRP.

3 value of the O_NO_ANSWER_TIMER field in table CAINPARM.

Note: In order to conserve switch resources the O_No_Answer timer is not started under the following conditions: 1) the *O_No_Answer* trigger action is provisioned as IGNORE or LEAVE_TDP or 2) an ***O_No_Answer*** EDP is armed with an action of IGNORE.

NetworkBuilder cancels the timer when

- ***O_Answer*** is encountered, indicating the called party has answered
- ***O_Called_Party_Busy*** is encountered, indicating the called party was busy
- ***Network_Busy*** is encountered, indicating network congestion
- the calling party abandons the call, or the calling party is sent to treatment
- when reorigination indication is received

If NetworkBuilder does not cancel the timer, ***O_No_Answer*** is encountered.

Long call duration

The O_No_Answer timer interacts with the long call timer as follows:

- If the long call timer expires before the O_No_Answer timer, the O_No_Answer timer is cancelled.

O_No_Answer trigger/event (continued)

- If the O_No_Answer timer expires before the long call timer, ***O_No_Answer*** DP is encountered, and call processing checks the trigger or event action. If the action is QUERY, REQUEST, NEXTRTE, or NEXTCNRTE, the long call timer is cancelled.

  If the action is QUERY or REQUEST and the SCP returns a **`Continue`** response, call processing restarts the long call timer using the datafilled default.

  Note 1: When the PLAY_ANN option is provisioned in the ONOANSWR table and the long call timer is cancelled for the above listed reasons, the long call timer will be cancelled before the announcement specified by the PLAY_ANN option is played. The announcement specified by the PLAY_ANN option is not counted within the long call timer.

  Note 2: Refer to *UCS DMS-250 CAIN/FlexDial Interactions* for information on AXXESS agents interation with the long call timer.

O_No_Answer trigger/event (continued)

ATTENTION

The *O_No_Answer* trigger requires the CAIN0508 SOC option. The *O_No_Answer* event requires the CAIN0602 SOC option. Refer to Volume 5, Chapter 5, "NetworkBuilder SOC functionality," for more information.

Uses

When you provision NetworkBuilder services on the switch, major call processing decisions can be made by the SCP. You define the services offered on the SCP. Some examples of services that can be developed on the SCP for the *O_No_Answer* trigger are:

- Call Forwarding
- Rerouting on no answer

The *O_No_Answer* trigger provides the flexibility to route advance when the trigger criteria is met. This capability is useful when the call has triggered at a previous PIC, and the SCP provided multiple routing parameters in an **`Analyze_Route`** response message. The switch can use the routing parameters previously provided by the SCP instead of sending another query.

The following example illustrates a call with enabled *Customized_Dialing_Plan* and *O_No_Answer* triggers:

1 The call triggers at the *Customized_Dialing_Plan* trigger, queries the SCP, and receives an **`Analyze_Route`** message containing the ***`PrimaryTrunkGroup`*** and ***`AlternateTrunkGroup`*** parameters.

2 NetworkBuilder uses the ***`PrimaryTrunkGroup`*** parameter to identify a route list in one of the routing tables (TANDMRTE or TERMRTE).

3 The switch chooses an agent from the route list, selects an idle circuit, and starts the O_No_Answer timer.

4 The O_No_Answer timer expires prior to call answer.

5 Since additional CAIN routing parameters are available, NetworkBuilder checks the trigger table to see if the RTEAVAIL criteria is provisioned against the CAIN group. If provisioned, the call can query the SCP or perform a routing parameter advance.

O_No_Answer trigger/event (continued)

Supported originating agencies

The *O_No_Answer* trigger/event supports the following originating agencies:

- DAL
- FGD
- PRI
- SS7 Inter-IMT
- SS7 Global-IMT
- AXXESS

Note: Refer to *UCS DMS-250 CAIN/FlexDial Interactions* for more information on support for AXXESS agents.

Subscribing to the O_No_Answer trigger

Subscription to the *O_No_Answer* trigger is available on a

- SCP-determined basis
- address basis (table STDPRTCT)
- authorization code basis (tables AUTHCODU, AUTHCDU2, AUTHCDU3, AUTHCDU4, or AUTHCDU5)
- ANI basis (table ANISCUSP, or tables ANIVAL and UNIPROF)
- agent basis (table TRKGRP or CALLATTR)
- office basis (table CAINPARM)

Trigger/Event evaluation

NetworkBuilder checks for CAIN routing parameters and standard routing parameters to determine the routing status. Following are routing status possibilities for an *O_No_Answer* trigger/event:

- RTEAVAIL – indicates that additional routing information is available to the switch for call completion. When the switch fails to seize an outgoing trunk it automatically attempts to route using the next route available without trigger or event evaluation.
- RTESDONE – indicates that all routes have been attempted and no additional routes are available to the switch for call completion.

  Note: The RTESDONE routing criteria is evaluated when either of the following occur:

O_No_Answer trigger/event (continued)

— There is no matching RTEAVAIL tuple/extension parameter for which the action can be performed (for example, when the provisioned action is NEXTCNRTE but there are only standard route list choices remaining).

— There are no RTEAVAIL tuples provisioned for any of the CAIN groups available for this trigger, or the RTEAVAIL action is IGNORE. This type of RTESDONE is evaluated when there are no more CAIN routing choices to evaluate.

ATTENTION

When evaluating multiple CAIN groups for this trigger, if some are provisioned with an action of NEXTRTE and some are provisioned with an action of NEXTCNRTE, the evaluation of RTESDONE will not occur until all routes (standard and NetworkBuilder) have been attempted.

Note: Refer to Volume 3, Chapter 10, "Incoming CAIN messages," for more information on routing.

Trigger evaluation

1 The current routing status is compared to the datafilled trigger criteria in table ONOANSWR.

2 Identified digits are checked against the datafilled range associated with the appropriate CAIN group.

Note 1: The digit type (information, ANI, XLAADDR, address, or CIC) are identified through datafill within the ONOANSWR table.

Note 2: Call processing cannot match trigger criteria for a digit type of ADDR if the call queried at *Off_Hook_Immediate* and the SCP returned an **`Analyze_Route`** message.

Trigger actions

NetworkBuilder call processing supports the following actions for the *O_No_Answer* trigger:

- IGNORE – NetworkBuilder continues checking the remaining subscription methods for *O_No_Answer.* If datafill does not enable the trigger, call processing continues through the call model and the user continues to hear ringing.
- QUERY – A query message is built and sent to the SCP. QUERY can be provisioned whether or not all routing options have been exhausted. The SCP analyzes the call and assists the switch with call processing.

Note: If the QUERY action is provisioned, the PLAY_ANN option can also be used to provide an announcement when the call is route advanced.

- NEXTRTE – Call processing routes using the next routing option according to the following precedences:
  - — 1. The switch route advances and attempts the next route available in the route list.
  - — 2. The switch performs a CAIN routing parameter advance and attempts to route accordingly.

  Note 1: Provision NEXTRTE only when the criteria is datafilled as RTEAVAIL.

  Note 2: Use of the NEXTRTE action allows for call control based on trigger table provisioning without SCP interaction.

  Note 3: If the NEXTRTE action is provisioned, the PLAY_ANN option can also be used to provide an announcement when the call is route advanced.
- NEXTCNRTE – Call processing attempts to route using the next CAIN routing parameter.

  Note 1: Provision NEXTCNRTE only when the criteria is datafilled as RTEAVAIL.

  Note 2: This action is only considered valid when there are CAIN routing parameters remaining.

  Note 3: If the NEXTCNRTE action is provisioned, the PLAY_ANN option can also be used to provide an announcement when the call is route advanced.
- LEAVE_TDP – NetworkBuilder call processing exits the ***O_No_Answer*** TDP with no further evaluation, and the user continues to hear ringing.
- CONT_NOTRIG – NetworkBuilder call processing exits the ***O_No_Answer*** TDP and prevents any further NetworkBuilder interaction for the call. This prevention is reset upon reorigination.

Event evaluation

When the ***O_No_Answer*** EDP is encountered and active there are several actions the switch may take:

- If an `oNoAnswerActions` extension parameter was received with the **`Request_Report_BCM_Event`** message, the action taken is based on the current routing status.

O_No_Answer trigger/event (continued)

- If an `oNoAnswerActions` extension parameter was not received with the **`Request_Report_BCM_Event`** message, but the active CAIN group has the `oNoAnswerActions` extension parameter datafilled in table CAINXDFT, the action taken is based on the current routing status.
- If the **`Request_Report_BCM_Event`** component was received without the `oNoAnswerActions` extension parameter and the `oNoAnswerActions` extension parameter is not datafilled for the active CAIN group in table CAINXDFT, an **`O_No_Answer`** EDP-Request message is sent to the SCP. The next event list (NEL) is cleared and the conversation transaction is maintained. From this point forward, call processing proceeds as if an **`O_No_Answer`** TDP-Request message were being sent.

Event actions

NetworkBuilder call processing supports the following actions for the *O_No_Answer* event:

- IGNORE – NetworkBuilder call processing exits ***O_No_Answer*** with no further evaluation, and the user continues to hear ringing. All active EDPs remain active.
- REQUEST – The next event list (NEL) is discarded, and a request message is built and sent to the SCP. The SCP analyzes the call and assists the switch with call processing.
- NEXTRTE – Call processing routes using the next routing option according to the following precedences:
  - — 1. The switch route advances and attempts the next route available in the route list.
  - — 2. The switch performs a CAIN routing parameter advance and attempts to route accordingly.

  All active EDPs remain active.

  Note: NEXTRTE is supported only when the routing status is RTEAVAIL.

- NEXTCNRTE – Call processing attempts to route using the next CAIN routing parameter. All active EDPs remain active.

  Note 1: NEXTCNRTE is supported only when the routing status is RTEAVAIL.

  Note 2: This action is only considered valid when there are CAIN routing parameters remaining.

Error actions are not identified for the *O_No_Answer* trigger or event. The switch applies AINF treatment for fatal application errors.

O_No_Answer trigger/event (continued)

Options

The *O_No_Answer* trigger supports the following options:

- GT – to identify the global title used to identify the SCP handling the query.
- T1OVFLGT – to identify the specific SCP to query on overflow.
- ACGOVFLGT – to identify the global title to use for requerying when a query is blocked by an ACG control
- VERSION – controls the CAIN protocol version for outgoing messages

  Note: The value provisioned in the VERSION option overrides the version provisioned in the CAIN_PROTOCOL_VERSION parameter of table CAINPARM on a transaction-by-transaction basis. Additionally, the VERSION option applies to both the outgoing and incoming messages within the same transaction. (Subsequent queries will use the setting provisioned in the CAIN_PROTOCOL_VERSION or provisioned setting of the VERSION option in the trigger table performing the subsequent query.)

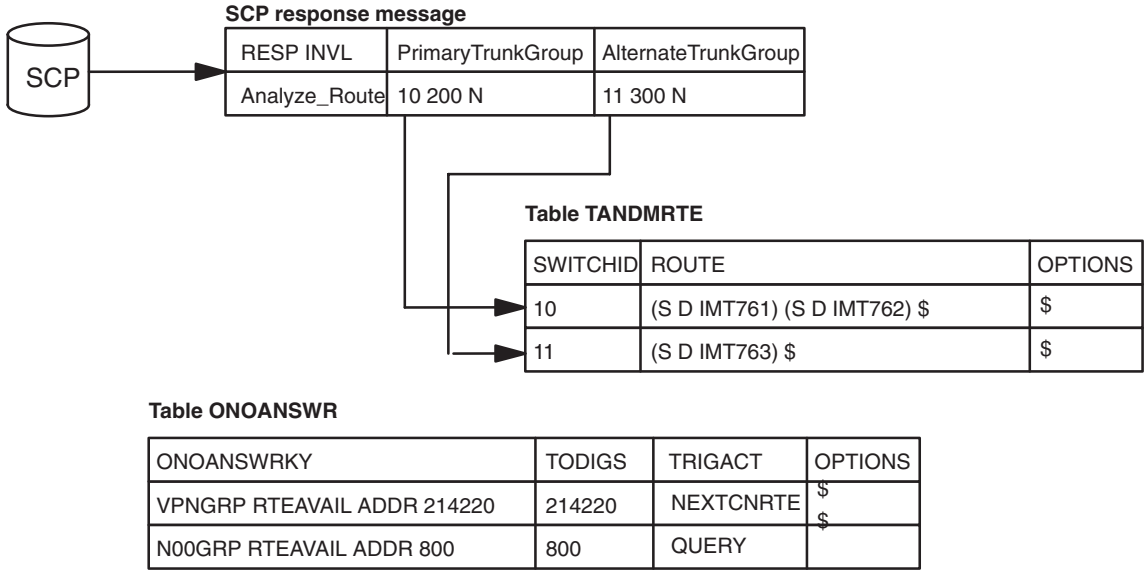
- PLAY_ANN – provides an announcement, when the O_No_Answer timer expires and the a call is rerouted (route advanced). The PLAY_ANN option is only allowed when the trigger action is QUERY, NEXTRTE, or NEXTCNRTE.
- STREAM – to control the protocol stream on a per-trigger tuple basis. The value of the STREAM option controls the set of parameters that are sent in NetworkBuilder messages.

Route Advancing

The following figure describes how NetworkBuilder route advances using the *O_No_Answer* trigger table (ONOANSWR). Route advancing with the *O_No_Answer* event is handled the same as for the *O_No_Answer* trigger, except that the next event list (NEL) is checked rather than ONOANSWR trigger table. See Volume 3, Chapter 10, “Incoming CAIN messages,” for more information on direct termination routing.

O_No_Answer trigger/event (continued)

Route advance with the O_No_Answer trigger



1 The switch receives an **`Analyze_Route`** message containing the ***`PrimaryTrunkGroup`*** and ***`AlternateTrunkGroup`*** parameters.

2 The switch attempts to complete the call using the route index identified by the ***`PrimaryTrunkGroup`*** parameter.

3 The switch seizes the trunk (IMT761) and sends an IAM message.

4 The switch checks to see if the call subscribes to NetworkBuilder services and if the *O_No_Answer* trigger is enabled. If subscribed and enabled, the O_No_Answer timer is started.

5 The O_No_Answer timer expires before the call is answered.

6 The switch accesses the ONOANSWR table and checks the trigger action.

   a If CAINGRP=VPNGRP and other criteria is met, the switch route advances and attempts routing using the next CAIN routing parameter (IMT763).

   b If CAINGRP=N00GRP and other criteria is met, NetworkBuilder discards the routing information received in the previous query and sends an **`O_No_Answer`** message to the SCP.

7 If the O_No_Answer trigger is not enabled, the call is processed normally as it would without Networkbuilder service interaction.

O_No_Answer TDP-Request

An **O_No_Answer** TDP-Request (query) is sent to the SCP in a query package, with a component type of Invoke_Last. The following table defines the parameters and usage requirements for the parameters the **O_No_Answer** TDP-Request may contain. Refer to Volume 3, Chapter 8, "Outgoing CAIN message parameters," for detailed descriptions of the request parameters.

Note: Parameters and extension parameters for AXXESS agents are handled differently. Refer to *UCS DMS-250 CAIN/FlexDial Interactions* for more information.

O_No_Answer TDP-Request message parameters

Parameter	Usage	Definition
UserID	Required	Contains the network identity of the originating agent
BearerCapability	Required	Contains the bearer capability of the call when the message is built
CalledPartyID	Optional	Contains the known address
Lata (Note 6)	Optional	Contains the call's Local Access and Transport Area (LATA)
TriggerCriteriaType	Optinoal	Contains oNoAnswer
ChargeNumber (Note 5)	Optional	Contains the billing number that would be used to populate the CDR at this point in call processing

Note 1: The `universalAccess` extension parameter is included in the query (a flat 10 digits) when parameter CAIN_PROTOCOL_VERSION is V0. When the CAIN_PROTOCOL_VERSION is set to V1 or higher, the parameter contains up to 24 digits including a nature of address and numbering plan similarly to the ***CalledPartyID***.
Note 2: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS07 or higher.
Note 3: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS08 or higher.
Note 4: The parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS09 or higher.
Note 5: When the CAIN_PROTOCOL_VERSION is set to V4 or higher non-standard charge numbers (CARD, AUTH, ACCT, PIN, and N00) no longer populate the ***ChargeNumber*** parameter and all standard NOAs are supported.
Note 6: The parameter is supported only on DAL, FGD, and AXXESS agencies and the parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS11 or higher.
Note 7: Refer to Volume 3, Chapter 8, "Outgoing CAIN message parameters" for information on the population of this parameter when the CAIN_PROTOCOL_VERSION is set to V5 or higher.

—continued—

O_No_Answer trigger/event (continued)

O_No_Answer TDP-Request message parameters (continued)

Parameter	Usage	Definition
CallingPartyID (Note 7)	Optional	Contains the one of the following (listed in order of precedence): For SS7 FGD, SS7 Inter-IMT, and SS7 Global-IMT calls: Calling_Party_Address from ISUP message, when available For FGD and AXXESS calls: valid ANI (information digits are not passed in this parameter) ***Note:*** Refer to *UCS DMS-250 CAIN/FlexDial Interactions* for more information on AXXESS agents. For PRI calls: CLID when available Valid SNPA value from table TRKGRP Default SNPA value from table CAINPARM
ChargePartyStationType (Note 2)	Optional	Contains the information digits for the call
Carrier	Optional	Contains the dialed CIC or the CIC value (with an indication of: Selected CIC presubscribed and not input by calling party) from table TRKGRP1.
ACGEncountered	Optional	Contains the ACG control encountered when the ACG control expires and a query is allowed to be sent

Note 1: The `universalAccess` extension parameter is included in the query (a flat 10 digits) when parameter CAIN_PROTOCOL_VERSION is V0. When the CAIN_PROTOCOL_VERSION is set to V1 or higher, the parameter contains up to 24 digits including a nature of address and numbering plan similarly to the ***CalledPartyID***.
Note 2: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS07 or higher.
Note 3: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS08 or higher.
Note 4: The parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS09 or higher.
Note 5: When the CAIN_PROTOCOL_VERSION is set to V4 or higher non-standard charge numbers (CARD, AUTH, ACCT, PIN, and N00) no longer populate the ***ChargeNumber*** parameter and all standard NOAs are supported.
Note 6: The parameter is supported only on DAL, FGD, and AXXESS agencies and the parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS11 or higher.
Note 7: Refer to Volume 3, Chapter 8, “Outgoing CAIN message parameters” for information on the population of this parameter when the CAIN_PROTOCOL_VERSION is set to V5 or higher.

—continued—

O_No_Answer trigger/event (continued)

O_No_Answer TDP-Request message parameters (continued)

Parameter	Usage	Definition
ExtensionParameter (Note 2)	Optional	Extension parameters require the CAIN0200 SOC option.
busyRoute	Optional	Contains the number of routing attempts for a call at the *O_No_Answer* trigger. When an outpulse number is available, it is included as the routing number. If the call was routed using a trunk group and SWID from the ***PrimaryTrunkGroup***, ***AlternateTrunkGroup***, or ***SecondAlternateTrunkGroup*** parameter, the route index is included.
universalAccess (Note 1)	Optional	Contains the universal access number dialed by the caller to obtain switch dial tone.
cainGroup	Optional	Contains the group number (field GRPNUM, table CAINGRP) for the CAIN group associated with the trigger when field EXTPARM in table CAINGRP contains CAINGRP.
origTrunkInfo	Optional	Contains the originating trunk group number, trunk type, and trunk member number when field EXTPARM in table CAINGRP contains ORGTINFO.
termTrunkInfo	Optional	Contains the terminating trunk group number, trunk type, and trunk member number.
netinfo (Note 2)	Optional	Contains external network ID, network customer group ID, and network class of service.

Note 1: The universalAccess extension parameter is included in the query (a flat 10 digits) when parameter CAIN_PROTOCOL_VERSION is V0. When the CAIN_PROTOCOL_VERSION is set to V1 or higher, the parameter contains up to 24 digits including a nature of address and numbering plan similarly to the ***CalledPartyID***.
Note 2: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS07 or higher.
Note 3: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS08 or higher.
Note 4: The parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS09 or higher.
Note 5: When the CAIN_PROTOCOL_VERSION is set to V4 or higher non-standard charge numbers (CARD, AUTH, ACCT, PIN, and N00) no longer populate the ***ChargeNumber*** parameter and all standard NOAs are supported.
Note 6: The parameter is supported only on DAL, FGD, and AXXESS agencies and the parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS11 or higher.
Note 7: Refer to Volume 3, Chapter 8, “Outgoing CAIN message parameters” for information on the population of this parameter when the CAIN_PROTOCOL_VERSION is set to V5 or higher.

—continued—

O_No_Answer trigger/event (continued)

O_No_Answer TDP-Request message parameters (continued)

Parameter	Usage	Definition
t1Overflow	Optional	Indicates to the SCP that an overflow occurred on the query to the initial SCP.
subscriptionInfo (Note 3)	Optional	Contains the digit type the switch triggered on and which subscription method was in use when the query occurred
jurisdictionInfo (Note 3)	Optional	Contains the originating switch's LRN.
collectedAddress (Note 3)	Optional	Contains the address collected from the incoming agent message, through subscriber dialing, through datafilled hotline digits, or through the *O_Feature_Requested* trigger.
switchID (Note 4)	Optional	Contains the switch ID
billingNumber (Note 4)	Optional	Contains the non-standard charge number
acgRequery	Optional	When present, indicates to the SCP that the current query is a re-query message resulting from the original query being blocked by an ACG control

Note 1: The universalAccess extension parameter is included in the query (a flat 10 digits) when parameter CAIN_PROTOCOL_VERSION is V0. When the CAIN_PROTOCOL_VERSION is set to V1 or higher, the parameter contains up to 24 digits including a nature of address and numbering plan similarly to the ***CalledPartyID***.
Note 2: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS07 or higher.
Note 3: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS08 or higher.
Note 4: The parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS09 or higher.
Note 5: When the CAIN_PROTOCOL_VERSION is set to V4 or higher non-standard charge numbers (CARD, AUTH, ACCT, PIN, and N00) no longer populate the ***ChargeNumber*** parameter and all standard NOAs are supported.
Note 6: The parameter is supported only on DAL, FGD, and AXXESS agencies and the parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS11 or higher.
Note 7: Refer to Volume 3, Chapter 8, "Outgoing CAIN message parameters" for information on the population of this parameter when the CAIN_PROTOCOL_VERSION is set to V5 or higher.

—end—

Once the request message is built, NetworkBuilder formats the message and sends it to the message encoder. Once encoded, the message is sent to the SCP.

O_No_Answer EDP-Request

An **O_No_Answer** EDP-Request message is sent to the SCP in a conversation package with a component type of Invoke_Last. The following table defines the parameters and usage requirements for the parameters the **O_No_Answer** EDP-Request may contain. Refer to Volume 3, Chapter 8, "Outgoing CAIN message parameters," for detailed descriptions of the request parameters. Refer to Volume 3, Chapter 3, "Event processing," for detailed descriptions of EDPs.

O_No_Answer EDP-Request message parameters

Parameter	Usage	Definition
UserID	Required	Contains the network identity of the originating agent
BearerCapability	Required	Contains the bearer capability of the call when the message is built
NotificationIndicator	Optional	Contains the value FALSE to identify the message type as Request.
ExtensionParameter	Optional	***ExtensionParameter***s require the CAIN0200 SOC option
busyRoute	Optional	Contains the number of routing attempts for a call at the *O_No_Answer* event. When an outpulse number is available, it is included as the routing number. If the call was routed using a trunk group and SWID from the ***PrimaryTrunkGroup***, ***AlternateTrunkGroup***, or ***SecondAlternateTrunkGroup*** parameter, the route index is included.
termTrunkInfo	Optional	Contains the terminating trunk group number, trunk type, and trunk member number
—end—		

Once the request message is built, NetworkBuilder formats the message and sends it to the message encoder. Once encoded, the message is sent to the SCP.

SCP response processing

The following messages are supported in response to an **O_No_Answer** TDP- or EDP-Request message:

- **Analyze_Route**

O_No_Answer trigger/event (continued)

- **Send_To_Resource**
- **Continue**

Note 1: EDPs may be armed through the **Request_Report_BCM_Event** component with the **Analyze_Route** message.

Note 2: EDPs may NOT be armed when the switch receives the **Request_Report_BCM_Event** component with the **Continue** message in response to the **O_No_Answer** TDP- or EDP-Request message.

Note 3: Once the call enters conversational digit collection, the **Continue** message is not a valid SCP response.

Note 4: A **Termination_Notification** may be requested through the **Send_Notification** component with the SCP response message. (The switch processes the **Send_Notification** component whether or not the called party answers before the switch receives a response.)

Note 5: An **ACG** component may be sent with the SCP response message. (The switch processes the **ACG** component whether or not the called party answers before the switch receives a response.)

Notes 6–10 apply when a response is received from the SCP after the called party answers.

Note 6: If the called party answers before an SCP response is received, the connection to the called party is completed and the switch continues waiting for an SCP response.

Note 7: If the switch receives a call-related **Analyze_Route** or **Send_To_Resource** message in a Response package, the switch closes the transaction and ignores the call-related message. (The switch processes non-call-related components.)

Note 8: If the switch receives an **Continue** message in a Response package, the switch closes the transaction and processes the response.

Note 9: If the switch receives a **Send_To_Resource** message in a Conversation package, the switch sends a **Resource_Clear** message with the ***ClearCause*** parameter set to `calledPartyAnswered`.

Note 10: If the switch receives an **Analyze_Route** or **Continue** message in a conversation package, the switch sends a **Close** message with the ***CloseCause*** parameter set to `calledPartyAnswered`.

The PLAY_ANN option may be provisioned in table ONOANSWR. (Only uninterruptible prompts are supported. Only in-switch announcements are allowed.) The tone or announcement provisioned in table ONOANSWR is played prior to routing if the SCP's response to the **O_No_Answer** TDP- or EDP-Request message is **Analyze_Route**, unless the `callBranding`

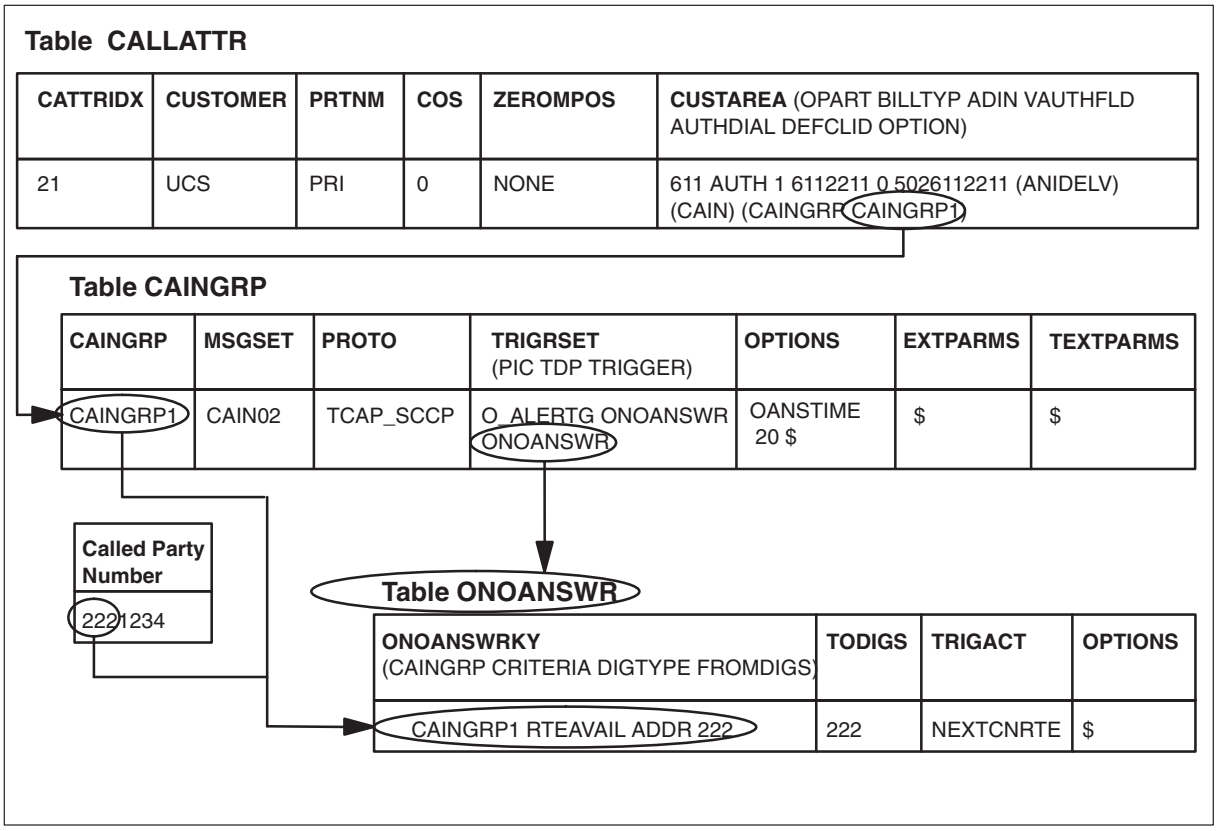
extension parameter is included in the **`Analyze_Route`** message or is provisioned in table CAINXDFT for the current CAIN group.

Refer to Volume 3, Chapter 10, "Incoming CAIN messages," for specific information regarding response messages; refer to Volume 3, Chapter 12, "Incoming CAIN message parameters," for detailed descriptions of the response parameters; refer to Volume 3, Chapter 3, "Event processing," for EDP specific message information.

Datafill

The following figure shows how a subscription table interacts with the *O_No_Answer* trigger table (ONOANSWR).

Subscription-ONOANSWR table interaction



Provisioning the O_No_Answer trigger

At the CI prompt

1 Provision the originating agent as CAIN-capable.

O_No_Answer trigger/event (end)

2 Subscribe to a CAIN group (table STDPRTCT, authcode tables, ANI tables, TRKGRP, CALLATTR, or CAINPARM).

Note: CAIN group subscription is handled differently for AXXESS agents. Refer to *UCS DMS-250 CAIN/FlexDial Interactions*.

3 Datafill CAIN group trigger subscription to the (O_ALERTG ONOANSWR ONOANSWR) trigger set (table CAINGRP).

4 Enter table ONOANSWR.

5 Define the trigger criteria for a CAIN group by using the following format:

>ADD onoanswrky todigs trigact options
where

onoanswrky is comprised of three subfields: CAINGRP, CRITERIA, DIGTYPE, FROMDIGS, where

CAINGRP	is the CAIN group that enables the trigger (from table CAINGRP).
CRITERIA	is the routing status (RTEAVAIL or RTESDONE).
DIGTYPE	is the digit type being analyzed (INFO, ANI, XLAADDR, ADDR, CIC).
FROMDIGS	is the first number used to define the digit range (0 to 9, *, #).

todigs is the second number used to define the digit range (0 to 9, *, #).

trigact is is the trigger action taken when the address is within the FROMDIGS-TODIGS range (IGNORE, QUERY, NEXTRTE, NEXTCNRTE, LEAVE_TDP, CONT_NOTRIG).

options is only allowed when ACTION is QUERY. Enter up to 6 options: GT, T1OVFLGT, ACGOVFLGT, VERSION, PLAY_ANN, or STREAM. When ACTION is not QUERY enter $.

Sample entry: **>ADD caingrp1 rteavail addr 222 222 nextcnrte $**

O_No_Answer trigger criteria is defined.

Associated OMs

CAINMSGS, CAINTRIG, CAINAGOM

O_Active and O_Suspended PICs

Call processing enters **O_Active** when a call is connected and in the talking state. Call processing re-enters the **O_Active** PIC when an SS7 RESUME (RES) message is received on the UCS DMS-250 switch

Call processing enters **O_Suspended** when an SS7 SUSPEND (SUS) message is received on the UCS DMS-250 switch.

O_Disconnect EDP

Call processing encounters the ***O_Disconnect*** EDP during the **O_Active** and **O_Suspended** PICs when EDPs are active and a party disconnects.

When reorigination indication is received, the switch encounters the ***O_Disconnect*** EDP before the ***O_Mid_Call*** TDP.

O_Mid_Call EDP

The switch encounters the ***O_Mid_Call*** EDP at the **O_Active** or **O_Suspended** PICs when the Timeout timer expires before the calling party is disconnected from the called party. The *Timeout* event is evaluated.

Note: The Timeout timer is started at the ***O_Answer*** EDP.

The switch encounters the ***O_Mid_Call*** EDP at the **O_Alerting** and **O_Active** PICs during four call configurations:

- CC2 (stable two-party call)
- CC4 (three-party call in the setup phase)
- CC6 (party on hold)
- CC10 (stable multi-party call)

During CC2, the switch encounters the ***O_Mid_Call*** EDP when the passive leg (***LegID*** 1) presses the asterisk "*" key for 40 milliseconds.

During CC4, CC6, and CC10, the switch encounters the ***O_Mid_Call*** EDP when the controlling leg (***LegID*** 0) presses the asterisk "*" key for more than 40 milliseconds.

The switch detects the *Switch_Hook_Flash* event.

Note: The ***O_Mid_Call*** EDP for the *Switch_Hook_Flash* event is invalid for the **O_Suspended** PIC.

O_Mid_Call TDP

The ***O_Mid_Call*** TDP can be encountered at the **O_Active** or **O_Suspended** PICs when reorigination indication is received at the UCS DMS-250 switch. The *O_IEC_Reorigination* trigger is evaluated.

O_Disconnect event

Uses

This event is detected when a party disconnects. The *O_Disconnect* event can only be detected after the call has been answered and the call has entered the **O_Active** or **O_Suspended** PICs. When you provision NetworkBuilder services on the switch, the SCP can make major call processing decisions. You define the services offered on the SCP. Debit card services and take-back and transfer are examples of services that can be developed on the SCP for the *O_Disconnect* event.

Note: The *O_Disconnect* event is not detected in SS7 RLT scenarios when an operator disconnects from the calling party to route the calling party to another party.

Supported originating agencies

The *O_Disconnect* event supports the following originating agencies:

- DAL
- FGD
- PRI
- SS7 Inter-IMT
- SS7 Global-IMT
- AXXESS

  Note: Refer to *UCS DMS-250 CAIN/FlexDial Interactions* for more information on support for AXXESS agents.

Event evaluation and actions

The **`Request_Report_BCM_Event`** message contains the ***`EDPRequest`*** and ***`EDPNotification`*** parameters. The *O_Disconnect* event can be present in either parameter.

The software optionality control (SOC) CAIN0800 controls the ***O_Disconnect*** EDP–Notification and EDP–Request messages.

If the ***O_Disconnect*** EDP is armed to send an **`O_Disconnect`** EDP-Notification message and the CAIN0800 SOC is active, the following occurs:

- An **`O_Disconnect`** EDP-Notification is sent to the SCP.
- If the *Timeout* event is active, the Timeout timer is cancelled.

O_Disconnect event (continued)

- A **Close** message is sent to the SCP with a ***CloseCause*** value of callTerminated. No SCP response is expected.

  Note: During a multi-party call, the switch does not send a **Close** message.

- When the *O_Disconnect* event is detected due to the switch receiving reorigination indication during a two-party call, the ***O_Mid_Call*** TDP is encountered immediately after the **Close** message is sent to the SCP.

Refer to Volume 3, Chapter 3, "Event processing," for more information on population of the **Close** message.

If the ***O_Disconnect*** EDP is armed to send an **O_Disconnect** EDP-Request message and the CAIN0800 SOC is active, the switch builds and sends the **O_Disconnect** EDP-Request message.

O_Disconnect EDP-Notification and EDP-Request

The following table defines the parameters and usage requirements for the parameters the **O_Disconnect** EDP-Notification and EDP-Request messages may contain. Refer to Volume 3, Chapter 8, "Outgoing CAIN message parameters," for detailed descriptions of parameters.

Note: The **O_Disconnect** EDP-Request message never contains extension parameters.

Table 9-1
O_Disconnect message parameters

Parameter	Usage	Definition
UserID	Required	Contains the network identity of the originating agent
BearerCapability	Required	Contains the bearer capability of the call when the message is built
NotificationIndicator	Optional	Identifies the message type as notification or request
ExtensionParameter	Optional	***ExtensionParameter***s require the CAIN0200 SOC option
Note: The **O_Disconnect** EDP-Request message does not support the ***ExtensionParameter***.		
—continued—		

Table 9-1
O_Disconnect message parameters (continued)

Parameter	Usage	Definition
termTrunkInfo	Optional	Contains the terminating trunk group number, trunk type, and trunk member number
connectTime	Optional	Indicates how much time has elapsed since the call was answered, in minutes, seconds, and tenths of seconds
LegID	Optional	Identifies a leg in a call segment
PointInCall	Optional	Contains current point in call of the call
Note: The **O_Disconnect** EDP-Request message does not support the ***ExtensionParameter***.		
—end—		

Once NetworkBuilder builds the notification or request message, NetworkBuilder formats the message and sends it to the message encoder. When the message is encoded, the switch sends the message to the SCP. For the notification message, the switch does not expect a response from the SCP.

SCP response messages

The following response messages are supported for the **O_Disconnect** EDP-Request message:

- **Connect_To_Resource** message
- **Disconnect** message
- **Disconnect_Leg** message
- **Merge_Call** message

NetworkBuilder arms EDPs if the switch receives a **Request_Report_BCM_Event** component within the **Disconnect_Leg** or **Merge_Call** messages.

Refer to Volume 3, Chapter 10, “Incoming CAIN messages,” for specific information regarding response messages; refer to Volume 3, Chapter 12, “Incoming CAIN message parameters,” for detailed descriptions of the response parameters; refer to Volume 3, Chapter 3, “Event processing,” for EDP-specific message information.

Timeout event

Uses

This event is used to indicate when the Timeout timer expires. When NetworkBuilder services are provisioned on the switch, the SCP can make major call processing decisions. The services offered are defined on the SCP. Debit card service is an example of a service that can be developed on the SCP for the *Timeout* event.

If the *Timeout* event is requested, the switch starts the Timeout timer when the call is answered. The length of the timer is determined by the value, in minutes, of the ***TimeoutTimer*** parameter received in the **Request_Report_BCM_Event** component, or from the value of the TIMEOUT_TIMER parameter in table CAINPARM (if no ***TimeoutTimer*** parameter is received or the received value is invalid).

When CAIN0801 Mid-Call Services 2 SOC option is enabled, the Timeout timer can be restarted if the *Timeout* event is requested again after the switch sends a **CTR_Clear** following a **Connect_To_Resource** at the ***O_Mid_Call*** EDP. The timer's duration is determined by the ***TimeoutTimer*** parameter value received in the second **Request_Report_BCM_Event** component or from the value of the TIMEOUT_TIMER parameter in table CAINPARM.

Note: The ***O_Mid_Call*** EDP for the *Timeout* and *Switch_Hook_Flash* events are not allowed to be armed at the same time. If NetworkBuilder arms the ***O_Mid_Call*** EDP for the two events simultaneously, the switch ignores the **O_Mid_Call** EDP for the *Switch_Hook_Flash* event.

Supported originating agencies

The *Timeout* event supports the following originating agencies:

- DAL
- FGD
- PRI
- SS7 Inter-IMT
- SS7 Global-IMT
- AXXESS

  Note: Refer to *UCS DMS-250 CAIN/FlexDial Interactions* for more information on support for AXXESS agents.

Event evaluation and actions

When the *Timeout* event is active, NetworkBuilder directs the switch to send the **Timeout** EDP-Request message to the SCP.

Timeout EDP-Request

The following table defines the parameters and usage requirements for the parameters the **Timeout** EDP-Request may contain. Refer to Volume 3, Chapter 8, "Outgoing CAIN message parameters," for detailed descriptions of parameters.

Timeout EDP-Request message parameters

Parameter	Usage	Definition
UserID	Required	Contains the network identity of the originating agent
BearerCapability	Required	Contains the bearer capability of the call when the message is built
CcID	Optional	Contains the call configuration identifier
ExtensionParameter	Optional	***ExtensionParameter***s require the CAIN0200 SOC option.
termTrunkInfo	Optional	Contains the terminating trunk group number, trunk type, and trunk member number
connectTime	Optional	Indicates how much time has elapsed since the call was answered, in minutes, seconds, and tenths of seconds
NotificationIndicator	Optional	Contains the value FALSE to identify the message type as Request.
—end—		

Once the request message is built, NetworkBuilder formats the message and sends it to the message encoder. Once encoded, the message is sent to the SCP.

SCP response processing

The **Disconnect** and **Connect_To_Resource** messages are supported in response to a **Timeout** EDP-Request message.

Note: When the switch sends a **CTR_Clear** message while the call is at the *Timeout* event, **Disconnect**, **Continue**, and **Connect_To_Resource** response messages are supported. Although the CAIN protocol allows the **Analyze_Route** and **Collect_Information** messages in response to **CTR_Clear**, a fatal application error occurs when one of these messages is received at the *Timeout* event.

Timeout event (continued)

Connect_To_Resource SCP response message

The **Connect_To_Resource** message requires the CAIN0801 Mid Call Services 2 SOC option to be enabled.

- If at the *Timeout* event the switch receives **Connect_To_Resource** message in a response package and the CAIN0801 is not enabled, the switch applies AINF treatment to the originator and idles the terminator.
- If at the *Timeout* event the switch receives **Connect_To_Resource** message in a conversation package and the CAIN0801 is not enabled, the switch sends a conversational **CTR_Clear** message with the ***ClearCause*** parameter value set to `taskRefused`.

Refer to Volume 3, Chapter 10, "Incoming CAIN messages," for specific information regarding response messages; refer to Volume 3, Chapter 12, "Incoming CAIN message parameters," for detailed descriptions of the response parameters; refer to Volume 3, Chapter 3, "Event processing," for EDP specific message information.

Application errors

The following table lists application errors that may occur and are associated with the *Timeout* event.

Timeout event (continued)

Timeout application errors

Error type	Package	Log generated	Reported to SCP?	Error action performed
Connect_To_ Resource received with ***ResourceType*** set to `Play Announcement`	Conversation	NA	No	Switch processes the **Connect_To_ Resource** as a `Play Announcement` request followed by a **Disconnect**
Connect_To_ Resource received with ***ResourceType*** set to `Flex Parameter Block`	Conversation	NA	No	Switch sends a **CTR_Clear** in conversation. ***ClearCause*** is set to `taskRefused` and the call is not cleared toward the controlling or passive leg.
Connect_To_ Resource received with ***ResourceType*** set to an expected value	Conversation	CAIN200	Yes	Switch sends a **CTR_Clear** in conversation. ***ClearCause*** is set to `abort` and the call is not cleared toward the controlling or passive leg.
Connect_To_ Resource received with ***ResourceType*** set to `Play Announcement and Collect Digits`	Response	CAIN200	Yes	Switch sends a **CTR_Clear** in conversation. ***ClearCause*** is set to `abort` and the call is not cleared toward the controlling or passive leg.
Connect_To_ Resource received with ***ResourceType*** set to `Play Announcement` and the ***Disconnect Flag*** is missing	Response	NA	No	Switch continues the interaction as if the ***DisconnectFlag*** were present.

Note: Refer to *UCS DMS-250 Logs Reference Manual* for information on logs.

—continued—

Timeout event (end)

Timeout application errors (continued)

Error type	Package	Log generated	Reported to SCP?	Error action performed
Connect_To_ Resource received and CAIN0801 not enabled	Conversation	CAIN102	Yes	Switch sends a **CTR_Clear** in conversation. ***ClearCause*** is set to `taskRefused` and the call is not cleared toward the controlling or passive leg.
Connect_To_ Resource received and CAIN0801 not enabled	Response	CAIN102	No	Switch applies AINF to the originator and idles the terminator.
Analyze_Route or **Collect_ Information** received in response to **CTR_Clear**	Conversation	CAIN200	Yes	Switch applies AINF to the originator and idles the terminator.
Analyze_Route or **Collect_ Information** received in response to **CTR_Clear**	Response	CAIN200	No	Switch applies AINF to the originator and idles the terminator.

Note: Refer to *UCS DMS-250 Logs Reference Manual* for information on logs.

—end—

Switch_Hook_Flash event

ATTENTION

The *Switch_Hook_Flash* event requires the CAIN0802 SOC option. Refer to Volume 5, Chapter 5, "NetworkBuilder SOC functionality," for more information.

Uses

The *Switch_Hook_Flash* event is detected when one of the following occurs:

- the passive leg (***LegID*** 1) presses the asterisk "*" key for 40 milliseconds during call configuration (CC) 2 (stable two-party call)
- the controlling leg (***LegID*** 0) presses the asterisk "*" key for 40 milliseconds during CC4 (three-party call in the setup phase), CC6 (party on hold), or CC10 (stable multi-party call)

The *Switch_Hook_Flash* event can only be detected after the switch has located an idle trunk member in the terminating trunk group.

When you provision NetworkBuilder services on the switch, the SCP can make major call processing decisions. You define the services offered on the SCP. The *Switch_Hook_Flash* event provides the take-back and transfer service. The take-back and transfer service is an N00 service and allows a terminating party to transfer a calling party to a third party or conference a third party into an existing two-party call.

Note 1: The ***O_Mid_Call*** EDP-Request message for the *Switch_Hook_Flash* event and the **Timeout** EDP-Request message cannot be armed at the same time. If the two messages are armed simultaneously, the switch generates a CAIN101 log and the ***O_Mid_Call*** EDP-Request message for the *Switch_Hook_Flash* event is ignored.

Note 2: In-switch reorigination and the *O_IEC_Reorigination* trigger are disallowed once the ***O_Mid_Call*** EDP for the *Switch_Hook_Flash* event has been enabled.

Note 3: For more information on call configurations, refer to Volume 3, Chapter 4, "Call Configuration Model."

Supported originating agencies

The *Switch_Hook_Flash* event supports the following originating agencies:

- DAL
- FGD
- PRI

Switch_Hook_Flash event (continued)

- SS7 Inter-IMT
- SS7 Global-IMT

Event evaluation and actions

Call processing detects the *Switch_Hook_Flash* event when the following conditions are met:

- the *Switch_Hook_Flash* event is active
- the switch has located an idle trunk member in the terminating trunk group
- a valid call leg in a valid call configuration presses the asterisk "*" key for 40 milliseconds or more (upto 300 milliseconds) .

  Note: Refer to the "Uses" section to see which call leg is valid in which call configurations.

The `switchHookFlashEnabledLegs` extension parameter indicates which call leg or legs the switch monitors for a switch hook flash. Refer to Volume 3, Chapter 12, "Incoming CAIN message parameters," for more information on the `switchHookFlashEnabledLegs` extension parameter.

When the switch detects the *Switch_Hook_Flash* event, the switch sends an **`O_Mid_Call`** EDP-Request message.

O_Mid_Call EDP-Request

The following table defines the **`O_Mid_Call`** EDP-Request message's parameters and the parameters' usage requirements.

Table 9-1
O_Mid_Call EDP-Request message parameters

Parameter	Usage	Definition
UserID	Required	Contains the network identity of the originating agent (Note)
BearerCapability	Required	Contains the bearer capability of the call when the message is built
CcID	Optional	Contains the call configuration identifier
NotificationIndicator	Optional	Contains the value FALSE to identify the message type as Request
ExtensionParameter	Optional	***ExtensionParameter***s require the CAIN0200 SOC option
termTrunkInfo	Optional	Contains the terminating trunk group number, trunk type, and trunk member number
Note: The ***UserID*** parameter is populated with the value of the user which invokes the service. Currently, only the terminator is capable of subscribing to the ***Switch_Hook_Flash*** EDP. Therefore, the ***UserID*** parameter is populated with the value for the terminator during call configuration (CC) 2 (stable two-party call). In call configurations following CC2, the ***UserID*** parameter is populated with the value for the controlling leg. Refer to Volume 3, Chapter 4, “Call Configuration Model,” for more information.		
—end—		

Once the request is built, NetworkBuilder formats the message and sends it to the message encoder. Once encoded, the message is sent to the SCP.

SCP response processing

The following messages are supported in response to an ***O_Mid_Call*** EDP-Request message for the *Switch_Hook_Flash* event:

- **Connect_To_Resource** message
- **Disconnect** message
- **Disconnect_Leg** message
- **Merge_Call** message
- **Originate_Call** message

NetworkBuilder arms EDPs when the switch receives a **Request_Report_BCM_Event** component within the **Disconnect**, **Disconnect_Leg** and **Merge_Call**, **Originate_Call** messages.

Switch_Hook_Flash event (end)

Refer to Volume 3, Chapter 10, "Incoming CAIN messages," for specific information regarding response messages; refer to Volume 3, Chapter 12, "Incoming CAIN message parameters," for detailed descriptions of the response parameters; refer to Volume 3, Chapter 3, "Event processing," for EDP-specific message information.

Connect_To_Resource

The **`Connect_To_Resource`** message requires the CAIN0801 Mid Call Services 2 SOC option to be enabled.

If at the *Switch_Hook_Flash* event the switch receives **`Connect_To_Resource`** message in a conversation package and the CAIN0801 is not enabled, the switch sends a conversational **`CTR_Clear`** message with the ***`ClearCause`*** parameter value set to `taskRefused`.

O_Mid_Call TDP

Call processing encounters the ***O_Mid_Call*** TDP when a reorigination indication is received at the UCS DMS-250 switch.

Authorization and allocation of reorigination resources occurs between the **Analyze_Information** PIC and the **Select_Route** PIC. The *O_IEC_Reorigination* trigger is evaluated twice during a call. First, for reorigination setup, to determine if NetworkBuilder needs to detect reorigination scenarios. Second, after reorigination indication is received to determine if the trigger criteria is met.

NetworkBuilder control of reorigination

O_Mid_Call provides the ability to override all switch reorigination controls. Table CAINGRP allows NetworkBuilder to control manual and auto reorigination through the OVRREORG option.

When the *O_IEC_Reorigination* trigger is first evaluated during reorigination setup where the OVRREORG option is datafilled against a CAIN group and the *O_IEC_Reorigination* trigger is enabled, table OIECREO is evaluated. If the trigger action is BLOCK, QUERY, or CONT_NOTRIG, reorigination is allowed regardless of any switch feature.

Once reorigination indication is received, table OIECREO is evaluated for the second time. If the trigger criteria is not met, and reorigination setup determines that NetworkBuilder controls reorigination, the BLOCK trigger action is assumed. If the trigger criteria is not met, and reorigination setup determines that NetworkBuilder is not controlling reorigination, reorigination is handled through normal switch features.

Note: Only **`Send_To_Resource`** reorigination is supported by NetworkBuilder for overriding. When the REORIG_RECEIVERS office parameter in table OFCVAR is set to DTMF_ONLY, then NetworkBuilder does not control reorigination.

Figure 9-1 provides reorigination control scenarios and resulting actions.

O_IEC_Reorigination trigger (continued)

Figure 9-1
Reorigination control scenarios

Reorigination allowed based on in-switch logic and datafill	Subscribed to O_IEC_Reorigination trigger	OVRREORG option datafilled against CAINGRP	Match found in table OIECREO and trigger action is QUERY, BLOCK or CONT_NOTRIG	Resulting actions
N	N	N	N	Reorigination is handled through normal switch features.
Y	N	N	N	Reorigination is handled through normal switch features.
N	Y	N	N	Reorigination is not allowed for the call.
N	Y	Y	N	Reorigination is not allowed for the call.
N	Y	Y	Y	Reorigination is allowed because the OVRREORG is set against the CAINGRP and a match is found in table IECREO. When a reorigination indication is received, the trigger action is then performed.
Y	Y	N	N	Reorigination is allowed, but there is no CAIN interaction since no match is found in table OIECREO.
Y	Y	N	Y	Reorigination is allowed and the trigger action is performed when a reorigination indication is received.
Y	Y	Y	N	Reorigination is allowed, but there is no CAIN interaction since no match is found in table OIECREO.
Y	Y	Y	Y	Reorigination is allowed and the trigger action is performed when a reorigination indication is received. The reorigination is controlled by CAINGRP datafill.

ATTENTION

The *O_IEC_Reorigination* trigger requires the CAIN0509 SOC option. Refer to Volume 5, Chapter 5, "NetworkBuilder SOC functionality," for more information.

Uses

When you provision NetworkBuilder services on the switch, the SCP can make major call processing decisions. You define the services offered on the SCP. An example of a service that can be developed on the SCP for the *O_IEC_Reorigination* trigger is reorigination.

O_IEC_Reorigination trigger (continued)

Supported originating agencies

The *O_IEC_Reorigination* trigger supports the following originating agencies:

- DAL
- FGD
- SS7 Inter-IMT
- AXXESS

Note: Refer to *UCS DMS-250 CAIN/FlexDial Interactions* for more information on support for AXXESS agents.

Subscribing to the O_IEC_Reorigination trigger

Subscription to the *O_IEC_Reorigination* trigger is available on a

- SCP-returned basis
- address basis (table STDPRTCT)
- authorization code basis (tables AUTHCODU, AUTHCDU2, AUTHCDU3, AUTHCDU4, or AUTHCDU5)
- ANI basis (table ANISCUSP, or tables ANIVAL and UNIPROF)
- agent basis (table TRKGRP)
- office basis (table CAINPARM)

Note 1: An SCP-returned CAIN subscription group is received in **`Analyze_Route`** messages. When a reorigination occurs after an **`Analyze_Route`** is received, the SCP-returned CAIN group is available for any reoriginated call.

Note 2: CAIN group subscription for AXXESS agents is handled differently. Refer to *UCS DMS-250 CAIN/FlexDial Interactions*.

Trigger evaluation

NetworkBuilder checks the *O_IEC_Reorigination* trigger table (OIECREO) and evaluates the identified digits against the datafilled range associated with the appropriate CAIN group.

Note: The digit type (information, ADIN, ANI, address, or CIC) is identified through datafill within the OIECREO table.

O_IEC_Reorigination trigger (continued)

Trigger actions

NetworkBuilder call processing supports the following actions for the *O_IEC_Reorigination* trigger:

- IGNORE – NetworkBuilder continues checking the next CAIN group subscription. When no more groups exist, reorigination is handled by normal switch features.
- BLOCK – The terminator is released. All appropriate reorigination call processing data is reset. AINF treatment is applied to the second call.
- QUERY – The switch builds an **`O_Mid_Call`** query and sends it to the SCP. The terminator is released. All appropriate reorigination call processing data is reset.
- LEAVE_TDP – NetworkBuilder call processing exits the TDP with no further evaluation. Reorigination is handled by normal switch features.
- CONT_NOTRIG – NetworkBuilder call processing exits the TDP and prevents any further NetworkBuilder interaction for the second call. Reorigination is handled by normal switch features.
- COLLINFO – NetworkBuilder call processing exits the TDP with no further evaluation. Reorigination is handled by normal switch features.

Options

The *O_IEC_Reorigination* trigger supports the following options:

- BUFFER – to activate digit buffering while the SCP is queried. Digits are used only for a subsequent **`Connect_To_Resource`** request.
- GT – to identify the global title used to identify the SCP handling the query.
- VERSION – controls the CAIN protocol version for outgoing messages

  Note: The value provisioned in the VERSION option overrides the version provisioned in the CAIN_PROTOCOL_VERSION parameter of table CAINPARM on a transaction-by-transaction basis. Additionally, the VERSION option applies to both the outgoing and incoming messages within the same transaction. (Subsequent queries will use the setting provisioned in the CAIN_PROTOCOL_VERSION or provisioned setting of the VERSION option in the trigger table performing the subsequent query.)

- STREAM – to control protocol stream on a per-trigger tuple basis. The value of the STREAM option controls the set of parameters that are sent in NetworkBuilder messages.

O_Mid_Call TDP-Request

An **`O_Mid_Call`** TDP-Request (query) is sent to the SCP in a query package, with a component type of Invoke_Last. The following table defines the parameters and usage requirements for the parameters the **`O_Mid_Call`** TDP-Request may contain. Refer to Volume 3, Chapter 8, "Outgoing CAIN message parameters," for detailed descriptions of the request parameters.

Note: Parameters and extension parameters for AXXESS agents are handled differently. Refer to *UCS DMS-250 CAIN/FlexDial Interactions* for more information.

O_Mid_Call TDP-Request message parameters for O_IEC_Reorigination

Parameter	Usage	Definition
UserID	Required	Contains the network identity of the originating agent
BearerCapability	Required	Contains the bearer capability of the call when the message is built
TriggerCriteriaType	Optional	Contains OIECReorigination
LegID	Optional	Contains the value 0 which indicates the controlling leg
PointInCall	Optional	Contains the point in call (PIC) where the trigger was checked
CcID	Optional	Contains the call configuration identifier `originatingSetup`

Note 1: The `universalAccess` extension parameter is included in the query (a flat 10 digits) when parameter CAIN_PROTOCOL_VERSION is V0. When CAIN_PROTOCOL_VERSION is set to V1 or higher, the parameter contains up to 24 digits including a nature of address and numbering plan similarly to the ***CalledPartyID***.
Note 2: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS06 or higher.
Note 3: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS07 or higher.
Note 4: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS08 or higher.
Note 5: This parameter is not populated when the CAIN_PROTOCOL_VERSION is set to V3 or higher, and is therefore not sent.
Note 6: The parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS09 or higher.
Note 7: When the CAIN_PROTOCOL_VERSION is set to V4 or higher non-standard charge numbers (CARD, AUTH, ACCT, PIN, and N00) no longer populate the ***ChargeNumber*** parameter and all standard NOAs are supported.
Note 8: Refer to Volume 3, Chapter 8, "Outgoing CAIN message parameters" for information on the population of this parameter when the CAIN_PROTOCOL_VERSION is set to V5 or higher.

—continued—

O_IEC_Reorigination trigger (continued)

O_Mid_Call TDP-Request message parameters for O_IEC_Reorigination (continued)

Parameter	Usage	Definition
ChargeNumber (Note 7)	Optional	Contains the billing number that would be used to populate the CDR at this point in call processing
CallingPartyID (Note 8)	Optional	Contains one of the following (listed in order of precedence): For SS7 FGD, SS7 Inter-IMT, SS7-Global IMT, and AXXESS calls: Calling_Party_Address from ISUP message, when available For FGD and AXXESS calls: valid ANI (information digits are not passed in this parameter) ***Note:*** Refer to *UCS DMS-250 CAIN/FlexDial Interactions* for more information on AXXESS agents. Valid SNPA value from table TRKGRP Default SNPA value from table CAINPARM
ChargePartyStationType (Note 3)	Optional	Contains the information digits for the call
Carrier	Optional	Contains the dialed CIC or the CIC value (with an indication of: Selected CIC presubscribed and not input by calling party) from table TRKGRP.
AccessCode(Note 5)	Optional	Contains the account code, when available

Note 1: The `universalAccess` extension parameter is included in the query (a flat 10 digits) when parameter CAIN_PROTOCOL_VERSION is V0. When CAIN_PROTOCOL_VERSION is set to V1 or higher, the parameter contains up to 24 digits including a nature of address and numbering plan similarly to the ***CalledPartyID***.
Note 2: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS06 or higher.
Note 3: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS07 or higher.
Note 4: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS08 or higher.
Note 5: This parameter is not populated when the CAIN_PROTOCOL_VERSION is set to V3 or higher, and is therefore not sent.
Note 6: The parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS09 or higher.
Note 7: When the CAIN_PROTOCOL_VERSION is set to V4 or higher non-standard charge numbers (CARD, AUTH, ACCT, PIN, and N00) no longer populate the ***ChargeNumber*** parameter and all standard NOAs are supported.
Note 8: Refer to Volume 3, Chapter 8, “Outgoing CAIN message parameters” for information on the population of this parameter when the CAIN_PROTOCOL_VERSION is set to V5 or higher.

—continued—

O_IEC_Reorigination trigger (continued)

O_Mid_Call TDP-Request message parameters for O_IEC_Reorigination (continued)

Parameter	Usage	Definition
CollectedAddressInfo	Optional	Contains the address collected from the incoming agent (from the IAM, subscriber dialing, or datafilled hotline digits)
VerticalServiceCode (Note 2)	Optional	Contains feature codes dialed by the subscriber
ACGEncountered	Optional	Contains the ACG control encountered when the ACG control expires and a query is allowed to be sent
ExtensionParameter (Note 3)	Optional	Extension parameters require the CAIN0200 SOC option.
universalAccess (Note 1)	Optional	Contains the universal access number dialed by the caller to obtain switch dial tone.
cainGroup	Optional	Contains the group number (field GRPNUM, table CAINGRP) for the CAIN group associated with the trigger when field EXTPARM in table CAINGRP contains CAINGRP.
origTrunkInfo	Optional	Contains the originating trunk group number, trunk type, and trunk member number when field EXTPARM in table CAINGRP contains ORGTINFO.
treatment (Note 2)	Optional	Contains the treatment set by regular call processing before the query was sent

Note 1: The universalAccess extension parameter is included in the query (a flat 10 digits) when parameter CAIN_PROTOCOL_VERSION is V0. When CAIN_PROTOCOL_VERSION is set to V1 or higher, the parameter contains up to 24 digits including a nature of address and numbering plan similarly to the ***CalledPartyID***.
Note 2: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS06 or higher.
Note 3: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS07 or higher.
Note 4: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS08 or higher.
Note 5: This parameter is not populated when the CAIN_PROTOCOL_VERSION is set to V3 or higher, and is therefore not sent.
Note 6: The parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS09 or higher.
Note 7: When the CAIN_PROTOCOL_VERSION is set to V4 or higher non-standard charge numbers (CARD, AUTH, ACCT, PIN, and N00) no longer populate the ***ChargeNumber*** parameter and all standard NOAs are supported.
Note 8: Refer to Volume 3, Chapter 8, “Outgoing CAIN message parameters” for information on the population of this parameter when the CAIN_PROTOCOL_VERSION is set to V5 or higher.

—continued—

O_IEC_Reorigination trigger (continued)

O_Mid_Call TDP-Request message parameters for O_IEC_Reorigination (continued)

Parameter	Usage	Definition
`netinfo` (Note 3)	Optional	Contains external network ID, network customer group ID, and network class of service.
`subscriptionInfo` (Note 4)	Optional	Contains the digit type the switch triggered on and which subscription method was in use when the query occurred
`numReorig`	Optional	Contains the number of times a user has reoriginated.
`jurisdictionInfo` (Note 4)	Optional	Contains the originating switch's LRN.
`switchID` (Note 6)	Optional	Contains the switch ID
`accountCode` (Note 6)	Optional	Contains the account code when available
`billingNumber` (Note 6)	Optional	Contains the non-standard charge number
`acgRequery`	Optional	When present, indicates to the SCP that the current query is a re-query message resulting from the original query being blocked by an ACG control

Note 1: The `universalAccess` extension parameter is included in the query (a flat 10 digits) when parameter CAIN_PROTOCOL_VERSION is V0. When CAIN_PROTOCOL_VERSION is set to V1 or higher, the parameter contains up to 24 digits including a nature of address and numbering plan similarly to the ***`CalledPartyID`***.
Note 2: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS06 or higher.
Note 3: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS07 or higher.
Note 4: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS08 or higher.
Note 5: This parameter is not populated when the CAIN_PROTOCOL_VERSION is set to V3 or higher, and is therefore not sent.
Note 6: The parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS09 or higher.
Note 7: When the CAIN_PROTOCOL_VERSION is set to V4 or higher non-standard charge numbers (CARD, AUTH, ACCT, PIN, and N00) no longer populate the ***`ChargeNumber`*** parameter and all standard NOAs are supported.
Note 8: Refer to Volume 3, Chapter 8, "Outgoing CAIN message parameters" for information on the population of this parameter when the CAIN_PROTOCOL_VERSION is set to V5 or higher.

—end—

Once the request message is built, NetworkBuilder formats the message and sends it to the message encoder. Once encoded, the message is sent to the SCP.

O_IEC_Reorigination trigger (continued)

SCP response processing

The following response messages are supported:

- **Analyze_Route**
- **Disconnect**
- **Collect_Information**
- **Connect_To_Resource**

Note 1: EDPs may be armed through the **Request_Report_BCM_Event** component with the **Analyze_Route** and **Collect_Information** messages in response to the **O_Mid_Call** TDP-Request message.

Note 2: A **Termination_Notification** may be requested through the **Send_Notification** component with the SCP response message.

Note 3: An **ACG** message may be sent with the SCP response message.

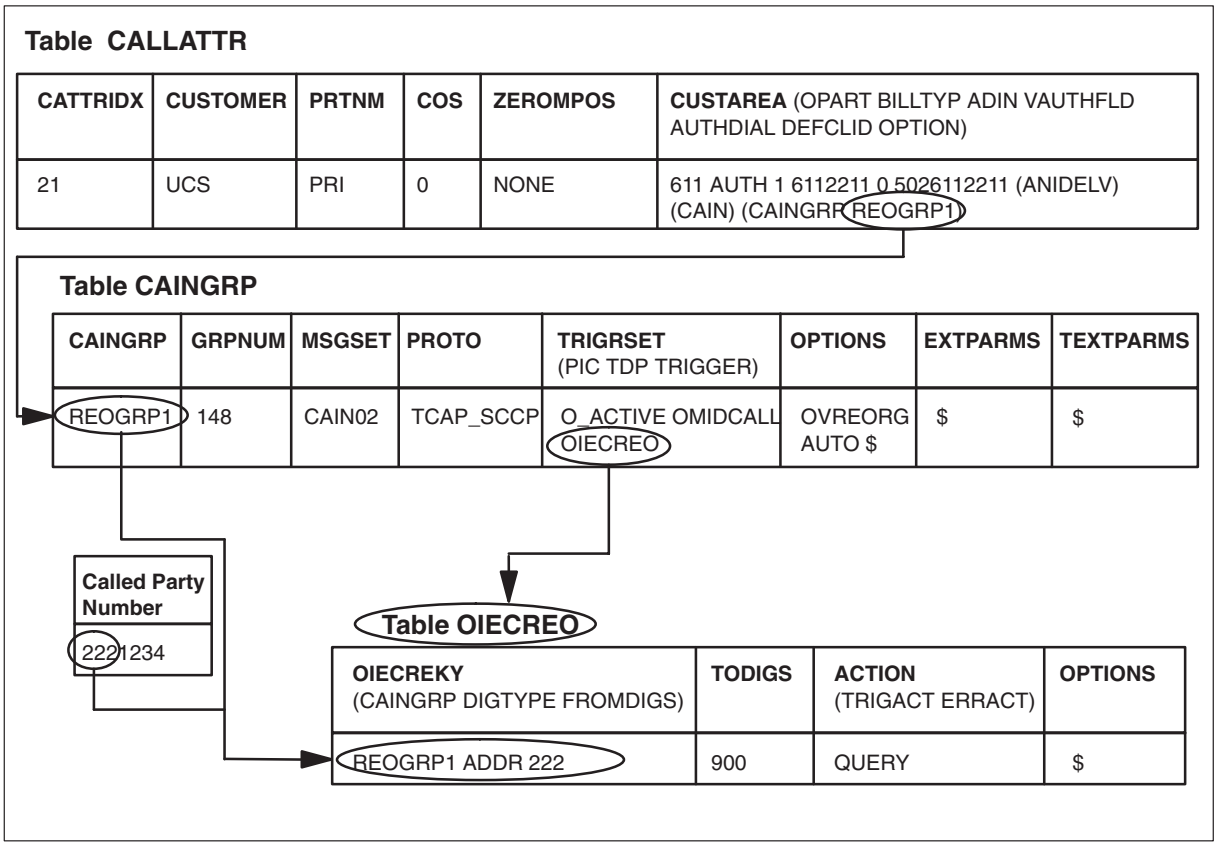
Refer to Volume 3, Chapter 10, "Incoming CAIN messages," for specific information regarding response messages; refer to Volume 3, Chapter 12, "Incoming CAIN message parameters," for detailed descriptions of the response parameters; refer to Volume 3, Chapter 3, "Event processing," for EDP specific message information.

Datafill

The following figure shows how a subscription table interacts with the *O_IEC_Reorigination* trigger table (OIECREO).

O_IEC_Reorigination trigger (continued)

Subscription-OIECREO table interaction



Provisioning NetworkBuilder control of reorigination

At the CI prompt

1 Enter table CAINGRP.

Note: Table CAINGRP must be datafilled prior to defining NetworkBuilder subscription.

2 **>REP caingrp msgset proto trigrset options extparms textparms $**
where

caingrp is the name of the CAIN group (0–16 alphanumeric characters).
grpnum is the number associated with the CAIN group (0 to 4095).
msgset is the message set (CAIN02, IN1).
proto is the message protocol (TCAP_SCCP).
trigrset is the trigger when **msgset** is IN1.
is a multiple-entry vector comprised of three subfields (PIC, TDP, TRIGGER) when **msgset** is CAIN02. Enter O_ACTIVE OMIDCALL OIECREO, for NetworkBuilder reorigination control.
PIC is the point in call defined for this CAIN group.

O_IEC_Reorigination trigger (continued)

TDP is the trigger detection point defined for this CAIN group and PIC.

TRIGGER is the trigger being defined for this CAIN group, PIC, and TDP.

Note: A CAINGRP may subscribe to more than one trigger.

options is an options vector. Enter OVRREORG for NetworkBuilder reorigination control.The OVRREORG option includes the following subfields which specify the type of reorigination allowed:

AUTO indicates automatic reorigination.

MANUAL indicates manual reorigination. Manual reorigination requires a reorigination key to be datafilled (AST or OCT).

extparms identifies the extension parameters to be sent in the originating call model query messages.

textparms identifies the extension parameters to be sent in the terminating call model query messages.

Sample entry: **>REP reogrp1 147 cain02 tcap_sccp o_active omidcall oiecreo $ ovreorg manual ast $**

Sample entry: **>REP reogrp2 148 cain02 tcap_sccp o_active omidcall oiecreo $ ovreorg auto $**

NetworkBuilder control of reorigination is defined.

Provisioning the O_IEC_Reorigination trigger

At the CI prompt

1 Provision the originating agent as CAIN-capable.

2 Subscribe to a CAIN group (table STDPRTCT, authcode tables, ANI tables, TRKGRP, CALLATTR, or CAINPARM)

Note: CAIN group subscription is handled differently for AXXESS agents. Refer to *UCS DMS-250 CAIN/FlexDial Interactions*.

3 Datafill CAIN group trigger subscription to the (O_ACTIVE OMIDCALL OIECREO) trigger set (table CAINGRP).

4 Enter table OIECREO.

5 Define the trigger criteria for a CAIN group by using the following format:

>ADD oiecreky todigs action options
where

oiecreky is comprised of three subfields: CAINGRP, DIGTYPE, and FROMDIGS, where

CAINGRP is the CAIN group requiring *O_IEC_Reorigination* trigger criteria (from table CAINGRP).

O_IEC_Reorigination trigger (end)

DIGTYPE is the digit type being analyzed (INFO, ADIN, ANI, ADDR, CIC).

FROMDIGS is the first number used to define the range of the collected address.

todigs is the second number used to define the range of the collected address.

action is comprised of one subfield: TRIGACT, where

TRIGACT is the trigger action taken when the address is within the FROMDIGS-TODIGS range (BLOCK, IGNORE, QUERY, LEAVE_TDP, CONT_NOTRIG).

6 Datafill the options, where

options is only allowed when ACTION is QUERY. Enter up to 4 options: BUFFER , GT, VERSION, or STREAM. When the ACTION is not QUERY enter $.

Sample entry: **>ADD reogrp1 ADDR 223 900 query treat buffer $**

O_IEC_Reorigination trigger criteria is defined.

Associated OMs

CAINMSGS, CAINTRIG, CAINAGOM

T_Null PIC

Call processing enters **T_Null** when indication of a desire to deliver a call is received from the originator.

Termination_Attempt TDP

Call processing encounters ***Termination_Attempt*** when a terminator is selected.

Note: When the *Termination_Attempt* trigger is encountered, the switch has not yet determined whether the selected agent is idle or not.

The *Termination_Attempt* trigger may be encountered multiple times during the processing of a call due to route advancing. On each attempt to terminate, or route advance, this trigger may be encountered.

Termination_Attempt trigger (continued)

ATTENTION

The *Termination_Attempt* trigger requires the CAIN0510 SOC option. Refer to Volume 5, Chapter 5, "NetworkBuilder SOC functionality," for more information.

Uses

When you provision NetworkBuilder services on the switch, the SCP can make major call processing decisions. You define the services offered on the SCP. Caller ID delivery is an example of a service that can be developed on the SCP for the *Termination_Attempt* trigger.

Supported terminating agencies

The *Termination_Attempt* trigger supports the following terminating agencies:

- DAL
- FGB
- FGC
- FGD
- PRI
- IMT
- AXXESS

  Note: Refer to *UCS DMS-250 CAIN/FlexDial Interactions* for more information on support for AXXESS agents.

Subscribing to the Termination_Attempt trigger

Subscription to the *Termination_Attempt* trigger is available on a terminating agent basis by datafilling the TCAIN and TCAINGRP options against the terminating agent (through table TRKGRP, CALLATTR, or TRKFEAT).

Note: The originator for the call is not required to be CAIN-capable in order to support the *Termination_Attempt* trigger.

Trigger evaluation

NetworkBuilder checks the *Termination_Attempt* trigger table (TERMATT) and evaluates the identified digits against the datafilled range associated with the appropriate CAIN group.

Note: The digit type (XLAADDR) is identified through datafill within the TERMATT table.

Termination_Attempt trigger (continued)

Trigger actions

NetworkBuilder call processing supports the following actions for the *Termination_Attempt* trigger:

- BLOCK – AINF treatment is applied to the call originator. Refer to Volume 4, Chapter 1, "Conversational processing," for more information on interactions between the terminating call model and outstanding conversational message processing.
- IGNORE – NetworkBuilder exits the current level of CAINGRP subscription and continues executing the terminating call model.
- QUERY – Switch builds a **`Termination_Attempt`** query and sends it to the SCP. The SCP analyzes the call and assists the switch with call processing.
- LEAVE_TDP – NetworkBuilder call processing exits the ***Termination_Attempt*** TDP with no further evaluation, call processing continues through the call model.
- CONT_NOTRIG – NetworkBuilder call processing exits the ***Termination_Attempt*** TDP and prevents any further NetworkBuilder interaction with this instance of the terminating call model for the remainder of the call.

  Note 1: When a route advance occurs (due to NetworkBuilder processing, or standard in-switch processing) a new instance of the terminating call model is invoked. CONT_NOTRIG does not affect the ability to trigger in this situation.

  Note 2: CONT_NOTRIG, LEAVE_TDP, and IGNORE perform the same function for the ***Termination_Attempt*** TDP since there are no other terminating call model triggers supported.

Error actions

When a fatal application occurs during an SCP query, one of the following error actions is performed (as datafilled):

- TREAT – AINF treatment is applied to the call originator.
- ROUTE – An error action of ROUTE is processed as if an IGNORE trigger action had been encountered and the switch will attempt to terminate the call as if no NetworkBuilder processing had been performed.

Options

Termination_Attempt supports the following options:

- BUFFER – to activate digit buffering while the SCP is queried.

Termination_Attempt trigger (continued)

- GT – to identify the global title used to identify the SCP handling the query.
- T1OVFLGT – to identify the specific SCP to query on T1 overflow.
- ACGOVFLGT – to identify the global title to use for requerying when a query is blocked by an ACG control
- VERSION – controls the CAIN protocol version for outgoing messages

  Note: The value provisioned in the VERSION option overrides the version provisioned in the CAIN_PROTOCOL_VERSION parameter of table CAINPARM on a transaction-by-transaction basis. Additionally, the VERSION option applies to both the outgoing and incoming messages within the same transaction. (Subsequent queries will use the setting provisioned in the CAIN_PROTOCOL_VERSION or provisioned setting of the VERSION option in the trigger table performing the subsequent query.)

Termination_Attempt trigger (continued)

The following figure shows how a call progresses through the terminating call model, encounters ***Termination_Attempt*** and queries the SCP.

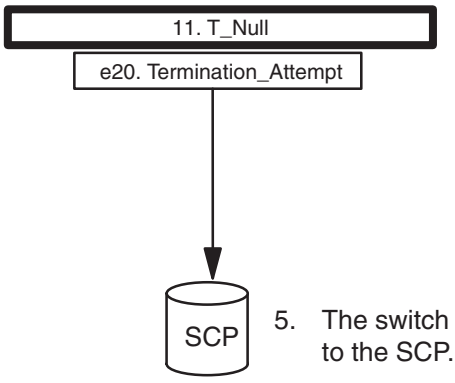
Figure 10-1
DAL termination call

DAL termination **Signalling:** FXO and FXS

Datafill:

Table TERMATT

TERMATTKY	TODIGS	ACTION	ERROR_ACTION	OPTIONS
TERMGRP1 XLADDR 214	214	QUERY	ROUTE	BUFFER $



1. The switch detects an indication of a desire to deliver a call from the originator and enters **T_Null**.
2. The switch encounters ***Termination_Attempt***.
3. Datafill enables *Termination_Attempt* for TERMGRP1.
4. Trigger criteria is met in table TERMATT.
5. The switch builds an **`Termination_Attempt`** query and sends it to the SCP.

Termination_Attempt TDP-Request

A **`Termination_Attempt`** TDP-Request (query) is sent to the SCP in a query package, with a component type of Invoke_Last. The following table defines the parameters and usage requirements for the parameters the **`Termination_Attempt`** query may contain. Refer to Volume 3, Chapter 8, "Outgoing CAIN message parameters," for detailed descriptions of the request parameters.

Note: Parameter and extension parameter population may differ for AXXESS agents. Refer to *UCS DMS-250 CAIN/FlexDial Interactions* for more information.

Termination_Attempt trigger (continued)

Table 10-1
Termination_Attempt TDP-Request message parameters

Parameter	Usage	Definition
UserID	Required	Contains the network identity of the terminating agent
BearerCapability	Required	Contains the bearer capability of the call when the message is built
CalledPartyID	Optional	Contains the translated address
TriggerCriteriaType	Optional	Contains `terminationAttempt`
ChargeNumber (Note 4)	Optional	Contains the digits in field VAUTHFLD of table TRKGRP, when available
CallingPartyID (Note 5)	Optional	Contains the one of the following (listed in order of precedence): For SS7 FGD, SS7 Inter-IMT, and SS7 Global-IMT calls: Calling_Party_Address from ISUP message, when available For PRI calls: CLID when available For FGD and AXXESS calls: valid ANI (information digits are not passed in this parameter) ***Note:*** Refer to *UCS DMS-250 CAIN/FlexDial Interactions* for more information on AXXESS agents. Valid SNPA value from table TRKGRP Default SNPA value from table CAINPARM

Note 1: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS07 or higher.
Note 2: The parameter requires the CAIN_PROTOCOL_STREAM to be UCS08 or higher.
Note 3: The parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS09 or higher.
Note 4: When the CAIN_PROTOCOL_VERSION is set to V4 or higher non-standard charge numbers (CARD, AUTH, ACCT, PIN, and N00) no longer populate the ***ChargeNumber*** parameter and all standard NOAs are supported.
Note 5: Refer to Volume 3, Chapter 8, “Outgoing CAIN message parameters” for information on the population of this parameter when the CAIN_PROTOCOL_VERSION is set to V5 or higher.
Note 6: The parameter is supported only on DAL, FGD, and AXXESS agencies and the parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS11 or higher.

—continued—

Termination_Attempt trigger (continued)

Table 10-1
Termination_Attempt TDP-Request message parameters (continued)

Parameter	Usage	Definition
ChargePartyStationType (Note 1)	Optional	Contains the information digits for the call
ACGEncountered	Optional	Contains the ACG control encountered when the ACG control expires and a query is allowed to be sent
ExtensionParameter	Optional	Extension parameters require the CAIN0200 SOC option.
cainGroup	Optional	Contains the group number (field GRPNUM, table CAINGRP) for the CAIN group associated with the trigger when field TEXTPARM in table CAINGRP contains CAINGRP
origTrunkInfo	Optional	Contains the originating trunk group number, trunk type, and trunk member number when field TEXTPARM in table CAINGRP contains ORGTINFO
reorigCall	Optional	Presence of this parameter indicates the call in progress is a result of reorigination
termTrunkInfo	Optional	Contains the terminating trunk group number, trunk type, and trunk member number when field TEXTPARM in table CAINGRP contains TERMTINFO
t1Overflow	Optional	Indicates to the SCP that an overflow occurred on the query to the initial SCP
subscriptionInfo (Note 2)	Optional	Contains the digit type the switch triggered on and which subscription method was in use when the query occurred

Note 1: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS07 or higher.
Note 2: The parameter requires the CAIN_PROTOCOL_STREAM to be UCS08 or higher.
Note 3: The parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS09 or higher.
Note 4: When the CAIN_PROTOCOL_VERSION is set to V4 or higher non-standard charge numbers (CARD, AUTH, ACCT, PIN, and N00) no longer populate the ***ChargeNumber*** parameter and all standard NOAs are supported.
Note 5: Refer to Volume 3, Chapter 8, “Outgoing CAIN message parameters” for information on the population of this parameter when the CAIN_PROTOCOL_VERSION is set to V5 or higher.
Note 6: The parameter is supported only on DAL, FGD, and AXXESS agencies and the parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS11 or higher.

—continued—

Termination_Attempt trigger (continued)

Table 10-1
Termination_Attempt TDP-Request message parameters (continued)

Parameter	Usage	Definition
`switchID` (Note 3)	Optional	Contains the switch ID
`billingNumber` (Note 3)	Optional	Contains the non-standard charge number
`acgRequery`	Optional	When present, indicates to the SCP that the current query is a re-query message resulting from the original query being blocked by an ACG control
Lata (Note 6)	Optional	Contains the call's Local Access and Transport Area (LATA)

Note 1: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS07 or higher.
Note 2: The parameter requires the CAIN_PROTOCOL_STREAM to be UCS08 or higher.
Note 3: The parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS09 or higher.
Note 4: When the CAIN_PROTOCOL_VERSION is set to V4 or higher non-standard charge numbers (CARD, AUTH, ACCT, PIN, and N00) no longer populate the ***ChargeNumber*** parameter and all standard NOAs are supported.
Note 5: Refer to Volume 3, Chapter 8, "Outgoing CAIN message parameters" for information on the population of this parameter when the CAIN_PROTOCOL_VERSION is set to V5 or higher.
Note 6: The parameter is supported only on DAL, FGD, and AXXESS agencies and the parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS11 or higher.

—end—

Once the query message is built, NetworkBuilder formats the message and sends it to the message encoder. Once encoded, the message is sent to the SCP.

SCP response processing

NetworkBuilder supports the following response messages:

- **`Authorize_Termination`**
- **`Send_To_Resource`**
- **`Disconnect`**

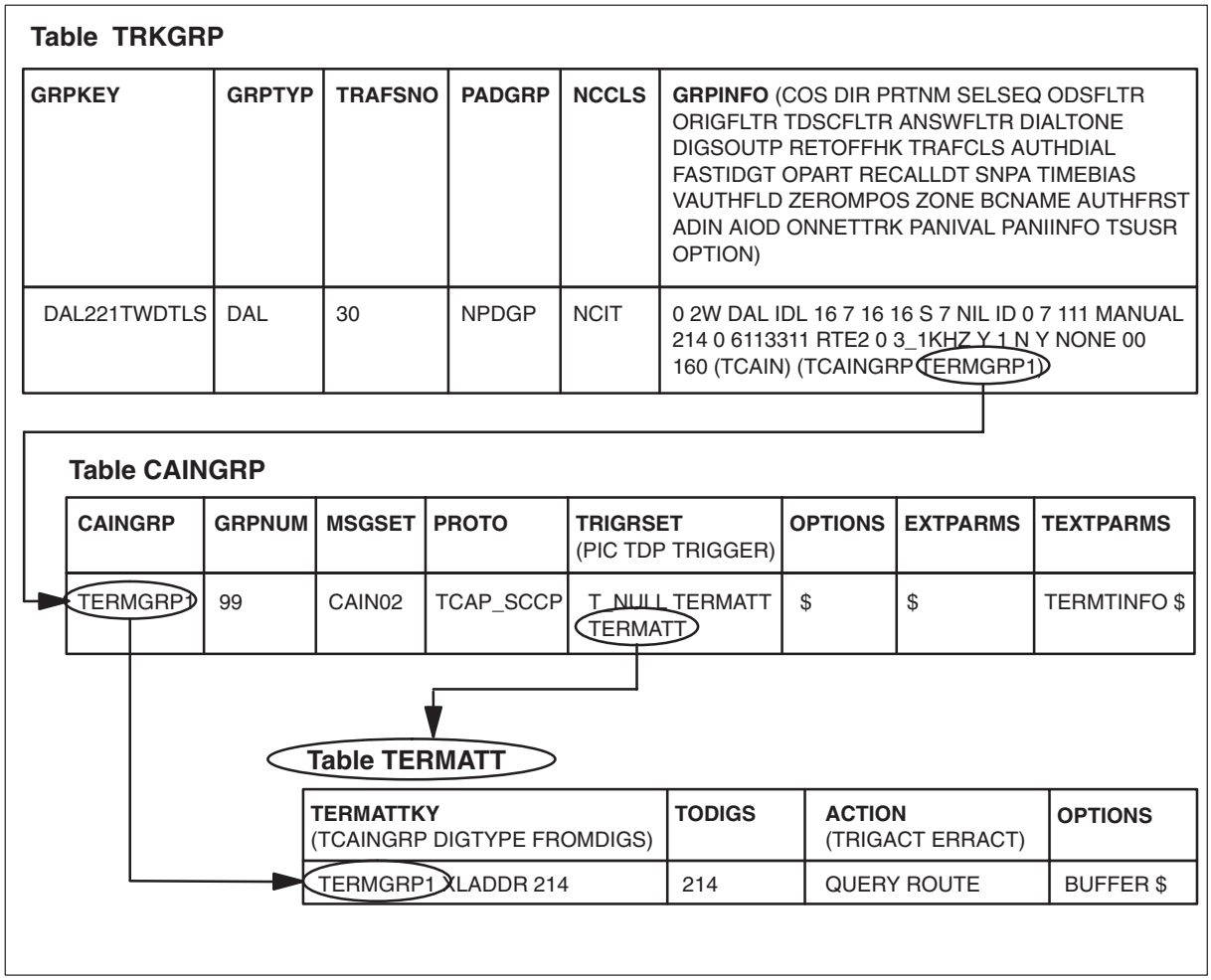
Note 1: A **`Termination_Notification`** may be requested through the **`Send_Notification`** component with the SCP response message.

Note 2: An **`ACG`** message may be sent with the SCP response message.

Datafill

The following figure shows how a subscription table interacts with the *Termination_Attempt* trigger table (TERMATT).

Figure 10-2
Subscription-TERMATT table interaction



Provisioning the Termination_Attempt trigger

At the CI prompt

1 Provision the terminating agent as TCAIN-capable.

2 Subscribe to a terminating CAIN group (table TRKGRP, CALLATTR, or TRKFEAT).

Note: CAIN group subscription is handled differently for AXXESS agents. Refer to *UCS DMS-250 CAIN/FlexDial Interactions*.

Termination_Attempt trigger (end)

3 Datafill CAIN group trigger subscription to the (T_NULL TERMATT TERMATT) trigger set (table CAINGRP).

4 Enter table TERMATT.

5 Define the trigger criteria for a terminating CAIN group by using the following format:

>ADD termattky action options
where

termattky is comprised of three subfields, where
- TCAINGRP is the terminating CAIN group trigger criteria (from table CAINGRP).
- DIGTYPE is the digit type being analyzed (XLAADDR).
- FROMDIGS is vector of up to 24 of digits (0 to 9).

action is comprised of one subfield: TRIGACT, where
- TRIGACT is the trigger action taken (BLOCK, IGNORE, QUERY, LEAVE_TDP, CONT_NOTRIG).

If you datafill TRIGACT as:	Go to:
BLOCK	step 7
IGNORE	step 7
QUERY	step 6
LEAVE_TDP	step 7
CONT_NOTRIG	step 7

6 Datafill the ERRACT refinement, where

ERRACT is the error action taken when a fatal application error occurs (TREAT, ROUTE).

7 Datafill the OPTIONS refinement, where

options is only allowed when ACTION is QUERY. Enter up to 6 options: BUFFER, GT, T1OVFLGT, ACGOVFLGT, VERSION, or STREAM. When the ACTION is not QUERY enter $.

Sample entry: **>ADD termgrp1 xlaaddr 214 214 query route buffer $**

Termination_Attempt trigger criteria is defined.

Associated OMs

CAINTRIG, CAINMSGS, CAINAGOM

Appendix A
Service migration

In moving to a CAIN-based network, you will be off-loading some of the normal call processing functions to the SCP. Preparing your SCP to perform these functions requires a thorough knowledge of the function, the switch requirements, the data provided to the SCP from the switch, and an idea of what special services you plan to offer.

This appendix explains the migration from an IN/1-based N00 service to a CAIN-based N00 service. The same methodology used in the migration example, can be used in preparing to migrate other services.

N00 services

Service access codes (also known as N00 services) are usually used to access a service instead of a person. These services are usually implemented on 500, 700, 800, 888, or 900 numbers, although any number can be specified for N00 services.

Initially, N00 numbers were translated in-switch and routed to a national number. With the implementation of IN/1 services, these translations were off-loaded to an IN/1 database (also known as a data control point [DCP]).

Carrier AIN takes the evolution of N00 services one step further by allowing call processing (translations and routing decisions) to be performed at an intelligent SCP.

Universal International Free Phone (UIFN) can be done and is illustrated and in the sections that follow.

Current IN/1 functionality

IN/1 processing translates the N00 number received during setup messaging at the IN/1 DCP. The data is passed from the switch to the DCP using TCAP CCS7 messaging.

A DCP query is initiated when field PRETRTE (table STDPRTCT, subtable STDPRT) is datafilled as ES SACREMOT.

CAIN functionality

CAIN functionality provides the following benefits over current IN/1 functionality:

- Additional data is supplied to the SCP.
- An extensive trigger checking logic similar to translations.
- The ability to return the call to a pre-query state (for ***Info_Analyzed*** triggers)
- The ability to return multiple route choices.
- The ability to apply a customized announcement or tone on a per-call basis.
- Customized branding before routing.
- Provisionable actions for the switch to take on fatal application errors.
- One TCAP subsystem handling multiple services (such as N00, VPN, and ANI screening).

Provisioning N00 services

This section addresses N00 services provisioning for IN/1 and CAIN implementations.

For IN/1

Datafill table STDPRTCT, subtable STDPRT, field PRETRTE as ES SACREMOT. Provision the DCP response timer in table OFCVAR, parameter N00_DCP_RESPONSE_TIMEOUT. Provision the no answer timer in table OFCVAR, parameter N00_NO_ANSWER_TIMER.

For CAIN

The following procedure is an example of how to set up the switch to query for the following:

- Originating agent: DAL221TWDTLS
- Called party: 800-555-1111

Implementing CAIN services

1 Determine which trigger set will work best for your N00 service.

Use one of the ***Info_Collected*** or ***Info_Analyzed*** triggers for N00 services. ***Info_Collected*** triggers are good choices because, digit collection has occurred, but the in-switch translations have not. Since your SCP service is providing translations, there is no need for the switch to also perform the translation. Also, no IN/1 query is performed.

At the CI prompt

2 Enter table CAINGRP and create a new CAIN group using the ***Info_Collected*** trigger set:

Sample entry: **>ADD n00grp 1 cain02 tcap_sccp collinfo infocoll siotrk collinfo infocoll pribchnl**

3 Enter table TRKGRP and provision the required originating agents as CAIN-capable by specifying the CAIN option.

Note 1: *PRI_B-Channel* trigger requires the appropriate call attribute (table CALLATTR) be provisioned as CAIN-capable.

Note 2: AXXESS agents require the appropriate agent (table TRKFEAT) be provisioned as CAIN-capable.

4 Identify the best subscription method to use for the N00 service on your SCP.

Office subscription (table CAINPARM, parameter CAIN_OFFICE_GROUP) is a good choice when you want all incoming calls on CAIN-capable agents to be able to access the N00 services on your SCP.

Agent subscription (through table TRKGRP or CALLATTR) is a good choice when you want to specify incoming calls on particular agents as being able to access the N00 services on your SCP.

5 Enter table SIOTRK and set up the trigger criteria.

Sample entry: **>ADD n00grp addr 800 800 query sio_n00 cain_addr_gt $**

UIFN Sample entry: **>ADD n00grp addr 011800 011800 query sio_n00 cain_addr_gt $**

Note: In most cases, you will want calls requiring N00 services on the SCP to trigger off the address.

For Info_Collected

6 Enter table STDPRTCT, subtable STDPRT and datafill for remote translations:

Sample entry: **>ADD 800 800 es sacremot $**

UIFN Sample entry: **>ADD 011800 011800 es sacremot $**

Note: Datafill ES SACREMOT in table STDPRTCT, so that additional digit collection (due to ANI or authcode provisioning) is not performed. Therefore, subscribed calls will always trigger since IN/1 queries don't occur until **Analyze_Information**.

For Info_Analyzed

7 Enter table STDPRTCT, subtable STDPRT and datafill for remote translations:

Sample entry: **>ADD 800 800 ct offnet $**

UIFN Sample entry: **>ADD 011800 011800 nt dd 3 in $**

Note: Datafill CT OFFNET or NT in table STDPRTCT, so that an IN/1 query won't be performed on the call. By provisioning table STDPRTCT in this manner, additional digit collection (as provisioned by ANI or authcode) is still performed.

8 Enter table CAINPARM and provision the following parameters:

- ACG_OVERFLOW_GT
- ACG_TREATMENT
- CAIN_DEFAULT_GT (valid values are: CAIN_CLID_GT, CAIN_ADDR_GT, CAIN_FEAT_GT)

  Note: When CAIN_DEFAULT_GT is set to CAIN_CLID_GT, the CLID_GT_FORMAT parameter also needs to be provisioned. When the CAIN_DEFAULT_GT parameter is set to CAIN_ADDR_GT, the ADDR_GT_FORMAT parameter also needs to be provisioned.

- CAIN_DEFAULT_OVERFLOW_GT
- CAIN_PROTOCOL_STREAM
- CAIN_PROTOCOL_VERSION
- CAIN_T1_TIMEOUT (valid range is 0–30)
- INFOANALYZED_FOR_RLT
- NUM_CAIN_EXT_BLOCKS (valid range is 0–32767)
- NUM_CAIN_ECCBS
- NUM_FRAMEWORK_EXT_BLOCKS (valid range is 0–32767)
- NUM_SEND_NOTIFICATION_EXT_BLOCKS
- SEND_CARRIER_FROM_TRKGRP

Note: Refer to Chapter 2, “Provisioning CAIN,” for more information.

9 Enter table VAMPTRID and provision the VAMP resources:

Sample entry: **>REP cain02 30 30 30 512**

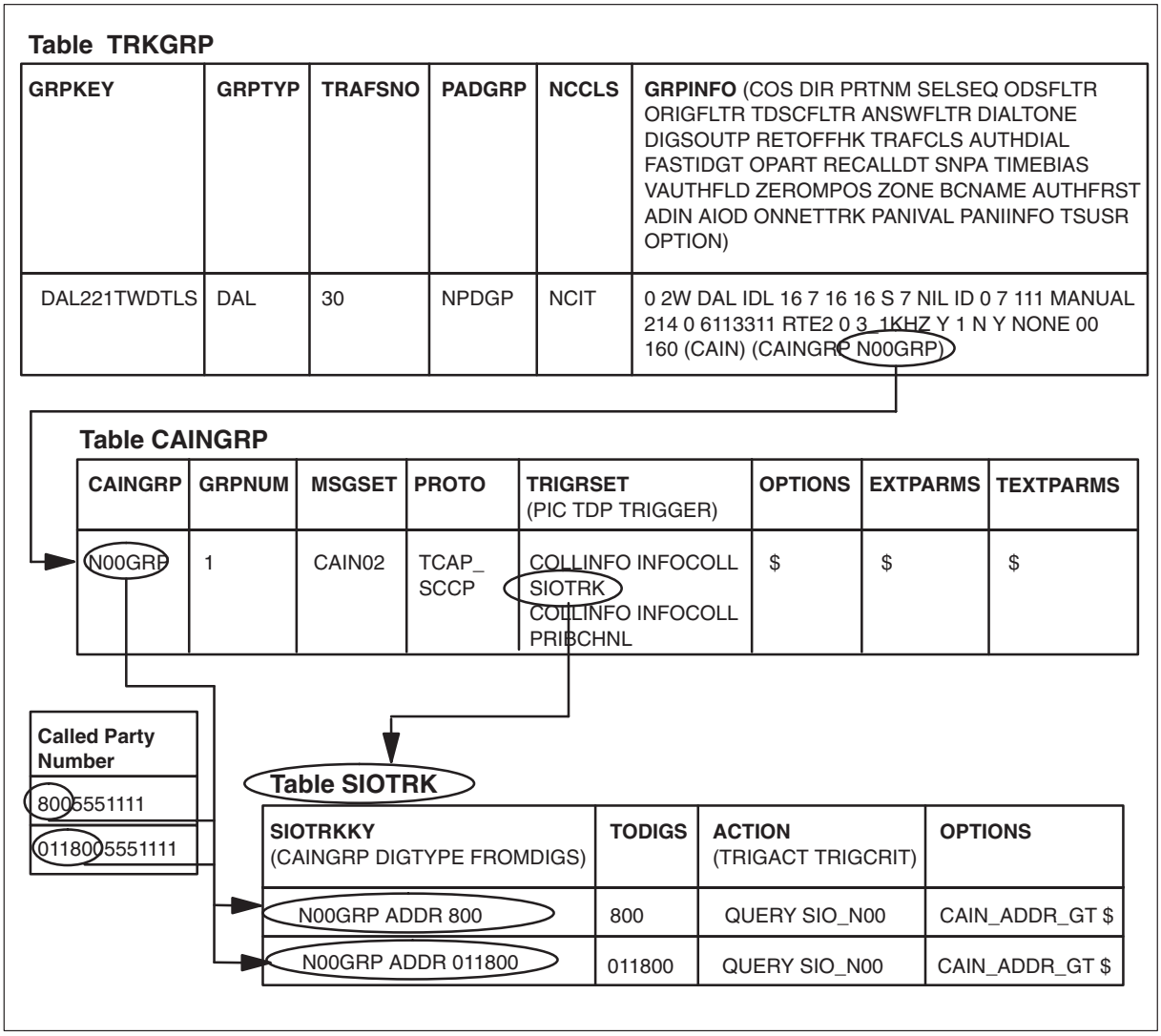
Note: Refer to Chapter 2, “Provisioning CAIN, Step 5: Define resource allocation requirements,” for more information.

10 Define the CCS7 subsystem.

Note: Refer to Chapter 2, “Provisioning CAIN,” for more information.

The following figure shows the interaction between the tables for the current scenario.

Tables



Data sent to the SCP

The switch sends data to the SCP for information required to complete the call.

IN/1

For IN/1 interaction, the switch sends the following to the DCP:

IN/1 query message parameters

Parameter	Definition
Dialed Number	Contains the subscriber dialed number
ANI	Contains the 3-, 6-, or 10-digit ANI or CLID received on the originating agent. If the ANI is not available, a PANI may be constructed and delivered. If neither an ANI nor a PANI is available, the trunk group SNPA is sent as a 3-digit ANI value.
—end—	

The DCP processes the data and returns information required to complete the call.

CAIN

For CAIN interaction, the switch sends the data associated with the request message. Continuing with the example, trigger criteria is met for the N00 number (8005551111) at *Shared_Interoffice_Trunk*. The switch builds an **`Info_Collected`** TDP-Request which includes the following data:

Note: CAIN parameters and extension parameters for AXXESS agents are handled differently. Refer to *UCS DMS-250 CAIN/FlexDial Interactions* for more information.

Info_Collected TDP-Request message parameters for Shared_Interoffice_Trunk

Parameter	Usage	Definition
UserID	Required	Contains the network identity of the originating agent
BearerCapability	Required	Contains the bearer capability of the call when the message is built
ChargeNumber (Note 5)	Optional	Contains the billing number that would be used to populate the CDR at this point in call processing
Carrier	Optional	Contains the dialed CIC or the CIC value (with an indication of: Selected CIC presubscribed and not input by calling party) from table TRKGRP1.
Lata (Note 7)	Optional	Contains the call's Local Access and Transport Area (LATA)
TriggerCriteriaType	Optional	Contains sharedIOTrunk, SIO_CIC, SIO_INFO, SIO_ANI, SIO_ADDR, SIO_ADIN, SIO_N00, or SIO_INTL

Note 1: The `universalAccess` extension parameter is included in the query (a flat 10 digits) when parameter CAIN_PROTOCOL_VERSION is V0. When the CAIN_PROTOCOL_VERSION is set to V1 or higher, the parameter contains up to 24 digits including a nature of address and numbering plan similarly to the ***CalledPartyID***.
Note 2: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS07 or higher.
Note 3: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS08 or higher.
Note 4: This parameter is not populated when the CAIN_PROTOCOL_VERSION is set to V3 or higher, and is therefore not sent.
Note 5: The parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS09 or higher.
Note 6: When the CAIN_PROTOCOL_VERSION is set to V4 or higher non-standard charge numbers (CARD, AUTH, ACCT, PIN, and N00) no longer populate the ***ChargeNumber*** parameter and all standard NOAs are supported.
Note 7: The parameter is supported only on DAL, FGD, and AXXESS agencies and the parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS11 or higher.
Note 8: Refer to Volume 3, Chapter 8, "Outgoing CAIN message parameters," for information on the population of this parameter when the CAIN_PROTOCOL_VERSION is set to V5 or higher.

—continued—

Info_Collected TDP-Request message parameters for Shared_Interoffice_Trunk (continued)

Parameter	Usage	Definition
CallingPartyID (Note 8)	Optional	Contains the one of the following (listed in order of precedence): For SS7 FGD calls: Calling_Party_Address from ISUP message, when available For FGD and AXXESS calls: valid ANI (information digits are not passed in this parameter) ***Note:*** Refer to *UCS DMS-250 CAIN/FlexDial Interactions* for more information on AXXESS agents. Valid SNPA value from table TRKGRP Default SNPA value from table CAINPARM
ChargePartyStationType (Note 2)		Contains the information digits for the call
AccessCode (Note 4)	Optional	Contains the account code, when available
CollectedAddressInfo	Optional	Contains the address collected from the incoming agent (from the IAM, subscriber dialing, or datafilled hotline digits)
CollectedDigits (Note 4)	Optional	Contains the collected PIN digits, when available
VerticalServiceCode	Optional	Contains feature codes dialed by the subscriber

Note 1: The `universalAccess` extension parameter is included in the query (a flat 10 digits) when parameter CAIN_PROTOCOL_VERSION is V0. When the CAIN_PROTOCOL_VERSION is set to V1 or higher, the parameter contains up to 24 digits including a nature of address and numbering plan similarly to the ***CalledPartyID***.
Note 2: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS07 or higher.
Note 3: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS08 or higher.
Note 4: This parameter is not populated when the CAIN_PROTOCOL_VERSION is set to V3 or higher, and is therefore not sent.
Note 5: The parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS09 or higher.
Note 6: When the CAIN_PROTOCOL_VERSION is set to V4 or higher non-standard charge numbers (CARD, AUTH, ACCT, PIN, and N00) no longer populate the ***ChargeNumber*** parameter and all standard NOAs are supported.
Note 7: The parameter is supported only on DAL, FGD, and AXXESS agencies and the parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS11 or higher.
Note 8: Refer to Volume 3, Chapter 8, “Outgoing CAIN message parameters,” for information on the population of this parameter when the CAIN_PROTOCOL_VERSION is set to V5 or higher.

—continued—

Info_Collected TDP-Request message parameters for Shared_Interoffice_Trunk (continued)

Parameter	Usage	Definition
ACGEncountered	Optional	Contains the ACG control encountered when the ACG control expires and a query is allowed to be sent
ExtensionParameter (Note 2)	Optional	Extension parameters require the CAIN0200 SOC option.
universalAccess (Note 1)	Optional	Contains the original address when the *O_Feature_Requested* ADDR collectible is executed or the universal access number dialed by the caller to obtain switch dial tone. Since this number is not the actual address for the call, it is not transported in ***CollectedAddressInfo*** or ***CalledPartyID***.
cainGroup	Optional	Contains the group number (field GRPNUM, table CAINGRP) for the CAIN group associated with the trigger when field EXTPARM in table CAINGRP contains CAINGRP.
adin	Optional	Contains the authorization code database index (field ADIN, table TRKGRP, DAL and FGD agencies) associated with the originating agency when field EXTPARM in table CAINGRP contains ADIN. ***Note:*** The adin extension parameter is not supported for AXXESS agents.

Note 1: The universalAccess extension parameter is included in the query (a flat 10 digits) when parameter CAIN_PROTOCOL_VERSION is V0. When the CAIN_PROTOCOL_VERSION is set to V1 or higher, the parameter contains up to 24 digits including a nature of address and numbering plan similarly to the ***CalledPartyID***.
Note 2: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS07 or higher.
Note 3: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS08 or higher.
Note 4: This parameter is not populated when the CAIN_PROTOCOL_VERSION is set to V3 or higher, and is therefore not sent.
Note 5: The parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS09 or higher.
Note 6: When the CAIN_PROTOCOL_VERSION is set to V4 or higher non-standard charge numbers (CARD, AUTH, ACCT, PIN, and N00) no longer populate the ***ChargeNumber*** parameter and all standard NOAs are supported.
Note 7: The parameter is supported only on DAL, FGD, and AXXESS agencies and the parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS11 or higher.
Note 8: Refer to Volume 3, Chapter 8, “Outgoing CAIN message parameters,” for information on the population of this parameter when the CAIN_PROTOCOL_VERSION is set to V5 or higher.

—continued—

Info_Collected TDP-Request message parameters for Shared_Interoffice_Trunk (continued)

Parameter	Usage	Definition
`origTrunkInfo`	Optional	Contains the originating trunk group number, trunk type, and trunk member number when field EXTPARM in table CAINGRP contains ORGTINFO.
`treatment`	Optional	Contains the treatment set by regular call processing before the query was sent
`reorigCall`	Optional	Presence of this parameter indicates the call in progress is a result of reorigination
`univIdx`	Optional	Used by SS7 Global-IMT agents only. Contains the universal translations scheme to be used for the call.
`netinfo` (Note 2)	Optional	Contains external network ID, network customer group ID, and network class of service.
`t1Overflow`	Optional	Indicates to the SCP that an overflow occurred on the query to the initial SCP.
`lnpReceived` (Note 2)	Optional	Presence of this parameter indicates LNP information was received from a previous switch.
`subscriptionInfo` (Note 3)	Optional	Contains the digit type triggered on for the query and the method to which the triggering CAIN group subscribed.
`jurisdictionInfo` (Note 3)	Optional	Contains the originating switch's LRN.

Note 1: The `universalAccess` extension parameter is included in the query (a flat 10 digits) when parameter CAIN_PROTOCOL_VERSION is V0. When the CAIN_PROTOCOL_VERSION is set to V1 or higher, the parameter contains up to 24 digits including a nature of address and numbering plan similarly to the ***`CalledPartyID`***.
Note 2: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS07 or higher.
Note 3: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS08 or higher.
Note 4: This parameter is not populated when the CAIN_PROTOCOL_VERSION is set to V3 or higher, and is therefore not sent.
Note 5: The parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS09 or higher.
Note 6: When the CAIN_PROTOCOL_VERSION is set to V4 or higher non-standard charge numbers (CARD, AUTH, ACCT, PIN, and N00) no longer populate the ***`ChargeNumber`*** parameter and all standard NOAs are supported.
Note 7: The parameter is supported only on DAL, FGD, and AXXESS agencies and the parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS11 or higher.
Note 8: Refer to Volume 3, Chapter 8, "Outgoing CAIN message parameters," for information on the population of this parameter when the CAIN_PROTOCOL_VERSION is set to V5 or higher.

—continued—

Info_Collected TDP-Request message parameters for Shared_Interoffice_Trunk (continued)

Parameter	Usage	Definition
`switchID` (Note 6)	Optional	Contains the switch ID
`accountCode` (Note 6)	Optional	Contains the account code when available
`pinDigits` (Note 6)	Optional	Contains the collected pin digits when available
`billingNumber` (Note 6)	Optional	Contains the non-standard charge number

Note 1: The `universalAccess` extension parameter is included in the query (a flat 10 digits) when parameter CAIN_PROTOCOL_VERSION is V0. When the CAIN_PROTOCOL_VERSION is set to V1 or higher, the parameter contains up to 24 digits including a nature of address and numbering plan similarly to the ***`CalledPartyID`***.
Note 2: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS07 or higher.
Note 3: This parameter requires the CAIN_PROTOCOL_STREAM to be UCS08 or higher.
Note 4: This parameter is not populated when the CAIN_PROTOCOL_VERSION is set to V3 or higher, and is therefore not sent.
Note 5: The parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS09 or higher.
Note 6: When the CAIN_PROTOCOL_VERSION is set to V4 or higher non-standard charge numbers (CARD, AUTH, ACCT, PIN, and N00) no longer populate the ***`ChargeNumber`*** parameter and all standard NOAs are supported.
Note 7: The parameter is supported only on DAL, FGD, and AXXESS agencies and the parameter requires the CAIN_PROTOCOL_STREAM to be set to UCS11 or higher.
Note 8: Refer to Volume 3, Chapter 8, “Outgoing CAIN message parameters,” for information on the population of this parameter when the CAIN_PROTOCOL_VERSION is set to V5 or higher.

—end—

SCP service options

Once the SCP receives the **`Info_Collected`** message, any of the following services can be performed such as:

- routing control
- advanced screening services based on the received subscriber data
- advanced translations
- additional subscriber interaction for digit collection

Data received from the SCP

Once SCP processing is complete, the SCP returns data to assist the switch in completing the call.

IN/1 successful translation

When the IN/1 translations are successful, the DCP returns the following data:

IN/1-returned data

Returned from DCP	Contents	Switch
satellite restrictions	0 or 1	A value of 1 indicates the call should not terminate to a satellite-based trunk
OPART and TPART	000–999 00–99	Forms an STS using table PARTOSTS
translated number	7 or 10 digit address	The translated number and the STS (formed from the OPART and TPART) access table HNPACONT to determine a routing list.
nature of number	Identifies ONNET or OFFNET numbers	Used for multi-COS screening and general call processing
CPI provided	0 or 1	Indicates whether ANI should be delivered
billing indicator	Indicates if the called party or calling party is billed for the call	If the called party is identified as the billed number, the service access code is placed in the BILLNUM field of the CDR. The switch blocks reorigination for called party billed N00 calls. If the calling party is identified as the billed number, the BILLNUM and ANISP fields of the CDR are unchanged.
multi-COS screening index	0–2047	Provides an index into table MULTICOS for multi-COS screening

IN/1 failed translation

When the IN/1 translations fail, the DCP returns a message instructing the switch to disconnect the call and apply one of the following treatments:

- VACT – indicates the N00 number was not recognized (for example, translations failed)
- N00B – indicates that the call should be blocked for the specified number

CAIN successful interaction

For simple N00 services (that map directly to current IN/1 functionality), the SCP should return an **`Analyze_Route`** message containing the following parameters:

Analyze_Route parameters

Parameter	Contents
ChargeNumber	For calling party billed, echo the ***ChargeNumber***. For called party billed, return the service access code with the nature of address set to N00 and the numbering plan set to PRVT.
CalledPartyID	Return the translated address of the called party. The switch uses translated number with the STS to access table HNPACONT and determine a routing list. ***Note:*** The SCP can return international numbers for the translated number.
ExtensionParameter	Extension parameters require the CAIN0200 SOC option.
servTranslationScheme	Return the STS to be used in routing the call
satRestriction	Presence indicates the call should not terminate to a satellite-based trunk
classOfSvc	Return an index into table MULTICOS for COS screening
reorigAllowed	When you return the called party as the billed party, return this parameter with a value of 0. ***Note:*** SCP-directed reorigination is subject to in-switch overrides.

CAIN failed interaction

When the SCP determines that the call should not continue, send one of the following to the switch:

- **Disconnect** containing the `treatment` extension parameter. (When a treatment is not specified, the switch applies AIND to the call.)
- **Send_To_Resource** or **Connect_To_Resource** containing ***DisconnectFlag*** and the `treatment` extension parameter. (When a treatment is not specified, the switch applies AIND to the call.)

Note: The SCP can specify up to three uninterruptible announcements to be played in the **Send_To_Resource** or **Connect_To_Resource** message.

CAIN extended functionality

CAIN offers the following extended functionalities:

- Advanced routing abilities
- Reorigination control
- Multi-COS screening

- DNIS
- Specify calling party presentation
- Network busy route advancing
- User busy route advancing
- No answer route advancing
- Network queuing
- Branding
- Digit collection
- Debit card services
- Terminating services
  - — Caller ID delivery with minimal delivery services tied up
- ACG

Advanced routing abilities

You gain more control over routing because the SCP can return up to eight route lists in the following parameters for an **`Analyze_Route`**:

- ***`PrimaryTrunkGroup`***
- ***`AlternateTrunkGroup`***
- ***`SecondAlternateTrunkGroup`***
- ***`Carrier`***
- ***`AlternateCarrier`***
- ***`SecondAlternateCarrier`***
- ***`CallingPartyID`***
- ***`GenericAddressList`***'s `OverflowRoutingNo`

The switch performs direct termination routing through the use of two tables: TANDMRTE (Tandem Routing) and TERMRTE (Termination Routing). Table TANDMRTE directs the call from switch to switch until the terminating switch is reached. Then, table TERMRTE directs the call to the terminating agency.

Note: Refer to Volume 3, Chapter 10, "Incoming CAIN messages," for more information on the **`Analyze_Route`** message, standard routing, and direct termination routing.

IN/1 equivalent

None

Reorigination control

The SCP can control reorigination by returning the `reorigAllowed` extension parameter in an **`Analyze_Route`** message. Returning a value of 0 (zero) blocks reorigination. The SCP can also specify the number of times a call may reoriginate (0–99). SCP-specified reorigination is subject to in-switch reorigination enabling logic overrides.

ATTENTION
When billing the called party, reorigination should be blocked.

When `reorigAllowed` is not returned, in-switch reorigination logic applies to the call.

IN/1 equivalent

IN/1 can only control reorigination through the Calling/Called Party Billed flag. When the flag indicates Called Party Billed, all reorigination is blocked.

Multi-COS screening

Multi-COS screening is performed on CAIN calls by returning the `classOfSvc` extension parameter or datafilling a default index in table CAINXDFT.

Note: COS screening can be performed before querying the SCP.

The following types of screening are provided:

- Call type – capability to allow/disallow calls based on the call type (international, OFFNET, or ONNET). If the ***`CallingPartyID`*** parameter is provided, the new call type (set by NOA) is used for screening.
- Destination number – capability to allow/disallow calls based upon the destination number. If the ***`CallingPartyID`*** parameter is provided, the new number is used for screening.
- Time of day – capability to allow/disallow calls based on the time of day. CAIN does not affect this type of screening.

If screening blocks the call, the appropriate COS treatment is applied. Otherwise, CAIN continues to process the **`Analyze_Route`** message and performs the appropriate routing.

Interaction

The parameter value indexes table MULTICOS for class of service screening.

When COS screening fails, the action specified by COS screening is performed and the **Analyze_Route** routing parameters are discarded. In this case, the route index determined through in-switch translations is used to route the call.

When successful COS screening occurs, the CAIN framework continues to process the **Analyze_Route** message in order to identify a route index.

IN/1 equivalent

IN/1 performs multi-COS screening when the response indicates calling party billing. Multi-COS screening is performed based on the MLTCOSID identified through existing call process, using the original N00 address. COS override is not supported. Multi-COS screening is not performed on called party billed calls.

Dialed number inward service

By moving all dialed number inward service (DNIS) activities to the CAIN SCP, data is centralized and datafill coordination becomes easier. Return the ***OutpulseNumber***, and the ***GenericAddressList***'s `AlternateOutpulseNo`, and the `SecondAlternateOutpulseNo` to determine DNIS routing.

CAIN allows multiple non-standard routing numbers to be outpulsed using tables TANDMRTE and TERMRTE to increase flexibility.

IN/1 equivalent

The translated number is datafilled in table HNPACONT using non-standard routing as follows:

```
> 20 (N D EAN861C7LP00 10 8005551234 Y) $
```

Non-standard routing is also available with CAIN.

Note: Refer to Chapter 2, "Provisioning CAIN" and Volume 3, Chapter 10, "Incoming CAIN messages," for more routing information.

Specify ANI delivery

ANI delivery is controlled by the CAIN presentation indicator in the ***ChargeNumber*** parameter and the following:

- Delivery of the ***ChargeNumber*** and OLI parameters in the outgoing IAM message is controlled by feature AD6849 (UCS ANI Delivery Enhancements). (Refer to the *UCS05 Software Release Document*, feature AD6849, for more information.) This feature introduces controls on parameter delivery based on the originating and terminating agencies. The following values can be datafilled for both originating and terminating agencies:

— ALWAYS – deliver ***CallingPartyID***, ***ChargeNumber***, and OLI when available

— NEVER – do not deliver ***CallingPartyID***, ***ChargeNumber***, or OLI

— CPNONLY – deliver only the ***CallingPartyID*** when available

— CGNONLY – deliver only the ***ChargeNumber*** and OLI when available

CAIN overrides the controls placed on originating agencies. Therefore, the controls placed on the originating agency are not used in determining parameter delivery. This override has the same effect as datafilling ALWAYS for the originating agency. The terminating controls are still functional.

- Delivery of the ***ChargeNumber*** parameter is subject to in-switch feature restrictions.
- Table RTEATTR provides a way to control the ***CallingPartyID*** and ***ChargeNumber*** delivery based on a terminating route. The controls in table RTEATTR, when provisioned, override any pre-existing controls.

  When the ***ChargeNumber*** is included in table RTEATTR, the ***ChargeNumber*** and OLI are delivered when:

  — A ***ChargeNumber*** is available.

  — No ***CallingPartyID*** is being delivered or the ***CallingPartyID***'s address is different than the ***ChargeNumber***'s address.

  When the ***CallingPartyID*** is included in table RTEATTR, the ***CallingPartyID*** and OLI are delivered when:

  — A ***CallingPartyID*** is available.

  — Excluding the ***CallingPartyID*** will stop delivery of the appropriate parameters.

  Note: (Refer to the *UCS06 Software Release Document*, feature AD8792, for more information.)

IN/1 equivalent

None

Network busy route advancing

In addition to the data received from a simple N00 query, the SCP can return the `cainGroup` extension parameter to control a network busy scenario. To prepare the switch to receive a CAIN group for a network busy scenario, perform the following procedure.

1 Enter table CAINGRP and create a new CAIN group using the Network_Busy trigger set:

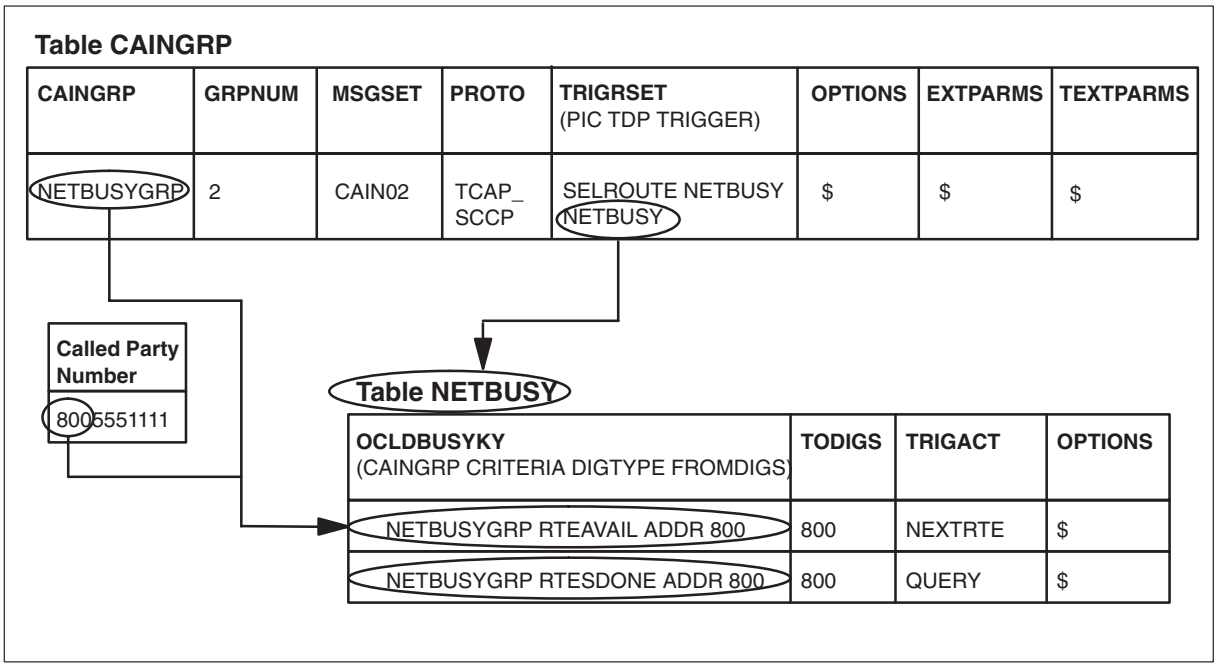
Sample entry: **>ADD netbusygrp 2 cain02 tcap_sccp selroute netbusy netbusy $ $**

2 Enter table NETBUSY and set up the trigger criteria.

Sample entry: **>ADD netbusygrp rteavail addr 800 800 nextrte $ $**
> ADD netbusygrp rtesdone addr 800 800 query $ $

Note: The call can presubscribe to the Network_Busy trigger set.

Subscription-NETBUSY table interaction



CAIN allows the following:

- ability to control route advancing through a TRIGACT of NEXTRTE when additional routes are available

  Note: NEXTRTE directs call processing to route advance to the next route available in the route list. If all routes have been attempted, directs the switch to perform a CAIN routing parameter advance.

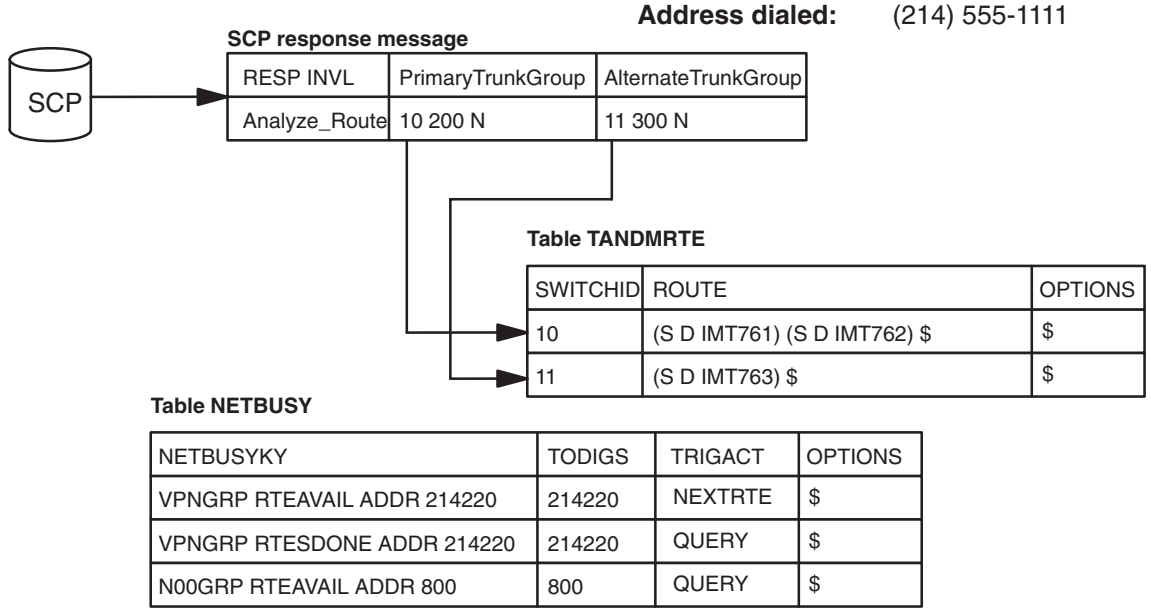
- ability to control route advancing through a TRIGACT of NEXTCNTRTE when additional routes are available

  Note: NEXTCNRTE directs call processing to route advance to the next CAIN routing parameter, skipping any remaining route choices within the current routing list.

- query as datafilled, whether or not all routes have been attempted (controlled by RTEAVAIL and RTESDONE)

The following figure describes how the CAIN framework route advances using the *Network_Busy* trigger table (NETBUSY). See Volume 3, Chapter 10, "Incoming CAIN messages," for more information on direct termination routing.

Route advance with the Network_Busy trigger



1 The switch receives an **`Analyze_Route`** message containing the ***`PrimaryTrunkGroup`*** and ***`AlternateTrunkGroup`*** parameters.

2 The switch attempts to complete the call using the route index identified by the ***`PrimaryTrunkGroup`*** parameter.

3 The switch seizes the trunk (IMT761) and sends an IAM message.

4 The switch receives a release message with a cause value indicating a Network Busy condition.

5 The switch checks to see if the call subscribes to CAIN services and if the *Network_Busy* trigger is enabled. If subscribed and enabled, the switch accesses the NETBUSY table.

   a If CAINGRP=VPNGRP and digits match, the switch route advances and attempts routing using the next trunk (IMT762). If routing over IMT762 is attempted and a Network Busy condition is indicated, the next CAIN routing parameter is tried.

   b If CAINGRP=N00GRP and digits match, the switch suspends call processing and sends a **`Network_Busy`** query message to the SCP.

IN/1 equivalent

Office parameter N00_BUSY_GNCT_ROUTE_ADV controls route advancing for N00 calls when busy conditions occur. When this parameters is set to Y and the switch receives a specific PRI or ISUP release cause, the switch route advances to the next trunk group in the route list. Once all routes are attempted, the switch applies treatment to the call.

IN/1 does not distinguish between user busy or network busy scenarios.

User busy route advancing

In addition to the data received from a simple N00 query, the SCP can return the `cainGroup` extension parameter to control a user busy scenario. To prepare the switch to receive a CAIN group for a user busy scenario, perform the following procedure.

1 Enter table CAINGRP and create a new CAIN group using the O_Called_Party_Busy trigger set:

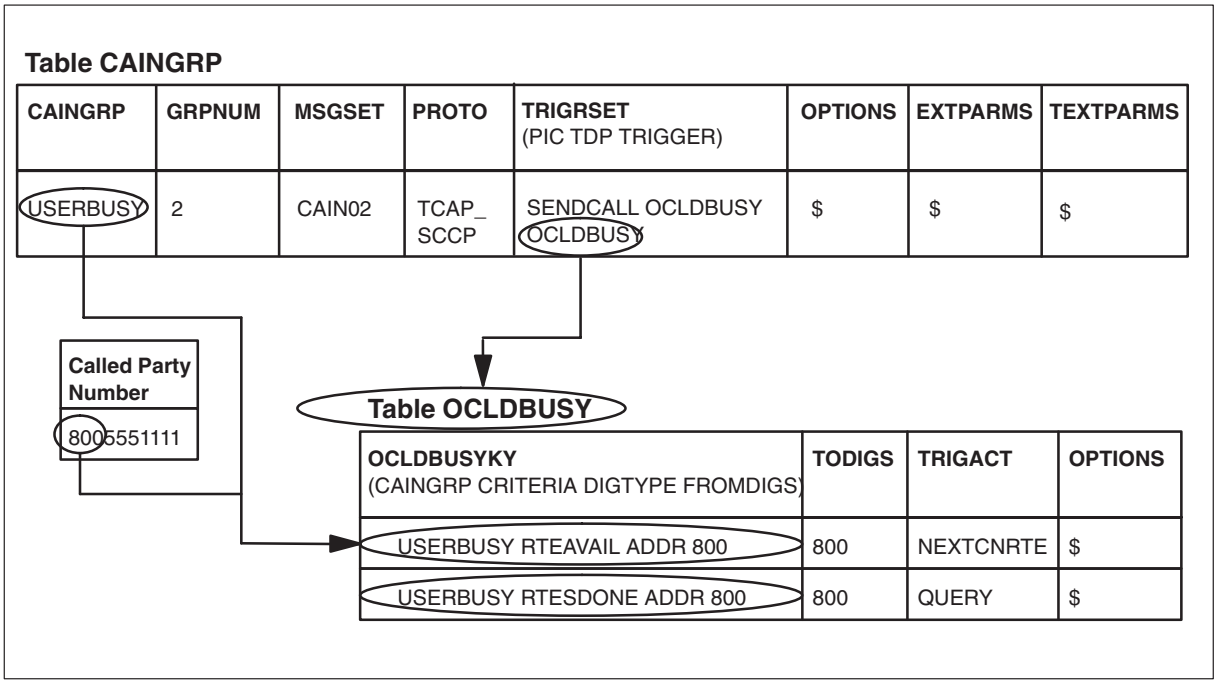
Sample entry: **>ADD userbusy 2 cain02 tcap_sccp sendcall ocldbusy ocldbusy $**

2 Enter table OCLDBUSY and set up the trigger criteria.

Sample entry: **>ADD userbusy rteavail addr 800 800 nextcnrte $**
> ADD userbusy rtesdone addr 800 800 query $

Note: The call can presubscribe to the Network_Busy trigger set.

Subscription-OCLDBUSY table interaction



CAIN allows the following:

- ability to control route advancing through a TRIGACT of NEXTRTE when additional routes are available

  Note: NEXTRTE directs call processing to route advance to the next route available in the route list. If all routes have been attempted, directs the switch to perform a CAIN routing parameter advance.

- ability to control route advancing through a TRIGACT of NEXCNTRTE when additional routes are available

  Note: NEXTCNRTE directs call processing to route advance to the next CAIN routing parameter, skipping any remaining route choices within the current routing list.

- query as datafilled, whether or not all routes have been attempted (controlled by RTEAVAIL and RTESDONE)

The following figure describes how the CAIN framework route advances using the *O_Called_Party_Busy* trigger table (OCLDBUSY). See Volume 3, Chapter 10, “Incoming CAIN messages,” for more information on direct termination routing.

Route advance with the O_Called_Party_Busy trigger

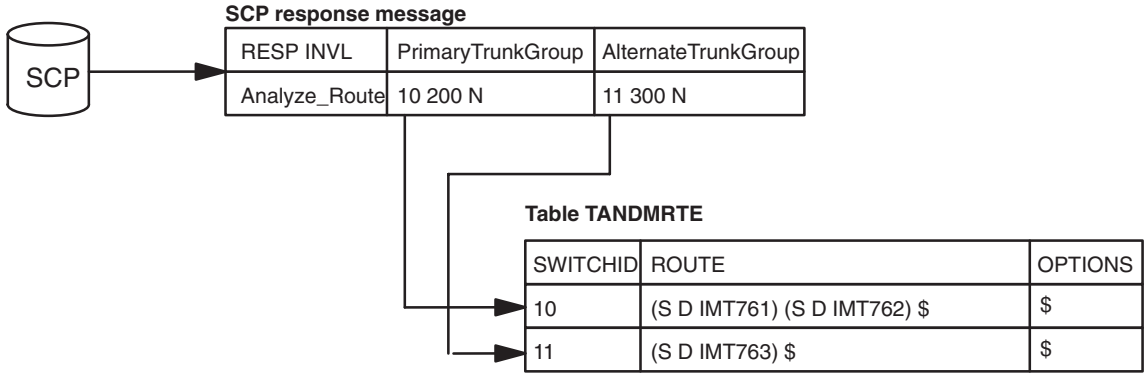


Table TANDMRTE

SWITCHID	ROUTE	OPTIONS
10	(S D IMT761) (S D IMT762) $	$
11	(S D IMT763) $	$

Table OCLDBUSY

OCLDBUSYKY	TODIGS	TRIGACT	OPTIONS
VPNGRP RTEAVAIL ADDR 214220	214220	NEXTCNRTE	$
VPNGRP RTESDONE ADDR 214220	214220	QUERY	$
N00GRP RTEAVAIL ADDR 800	800	QUERY	$

6 The switch receives an **`Analyze_Route`** message containing the ***`PrimaryTrunkGroup`*** and ***`AlternateTrunkGroup`*** parameters.

7 The switch attempts to complete the call using the route index identified by the ***`PrimaryTrunkGroup`*** parameter.

8 The switch seizes the trunk (IMT761) and sends an IAM message.

9 The switch receives a release message with a cause value indicating a User Busy condition.

10 The switch checks to see if the call subscribes to CAIN services and if the *O_Called_Party_Busy* trigger is enabled. If subscribed and enabled, the switch accesses the OCLDBUSY table.

   a If CAINGRP=VPNGRP and other criteria is met, the switch route advances and attempts routing using the next CAIN route choice (IMT763).

   b If CAINGRP=N00GRP and other criteria is met, the switch suspends call processing and sends a **`O_Called_Party_Busy`** query message to the SCP.

IN/1 equivalent

Office parameter N00_BUSY_GNCT_ROUTE_ADV controls route advancing for N00 calls when busy conditions occur. When this parameters is set to Y and the switch receives a specific PRI or ISUP release cause, the switch route advances to the next trunk group in the route list. Once all routes are attempted, the switch applies treatment to the call.

IN/1 does not distinguish between user busy or network busy scenarios.

No answer route advancing

CAIN controls a no answer scenario at the ***O_No_Answer*** TDP. When the terminating switch applies ringing, the CAIN switch starts the O_No_Answer timer (when provisioned on the CAIN group). If the timer expires before the called party answers, route advancing or SCP querying is performed. You can gain further control of the call by returning different CAIN groups, each with the O_No_Answer timer set for varying lengths.

Include the *O_No_Answer* trigger set in any existing CAIN group to enable the *O_No_Answer* trigger.

1 Enter table CAINGRP and change a CAIN group to include the *O_No_Answer* trigger set:

Sample entry: **>CHA n00grp 1 cain02 tcap_sccp collinfo infocoll siotrk collinfo infocoll pribchnl o_alertg onoanswr onoanswr $**

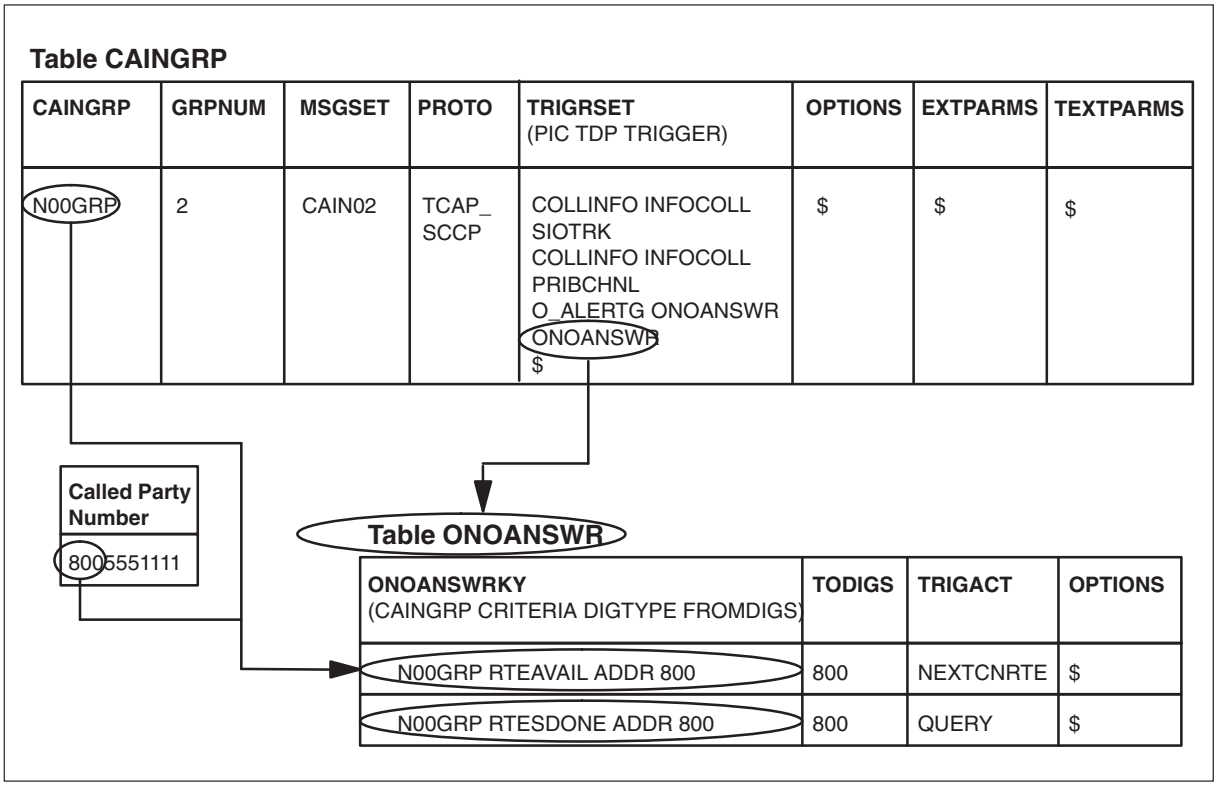
2 Enter table ONOANSWR and set up the trigger criteria.

Sample entry: **>ADD n00grp rteavail addr 800 800 nextcnrte $**
> ADD n00grp rtesdone addr 800 800 query $

3 Enter table CAINPARM and set the O_No_Answer timer (1–120 seconds).

Note: The SCP can also return the `cainGroup` extension parameter to control a no answer scenario.

Subscription-ONOANSWR table interaction



CAIN allows the following:

- ability to control route advancing through a TRIGACT of NEXTRTE when additional routes are available

  Note: NEXTRTE directs call processing to route advance to the next route available in the route list. If all routes have been attempted, directs the switch to perform a CAIN routing parameter advance.

- ability to control route advancing through a TRIGACT of NEXCNTRTE when additional routes are available

  Note: NEXTCNRTE directs call processing to route advance to the next CAIN routing parameter, skipping any remaining route choices within the current routing list.

- query as datafilled, whether or not all routes have been attempted (controlled by RTEAVAIL and RTESDONE)

- ability to play an announcement (PLAY_ANN option in table ONOANSWR) if the O_No_Answer timer expires and the a call is rerouted (route advanced) when the trigger action is QUERY, NEXTRTE, or NEXTCNRTE

The following figure describes how the CAIN framework route advances using the *O_No_Answer* trigger table (ONOANSWR). See Volume 3, Chapter 10, "Incoming CAIN messages," for more information on direct termination routing.

Route advance with the O_No_Answer trigger

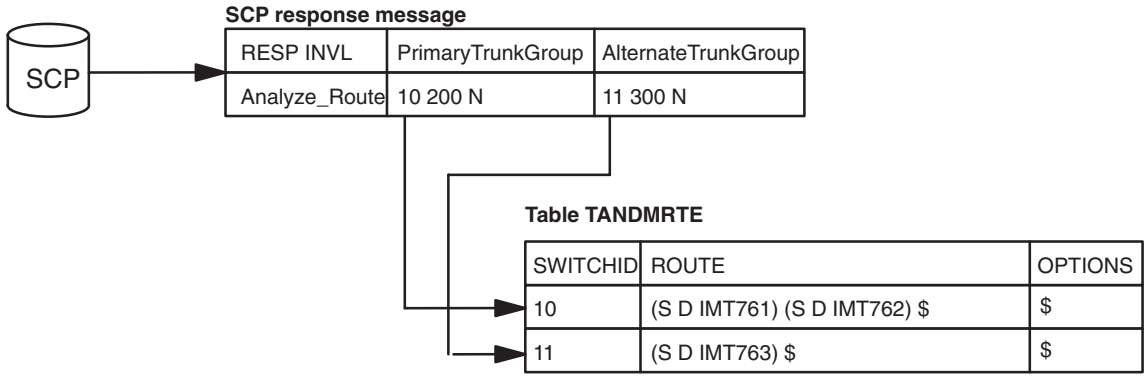


Table ONOANSWR

ONOANSWRKY	TODIGS	TRIGACT	OPTIONS
VPNGRP RTEAVAIL ADDR 214220	214220	NEXTCNRTE	$
VPNGRP RTESDONE ADDR 214220	214220	QUERY	$
N00GRP RTEAVAIL ADDR 800	800	QUERY	$

11 The switch receives an **`Analyze_Route`** message containing the ***`PrimaryTrunkGroup`*** and ***`AlternateTrunkGroup`*** parameters.

12 The switch attempts to complete the call using the route index identified by the ***`PrimaryTrunkGroup`*** parameter.

13 The switch seizes the trunk (IMT761) and sends an IAM message.

14 The switch checks to see if the call subscribes to CAIN services and if the *O_No_Answer* trigger is enabled. If subscribed and enabled, the O_No_Answer timer is started.

15 The O_No_Answer timer expires before the call is answered.

16 The switch accesses the ONOANSWR table and checks the trigger action.

   a If CAINGRP=VPNGRP and other criteria is met, the switch route advances and attempts routing using the next CAIN routing parameter (IMT763).

   b If CAINGRP=N00GRP and other criteria is met, the CAIN framework discards the routing information received in the previous query and sends an **`O_No_Answer`** message to the SCP.

IN/1 equivalent

Office parameter N00_NO_ANSWER_TIMER controls route advancing for N00 calls when no answer conditions occur. If the time expires prior to answer, the switch route advances to the next group in the route list. Once all routes are attempted, the timer is cancelled and the phone is allowed to ring.

Network queuing

Network queuing allows calls to be placed in a queue until the call can be completed. Set up queuing by returning a

- **`Send_To_Resource`** (containing Play Announcement and Collect Digits) in a conversation package requesting collection of zero digits

  Specify a resource announcement that will play until the SCP sends a **`Cancel_Resource_Event`** or the announcement times out and the switch sends a **`Resource_Clear`** with a ***`ClearCause`*** of `normal`.

  Note: Resources played during network queuing might include a message such as: "We're sorry, all operators are currently busy. Please continue holding and your call will by our next available operator."

- **`Send_To_Resource`** (with ***`DestinationAddress`***) in a conversation package

  The switch routes the call to an IP that can play music instead of announcements only. The resource continues to play until the SCP sends a **`Cancel_Resource_Event`**. The switch responds by sending a **`Resource_Clear`** with a ***`ClearCause`*** of `resourceCancelled`.

  Note: This call can be billed for the time at the IP by using the ***`AMAMeasure`*** parameter.

IN/1 equivalent

None.

Branding

The SCP can include the `callBranding` extension parameter in an **`Analyze_Route`** message. Including a specific resource in an **`Analyze_Route`** message, you can provide specific announcements for each of your customers. The switch will play the announcement or tone specified in the `callBranding` extension parameter.

IN/1 equivalent

Datafill an announcement or tone CLLI in the routing list.

Digit collection

Implementing N00 through CAIN allows subscriber interaction through SCP-controlled digit collection for data such as additional validation numbers. The SCP initiates digit collection by returning a **`Send_To_Resource`** or **`Connect_To_Resource`** message in a conversation package. Collected digits are returned to the SCP in a **`Resource_Clear`** or **`CTR_Clear`** message. Any digits collected can be returned to the switch in an **`Analyze_Route`** message in the ***`AMADigitsDialedWC`*** parameter; additional billing data can also be returned in the `billSequenceNumber` extension parameter.

The SCP initiates digit collection by returning a **`Send_To_Resource`** or **`Connect_To_Resource`** message in a conversation package that contains the following parameters:

Play announcement and collect digits parameters

Parameter	Usage	Definition
`ResourceType`	Required	Contains Play Announcement and Collect Digits
`StrParameterBlock`	Required	Contains a Play Announcement and Collect Digits tag and one announcement digit block (Note 1):
Note 1: Announcements are played in the order received, uninterruptible announcements are played first, followed by the interruptible announcements. ***Note 2:*** The switch ignores any information digits received.		

—continued—

Play announcement and collect digits parameters (continued)

Parameter	Usage	Definition
		• AnnouncementBlock encoded as an AnnouncementDigitBlock – contains the following: — MaximumDigits – identifies the number of digits required – Fixed indicates that the switch should collect the specified number of digits. – Variable indicates that the switch should collect collect the number of digits specified in the range provided by the SCP. For example, collect 0 to 10 digits. — UninterAnnounceBlock – contains a Play Announcement tag and up to 3 uninterruptible, audible tone or announcement resource identifiers. The announcement elements contain a resource identifier (index into table CAINRSRC). — InterAnnounceBlock – contains a Play Announcement tag and up to 3 interruptible, audible tone or announcement resource identifiers. The announcement elements contain a resource identifier (index into table CAINRSRC).
AnswerIndicator	Optional	Presence instructs the switch to provide answer supervision to the originating agent while the caller is connected to the resource. The switch sends answer indication to the caller in response to the Play Announcement request if answer indication has not already been sent. ***Note 1:*** ***AnswerIndicator*** is used for SS7 and PRI originators. ***Note 2:*** ***AnswerIndicator*** does not affect billing (internal resources only) at the querying switch.
Note 1: Announcements are played in the order received, uninterruptible announcements are played first, followed by the interruptible announcements. ***Note 2:*** The switch ignores any information digits received.		
—end—		

The switch returns the collected digits in a **Resource_Clear** containing the following parameters:

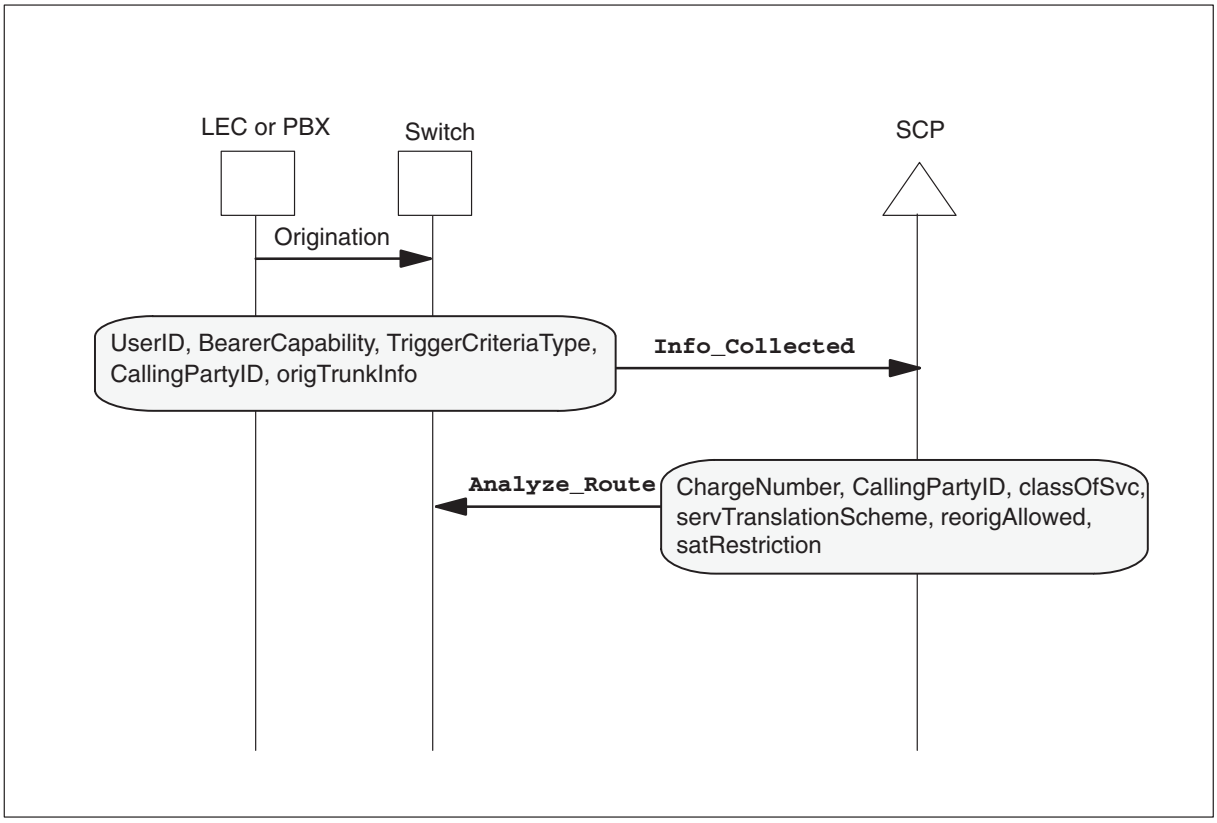
Resource_Clear parameters

Parameter	Definition
ClearCause	Indicates reason a connection between a caller and a resource was terminated
CollectedDigits	Contains the digits collected during SCP digit collection
FailureCause	Indicates reason a failure occurred

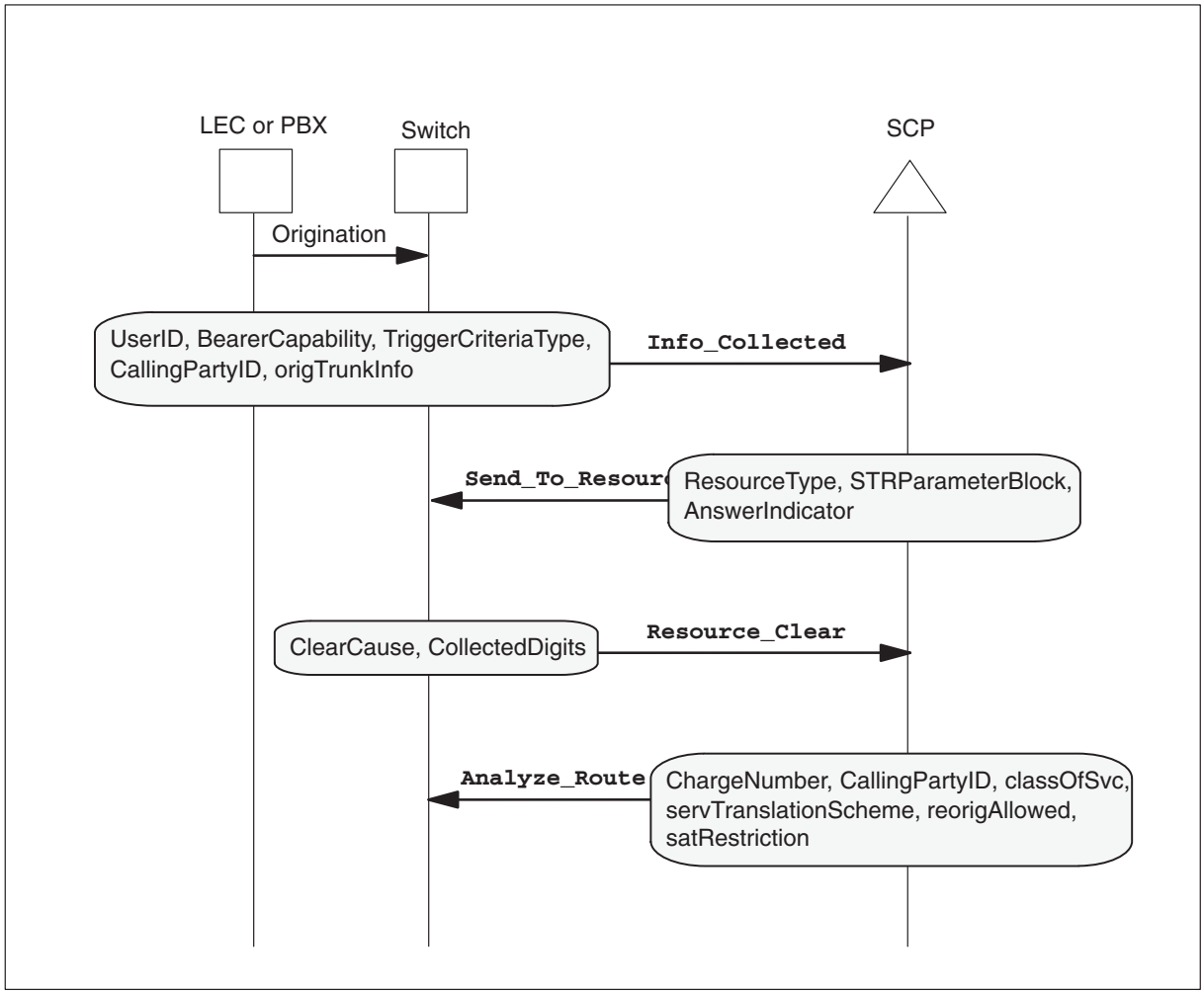
Example

The following chart shows the contents of the messages between the switch and the CAIN SCP.

Analyze_Route



Send_To_Resource



Debit card services

NetworkBuilder can be used to implement debit, pre-paid card services using the *Timeout* and *O_Disconnect* events as described in the following call example.

1 A subscriber dials an N00 number to use a debit card.

2 The switch evaluates the *Shared_Interoffice_Trunk* trigger and sends an **`Info_Collected`** query to the SCP.

3 The SCP responds with a conversational **`Send_To_Resource`** message requesting validation digits from the subscriber.

4 The switch plays the specified resource, collects digits from the user and reports them to the SCP in a **`Resource_Clear`** message.

5 The SCP replies to the second **Resource_Clear** with a conversational package containing an **Analyze_Route** message and a **Request_Report_BCM_Event** component which arms the *Timeout* and *O_Disconnect* events. The SCP should also include a ***TimeoutTimer*** parameter containing the amount the time left on the subscriber's card.

6 The switch then routes the call.

7 If the called party answers, the two parties remain connected until the Timeout timer expires, one of the parties hangs up, or reorigination occurs.

8 The switch sends either a **Timeout** message (if the Timeout timer expired) or an **O_Disconnect** message (in all other cases) to the SCP.

9 The SCP deducts the amount of time specified in the message's `connectTime` extension parameter from the user's card. If the switch sent a **Timeout** message, the SCP returns a **Disconnect** message (to end the call) or a **Connect_To_Resource** message to the switch.

10 If the switch receives a **Connect_To_Resource** message, it plays the specified announcement. For example, the announcement could inform the subscriber that the card's time is about to expire, or it could describe how to recharge the card. The **Connect_To_Resource** message could also instruct the switch to connect the user to an Intelligent Peripheral (IP) which could allow the user to recharge the card.

11 If the **Connect_To_Resource** message is sent in conversation, the switch sends a conversational **CTR_Clear** message to the SCP after the announcement has been played. The SCP may respond to the **CTR_Clear** with:

- a **Disconnect** message to end the call
- a **Continue** message re-arming the *Timeout* and *O_Disconnect* events (if there is time remaining on the card)
- another **Connect_To_Resource** message

IN/1 equivalent

None.

CAIN ACG

CAIN ACG provides the following benefits over IN/1 ACG:

- Larger control list based on provisioning of table VAMPTRID
- Separate control lists for SCP and SOCC controls
- Ability to re-query to a new SCP in the event that a query is blocked by an ACG control
- Ability to specify the treatment applied to a call blocked by an ACG control

- Global ACG controls through the GLOBAL command in the ACGCNTRL CI
- Removal of large number of controls through the **ACG_Global_Ctrl_Restore** message or the RESET command in the ACGCNTRL CI
- Larger range of ACG GAP Intervals
- Separate GAP Intervals for SOCC and SCP controls
- Able to receive unidirectional **ACG** messages
- Notification is sent to the SCP in an **ACG_Overflow** message, when a control list overflows
- Notification is sent to the SCP in an ***ACGEncountered*** parameter, when a control is encountered (but not applied)

IN/1 equivalent

ACG controls are added to a single control list, to reduce the number of outgoing queries from the switch to the SCP. Controls that are blocked by an ACG control are sent to treatment.

CAIN terminating services

CAIN terminating services provide caller ID delivery with minimal deliver services tied up. The terminating switch is queried to determine if caller ID information is requested. Since the query does not occur until the terminating switch is reached, network congestion is reduced because the caller ID information does not have to be passed through the network.

IN/1 equivalent

None

Appendix B
Engineering guidelines

This appendix contains the engineering guidelines you should consider when provisioning your switch for Carrier AIN.

General engineering rules

Consider the following when provisioning your switch:

- CAIN groups should be used judiciously. Where possible multiple services/triggers should be combined into a single group.
- Engineer CAIN extension block parameters when introducing NetworkBuilder for the first time. Both CAIN and TCAIN services need to be considered.
- High traffic services such as N00 should use global titles to route to a dedicated SCP via a dedicated linkset. This will help avoid SS7/SCP congestion.
- When converting a subscriber address for use by CAIN, it may be advisable to datafill the number as CT OFFNET in table STDPRTCT to minimize feature interactions.
- For standardized digit collection, the *O_Feature_Requested* trigger is much more efficient than conversation.
- Using CAIN on SS7 originators will increase the holding time for ISUP extension blocks. These blocks are engineered in table OFCENG (NUM_ISUP_EXT_BLKS).
- The switch allows a maximum of 32,768 permutations of CAINGRP/DIGTYPE combinations in the trigger tables. For example, each of the following tuples (from table SPECDIG) count as one permutation:

  — > CAINGRP2 ADDR 214 214 QUERY SDS_ADDR TREAT $

  — > CAINGRP2 INFO 11 11 QUERY SDS_INFO ROUTE $

  — > CAINGRP2 ADDR 800 800 QUERY SDS_N00 ROUTE $

  — > CAINGRP2 ADDR 011 011 IGNORE $

— > CAINGRP2 ANI 684 684 BLOCK $

— > CAINGRP2 CIC 333 333 QUERY SDS_CIC TREAT $

CCS7 links

Table 1 shows the number of physical CCS7 links to the STP and/or the SCP required to accomodate the inclusion of CAIN traffic on a UCS DMS-250 network. These numbers assume the following:

- an average CAIN message size is less than or equal to 80 bytes (not including header data)
- CCS7 links are provisioned for 1 + 1 redundancy
- CCS7 link maximum occupancy rate is 80%

Table 1
Physical CCS7 links required

# CCS7 links	#CAIN calls within an hour		
	80 bytes	**150 bytes**	**225 bytes**
2	190,000	115,000	72,000
4	190,000 – 360,000	115,000 – 230,000	72,000 – 144,000
6	380,000 – 570,000	230,000 – 345,000	144,000 – 216,000
8	na	345,000 – 460,000	216,000 – 288,000
10	na	460,000 – 575,000	288,000 – 360,000
12	na	na	360,000 – 432,000
14	na	na	432,000 – 504,000

STR-Connections

The ISUP extension block stores information received in an incoming IAM message for later use in a call. It is allocated upon receipt of an IAM message, and is typically deallocated when the call is answered.

During an STR-Connection to an IP, the ISUP extension block is not deallocated when the IP answers. The extension block is still needed, since the switch may establish a second leg following the STR-Connection. When a second leg is established, the ISUP extension block is deallocated when the called party answers.

Since the average holding time may be longer for an STR-Connection, the number of ISUP extension blocks may need to be increased (Office parameter NUM_ISUP_EXT_BLKS in table OFCENG).

DTMF UTR and ANNC UTR considerations

Announcement planning and engineering as well as determining service requirements are in the *Digital Recorded Announcement Machine DRAM and EDRAM Guide*. Additional detailed planning and engineering information can be found in the *DMS-100 Provisioning Manual* and the *Digital Recorded Announcement Machine Administration Guide*.

Refer to the Capacity Engineering Manual for capacity factors for receivers, tones, and announcements.

Service circuit capacity

Service circuits are those switch components that are provisioned on an office basis and are in a common pool to serve all subscribers. The circuits discussed in this chapter include the following:

- receivers (MF, DTMF, and UTR)
- three-port/six-port conference circuits
- special tones
- announcements

Monitoring service circuit capacity factors

OM Group UTR applies to the universal tone receivers. Sufficient studies should be taken to assure that the holding times and occupancy rates developed are not the result of any unusual occurrence in the office traffic load.

In addition to the OMs associated with receiver capacity performance, receivers are measured for dial tone delay by the dial tone speed recording (DTSR) OM group and incoming start-dial-delay by the ISDD OM group. From the DTSR, dial tone delay (over 3 seconds) percentages are obtained. In a similar fashion, incoming start-dial-delay (over 3 seconds to attach a receiver) percentages are produced. Both of these measurements can be used to determine if receiver capacity is adequate to meet service objectives for current traffic and projected loads to the end of design date.

Service circuit occupancy

In all cases, occupancy on individual groups of service circuits, such as receivers, tone circuits, conference circuits is limited to 75 percent on an HDBH load level basis. The requirements are expected to be controlling requirements for large groups of service circuits. For smaller groups of

service circuits, the following definitions provide the basis for service circuit delay and blocking capacity.

In general, the quantities of these system components are engineered to cause very little blocking or delay to customers and to cause minimal queueing even under peak loads. Central processor usage for each call does not increase substantially during peak load conditions. The service circuits grade-of-service based on blocking and delay criteria are shown in Table 2.

Table 2
Service circuits blocking and delay criteria

Service circuit	Criteria	ABSBH	HDBH
DTMF receiver	DTD > 0 seconds DTD > 3 seconds	1 % 0.1%	5% 1%
MF receiver	DTD > 0 seconds DTD > 3 seconds	0.1% 1%	1% 5%
Announcement circuits	Blocking	1%	5%
Tone circuits	Blocking	1%	5%

Note: UTR receiver uses similar grade-of-service as DTMF or MF receivers.

Receivers

Service circuits in the DMS-100 Family system are common equipment units which include dual-tone multifrequency (DTMF) receivers, multifrequency (MF) receivers, and universal tone (UT) receivers. The DTMF receiver is used to convert dual tone multi-frequency address signals from the customer to machine readable codes. The DTMF receiver connection is from a customer line; through an LCM/LGC, and then through the network to a DTMF receiver on a trunk module (TM) or maintenance trunk module (MTM).

The multifrequency (MF) receiver is used to convert multi-frequency signals over a trunk to a machine readable code. The MF receiver connection is from an incoming trunk (through a digroup or trunk module); then through the network to an MF receiver on a TM or MTM.

The universal tone receiver (UTR) is used to collect and decode both DTMF and MF address signals and to report the decoded address digits to the central control by means of the peripheral signalling processor. It eliminates the need to establish a network path to a DTMF or MF receiver in an MTM.

The UTR is located in the LGC, LTC, DTC, or RSC peripheral. Two connection attempts will be made on randomly chosen DTMF or MF receivers, mounted on an MTM, if blocking occurs. The length of time taken to establish a connection to a receiver determines its delay criteria. To minimize blocking probability, the receivers are spread over the available TMs and MTMs.

Receiver capacity

Holding times and percent occupancy are the most significant capacity factors of receivers. These factors should be studied on a regular basis to determine if the holding times are in the expected values used for provisioning the office. Significant increases are usually due to unanticipated changes in the call mix. Occupancy rates that exceed the forecast are usually due to calling growth rates increasing faster than expected. Holding times and occupancy rates that exceed the forecast reflect an increase in the number of delays greater than 3 seconds.

Evaluating performance

Receivers should be studied during the component busy hour. Incoming and outgoing calls may have separate busy hours if the office traffic loads are affected by factors such as a business that receives large numbers of incoming calls due to the nature of the business.

Calculations used to evaluate performance

In addition to holding time determination, calculations are included here to determine the percentage of the receiver capacity used and the overflow rates during the study period. These calculations apply to both MF and Digitone receivers and are the most significant measurements for the evaluation process as these are the measurements on which the engineering capacity tables are based. The DTS and ISDD percentages of delay results are computed automatically and require no further calculations.

$$\text{Digitone receiver holding time (in seconds)} = \frac{\text{RCVTRU (Digitone)} \times 10}{\text{RCVSZRS}}$$

Note: RCVTRU is a fast scan (10 second) register

$$\text{MF receiver holding time (in seconds)} = \frac{\text{RCVTRU (MF)} \times 10}{\text{RCVSZRS}}$$

Note: RCVTRU is a fast scan (10 second) register

```
 % of receiver             Sum of the RCV traffic use
 capacity used =        ---------------------------------  *  100
                        Engineered (MF or Digitone)capacity


                        RCVQOVFL
% receiver overflow =   --------  *  100
                        RCVSZRS
```

Note: RCVSZRS does not include calls abandoned while in queue.

```
                  UTRQOVFL
% UTR overflow =  --------  * 100
                  UTRSZRS
```

Note: UTRSZRS does not include calls abandoned while in queue.

```
                                    UTRTRU x 10
UTR holding time (in seconds) =     -----------
                                      UTRSZRS
```

Note: UTRTRU is a fast scan (10 second) register

Traffic capacity

The MF receivers and Digitone receivers are provisioned based on holding times and the selected probability of delay greater than 3 seconds. Northern Telecom holding times are shown in Tables 3 through 6. Holding times reflect call mix (call type and number of digits expected), as they are a weighted average (in proportion to the volume of calls offered in each category) of all the holding times. Significant increases in holding times due to changes in call mix may result in receiver shortages. Some operating companies do not use holding times, but use Poisson tables for provisioning purposes (Tables 3 through 6).

Table 3
Digitone receiver holding times

Calls	Holding times
0 operator	2.3 seconds (Note 1)
3-digit service code	3.1 seconds
7-digit number	7.9 seconds (Note 2)

Note 1: When special toll call (0+ service is provided, add 4 seconds holding time to operator calls for the time-out feature used to discriminate between 0+ and 0- calls.
Note 2: Add 0.8 seconds for each digit dialed for toll call prefixes.

Table 3
Digitone receiver holding times

Calls	Holding times
10-digit number	11.3 seconds (Note 2)
False attempt	3.0 seconds
Note 1: When special toll call (0+ service is provided, add 4 seconds holding time to operator calls for the time-out feature used to discriminate between 0+ and 0- calls. ***Note 2:*** Add 0.8 seconds for each digit dialed for toll call prefixes.	

Table 4
MF receiver holding times

Call type	4 digits	Additional digit
Incoming MF call (all types)	1.9 seconds	0.14 seconds
Key pulsing switchboard	5.7 seconds	0.60 seconds
Note: Add 0.9 seconds for MF incoming calls and 2.32 seconds for DP incoming calls outpulsing ANI information.		

Table 5
MF receiver – CCS capacity

Number of receivers	HT = 7 seconds		HT=8 seconds		HT=9 seconds	
	0.001	0.01	0.001	0.01	0.001	0.01
4	36	30	38	52	26	47
8	143	184	126	168	115	157
12	269	318	244	297	226	282
16	402	386	371	431	348	413

Table 6
Digitone receiver – CCS capacity

Number of receivers	HT = 7 seconds		HT=8 seconds		HT=9 seconds	
	0.001	0.01	0.001	0.01	0.001	0.01
4	23	42	22	40	21	39
8	102	43	98	138	94	134
12	204	260	197	253	191	247
16	318	386	308	376	300	369
20	439	516	426	505	416	495
24	565	648	549	636	537	624
28	694	756	676	756	661	756
32	825	864	805	864	488	864
36	957	972	936	972	918	972
40	1080	1080	1068	1080	1049	1080
44	1188	1188	1188	1188	1188	1188
48	1296	1296	1296	1296	1296	1296
52	1404	1404	1404	1404	1404	1404

Universal tone receivers

The universal tone receiver (UTR) is an optional card in the PM. If the UTR is not included in a specific PM, the central control can establish a network connection between that PM and one that has a UTR. The UTR is a 32-channel tone receiver. Thirty channels detect a variety of tones, including dual-tone multifrequency (DTMF) for lines and multifrequency (MF) for trunks. Tone samples are switched onto the parallel speech bus by the time switch and are collected by the UTR at the appropriate time slots. The UTR analyzes the samples and identifies the tones. The results are then sent to the signaling processor.

The UTR plugs into designated slots in any of the following configurations of the common peripheral controller shelves:

- DTC - digital trunk controller
- IDTC - international digital trunk controller

Tone circuits

The switch provides a wide variety of tone treatments. These tones are typically provided in the peripheral modules (for example, the MTM). Tone generator circuit cards provide the tones as required; for example, the NT3X68 card provides tones for general purposes that include the following:

- permanent signal
- conference calling
- dual-tone multifrequency (DTMF) signaling
- call waiting

In general, it is expected that if sufficient tone capacity is provided, then no shortage will exist and no overflows would be encountered under normal circumstances. To determine which tones are provided, refer to the translations tables: TONES, STN (special tones), and SVRCKT (service circuits).

NCCBS

NCCBS is an office parameter in table OFCENG which specifies the number of CCBs required for a switch. A CCB is a software register that contains information such as the identity of the calling and called appearances. It is associated with a call throughout its duration. This parameter is required for all switches.

The parameter value determines the maximum number of simultaneous calls.

The DMS packet handler uses this parameter to set up packet calls. Each packet call requires one CCB.

The recommended minimum value for this parameter is 2000.

CAUTION

Possible service interruption

Changes to this parameter have an affect on the site's overall ECCB pool. This could have an impact on the volume of calls that the switch can support. Any changes to this parameter should be made only after consulting with your Nortel Networks support.

Engineering call condense blocks for CAIN

Use the following formulas to estimate the number of call condense blocks you will need to provision for parameter NCCBS when using CAIN.

NCCBS = (SwitchCallRate +CAINMultiPartyCallRate) * Holding time *1.62

SwitchCallRate = BHCA ÷ 3600

CAINCallRate = CBHCA ÷ 3600

CAINmultiPartyCallRate = CainCallRate x %MultiPartyCalls x %MultiPartySetupTime

Variable definitions

CBHCA – Your estimate of the call volume on the switch on CAIN capable agents expressed in busy-hour call attempts.

BHCA – Your estimate of the call volume on the switch, expressed in busy-hour call attempts

Holding time – Your estimate of average call holding time in seconds (recommended default = 200)

%MultiPartyCalls – Your estimate of the percentage of CAIN calls which utilize multi-call party handling

%MultiPartySetupTime – Your estimate of the percentage of the total call time it will take to originate a call to the 3rd party and merge that party into the existing call.

3600 – represents the number of seconds in an hour

1.62 – Factor based on high day busy hour calculations

Example

For this example, you have gathered the following data:

- Each call requires one CCB
- BHCA=400, 000 calls per hour
- CBHCA=200,000 calls per hour
- Holding time=200 seconds
- You estimate 50% of CAIN Calls to use multi party handling
- You estimate 10% of call time as MultiPartySetupTime

therefore:

SwitchCall Rate = BHCA ÷ 3600 = 400, 000 ÷ 3600 = 111

CAINCall Rate = CBHCA ÷ 3600 = 200,000 ÷ 3600 = 56

CAINMultiPartyCall Rate = CainCallRate x %MultiPartyCalls x %MultiPartySetupTime = (56) x .(50) x (.10) = 3 calls/sec

therefore:

NumberCAINCallCondenseBlocksRequired = (SwitchCallRate +CAINMultiPartyCallRate) * Holding time *1.62 = (111 + 3) * 200 *1.62 = 36936 call condense blocks

Therefore, you should provision 36936 call condense blocks in the NCCBS parameter.

NUM_CAIN_ECCBS

NUM_CAIN_ECCBS is an office parameter in table CAINPARM. This office parameter is used to allocate pools of extended call condense blocks (ECCBS) for CAIN. These extended call condense blocks, which store call data, are allocated during origination on an agent that has the CAIN option or during termination on an agent that has the TCAIN option. During call processing, this block stores data that must survive for the entire call process, such as for various CAIN group subscriptions. An additional block may be allocated if another call process is facilitated by the first. Currently, this is only applicable during multi call party handling.

Extended call condense blocks remain active for the entire call process, including reorigination. They are not deallocated until the call process is released. Because of this extended life, these blocks are allocated as a percentage of the NCCBS office parameter in table OFCENG.

Engineering CAIN extended call condense blocks

Use the following formulas to estimate the number of CAIN extended call condense blocks you will need to provision for parameter NUM_CAIN_ECCBS.

NumberCAINextCallCondenseBlocksRequired = (CAINcallRate+CAINMultipartyCall Rate) x Holding time x 1.62 x ExtendedCallCondenseBlocksRequiredPerCallProcess

where:

CAINcallRate = (CBHCA ÷ 3600)

CAINmultiPartyCallRate = CainCallRate x %MultiPartyCalls x %MultiPartySetupTime

ExtendedCallCondenseBlocksRequiredPerCall process = 1 (only one extended call condense block is required per call process)

Call rate calculation

The CAIN call rate is the number of calls per second that require SCP service logic. The multi–party call rate is the number of calls per second that

involve three or more simultaneous party connections. To estimate the switch, CAIN and multi-party call rates, use the following formulas:

CAINcallRate = (CBHCA ÷ 3600)

CAINmultiPartyCallRate = CainCallRate x %MultiPartyCalls x %MultiPartySetupTime

Variable definitions

CBHCA – Your estimate of the call volume on the switch on CAIN capable agents expressed in busy-hour call attempts

3600 represents the number of seconds in an hour (60 seconds/minute × 60 minutes/hour)

%MultiPartyCalls – Your estimate of the percentage of CAIN calls which utilize multi-call party handling

%MultiPartySetupTime – Your estimate of the percentage of the total call time it will take to originate a call to the 3rd party and merge that party into the existing call.

Holding Time – Your estimate of the average call holding time in seconds (recommended default=200)

1.62 – Factor based on high day busy hour calculations

Example

For this example, you have gathered the following data:

- Holding time = 200 seconds
- CBHCA = 200,000 calls per hour because you expect 50% of the busy-hour call attempts to be on CAIN capable agents
- You estimate 50% of CAIN Calls to use multi party handling
- You estimate 10% of call time as MultiPartySetupTime
- You know that you require one extension block per call process

Therefore,

CAINcallRate = (CBHCA ÷ 3600)
= (200,000 ÷ 3600)
= 56 calls/second

CAINMultiPartyCallRate = CainCallRate x %MultiPartyCalls x %MultiPartySetupTime = (56) x (.50) x (.10) = 3 calls/sec

---

ExtendedCallCondenseBlocksrequired = 1 (only one extended block is required per call process)

---

therefore:

NumberCAINextCallCondenseBlocksRequired=(CAINcallRate+CAIN multi Party Call Rate) x Holding time x 1.62 x ExtendedCallCondenseBlocksRequiredPerCall = (56 + 3) x 200 x 1.62 x 1 = 19116 extended call condense blocks

Therefore, you should provision 19116 extended call condense blocks in the NUM_CAIN_ECCBS parameter.

NUM_CAIN_EXT_BLOCKS

An extension block is a storage mechanism used to store feature data required for a single call. The CAIN extension block is used to store data needed for triggering, routing, and connecting to a resource in the originating call model.

ATTENTION

The switch requires CAIN extension blocks in order to query the SCP in the originating call model. The default is 260.

When a TDP is encountered in the originating call model, trigger criteria is met, and an action of QUERY or FEAT is provisioned, the CAIN framework determines if a CAIN extension block has been allocated. If not, a new CAIN extension block is allocated. CAIN extension blocks normally remain allocated until a call is answered. However, some scenarios, such as fatal application errors, may release the extension block early. Other scenarios such as those involving mid–call TDPs and EDPs may require the extension blocks to be held through the entire life of the call.

During a call process, more than one CAIN extension block may be allocated. However, for each call process there is never more than one CAIN extension block allocated at a single point in time. A call may contain at most 2 call processes so at most 2 extension blocks are allocated at one time. For example:

1 The switch encounters ***Info_Analyzed*** and trigger criteria is met with a QUERY action. CAIN extension blocks and framework extension blocks (discussed in the next section, NUM_FRAMEWORK_EXT_BLOCKS) are allocated and an **`Info_Analyzed`** message is sent to the SCP.

2 The SCP responds with a **`Continue`** message which is processed by the switch. Both extension blocks are deallocated and the call continues at **Select_Route**.

3 Later in the call, the switch encounters *Network_Busy* and trigger criteria is met with a QUERY action. CAIN and framework extension blocks are allocated and a **`Network_Busy`** message is sent to the SCP.

4 The SCP responds with an **`Analyze_Route`** message that is processed in order to identify a route index. It also arms the ***O_Mid_Call*** EDP for the *switchHookFlash* event. Call processing enters **Select_Route**.

5 The SSP detects the asterisk key being pressed by the operator and sends an **O_Mid_Call** message to the SCP.

6 The SCP responds by sending a **`Connect_To_Resource`** message to the SSP instructing it to collect the address digits of the third party.

7 The SSP sends a **`Resource_Clear`** message containing the address digits.

8 The SCP responds with an **`Originate_Call`** message instructing the SSP to create another call segment and arm the **`O_Disconnect`** and **`O_Abandon`** EDPs.

9 The operator goes on–hook causing an **`O_Abandon`** message to be sent to the SCP.

10 The SCP responds with a **`Merge_call`** message instructing the SSP to merge the two remaining legs of the call and rearm the **`O_Disconnect`** EDP.

11 The SSP sends an **`O_Disconnect`** message when one of the remaining parties goes on hook.

12 The SCP responds with a **`Disconnect`** message bringing down the call.

13 Both extension blocks are deallocated.

Engineering CAIN extension blocks

Use the following formulas to estimate the number of CAIN extension blocks you will need to provision for parameter NUM_CAIN_EXT_BLOCKS.

NumberCAINextBlocksRequired = CAINcallRate × HoldingTime × ExtBlocksRequiredPerCall

where:

CAINcallRate = (BHCA ÷ 3600) × Call%RequiringSCPserviceLogic

HoldingTime = (Q-R% × AvgQ-RTime) + (STRConn% × AvgSTRConnTime) + (MultReq% × AvgMultReqTime) + (IP% × AvgIPtime) + (MP% x AvgMultiPartyTime) +(Timeout% x AvgTimeoutTimerTime)

where:

AvgQ–RTime =T1timerValue +AvgCallAnswerTime

AvgSTRConnTime =T1timerValue +(STRmsgsInConvPkg x (AvgRsrcTime + T1timerValue)) + AvgCallAnswerTime

AvgMultReqTime = (AvgNumberQueriesPerCall × T1timerValue) +AvgCallSetupTime+ AvgCallAnswerTime

AvgMultiPartyTime= (AvgMultiPartyConnTime + AvgMultiPartySetupTime) x AvgNumber3rdPartyConn +T1timerValue

AvgIPTime = T1timerValue +STRmsgsInConvPkg x (AvgIPsetupTime + AvgRsrcTime + T1timerValue) + AvgCallAnswerTime

AvgTimeoutTimerTime =T1TimerValue+AvgCallAnswerTime + AvgNumberTimeoutEventsReq x (AvgTimeouttimerValue x 60 + T1timerValue)

ExtBlocksRequiredPerCall = 1 (only one extension block is required per call process)

CAIN call rate calculation

The CAIN call rate is the number of calls per second that require SCP service logic. To estimate the call rate, use the following formula:

CAINcallRate = (BHCA ÷ 3600) × Call%RequiringSCPserviceLogic

Variable definitions

BHCA – Your estimate of the call volume on the switch, expressed in busy-hour call attempts

Call%RequiringSCPserviceLogic – Your estimate of the percentage of BHCAs that require SCP service logic

3600 represents the number of seconds in an hour (60 seconds/minute × 60 minutes/hour)

Holding time calculation

The holding time refers to the amount of time in seconds that a CAIN extension block is allocated on a call. The holding time varies depending upon the call scenario. As there is an unlimited number of possible scenario combinations, you will estimate an average time based on the following basic scenarios:

- Query-Response – The switch queries the SCP and receives a response. The switch may send a subsequent EDP-Notification message, **`Close`** message, or both.
- STR or CTR Connection Inswitch – The switch queries the SCP and receives a **`Send_To_Resource`** or **`Connect_To_Resource`** message in a conversation package. The switch performs the requested operation and sends a **`Resource_Clear`** or **`CTR_Clear`** message to the SCP upon completion. This may be repeated several times until the switch receives a message in a response package.
- Multiple requests – The switch queries the SCP and receives an **`Analyze_Route`** or **`Continue`** message. Later in the call model the switch may send another TDP-Request or EDP-Request (at ***Network_Busy***, ***O_Called_Party_Busy***, or ***O_No_Answer***) and receive another response.
- STR- or CTR-Connection to an Intelligent Peripheral – The switch queries the SCP and receives a **`Send_To_Resource`** or **`Connect_To_Resource`** message (containing a ***`DestinationAddress`***) in a conversation package. A connection is established to an IP. Once the IP performs the requested function, the switch sends a **`Resource_Clear`** or **`CTR_Clear`** message to the SCP. This may be repeated several times until the switch receives a message in a response package.
- Multi Party–Multi Party call handling becomes active when the *switchHookFlash* event is armed at the ***O_Mid_Call*** EDP. This scenario requires a request message to be sent at a previous trigger in order to arm EDPs. A Multi Party call remains active until we either receive a Disconnect message from the SCP, only one agent remains on the call, or we return to call configuration 2 (refer to Volume 3, Chapter 4, “Call Configurations,” chapter for Multi call party handling) and the *switchHookFlash* event has not been rearmed .
- Timeout Requests – The timeout timer is activated when the call is answered if the timeout event is armed at the ***O_Mid_Call*** EDP. When the timeout timer expires, a **`Timeout`** message is sent to the SCP. The SCP’s only valid response is either a **`Disconnect`** or **`Connect_To_Resource`** message. If it is a **`Connect_To_Resource`** message, refer to the descriptions above for **`Connect_To_Resource`** interactions, otherwise the call will come down and the extension block will be deallocated.

Therefore, the holding time can be estimated by taking an average time based on the following basic scenarios:

HoldingTime = (Q-R% × AvgQ-RTime) + (STRConn% × AvgSTRConnTime) + (MultReq% × AvgMultReqTime) + (IP% × AvgIPtime) +(MP% x AvgMultiPartytime) + (Timeout% x AvgTimeoutTimerTime)

where:

AvgQ-RTime = T1timerValue + AvgCallAnswerTime

AvgSTRConnTime = T1timerValue + (STRmsgsInConvPkg × (AvgRsrcTime + T1timerValue)) + AvgCallAnswerTime

AvgMultReqTime = (AvgNumberQueriesPerCall × T1timerValue) + AvgCallSetupTime + AvgCallAnswerTime

AvgMultiPartyTime= (AvgMultiPartyConnTime + AvgMultiPartySetupTime) x AvgNumber3rdPartyConn +T1timerValue

AvgIPTime = T1timerValue +STRmsgsInConvPkg x (AvgIPsetupTime + AvgRsrcTime + T1timerValue) + AvgCallAnswerTime

AvgTimeoutTimerTime = T1TimerValue+AvgCallAnswerTime+ AvgNumberTimeoutEventsReq x (AvgTimeouttimerValue x 60 + T1timerValue)

Variable definitions

Q-R% – Your estimate of the percentage of the CAIN call rate that will be query-response scenarios

STRConn% – Your estimate of the percentage of the CAIN call rate that will require in-switch STRs

MP% – Your estimate of the percentage of CAIN call rate that will be multi party scenarios

MultReq% – Your estimate of the percentage of CAIN call rate that will be multiple request scenarios

IP% – Your estimate of the percentage of CAIN call rate that will require STR- or CTR-Connections to an IP

T1timerValue – Use the value datafilled in the CAIN_T1_TIMEOUT (table CAINPARM)

Timeout% – Your estimate of the percentage of the CAIN call rate that will require arming the Timeout EDP

AvgCallAnswerTime – Your estimate of the average time in seconds it takes to establish a call through the network and ring time prior to answer

STRmsgsInConvPkg –Your estimate of the number of **`Send_To_Resource`** or **`Connect_To_Resource`** packages that will be received in a conversation package.

AvgRsrcTime – Your estimate of the time, in seconds, required for user interaction (including the time connected to a resource (interruptible or uninterruptible), time required to collect the subscriber's dialed digits, time required due to reset dialing.)

AvgCallSetupTime – Your estimate of the time, in seconds, required to set up the call and reach the second DP. (When figuring the time for ***Network_Busy*** or ***O_Called_Party_Busy***, remember to include the time required for the busy condition to be recognized by the switch. For ***O_No_Answer***, remember to include the ***O_No_Answer*** timer value.)

AvgIPsetupTime –Your estimate of the time, in seconds, for the call to be answered by the IP (including the time required to establish the call through the network and ring time prior to answer)

Variable definitions

AvgMultPartyConnTime – Time in seconds it takes from the point where the switchHookFlash event was armed until the conversation has ended.

AvgNo3rdPartyConn – Number of 3rd party agents conferenced into the call.

AvgNoTimeoutEventsReq –The number of Timeout events requested by the SCP (ie. how many times the Timeout event is armed).

AvgTimeoutTimerValue –The number in minutes for the average Timeout length

AvgNumberQueriesPerCall – Your estimate of the average number of CAIN queries per call.

AvgMultiPartySetupTime –Time it takes, in seconds, from the point at which we receive the **`Originate_Call`** message until we merge the 3rd party into the existing call.

Example

For this example, you have gathered the following data:

- BHCA = 400,000 calls per hour
- You expect 5% of the BHCA to require CAIN services.
- You expect the call mix to be
  - 50% will be Query-Response scenarios
  - 20% will be In–switch STR scenarios
  - 5% will be Multiple requests with **`Network_Busy`** as the second request
  - 5% will require STR-Connections to an IP
  - 10% will require multi-party call handling
  - 10% will arm the ***O_Mid_Call*** EDP for the *Timeout* event
- You have datafilled CAIN_T1_TIMEOUT as 2 seconds
- You expect the average call answer time to be 24 seconds
- You expect to receive one **`Send_To_Resource`** or **`Connect_To_Resource`** message in a conversation package
- You expect the resource time to be 30 seconds for one user interaction
- You expect the call setup time to be 1 second
- You expect the IP setup time to be 5 seconds
- You expect the multi-party connection time to be 60 seconds
- You expect the multi-party setup time to be 5 seconds

- You expect the number of 3rd party connections to be one
- You expect the number of *Timeout* events requested to be one
- You expect the average timeout timer to be 15 minutes
- You know that you require one extension block per call process

Therefore,

CAINcallRate = (BHCA ÷ 3600) × Call%RequiringSCPserviceLogic
= (400,000÷ 3600) × .05
= 6 calls/second

AvgQ-RTime = T1timerValue + AvgCallAnswerTime
= 2 + 24
= 26 seconds

AvgSTRConnTime = T1timerValue + (STRmsgsInConvPkg × (AvgRsrcTime + T1timerValue))
+ AvgCallAnswerTime
= 2 + (1 × (30 + 2)) + 24
= 58 seconds

AvgMultReqTime = (AvgNumberQueriesPerCall× T1timerValue) +AvgCallSetuptime +
AvgCallAnswerTime
= (2 × 2) + 1 + 24
= 29 seconds

AvgMultiPartyTime = (AvgMultiPartyConnTime +AvgMultiPartySetupTime) x
AvgNumber3rdPartyConn +T1timervalue
= (60 +5) x 1 +2
= 67 seconds

AvgIPTime = T1timerValue +(STRMsgsInConvPkg x (AvgIPSetupTime + AvgRsrcTime
+ T1timerValue)) + AvgCallAnswerTime
= 2 +(1 x (5 + 30 + 2)) +24
= 63 seconds

AvgTimeoutTimerTime = T1TimerValue+AvgCallAnswerTime +AvgNumberTimeoutEventsReq x
(AvgTimeouttimerValue x 60 + T1timerValue)
= 2+24 +(1 x ((15 x 60) +2)) = 928 seconds

Therefore:

HoldingTime = (Q–R% x AvgQ–RTime) +(STRConn% x AvgSTRConnTime) +(MultReq% x
AvgMultReqTime) +(IP% x AvgIPTime) + (MP% x AvgMultiPartyTime) +
(Timeout% x AvgTimeoutTimerTime)
= (.50 x 26) +(.20 x 58) +(.05 x 29) +(.05 x 67) +(.10 x 63) +(.10 x 928)
= 13+ 11.60 +1.45+3.25+6.30+92.80= 128.40 seconds = 128 seconds

---

ExtBlocksRequiredPerCall = 1 (only 1 extension block is required per call process)

NumberCAINextBlocksRequired = CAINcallRate x HoldingTime x ExtBlocksrequiredPerCall
= 6 x 128 x 1
= 768 extension blocks

Therefore, you should provision 768 extension blocks in the NUM_CAIN_EXT_BLOCKS parameter.

Note: The value for extension blocks may be increased to provide a buffer against peak or abnormal traffic loads.

NUM_T_CAIN_EXT_BLOCKS

An extension block is a storage mechanism used to store feature data required for a single call. The T_CAIN extension block is used to store data

needed for triggering and routing in the terminating call model.

> **ATTENTION**
> The switch requires T_CAIN extension blocks in the Terminating call model in order to query the SCP. The default is 260.

When a TDP is encountered in the terminating call model, trigger criteria is met, and an action of QUERY is provisioned, the CAIN framework allocates a T_CAIN extension block. If not, a new T_CAIN extension block is allocated. T_CAIN extension blocks normally remain allocated until the required processing has occurred in the **T_Null** PIC. However, some scenarios, such as fatal application errors, may release the extension block early.

Engineering T_CAIN extension blocks

Use the following formulas to estimate the number of T_CAIN extension blocks you will need to provision for parameter NUM_T_CAIN_EXT_BLOCKS.

NumberTCAINextBlocksRequired = CAINcallRate × HoldingTime × ExtBlocksRequiredPerCall

where:

TCAINcallRate = (BHCA ÷ 3600) × Call%RequiringSCPserviceLogic

HoldingTime = (Q-R% × AvgQ-RTime) + (Conv% × AvgConvTime)

where:

AvgQ-RTime = T1timerValue + AvgCallAnswerTime

AvgConvTime = T1timerValue + (STRmsgsInConvPkg × (AvgRsrcTime + T1timerValue)) + AvgCallAnswerTime

ExtBlocksRequiredPerCall = 1 (only one extension block is required per call)

CAIN call rate calculation

The CAIN call rate is the number of calls per second that require SCP service logic. To estimate the call rate, use the following formula:

TCAINcallRate = (BHCA ÷ 3600) × Call%RequiringSCPserviceLogic

Variable definitions

BHCA – Your estimate of the call volume on the switch, expressed in busy-hour call attempts

Call%RequiringSCPserviceLogic – Your estimate of the percentage of BHCAs that require SCP service logic

3600 represents the number of seconds in an hour (60 seconds/minute × 60 minutes/hour)

TCAINcallRate – Your estimate of the number of terminating calls

Holding time calculation

The holding time refers to the amount of time in seconds that a T_CAIN extension block is allocated on a call. The holding time varies depending upon the call scenario. As there are many possible scenario combinations, you will estimate an average time based on the following basic scenarios:

- Query-Response – The switch queries the SCP and receives a response. The switch may send a subsequent **`Close`** message.
- Conversation – The switch queries the SCP and receives a **`Send_To_Resource`** message in a conversation package. The switch performs the requested operation and sends a **`Resource_Clear`** message to the SCP upon completion. This may be repeated several times until the switch receives a message in a response package.

HoldingTime = (Q-R% × AvgQ-RTime) + (Conv% × AvgConvTime)

where:

AvgQ-RTime = T1timerValue + AvgCallAnswerTime

AvgConvTime = T1timerValue + (STRmsgsInConvPkg × (AvgRsrcTime + T1timerValue)) + AvgCallAnswerTime

Variable definitions

Q-R% – Your estimate of the percentage of the CAIN call rate that will be query-response scenarios

Conv% – Your estimate of the percentage of the CAIN call rate that will be conversation scenarios

T1timerValue – Use the value datafilled in the CAIN_T1_TIMEOUT (table CAINPARM)

AvgCallAnswerTime – Your estimate of the average time it takes, in seconds, to establish a call through the network and ring time prior to answer

STRmsgsInConvPkg –Your estimate of the number of **`Send_To_Resource`** packages that will be received in a conversation package.

AvgRsrcTime – Your estimate of the time, in seconds, required for user interaction (including the time connected to a resource (interruptible or uninterruptible), time required to collect the subscriber's dialed digits, time required due to reset dialing.)

Example

For this example, you have gathered the following data:

- BHCA = 400,000 calls per hour
- You expect 5% of the BHCA to require CAIN services.
- You expect the call mix to be
  - — 80% will be Query-Response scenarios
  - — 20% will be Conversation scenarios
- You have datafilled CAIN_T1_TIMEOUT as 2 seconds
- You expect the average call answer time to be 24 seconds
- You expect to receive one **`Send_To_Resource`** message in a conversation package
- You expect the resource time to be 30 seconds for one user interaction

- You expect the call setup time to be 1 second
- You know that you require one extension block per call

Therefore,

CAINcallRate	=	(BHCA ÷ 3600) × Call%RequiringSCPserviceLogic
	=	(400,000÷ 3600) × .05
	=	6 calls/second

AvgQ-RTime	=	T1timerValue + AvgCallAnswerTime
	=	2 + 24
	=	26 seconds
AvgConvTime	=	T1timerValue + (STRmsgsInConvPkg × (AvgRsrcTime + T1timerValue)) + AvgCallAnswerTime
	=	2 + (1 × (30 + 2)) + 24
	=	58 seconds

Therefore:

HoldingTime	=	(Q-R% × AvgQ-RTime) + (Conv% × AvgConvTime)
	=	(.80)(26) + (.20)(58)
	=	32 seconds

ExtBlocksRequiredPerCall	=	1 (only one extension block is required per call)

Therefore:

NumberTCAINextBlocksRequired	=	CAINcallRate × HoldingTime × ExtBlocksRequiredPerCall
	=	6 × 32 × 1
	=	192 extension blocks

Therefore, you should provision 192 extension blocks in the NUM_T_CAIN_EXT_BLOCKS parameter.

Note: The value for extension blocks may be increased to provide a buffer against peak or abnormal traffic loads.

NUM_FRAMEWORK_EXT_BLOCKS

An extension block is a storage mechanism used to store feature data required for a single call. The framework extension block is used to store data, such as state and transaction identifiers, needed by the CAIN messaging framework.

ATTENTION

This parameter should be engineered when NetworkBuilder is introduced for the first time.

ATTENTION

The switch requires framework extension blocks in order to query the SCP. The default is 1800.

For both the originating and terminating call models, when a TDP is encountered, trigger criteria is met, and an action of QUERY (or FEAT for the originating call model only) is provisioned, the CAIN framework determines if a framework extension block has been allocated. If not, a new framework extension block is allocated. Framework extension blocks normally remain allocated until a call is answered. The default is 1800. However, some scenarios, such as fatal application errors, may release the call early. Other scenarios such as those involving mid–call TDPs and EDPs may require the extension blocks to be held through the entire lifecycle of the call.

During a call, more than one CAIN framework extension block may be allocated. However, for each call process there is never more than one CAIN framework extension block allocated at a single point in time. A call may contain at most two call processes therefore at most two extension blocks can be allocated at one time.

In most cases, the CAIN and framework extension blocks are allocated and deallocated at the same time. However, one exception occurs during a **`Send_To_Resource`** or **`Connect_To_Resource`** connection to an intelligent peripheral (IP). When the IP answers, only the CAIN extension block is deallocated. The framework extension block remains allocated in order to store the data required to send a **`Resource_Clear`** or **`CTR_Clear`** message following the STR- or CTR-Connection. For example:

1 The switch encounters ***Info_Analyzed*** and trigger criteria is met with a QUERY action. CAIN and framework extension blocks are allocated and an **`Info_Analyzed`** message is sent to the SCP.

2 The SCP responds with a **`Send_To_Resource`** message containing a ***`DestinationAddress`*** parameter. The switch processes the message and establishes a connection to an IP.

3 When the IP answers, only the CAIN extension block is deallocated.

4 Once the IP performs its function and releases the call, a CAIN extension block is allocated. The switch sends a **`Resource_Clear`** message to the SCP.

5 The SCP responds with an **`Analyze_Route`** message that is processed in order to identify a route index. Call processing enters **Select_Route**.

6 When the call is answered, both extension blocks are deallocated.

Engineering CAIN framework extension blocks

Use the following formulas to estimate the number of CAIN extension blocks you will need to provision for parameter NUM_FRAMEWORK_EXT_BLOCKS.

NumberFrameworkExtBlocksRequired = CAINcallRate × HoldingTime × ExtBlocksRequiredPerCall

where:

CAINcallRate = (BHCA ÷ 3600) × Call%RequiringSCPserviceLogic

HoldingTime = (Q-R% × AvgQ-RTime) + (STRConv% × AvgSTRConvTime) + (MultReq% × AvgMultReqTime) + (IP% × AvgIPtime) +(MP% x AvgMultiPartyTime) + (Timeout% x AvgTimeoutTimerTime)

where:

AvgQ-RTime = T1timerValue + AvgCallAnswerTime

AvgSTRConvTime =T1timerValue + (STRmsgInConvPkg × (AvgRsrcTime + T1timerValue) + AvgCallAnswerTime

AvgMultReqTime = (AvgNumberQueriesPerCall × T1timerValue) +AvgCallSetupTime + AvgCallAnswerTime

AvgMultiPartyTime= (MultiPartyConnTime + AvgMultiPartySetupTime) x AvgNumber3rdPartyConn +T1timerValue

AvgIPTime = T1timerValue +(STRmsgsInConvPkg x (AvgIPsetupTime + AvgRsrcTime + T1 timerValue)) + AvgCallAnswerTime

AvgTimeoutTimerTime = T1TimerValue+AvgCallAnswerTime+AvgNumberTimeoutEventsReq x (AvgTimeouttimerValue x 60 + T1timerValue)

ExtBlocksRequiredPerCall =1 (only one extension block is required per call)

CAIN call rate calculation

The CAIN call rate is the number of calls per second that require SCP service logic. To estimate the call rate, use the following formula:

CAINcallRate	=	(BHCA ÷ 3600) × Call%RequiringSCPserviceLogic

Variable definitions

BHCA – Your estimate of the call volume on the switch, expressed in busy-hour call attempts

Call%RequiringSCPserviceLogic – Your estimate of the percentage of BHCAs that require SCP service logic

3600 represents the number of seconds in an hour (60 seconds/minute × 60 minutes/hour)

Holding time calculation

The holding time refers to the amount of time in seconds that a CAIN extension block is allocated on a call. The holding time varies depending upon the call scenario. As there is an unlimited number of possible scenario combinations, you will estimate an average time based on the following basic scenarios:

- Query-Response – The switch queries the SCP and receives a response. The switch may send a subsequent EDP-Notification message, **`Close`** message, or both.
- STR or CTR Connection Inswitch – The switch queries the SCP and receives a **`Send_To_Resource`** or **`Connect_To_Resource`** message in a conversation package. The switch performs the requested operation and sends a **`Resource_Clear`** or **`CTR_Clear`** message to the SCP upon completion. This may be repeated several times until the switch receives a message in a response package.
- Multiple requests – The switch queries the SCP and receives an **`Analyze_Route`** or **`Continue`** message. Later in the call model the switch may send another TDP-Request or EDP-Request (at ***Network_Busy***, ***O_Called_Party_Busy***, or ***O_No_Answer***) and receive another response.
- STR- or CTR-Connection to an Intelligent Peripheral – The switch queries the SCP and receives a **`Send_To_Resource`** or **`Connect_To_Resource`** message (containing a ***`DestinationAddress`***) in a conversation package. A connection is established to an IP. Once the IP performs the requested function, the switch sends a **`Resource_Clear`** or **`CTR_Clear`** message to the SCP. This may be repeated several times until the switch receives a message in a response package.
- Multi Party–Multi Party call handling becomes active when the *switchHookFlash* event is armed at the ***O_Mid_Call*** EDP. This scenario requires a request message to be sent at a previous trigger in order to arm EDPs. A Multi Party call remains active until we either receive a **`Disconnect`** message from the SCP, only one agent remains on the call,

or we return to call configuration 2 (refer to Volume 3, Chapter 4, “Call Configurations,” chapter for Multi call party handling) and the *switchHookFlash* event has not been rearmed .

- Timeout Requests – The timeout timer is activated when the call is answered if the timeout event is armed at the ***O_Mid_Call*** EDP. When the timeout timer expires a **`Timeout`** message is sent to the SCP. The SCPs only valid response is either a **`Disconnect`** or **`Connect_To_Resource`** message. If it is a **`Connect_To_Resource`** message, refer to the descriptions above for CTR interactions, otherwise the call will come down and the extension block will be deallocated.

Therefore, the holding time can be estimated by taking an average time based on the following basic scenarios:

HoldingTime = (Q-R% × AvgQ-RTime) + (STRConn% × AvgSTRConnTime) + (MultReq% × AvgMultReqTime) + (IP% × AvgIPtime) + (MP% x AvgMultiPartytime) +(Timeout% x AvgTimeoutTimerTime)

where:

AvgQ-RTime = T1timerValue + AvgCallAnswerTime

AvgSTRConnTime = T1timerValue + (STRmsgsInConvPkg × (AvgRsrcTime + T1timerValue)) + AvgCallAnswerTime

AvgMultReqTime = (AvgNumberQueriesPerCall × T1timerValue) + AvgCallSetupTime + AvgCallAnswerTime

AvgMultiPartyTime=(AvgMultiPartyConnTime + AvgMultiPartySetupTime) x AvgNumber3rdPartyConn +T1TimerValue

AvgIPTime = T1timerValue +(STRmsgsInConvPkg x (AvgIPsetupTime + AvgRsrcTime + T1timerValue)) + AvgCallAnswerTime

AvgTimeoutTimerTime = T1TimerValue+AvgCallAnswerTime+ AvgNumberTimeoutEventsReq x (AvgTimeouttimerValue x 60 + T1timerValue)

Variable definitions

Q-R% – Your estimate of the percentage of the CAIN call rate that will be query-response scenarios

STRConn% – Your estimate of the percentage of the CAIN call rate that will require inswitch STRs

MP% – Your estimate of the percentage of CAIN call rate that will be multi party scenarios

MultReq% – Your estimate of the percentage of CAIN call rate that will be multiple request scenarios

IP% – Your estimate of the percentage of CAIN call rate that will require STR- or CTR-Connections to an IP

T1timerValue – Use the value datafilled in the CAIN_T1_TIMEOUT (table CAINPARM)

Timeout% – Your estimate of the percentage of the CAIN call rate that will require arming the Timeout EDP.

AvgCallAnswerTime – Your estimate of the average time it takes, in seconds, to establish a call through the network and ring time prior to answer

STRmsgsInConvPkg –Your estimate of the number of **`Send_To_Resource`** or **`Connect_To_Resource`** packages that will be received in a conversation package.

AvgRsrcTime – Your estimate of the time, in seconds, required for user interaction (including the time connected to a resource (interruptible or uninterruptible), time required to collect the subscriber's dialed digits, time required due to reset dialing.)

AvgCallSetupTime – Your estimate of the time, in seconds, required to set up the call and reach the second DP. (When figuring the time for ***Network_Busy*** or ***O_Called_Party_Busy***, remember to include the time required for the busy condition to be recognized by the switch. For ***O_No_Answer***, remember to include the ***O_No_Answer*** timer value.)

AvgIPsetupTime –Your estimate of the time, in seconds, for the call to be answered by the IP (including the time required to establish the call through the network and ring time prior to answer)

AvgMultiPartyConnTime –Time it takes, in seconds, from the point when the *switchHookFlash* event was armed until the conversation has ended.

Variable definitions

AvgNo3rdPartyConn –Number of 3rd party agents conferenced into the call.

AvgNoTimeoutEventsReq –The number of Timeout events requested by the SCP (ie. how many times the Timeout event is armed).

AvgTimeoutTimerValue –The number in minutes for the average Timeout length

AvgNumberQueriesPerCall – Your estimate of the average number of CAIN queries per call.

AvgMultiPartySetupTime –Time it takes, in seconds, from the point at which we receive the **`Originate_Call`** message until we condense into a call with a single call segment.

Example

For this example, you have gathered the following data:

- BHCA = 400,000 calls per hour
- You expect 5% of the BHCA to require CAIN services.
- You expect the call mix to be
  - — 50% will be Query-Response scenarios
  - — 20% will be In-switch STR scenarios
  - — 5% will be Multiple requests with **`Network_Busy`** as the second request
  - — 5% will require STR-Connections
  - — 10% will require multi-party call handling
  - — 10% will arm the Timeout EDP
- You have datafilled CAIN_T1_TIMEOUT as 2 seconds
- You expect the average call answer time to be 24 seconds
- You expect to receive one **`Send_To_Resource`** or **`Connect_To_Resource`** message in a conversation package
- You expect the resource time to be 30 seconds for one user interaction
- You expect the call setup time to be 1 second
- You expect the IP setup time to be 5 seconds
- You expect the multi-party connection time to be 60 seconds
- You expect the multi-party setup time to be 5 seconds
- You expect the number of 3rd party connections to be one

- You expect the number of Timeout events requested to be one
- You expect the average timeout timer to be 15 minutes
- You know that you require one extension block per call

Therefore,

CAINcallRate = (BHCA ÷ 3600) × Call%RequiringSCPserviceLogic
= (400,000÷ 3600) × .05
= 6 calls/second

AvgQ-RTime = T1timerValue + AvgCallAnswerTime
= 2 + 24
= 26 seconds

AvgSTRConnTime = T1timerValue + (STRmsgsInConvPkg × (AvgRsrcTime + T1timerValue)) + AvgCallAnswerTime
= 2 + (1 × (30 + 2)) + 24
= 58 seconds

AvgMultReqTime = (AvgNumberQueriesPerCall× T1timerValue) +AvgCallSetuptime
+ AvgCallAnswerTime
= (2 × 2) + 1 + 24
= 29 seconds

AvgMultiPartyTime = (AvgMultiPartyConnTime +AvgMultiPartySetupTime) x AvgNumber3rdPartyConn +T1timervalue
= (60 +5) x 1 +2
= 67 seconds

AvgIPTime = T1timerValue +STRMsgsInConvPkg x (AvgIPSetupTime + AvgRsrcTime + T1timerValue) + AvgCallAnswerTime
= 2 +(1 x (5 +30 +2))+24
= 63 seconds

AvgTimeoutTimerTime = T1TimerValue+AvgCallAnswerTime +AvgNumberTimeoutEventsReq × (AvgTimeouttimerValue x 60 + T1timerValue)
= 2+24 +(1 x ((15 x 60) +2)) = 928 seconds

Therefore:

HoldingTime = (Q–R% x AvgQ–RTime) +(STRConn% x AvgSTRConnTime) +(MultReq% x AvgMultReqTime) +(IP% x AvgIPTime) + (MP% x AvgMultiPartyTime) + (Timeout% x AvgTimeoutTime)
= (.50 x 26) +(.20 x 58) +(.05 x 29) +(.05 x 67) +(.10 x 63) +(.10 x 928)
= 13+ 11.60 +1.45+3.25+6.30+92.80= 128.40 seconds = 128 seconds

---

ExtBlocksRequiredPerCall = 1 (only 1 extension block is required per call process)

NumberCAINextBlocksRequired = CAINcallRate x HoldingTime x ExtBlocksrequiredPerCall
= 6 x 128 x 1
= 768extension blocks

Therefore, you should provision 768 extension blocks in the NUM_FRAMEWORK_EXT_BLOCKS parameter.

Note: The value for extension blocks may be increased to provide a buffer against peak or abnormal traffic loads.

NUM_STR_EXT_BLOCKS

An extension block is a storage mechanism used to store feature data required for a single call. The STR extension block is used to store data needed for managing a connection to a switch resource.

When a **Send_To_Resource** or **Connect_To_Resource** message is received, the CAIN user interaction framework determines if an STR extension block has been allocated. If not, a new STR extension block is allocated. STR extension blocks normally remain allocated until a call is routed. However, some scenarios (such as fatal application errors) may release the extension block early. The default value of the NUM_STR_EXT_BLOCKS parameter is 260.

During a call, more than one STR extension block may be allocated. However, for each call there is never more than one STR extension block allocated at a single point in time.

For example:

1 The switch encounters ***Info_Analyzed*** and trigger criteria is met with a QUERY action. CAIN and framework extension blocks are allocated and an **Info_Analyzed** message is sent to the SCP.

2 The SCP responds with a **Send_To_Resource** conversation message, which is processed by the switch. The user interaction is completed and a **Resource_Clear** message with subscriber digits is returned to the SCP.

3 The SCP responds with an **Analyze_Route** message that is processed to identify a route index. Call processing enters **Select_Route**. The STR extension block is deallocated and the call continues at **Select_Route**.

4 Later in the call, the switch encounters ***Network_Busy*** and trigger criteria is met with a QUERY action. The SCP responds with a second **Send_To_Resource** conversation message. An STR extension block is allocated and the user interaction is processed by the switch, followed by a **Resource_Clear** message with subscriber digits.

5 The SCP responds with an **Analyze_Route** message that is processed in order to identify a route index. Call processing enters **Select_Route**.

6 The STR extension block is deallocated and the call continues at **Select_Route**.

Engineering STR extension blocks

Use the following formulas to estimate the number of STR extension blocks you will need to provision for parameter NUM_STR_EXT_BLOCKS.

NumberSTRextBlocksRequired = CAINcallRate × HoldingTime × ExtBlocksRequiredPerCall

where:

CAINcallRate = (BHCA ÷ 3600) × Call%RequiringSCPserviceLogic

HoldingTime = (ConvSTR% × AvgConvSTRTime) + (RespSTR% x AvgResourceTime)

where:

AvgConvSTRTime = AvgConvSTRmessages x (AvgResourceTime +T1timerValue) + AvgCallSetupTime

ExtBlocksRequiredPerCall = 1 (only one extension block is required per call process)

CAIN call rate calculation

The CAIN call rate is the number of calls per second that require SCP service logic. To estimate the call rate, use the following formula:

CAINcallRate = (BHCA ÷ 3600) × Call%RequiringSCPserviceLogic

Variable definitions

BHCA – Your estimate of the call volume on the switch, expressed in busy-hour call attempts

Call%RequiringSCPserviceLogic – Your estimate of the percentage of BHCAs that require SCP service logic

3600 represents the number of seconds in an hour (60 seconds/minute × 60 minutes/hour)

Holding time calculation

The holding time refers to the amount of time in seconds that an STR extension block is allocated on a call. The holding time varies depending upon the call scenario. As there is an unlimited number of possible scenario combinations, you will estimate an average time based on the three following basic scenarios:

- Response STR – The switch queries the SCP and receives a **`Send_To_Resource`** or **`Connect_To_Resource`** message in a response package with the disconnect flag. The switch performs the requested operation and applies AIND treatment and disconnects the call.
- Conversation STR– The switch queries the SCP and receives a **`Send_To_Resource`** or **`Connect_To_Resource`** message in a conversation package. The switch performs the requested operation and sends a **`Resource_Clear`** or **`CTR_Clear`** message to the SCP upon

completion. This may be repeated several times until the switch receives a message in a response package.

Therefore, the holding time can be estimated by taking an average of the three scenarios, using the following formulas:

HoldingTime = (ConvSTR% x AvgConvSTRtime) +(RespSTR% × AvgResourceTime)

where:

AvgConvSTRTime = AvgConvSTRmessages x (AvgResourceTime+T1timerValue) + AvgCallSetupTime

Variable definitions

ConvSTR% – Your estimate of the percentage of the CAIN call rate that will be conversation scenarios

RespSTR% – Your estimate of the percentage of the CAIN call rate that will be multiple request scenarios

T1timerValue – Use the value datafilled in the CAIN_T1_TIMEOUT (table CAINPARM)

AvgConvSTRmessages –Your estimate of the number of **`Send_To_Resource`** or **`Connect_To_Resource`** packages that will be received in a conversation package.

AvgRsrcTime – Your estimate of the time, in seconds, required for user interaction (including the time connected to a resource (interruptible or uninterruptible), time required to collect the subscriber's dialed digits, time required due to reset dialing)

AvgCallSetupTime – Your estimate of the time, in seconds, required to set up the call and reach the second DP. (When figuring the time for ***Network_Busy*** or ***O_Called_Party_Busy***, remember to include the time required for the busy condition to be recognized by the switch. For ***O_No_Answer***, remember to include the ***O_No_Answer*** timer value.)

Example

For this example, you have gathered the following data:

- BHCA = 400,000 calls per hour
- You expect 5% of the BHCA to require CAIN services.
- You expect the call mix to be
  - — 70% will be Query-Response scenarios
  - — 30% will be Conversation scenarios
- You have datafilled CAIN_T1_TIMEOUT as 2 seconds
- You expect the average call answer time to be 24 seconds
- You expect to receive one **`Send_To_Resource`** or **`Connect_To_Resource`** message in a conversation package
- You expect the resource time to be 30 seconds for one user interaction
- You know that you require one extension block per call

Therefore,

CAINcallRate = (BHCA ÷ 3600) × Call%RequiringSCPserviceLogic
= (400,000÷ 3600) × .05
= 6 calls/second

AvgConvSTRTime = AvgConvSTRmessages x (AvgResourceTime+T1timerValue) + AvgCallSetupTime
= 1 x (30+2) +1
= 33 seconds

HoldingTime = (ConvSTR% × AvgConvSTRTime) + (RespSTR% x AvgResourceTime)
= (.30 × 33) +(.70 x 30)
= 31 seconds

ExtBlocksRequiredPerCall = 1 (only one extension block is required per call process)

Therefore:

NumberSTRextBlocksRequired = CAINcallRate × HoldingTime × ExtBlocksRequiredPerCall
= 6 × 31 × 1
= 186 extension blocks

Therefore, you should provision 186 extension blocks in the NUM_STR_EXT_BLOCKS parameter.

Note: The value for extension blocks may be increased to provide a buffer against peak or abnormal traffic loads.

NUM_SEND_NOTIFICATION_EXT_BLOCKS

An extension block is a storage mechanism used to store feature data required for a single call. The send notification extension block is used to store data associated with requests for notification upon termination.

Engineering send notification extension blocks

Use the following formulas to estimate the number of send notification extension blocks you will need to provision for parameter NUM_SEND_NOTIFICATION_EXT_BLOCKS.

NumberSNExtBlocksRequired = CAINcallRate × HoldingTime × SNExtBlocksAllowedPerCall

where:

CAINcallRate = (BHCA ÷ 3600) × Call%RequiringSCPserviceLogic

HoldingTime = (Q-R% × AvgQ-RTime) + (Conv% × AvgConvTime) + AvgCallDuration

where:

AvgQ-RTime = T1timerValue + AvgCallAnswerTime

AvgConvTime = T1timerValue + (STRmsgsInConvPkg × (AvgRsrcTime + T1timerValue)) + AvgCallAnswerTime

SNExtBlocksAllowedPerCall = 3 (only three send notification extension blocks allowed per call)

CAIN call rate calculation

The CAIN call rate is the number of calls per second that require SCP service logic. To estimate the call rate, use the following formula:

CAINcallRate = (BHCA ÷ 3600) × Call%RequiringSCPserviceLogic

Variable definitions

BHCA – Your estimate of the call volume on the switch, expressed in busy-hour call attempts

Call%RequiringSCPserviceLogic – Your estimate of the percentage of BHCAs that require SCP service logic

3600 represents the number of seconds in an hour (60 seconds/minute × 60 minutes/hour)

Holding time calculation

The holding time refers to the amount of time in seconds that a send notification extension block is allocated on a call. The holding time varies depending upon the call scenario. As there are many possible scenario combinations, you will estimate an average time based on the following basic scenarios:

- Query-Response – The switch queries the SCP and receives a response. The switch may send a subsequent **`Close`** message.
- Conversation – The switch queries the SCP and receives a **`Send_To_Resource`** message in a conversation package. The switch performs the requested operation and sends a **`Resource_Clear`** message to the SCP upon completion. This may be repeated several times until

the switch receives a message in a response package.

- Call Duration – The extension block is held until the end of the call.

HoldingTime = (Q-R% × AvgQ-RTime) + (Conv% × AvgConvTime) + AvgCallDuration

where:

AvgQ-RTime = T1timerValue + AvgCallAnswerTime

AvgConvTime = T1timerValue + (STRmsgsInConvPkg × (AvgRsrcTime + T1timerValue)) + AvgCallAnswerTime

Variable definitions

Q-R% – Your estimate of the percentage of the CAIN call rate that will be query-response scenarios

Conv% – Your estimate of the percentage of the CAIN call rate that will be conversation scenarios

T1timerValue – Use the value datafilled in the CAIN_T1_TIMEOUT (table CAINPARM)

AvgCallAnswerTime – Your estimate of the average time it takes, in seconds, to establish a call through the network and ring time prior to answer

STRmsgsInConvPkg –Your estimate of the number of **`Send_To_Resource`** packages that will be received in a conversation package.

AvgRsrcTime – Your estimate of the time, in seconds, required for user interaction (including the time connected to a resource (interruptible or uninterruptible), time required to collect the subscriber's dialed digits, time required due to reset dialing.)

AvgCallDuration – Your estimate of the average call duration, in seconds, from the time the call is answered to the time the call ends.

Example

For this example, you have gathered the following data:

- BHCA = 400,000 calls per hour
- You expect 5% of the BHCA to require CAIN services.
- You expect the call mix to be
  - — 80% will be Query-Response scenarios
  - — 20% will be Conversation scenarios
- You have datafilled CAIN_T1_TIMEOUT as 2 seconds
- You expect the average call answer time to be 24 seconds

- You expect to receive one **`Send_To_Resource`** message in a conversation package
- You expect the resource time to be 30 seconds for one user interaction
- You expect the call setup time to be 1 second
- You expect the average call duration to be 180 seconds
- You know that you will allow 3 send notification extension blocks per call

Therefore,

CAINcallRate = (BHCA ÷ 3600) × Call%RequiringSCPserviceLogic
= (400,000÷ 3600) × .05
= 6 calls/second

AvgQ-RTime = T1timerValue + AvgCallAnswerTime
= 2 + 24
= 26 seconds

AvgConvTime = T1timerValue + (STRmsgsInConvPkg × (AvgRsrcTime + T1timerValue)) + AvgCallAnswerTime
= 2 + (1 × (30 + 2)) + 24
= 58 seconds

Therefore:

HoldingTime = (Q-R% × AvgQ-RTime) + (Conv% × AvgConvTime) + AvgCallDuration
= (.80)(26) + (.20)(58) + 180
= 212.40 =212 seconds

SNExtBlocksAllowedPerCall = 3 (three extension blocks are allowed per call)

Therefore:

NumberSNextBlocksRequired = CAINcallRate × HoldingTime × SNExtBlocksAllowedPerCall
= 6 × 212 × 3
= 3816 extension blocks

Therefore, you should provision 3816 extension blocks in the NUM_SEND_NOTIFICATION_EXT_BLOCKS parameter.

Note: The value for extension blocks may be increased to provide a buffer against peak or abnormal traffic loads.

NUM_FURNISHAMA_EXT_BLOCKS

An extension block is a storage mechanism used to store feature data required for a single call. The Furnish_AMA extension block indicates the number of CAIN Furnish_AMA extension blocks to be allocated for the UCS DMS-250 switch.

Engineering Furnish_AMA extension blocks

Use the following formulas to estimate the number of Furnish_AMA extension blocks you will need to provision for parameter NUM_FURNISHAMA_EXT_BLOCKS.

NUM_FurnishAMA_EXT_BLOCKS = MIN(CAINcallRate × HoldingTime × FurnishAMAExtBlocksAllowedPerCall,NO_OF_DMS250_REC_UNITS

where:

MIN(x,y) is a function that returns the lesser of values x and y

CAINcallRate = (BHCA ÷ 3600) × Call%RequiringSCPserviceLogic

HoldingTime = (Q-R% × AvgQ-RTime) + (Conv% × AvgConvTime) + AvgCallDuration

where:

AvgQ-RTime = T1timerValue + AvgCallAnswerTime

AvgConvTime = T1timerValue + (STRmsgsInConvPkg × (AvgRsrcTime + T1timerValue)) + AvgCallAnswerTime

FurnishAMAExtBlocksAllowedPerCall = 1 (only one Furnish_AMA extension block allowed per call)

Variable definitions

NO_OF_DMS250_REC_UNITS – Value of the NO_OF_DMS250_REC_UNITS parameter in table OFCENG

CAIN call rate calculation

The CAIN call rate is the number of calls per second that require SCP service logic. To estimate the call rate, use the following formula:

CAINcallRate = (BHCA ÷ 3600) × Call%RequiringSCPserviceLogic

Variable definitions

BHCA – Your estimate of the call volume on the switch, expressed in busy-hour call attempts

Call%RequiringSCPserviceLogic – Your estimate of the percentage of BHCAs that require SCP service logic

3600 represents the number of seconds in an hour (60 seconds/minute × 60 minutes/hour)

Holding time calculation

The holding time refers to the amount of time in seconds that a Furnish_AMA extension block is allocated on a call. The holding time varies depending upon the call scenario. As there are many possible scenario combinations, you will estimate an average time based on the following basic scenarios:

- Query-Response – The switch queries the SCP and receives a response. The switch may send a subsequent **`Close`** message.
- Conversation – The switch queries the SCP and receives a **`Send_To_Resource`** message in a conversation package. The switch performs the requested operation and sends a **`Resource_Clear`** message to the SCP upon completion. This may be repeated several times until the switch receives a message in a response package.
- Call Duration – The extension block is held until the end of the call.

HoldingTime = (Q-R% × AvgQ-RTime) + (Conv% × AvgConvTime) + AvgCallDuration

where:

AvgQ-RTime = T1timerValue + AvgCallAnswerTime

AvgConvTime = T1timerValue + (STRmsgsInConvPkg × (AvgRsrcTime + T1timerValue)) + AvgCallAnswerTime

Variable definitions

Q-R% – Your estimate of the percentage of the CAIN call rate that will be query-response scenarios

Conv% – Your estimate of the percentage of the CAIN call rate that will be conversation scenarios

T1timerValue – Use the value datafilled in the CAIN_T1_TIMEOUT (table CAINPARM)

AvgCallAnswerTime – Your estimate of the average time it takes, in seconds, to establish a call through the network and ring time prior to answer

STRmsgsInConvPkg –Your estimate of the number of **`Send_To_Resource`** packages that will be received in a conversation package.

AvgRsrcTime – Your estimate of the time, in seconds, required for user interaction (including the time connected to a resource (interruptible or uninterruptible), time required to collect the subscriber's dialed digits, time required due to reset dialing.)

AvgCallDuration – Your estimate of the average call duration, in seconds, from the time the call is answered to the time the call ends.

Example

For this example, you have gathered the following data:

- BHCA = 400,000 calls per hour
- You expect 5% of the BHCA to require CAIN services.
- You expect the call mix to be
  - — 80% will be Query-Response scenarios
  - — 20% will be Conversation scenarios
- You have datafilled CAIN_T1_TIMEOUT as 2 seconds
- You expect the average call answer time to be 24 seconds
- You expect to receive one **`Send_To_Resource`** message in a conversation package
- You expect the resource time to be 30 seconds for one user interaction
- You expect the call setup time to be 1 second
- You expext the average call duration to be 180 seconds
- You have obtain a value of 300 for the NO_OF_DMS250_REC_UNITS parameter in table OFCENG
- You know that you will allow 1 Furnish_AMA extension block per call

Therefore,

CAINcallRate = (BHCA ÷ 3600) × Call%RequiringSCPserviceLogic
= (400,000÷ 3600) × .05
= 6 calls/second

AvgQ-RTime = T1timerValue + AvgCallAnswerTime
= 2 + 24
= 26 seconds

AvgConvTime = T1timerValue + (STRmsgsInConvPkg × (AvgRsrcTime + T1timerValue))
+ AvgCallAnswerTime
= 2 + (1 × (30 + 2)) + 24
= 58 seconds

AvgCallDuration = T1timerValue +
=
=

Therefore:

HoldingTime = (Q-R% × AvgQ-RTime) + (Conv% × AvgConvTime) + AvgCallDuration
= (.80)(26) + (.20)(58) + 180
= 212.40 = 212 seconds

FurnishAMAExtBlocksAllowedPerCall = 1 (one extension block allowed per call)

Therefore:

NUM_FURNISHAMA_EXT_BLOCKS = CAINcallRate × HoldingTime × FurnishAMAExtBlocksAllowedPerCall
= 6 × 212 × 1
= 1272 extension blocks

OR

NUM_FURNISHAMA_EXT_BLOCKS = **NO_OF_DMS250_REC_UNITS** = 300 (default)

You should provision the lesser of the two values. Therefore, in this example, you should provision 300 extension blocks for the NUM_FURNISHAMA_EXT_BLOCKS parameter.

Note: The value for extension blocks may be increased to provide a buffer against peak or abnormal traffic loads.

NUMCPWAKE

NUMCPWAKE is a non-CAIN-specific office parameter in table OFCENG. This office parameter is used to determine the number of call processing timers allocated for use by the switch.

Engineering CAIN No Answer and Timeout timers

The NetworkBuilder O_No_Answer timer is one of the switch features that uses timers from this pool. The O_No_Answer timer is started at the ***O_Term_Seized*** event and deallocated at the ***O_Answer*** event or when the call ends.

Because the O_No_Answer timer causes more of these timer resources to be consumed, this office parameter needs to be increased when provisioning O_No_Answer services on the switch.

The NetworkBuilder Timeout timer is another switch feature that uses timers from this pool. If the *Timeout* event is requested, the Timeout timer is started at the *O_Answer* event and deallocated at the *O_Disconnect* or *Timeout* event. The O_No_Answer timer and Timeout timer are never active simultaneously for the call.

Use the following formulas to determine the maximum and minimum values for the timer values you will need to provision for parameter NUMCPWAKE.

Note: These formulas only take into account the CAIN-specific usage of call processing timers.

MaxTimerValue = MaxONoAnswerCPWAKE + MaxTimeoutCPWAKE

where:

MaxONoAnswerCPWAKE = (BHCA ÷ 3600) x (Calls%UsingONoAnswerTimer x AvgONoAnswerTime)
MaxTimeoutCPWAKE = (BHCA ÷ 3600) x (Calls%UsingTimeoutTimer x AvgTimeoutTime x 60)

MinTimerValue = MinONoAnswerCPWAKE + MinTimeoutCPWAKE

where:

MinONoAnswerCPWAKE = (BHCA ÷ 3600) x (Calls%UsingONoAnswerTimer x AvgAnswerTime)
MinTimeoutCPWAKE = (BHCA ÷ 3600) x (Calls%UsingTimeoutTimer x AvgActiveTime x 60)

Variable definitions

BHCA – Your estimate of the call volume on the switch, expressed in busy-hour call attempts

3600 represents the number of seconds in an hour (60 seconds/minute × 60 minutes/hour)

Calls%UsingONoAnswerTimer -Your estimate of the percentage of calls using O_No_Answer timer

Calls%UsingTimeoutTimer -Your estimate of the percentage of calls using Timeout timer

Calls%Answered -Your estimate of the percentage of calls answered before the O_No_Answer timer expires

Calls%NoAnswer -Your estimate of the percentage of calls not answered before the O_No_Answer timer expires

Calls%Disc -Your estimate of the percentage of calls using the Timeout timer which are disconnected while the timer is running

Calls%TimeoutExp -Your estimate of the percentage of calls using the Timeout timer which are still active when the timer expires

AvgAnswerTime -Your estimate of the average time, in seconds, until answer is received

AvgONoAnswerTime -Your estimate of the average time, in seconds, for the O_No_Answer timer

AvgTimeoutTime -Your estimate of the average time, in minutes, for the Timeout timer

AvgActiveTime -Your estimate of the average time, in minutes, for calls utilizing the Timeout timer to be in the active (talking) state

Example

For this example, you have gathered the following data:

- BHCA = 400,000 calls per hour
- You expect 10% of the BHCA to use the O_No_Answer timer
- You expect 2% of the BHCA to use the Timeout timer
- You expect the average call answer time to be 4 seconds
- You expect the average time for the O_No_Answer timer to be 15 seconds
- You expect the average Timeout time to be 15 minutes
- You expect the average Active time to be 10 minutes
- You expect that 80% of calls using the O_No_Answer timer are answered before the O_No_Answer timer expires
- You expect that 20% of calls using the O_No_Answer timer are not answered before the O_No_Answer timer expires
- You expect 70% of calls using the Timeout timer are disconnected before the timer expires
- You expect 30% of calls using the Timeout timer are still active when the timer expires

Therefore,

MaxONoAnswerCPWAKE = (BHCA ÷ 3600) x (Calls%UsingONoAnswerTimer x AvgONoAnswerTime)
= (400,000 ÷ 3600) x (.10 x 15)
= 167

MaxTimeoutCPWAKE = (BHCA ÷ 3600) x (Calls%UsingTimeoutTimer x AvgTimeoutTime x 60)
= (400,000 ÷ 3600) x (.02 x 15 x 60)
= 2000

Therefore:

MaxTimerValue = MaxONoAnswerCPWAKE + MaxTimeoutCPWAKE
= (167 + 2000)
= 2167

MinONoAnswerCPWAKE = (BHCA ÷ 3600) x (Calls%UsingONoAnswerTimer x AvgAnswerTime)
= (400,000 ÷ 3600) x (.10 x 4)
= 44

MinTimeoutCPWAKE = (BHCA ÷ 3600) x (Calls%UsingTimeoutTimer x AvgActiveTime x 60)
= (400,000 ÷ 3600) x (.02 x 10 x 60)
= 1333

Therefore:

MinTimerValue = MinONoAnswerCPWAKE + MinTimeoutCPWAKE
= (44 + 1333)
= 1377

It is recommended that you increase the NUMCPWAKE parameter by the amount of the Max Timer Value. This will insure that during high call traffic times, there will be enough no answer timers available.

If it is necessary, however, to engineer the NUMCPWAKE parameter more closely to the real value of timers used, then apply the following formula:

CloselyEngineeredValue = (MaxONoAnswerCPWAKE x Calls%NoAnswer) +
(MinONoAnswerCPWAKE x Calls%Answered) +
(MaxTimeoutCPWAKE x Calls%TimeoutExp) +
(MinTimeoutCPWAKE x Calls%Disc)
= (167 x .20) + (44 x .80) + (2000 x .30) + (1333 x .70)
= 1601

Therefore, to use a more closely engineered value for the NUMCPWAKE parameter, in this example you would increase the value by 1601.

VAMPTRID resources

Table VAMPTRID (Variable AIN Messaging Platform Transaction Identifiers) provides the following resources used in CAIN messaging.

- transaction identifier – used to establish and maintain a TCAP communication session between two applications
- component identifier – used to identify and correlate individual operations/requests between applications within the context of a transaction
- message buffers – used as internal workspace for encoding and decoding messages
- ACG controls – used to establish the number of ACG blocks (controls) that are allocated for an application (shared by the SCP and SOCC controls lists)

NetworkBuilder requires one transaction identifier, one component identifier (or more when using non-call related components), and two message buffers for each query generated by a CAIN call.

Engineering transaction identifier blocks

Use the following formulas to estimate the number of transaction identifier blocks you will need to provision.

NumberTransactionIdentifierBlocksRequired = QueryRate × HoldingTime × TransactionIdentifierBlocksRequiredPerQuery

where:

QueryRate = (BHCA ÷ 3600) × Call%RequiringSCPserviceLogic

HoldingTime = (Q-R% × AvgQ-RTime)+(STRConn% × AvgSTRConnTime)+(MultReq% × AvgMultReqTime)+(IP% × AvgIPTime)+(MP% × AvgMultiPartyTime)+(Timeout% × AvgTimeoutTimerTime) + (EDP-N% × AvgEDP-NTime)

where:

AvgQ-RTime = T1TimerValue

AvgSTRConnTime = T1TimerValue + (STRmsgsInConvPkg × (AvgRsrcTime + T1TimerValue)) + AvgCallAnswerTime

AvgMultReqTime = (AvgNumberQueriesPerCall × T1TimerValue) +AvgCallSetupTime

AvgIPTime = T1TimerValue+STRmsgsInConvPkg × (AvgIPsetupTime +AvgRsrcTime + T1Timervalue) + AvgCallAnswerTime

AvgMultiPartyTime = (MultiPartyConnTime + AvgMultiPartySetupTime) x AvgNumber3rdPartyConn + T1TimerValue

AvgTimeoutTimerTime = T1TimerValue+AvgCallAnswerTime+AvgNumberTimeoutEventsReq × (AvgTimeouttimerValue × 60 +T1TimerValue)

TransactionIdentifierBlocksRequiredPerQuery = 1 (only one transaction identifier block is required per query)

Query rate calculation

The query rate is the number of queries per second that require SCP service logic. To estimate the query rate, use the following formula:

QueryRate = (BHCA ÷ 3600) × Call%RequiringSCPserviceLogic

Variable definitions

BHCA – Your estimate of the call volume on the switch, expressed in busy-hour call attempts

Call%RequiringSCPserviceLogic – Your estimate of the percentage of BHCAs that require SCP service logic

3600 represents the number of seconds in an hour (60 seconds/minute × 60 minutes/hour)

Holding time calculation

The holding time refers to the amount of time in seconds that a transaction identifier block is allocated on a call. The holding time varies depending upon the call scenario. As there is an unlimited number of possible scenario combinations, you will estimate an average time based on the following

basic scenarios:

- Query-Response – The switch queries the SCP and receives a response. The switch may send a subsequent **`Close`** message.
- EDP-Notification – The switch queries the SCP and receives a response which arms EDPs. The switch may send zero or more subsequent EDP-Notification messages, but no EDP-Requests. When the switch determines that no more active EDPs are reachable, it sends a **`Close`** message.
- STR or CTR connection In-switch – The switch queries the SCP and receives a **`Send_To_Resource`** or **`Connect_To_Resource`** message in a conversation package. The switch performs the requested operation and sends a **`Resource_Clear`** or **`CTR_Clear`** message to the SCP upon completion. This may be repeated several times until the switch receives a message in a response package.
- Multiple requests – The switch queries the SCP and receives an **`Analyze_Route`**, **`Continue`**, or **`Collect_Information`** message. Later in the call model the switch may send another TDP-Request or EDP-Request (at *Network_Busy*, *O_Called_Party_Busy*, *O_No_Answer*, or *Timeout*) and receive another response.
- STR- or CTR-Connection to an IP– The switch queries the SCP and receives a **`Send_To_Resource`** or **`Connect_To_Resource`** message (containing a ***`DestinationAddress`***) in a conversation package. A connection is established to an IP. Once the IP performs the requested function, the switch sends a **`Resource_Clear`** or **`CTR_Clear`** message to the SCP and the SCP responds with a message in a response package.
- Multi Party–Multi Party call handling becomes active when the *switchHookFlash* event is armed at the ***O_Mid_Call*** EDP. This scenario requires a request message to be sent at a previous trigger in order to arm EDPs. A Multi Party call remains active until we either receive a Disconnect message from the SCP, only one agent remains on the call, or we return to call configuration 2 (refer to Volume 3, Chapter 4, "Call Configurations," chapter for Multi call party handling) and the *switchHookFlash* event has not been rearmed .
- Timeout Requests – The timeout timer is activated when the call is answered if the timeout event is armed at the ***O_Mid_Call*** EDP. When the timeout timer expires, a **`Timeout`** message is sent to the SCP. The SCP's only valid response is either a **`Disconnect`** or **`Connect_To_Resource`** message. If it is a **`Connect_To_Resource`** message, refer to the descriptions above for **`Connect_To_Resource`** interactions, otherwise the call will come down and the transaction identifier block will be deallocated.

Therefore, the holding time can be estimated by taking an average of the

scenarios, using the following formulas:

HoldingTime = (Q-R% × AvgQ-RTime) + (STRConn% × AvgSTRConnTime)
+ (MultReq% × AvgMultReqTime) + (IP% × AvgIPtime) + (MP% × AvgMultiPartyTime) + (Timeout% × AvgTimeoutTimerTime)
+ (EDP-N% × AvgEDP-NTime)

where:

AvgQ-RTime = T1timerValue

AvgEDP-NTime = T1timerValue + AvgCloseTime

AvgSTRConnTime = T1timerValue +(STRmsgsInConvPkg x (AvgRsrcTime
+ T1timerValue)) + AvgCallAnswerTime

AvgMultReqTime = (AvgNumberQueriesPerCall× T1timerValue) + AvgCallSetupTime

AvgMultiPartyTime = (AvgMultiPartyConnTime + AvgMultiPartySetupTime) x AvgNumber3rdPartyConn +T1timerValue

AvgIPTime = T1timerValue+STRmsgsInConvPkg × (AvgIPsetupTime+AvgRsrcTime + T1timerValue)
+ AvgCallAnswerTime

AvgTimeoutTimerTime= T1TimerValue+AvgNumberTimeoutEventsReq x (AvgTimeouttimerValue +T1timerValue)

Variable definitions

Q-R% – Your estimate of the percentage of the CAIN call rate that will be query-response scenarios

EDP-N% – Your estimate of the percentage of the CAIN call rate that will be EDP-Notification scenarios

MP% – Your estimate of the percentage of CAIN call rate that will be multi–party scenarios

STRConn% – Your estimate of the percentage of the CAIN call rate that will require in-switch STRs

MultReq% – Your estimate of the percentage of the CAIN call rate that will be multiple request scenarios

IP% – Your estimate of the percentage of CAIN call rate that will require STR- or CTR-Connections tp an IP

Timeout% – Your estimate of the percentage of CAIN call rate that will require arming the Timeout EDP.

T1timerValue – Use the value datafilled in the CAIN_T1_TIMEOUT (table CAINPARM)

STRmsgsInConvPkg –Your estimate of the number of **`Send_To_Resource`** or **`Connect_To_Resource`** packages that will be received in a conversation package.

AvgCloseTime –Your estimate of the time, in seconds, required for the call to reach the EDP at which the **`Close`** message is sent

AvgRsrcTime – Your estimate of the time, in seconds, required for user interaction (including the time connected to a resource (interruptible or uninterruptible), time required to collect the subscriber's dialed digits, time required due to reset dialing.)

AvgCallSetupTime – Your estimate of the time, in seconds, required to set up the call and reach the second DP. (When figuring the time for ***Network_Busy*** or ***O_Called_Party_Busy***, remember to include the time required for the busy condition to be recognized by the switch. For ***O_No_Answer***, remember to include the O_No_Answer timer value. For *Timeout*, remember to include the Timeout timer value.)

Variable definitions

AvgNumberQueriesPerCall –Your estimate of the average number of CAIN queries per call

AvgMultiPartySetupTime – Your estimate of the time, in seconds, it will take to originate a call to the 3rd party and merge that party into the existing call

AvgCallAnswerTime –Your estimate of the time, in seconds, it takes to establish a call through the network and ring time prior to answer

AvgIPTime –Your estimate of the time, in seconds, for the call to be answered by the IP

AvgMultiPartyConnTime –Time it takes, in seconds, from the point when the *switchHookFlash* event was armed until the conversation has ended.
AvgNo3rdPartyConn –Number of 3rd party agents conferenced into the call.

AvgNoTimeoutEventsReq –The number of Timeout events requested by the SCP (ie. how many times the Timeout event is armed).

AvgTimeoutTimerValue –The number in minutes for the average Timeout length

Example

For this example, you have gathered the following data:

- BHCA = 400,000 calls per hour
- You expect 5% of the BHCA to require CAIN services.
- You expect the call mix to be
  - 50% will be Query-Response
  - 5% will be EDP-Notification scenarios
  - 20% will be in-switch STR scenarios
  - 5% will be Multiple requests with **`Network_Busy`** as the second request
  - 5% will require STR- or CTR-Connections to an IP
  - 10% will require multi-party call handling
  - 10% will arm the Timeout EDP
- You have datafilled CAIN_T1_TIMEOUT as 2 seconds
- You expect to receive one **`Send_To_Resource`** or **`Connect_To_Resource`** message in a conversation package
- You expect the resource time to be 30 seconds for one user interaction
- You expect the call setup time to be 1 second
- You expect the IP setup time to be 5 seconds
- You expect the multi-party connection time to be 60 seconds
- You expect the multi-party setup time to be 5 seconds

- You expect the number of 3rd party connections to be one
- You expect the number of timeout events requested to be one
- You expect the average timeout timer to be 15 minutes
- You know that you require one transaction identifier block per query
- You expect the average **`Close`** time to be 15 seconds

Therefore,

QueryRate = (BHCA ÷ 3600) × Call%RequiringSCPserviceLogic
= (400,000÷ 3600) × .05
= 6 queries/second

AvgQ-RTime = T1timerValue
= 2
= 2 seconds

AvgEDP-NTime = (T1timerValue) + AvgCloseTime
= 2 + 15
= 17 seconds

AvgSTRConnTime = T1timerValue + (STRmsgsInConvPkg × (AvgRsrcTime + T1timerValue))
= 2 + (1 × (30 + 2)) + 24
= 58 seconds

AvgMultReqTime = (AvgNumberQueriesPerCall× T1timerValue) + AvgCallSetupTime
= (2 × 2) + 1
= 5 seconds

AvgMultiPartyTime = (MultiPartyConnTime + AvgMultiPartySetupTime) x AvgNumber3rdPartyConn +T1timerValue
= (60 +5) x 1 +2
= 67 seconds

AvgIPTime = T1timervalue + STRmsgsInConvPkg × (AvgIPsetupTime + AvgRsrcTime + T1timervalue)
+ AvgCallAnswerTime
= 2 +(1 × (5 +30 + 2) + 24 = 63 seconds

AvgTimeoutTimerTime = T1timerValue + AvgCallAnswerTime +AvgNumberTimeoutEventsReq × (AvgTimeouttimerValue +T1timerValue)
= 2 + 24 + (1 × ((15 × 60) +2))
= 928 seconds

therefore:

HoldingTime = (Q–R% x AvgQ–RTime) +(STRConn% x AvgSTRConnTime) +(EDP-N% x AvgEDP-NTime) + (MultReq% x AvgMultReqTime) +(IP% x AvgIPTime) +(MP% x AvgMultiPartyTime) +(Timeout% x AvgTimeoutTimerTime)
= (.60 × 2) +(.05 × 34) +(.05 × 17)+(.05 × 5) + (.05 x 63) + (.10 × 67) + (.10 × 928)
= (1.20) +(1.70) +(.85) +(.25) +(3.15) +(6.70) +(92.80)
= 106.65 seconds = 107 seconds

TransactionIdentifierBlocksRequiredPerQuery = 1 (only one transaction identifier block is required per query)

therefore:

NumberTransactionIdentifierBlocksRequired = QueryRate x HoldingTime x TransactionIdentifierBlocksRequired PerQuery
= 6 x 107 x 1=642 blocks

If the query rate is 6 queries per second, the holding time is 107 seconds, and the number of transaction identifier blocks required for each query is 1, then the number of blocks required is 642.

Note: This number (6) is correct provided each call queries only once.

Note: The value for transaction identifier blocks may be increased to provide a buffer against peak or abnormal traffic loads.

Note: Transaction identifier blocks are allocated in groups of 128. Therefore, in this example you would provision 768 transaction identifier blocks.

Engineering component identifier blocks

Use the following formulas to estimate the number of component identifier blocks you will need to provision.

NumberComponentIdentifierBlocksRequired = QueryRate × HoldingTime × AvgComponentIdentifierBlocksRequiredPerQuery

where:

QueryRate = (BHCA ÷ 3600) × Call%RequiringSCPserviceLogic

HoldingTime = (Q-R% × AvgQ-RTime) + (STRConn% × AvgSTRConnTime) + (MultReq% × AvgMultReqTime) + (IP% × AvgIPtime)

where:

AvgQ-RTime =T1TimerValue

AvgSTRConnTime = T1TimerValue + (STRmsgsInConvPkg × (AvgRsrcTime + T1TimerValue))

AvgMultReqTime = (AvgNumberQueriesPerCall × T1TimerValue) + AvgCallSetupTime

AvgIPTime = T1TimerValue + STRmsgsInConvPkg × (AvgIPSetupTime + AvgRsrcTime + T1TimerValue) + AvgCallAnswerTime

AvgComponentIdentifierBlocksRequiredPerQuery = 1 + EDP% + ACG% + SENDNOT% + FAMA%

Query rate calculation

The query rate is the number of queries per second that require SCP service logic. To estimate the query rate, use the following formula:

QueryRate = (BHCA ÷ 3600) × Call%RequiringSCPserviceLogic

Variable definitions

BHCA – Your estimate of the call volume on the switch, expressed in busy-hour call attempts

Call%RequiringSCPserviceLogic – Your estimate of the percentage of BHCAs that require SCP service logic

3600 represents the number of seconds in an hour (60 seconds/minute × 60 minutes/hour)

Holding time calculation

The holding time refers to the amount of time in seconds that a component identifier block is allocated on a call. The holding time varies depending upon the call scenario. As there is an unlimited number of possible scenario combinations, you will estimate an average time based on the four following basic scenarios:

- Query-Response – The switch queries the SCP and receives a response. The switch may send subsequent EDP-Notifications, **`Close`** message, or both.
- STR or CTR connection in-switch – The switch queries the SCP and receives a **`Send_To_Resource`** or **`Connect_To_Resource`** message in a conversation package. The switch performs the requested operation and sends a **`Resource_Clear`** or **`CTR_Clear`** message to the SCP upon completion. This may be repeated several times until the switch receives a message in a response package.
- Multiple requests – The switch queries the SCP and receives a response message. Later in the call model the switch may send another TDP-Request or EDP-Request (at *Network_Busy*, *O_Called_Party_Busy*, *O_No_Answer* , or *Timeout*) and receive another response.
- STR- or CTR-Connection to an IP– The switch queries the SCP and receives a **`Send_To_Resource`** or **`Connect_To_Resource`** message (containing a ***`DestinationAddress`***) in a conversation package. A connection is established to an IP. Once the IP performs the requested function, the switch sends a **`Resource_Clear`** or **`CTR_Clear`** message to the SCP and the SCP responds with a message in a response package.

Therefore, the holding time can be estimated by taking an average of the scenarios, using the following formulas:

HoldingTime = (Q-R% × AvgQ-RTime) + (STRConn% × AvgSTRConnTime)
+ (MultReq% × AvgMultReqTime) + (IP% × AvgIPTime)

where:

AvgQ-RTime = T1timerValue

AvgSTRConnTime = T1TimerValue + (STRmsgsInConvPkg × (AvgRsrcTime + T1timerValue))

AvgMultReqTime = (AvgNumberQueriesPerCall × T1timerValue) +AvgCallSetupTime

AvgIPTime = T1TimerValue + STRmsgsInConvPkg × (AvgIPSetupTime +AvgRsrcTime + T1TimerValue) + AvgCallAnswerTime

Variable definitions

Q-R% – Your estimate of the percentage of the CAIN call rate that will be query-response scenarios

STRConn% – Your estimate of the percentage of the CAIN call rate that will require in-switch STRs

MultReq% – Your estimate of the percentage of the CAIN call rate that will be multiple request scenarios

IP% – Your estimate of the percentage of CAIN call rate that will require STR- or CTR-Connections to an IP

T1TimerValue – Use the value datafilled in the CAIN_T1_TIMEOUT (table CAINPARM)

STRmsgsInConvPkg –Your estimate of the number of **`Send_To_Resource`** or **`Connect_To_Resource`** packages that will be received in a conversation package.

AvgRsrcTime – Your estimate of the time, in seconds, required for user interaction (including the time connected to a resource (interruptible or uninterruptible), time required to collect the subscriber's dialed digits, time required due to reset dialing.)

AvgNumberQueriesPerCall –Your estimate of the average number of CAIN queries per call

AvgIPsetupTime –Your estimate of the time, in seconds, for the call to be answered by the IP (including the time required to establish the call through the network and ring time prior to answer)

AvgCallAnswerTime –Your estimate of the time, in seconds, it takes to establish a call through the network and ring time prior to answer

AvgIPTime –Your estimate of the time, in seconds, for the call to be answered by the IP

Average component identifier blocks calculation

Every incoming **Request_Report_BCM_Event** (for EDPs), ACG, **Send_Notification**, or **Furnish_AMA_Information** component requires an additional component identifier block.

For EDP scenarios, the switch queries the SCP and receives a call-related component along with a **Request_Report_BCM_Event** component in a conversation package. If the switch reaches an EDP that is armed to send an EDP-Request, the switch sends the request and receives a response from the SCP. If no EDP-Request has been sent by the time the call is answered, the switch sends a **Close** message to the SCP; no response is expected.

For ACG scenarios, the switch queries the SCP and receives a response that contains an **ACG** component. The switch can also receive a unidirectional, non-call related message from the SCP containing an **ACG** component. The switch can also receive a non-call related **ACG_Global_Ctrl_Restore** request message from the SCP.

For **Send_Notification** scenarios, the switch queries the SCP and receives a response that contains a **Send_Notification** component. After the call is completed, the switch sends a **Termination_Notification** message.

For **Furnish_AMA_Information** scenarios, the SCP sends a Furnish_AMA_Information component to the switch in the same TCAP package with an **Analyze_Route** or **Send_To_Resource** message. The switch examines the structure of the component. After examining the component, the switch captures the contents of the billing information in the CDR.

It is possible that during the course of a single call, the switch may receive multiple Furnish_AMA_Information components. If this happens, the switch considers the most recent component to the correct one and overwrites any information from previous components.

To estimate the average component identifier blocks per query, use the following information:

AvgComponentIdentifierBlocksRequiredPerQuery = 1 + EDP% + ACG% +SENDNOT% + FAMA%

Variable definitions

EDP% – Your estimate of the percentage of the CAIN call rate that will be EDP scenarios

ACG% – Your estimate of the percentage of the CAIN call rate that will include **`ACG`** components

SENDNOT% – Your estimate of the percentage of the CAIN call rate that will include **`Send_Notification`** components

FAMA% – Your estimate of the percentage of the CAIN call rate that will include **`Furnish_AMA_Information`** components

Example

For this example, you have gathered the following data:

- BHCA = 400,000 calls per hour
- You expect 5% of the BHCA to require CAIN services.
- You expect the call mix to be
  - — 70% will be Query-Response
  - — 20% will be Conversation scenarios
  - — 5% will be Multiple requests with **`Network_Busy`** as the second request
  - — 5% will require STR- or CTR-Connections
  - — 1% will be EDP scenarios
  - — 3% will be ACG scenarios
  - — 2% will be **`Send_Notification`** scenarios
  - — 2% will be **`Furnish_AMA_Information`** scenarios

  Note: EDP, ACG, **`Send_Notification`**, and **`Furnish_AMA_Information`** scenarios may overlap with other scenarios and be counted in more than one category. Therefore, total percentages may not equal 100%.

- You have datafilled CAIN_T1_TIMEOUT as 2 seconds
- You expect to receive one **`Send_To_Resource`** or **`Connect_To_Resource`** message in a conversation package
- You expect the resource time to be 30 seconds for one user interaction
- You expect the call setup time to be 1 second
- You expect the IP setup time to be 5 seconds

Therefore,

QueryRate = (BHCA ÷ 3600) × Call%RequiringSCPserviceLogic
= (400,000÷ 3600) × .05
= 6 queries/second

AvgQ-RTime = T1timerValue
= 2
= 2 seconds

AvgSTRConnTime = T1timerValue + (STRmsgsInConvPkg × (AvgRsrcTime + T1timerValue))
= 2 + (1 × (30 + 2))
= 34 seconds

AvgMultReqTime = (AvgNumberQueriesPerCall × T1timerValue) + AvgCallAnswerTime
= (2 × 2) + 1
= 5 seconds

AvgIPTime = T1timerValue + STRmsgsInConvPkg × (AvgIPsetupTime +AvgRsrcTime + T1timerValue) + AvgCallAnswerTime
= 2 + (1 × (5+30 + 2)) + 24
= 63 seconds

Therefore:

HoldingTime = (Q-R% × AvgQ-RTime) + (STRConn% × AvgSTRConnTime) + (MultReq% × AvgMultReqTime) + (IP% × AvgIPtime)
= (.70)(2) + (.20)(34) + (.05)(5) + (.05)(63)
= 1.4 + 6.8 + .25 + 3.15 = 11.60 seconds = 12 seconds

AvgComponentIdentifierBlocksRequiredPerQuery =1 + EDP% + ACG% + SENDNOT% + FAMA%

Therefore:

NumberComponentIdentifierBlocksRequired = QueryRate × HoldingTime × AvgComponentIdentifierBlocksRequiredPerQuery
= 6 × 12× (1 + .01 + .03 + .02 +.02)
= 77.76 component identifier blocks

If the query rate is 6 queries/second, the holding time is 12 seconds, and the number of component identifier blocks required for each query is 1.08, then the number of blocks required is approximately 78.

Note: This number (6) is correct provided each call queries only once.

Note: The value for component identifier blocks may be increased to provide a buffer against peak or abnormal traffic loads.

Note: Component identifier blocks are allocated in groups of 64. Therefore, in this example you would provision 128 component identifier blocks.

Engineering message buffers

Use the following formulas to estimate the number of message buffers you will need to provision.

NumberMessageBuffersRequired = QueryRate × HoldingTime × MessageBuffersRequiredPerCall

where:

QueryRate = (BHCA ÷ 3600) × Call%RequiringSCPserviceLogic

HoldingTime = 1 second

MessageBuffersRequiredPerCall = (Q–R% x 2) + (Conv% x (AvgNumberQ–R sets x 2)) +(MultReq% x (AvgNumber Q–R sets x 2))

Query rate calculation

The query rate is the number of queries per second that require SCP service logic. To estimate the query rate, use the following formula:

QueryRate = (BHCA ÷ 3600) × Call%RequiringSCPserviceLogic

Variable definitions

BHCA – Your estimate of the call volume on the switch, expressed in busy-hour call attempts

Call%RequiringSCPserviceLogic – Your estimate of the percentage of BHCAs that require SCP service logic

3600 represents the number of seconds in an hour (60 seconds/minute × 60 minutes/hour)

The message buffers required per call can be estimated using the following formula:

MessageBuffersRequiredPerCall = (Q–R% x 2) + (Conv% x (AvgNumberQ–R sets x 2)) +(MultReq% x (AvgNumber Q–R sets x 2))

Variable definitions

Q-R% – Your estimate of the percentage of the CAIN call rate that will be query-response scenarios

Conv% – Your estimate of the percentage of the CAIN call rate that will be conversation scenarios

MultReq% – Your estimate of the percentage of the CAIN call rate that will be multiple request scenarios

AvgNumberQ-RSets –Your estimate of the number of query/response sets within a conversation.

Example

For this example, you have gathered the following data:

- BHCA = 400,000 calls per hour
- You expect 5% of the BHCA to require CAIN services.
- You expect the call mix to be
  - — 70% will be Query-Response scenarios
  - — 20% will be Conversation scenarios
  - — 10% will be Multiple requests with ***Network_Busy*** as the second request
- You know that the average number of query/response sets per conversation is 1.6.

Therefore,

QueryRate = (BHCA ÷ 3600) × Call%RequiringSCPserviceLogic
= (400,000÷ 3600) × .05
= 6 queries/second

HoldingTime = 1 second

MessageBuffersRequiredPerCallQuery = (Q–R% x 2) + (Conv% x (AvgNumberQ–R sets x 2)) +(MultReq% x (AvgNumber Q–R sets x 2))
= (.70 x 2) +(.20 x (1.6 x 2) +(.10 x (1.6 x2))
= (1.40) +(.64) +(.32)
= 3 message buffers

Therefore:

NumberMessageBuffersRequired = QueryRate × HoldingTime × MessageBuffersRequiredPerQuery
= 6 × 1 × 3
= 18 message buffers

If the query rate is 6 queries/second, the holding time is 1 second, and the number of message buffers required for each query is 3, then the number of message buffers required is 18.

Note: This number (6) is correct provided each call queries only once.

Note: The value for message buffers may be increased to provide a buffer against peak or abnormal traffic loads.

Note: Message buffers are allocated in groups of 64. Therefore, in this example you would provision 64 message buffers.

Engineering ACG blocks

One ACG block is used for each ACG control that is initiated on the control list. The number of ACG blocks that are allocated for an application are shared by the SCP and SOCC control lists on a first come, first served basis.

The ACG resources are not necessarily shared equally between the two control lists. For example, if 256 blocks are allocated and 256 SCP controls are initiated, there are no resources left for a SOCC control to be initiated until some control is removed or expire.

Use the following formula to estimate the number of ACG blocks you will need to provision.

NumberACGBlocksRequired = MaxNumberSimultaneousSCPControlsExpected + MaxNumberSimultaneousSOCCControlsExpected

where:

MaxNumberSimultaneousSCPControlsExpected = 150

MaxNumberSimultaneousSOCCControlsExpected = 100

Therefore,

NumberACGBlocksRequired = MaxNumberSimultaneousSCPControlsExpected + MaxNumberSimultaneousSOCCControlsExpected
= (150 + 100)
= 250 ACG blocks

In this example you would provision 250 ACG blocks.

NUMPERMEXT

When the switch receives a **`Connect_To_Resource`** or a **`Merge_Call`** message, call processing determines if the switch has allocated PORTPERM extension blocks. If the switch has not allocated PORTPERM extension blocks, the switch allocates four PORTPERM extension blocks. PORTPERM extension blocks normally remain allocated until the **`Connect_To_Resource`** interaction or **`Merge_Call`** processing is complete. If the **`Merge_Call`** processing transitions the call into call configuration 10, the switch holds PORTPERM extension blocks until one party releases. If

the call transitions to any other call configuration, the switch releases the PORTPERM extension blocks immediately. Certain scenarios, such as fatal application errors, can release the extension blocks early.

Engineering PORTPERM extension blocks

Use the following formulas to estimate the number of PORTPERM extension blocks you must provision for the NUMPERMEXT parameter.

NumberPortPermExtBlocksRequired = CAINcallRate × HoldingTime × ExtBlocksRequiredPerCall

where:

CAINcallRate = (BHCA ÷ 3600) × Call%RequiringSCPserviceLogic

HoldingTime = CTRHoldingTime + MCHoldingTime

CTRHoldingTime = (CTRCalls% x (CTRHoldTime x Avg#CTRsPerCall))

MCHoldingTime = (MCCalls% x (MCHoldTime x Avg#MCsPerCall))

where:

CTRHoldTime = AvgCallAnswerTime + AvgRsrcTime + AvgCallSetupTime

MCHoldTime = AvgMCTime + T1TimerValue

ExtBlocksRequiredPerCall = 4 (extension blocks required per conversational **`Connect_To_Resource`** and **`Merge_Call`** call)

CAIN call rate calculation

The CAIN call rate is the number of calls per second that require SCP service logic. To estimate the call rate, use the following formula:

CAINcallRate = (BHCA ÷ 3600) × Call%RequiringSCPserviceLogic

Variable definitions

BHCA – Your estimate of the call volume on the switch, expressed in busy-hour call attempts

Call%RequiringSCPserviceLogic – Your estimate of the percentage of BHCAs that require SCP service logic

3600 represents the number of seconds in an hour (60 seconds/minute × 60 minutes/hour)

Holding time calculation

The holding time refers to the amount of time, in seconds, that elapses between the allocation and deallocation of a PORTPERM extension block.

The holding time varies depending upon the call scenario. Since there is an an unlimited number of possible scenario combinations, you must estimate an average time. To estimate the average time, use the following basic scenarios:

- **`Connect_To_Resource`** – The switch queries the SCP and receives a **`Connect_To_Resource`** message. If the switch received the message in a conversation package, the switch performs the requested operation. When the switch finishes the operation, the switch sends a **`CTR_Clear`** message to the SCP. The switch and SCP repeat this process until the switch receives a message in a response package.
- **`Merge_Call`** – The switch queries the SCP and receives a **`Merge_Call`** message in a conversation package. The switch performs the requested operation and merges all agents on the call. If there are three agents on the call, the switch does not release the resources associated with the **`Merge_Call`** until the switch disconnects one of the agents. If there are only two agents on the call when the switch receives the **`Merge_Call`** message, the switch releases the resources immediately. The switch and SCP can repeat this process multiple times during the call.

Therefore, estimate the holding time by taking an average of the scenarios, using the following formulas:

HoldingTime = CTRHoldingTime + MCHoldingTime

CTRHoldingTime = (CTRCalls% x (CTRHoldTime x Avg#CTRsPerCall))
MCHoldingTime = (MCCalls% x (MCHoldTime x Avg#MCsPerCall))

where:

CTRHoldTime = AvgCallAnswerTime + AvgRsrcTime + AvgCallSetupTime

MCHoldTime = AvgMCTime + T1TimerValue

Variable definitions

CTRCalls% – Your estimate of the percentage of the CAIN call rate that will be **`Connect_To_Resource`** scenarios

MCCalls% – Your estimate of the percentage of the CAIN call rate that will be **`Merge_Call`** scenarios

T1TimerValue – Use the value datafilled in the CAIN_T1_TIMEOUT (table CAINPARM)

AvgMergeCallTime -Your estimate of the average time, in seconds, required for three-party interaction

AvgCallAnswerTime – Your estimate of the average time it takes, in seconds, to establish a call through the network and ring time prior to answer

AvgRsrcTime – Your estimate of the time, in seconds, required for user interaction (including the time connected to a resource (interruptible or uninterruptible), time required to collect the subscriber's dialed digits, time required due to reset dialing, IP connection time.)

AvgCallSetupTime – Your estimate of the time, in seconds, required to set up the call and reach the second TDP. (When figuring the time for ***Network_Busy*** or ***O_Called_Party_Busy***, remember to include the time required for the switch to recognize the busy condition. For ***O_No_Answer***, remember to include the time provisioned in table CAINGRP or CAINPARM.)

Avg#CTRsPerCall – The average number of **`Connect_To_Resource`** interactions processed during the course of one call.

Avg#MCsPerCall – The average number of **`Merge_Call`** interactions processed during the course of one call.

Example

For this example, you have gathered the following data:

- BHCA = 600,000 calls per hour
- You expect 5% of the BHCA to require CAIN services.
- You expect the call mix to be
  - — 20% will be Conversation scenarios that result in a **`Connect_To_Resource`** message
  - — 5% will be Conversation scenarios that result in a **`Merge_Call`** message
  - — the average number of **`Connect_To_Resource`** interactions per call is 1.3
  - — the average number of **`Merge_Call`** interactions per call is 1.1
- You have datafilled CAIN_T1_TIMEOUT as 2 seconds
- You expect the average call answer time to be 24 seconds
- You expect the resource time to be 30 seconds for one user interaction
- You expect the three-way merge call time to be 30 seconds
- You expect the call setup time to be 1 second
- You know that you will require four extension blocks per call

Note: This scenario serves only as an example of how to calculate the required number of PORTPERM extension blocks. It shows a large number of PORTPERM extension blocks are required. Your switch may not require as many PORTPERM extension blocks.

Therefore,

NumberPortPermExtBlocksRequired = CAINcallRate × HoldingTime × ExtBlocksRequiredPerCall

where:

CAINcallRate = (BHCA ÷ 3600) × Call%RequiringSCPserviceLogic
= (600,000 ÷ 3600) × .05
= 8 calls per second

CTRHoldTime = AvgCallAnswerTime + AvgRsrcTime + AvgCallSetupTime
= 24 + 30 + 1
= 55

MCHoldTime = AvgMCTime + T1TimerValue
= 30 + 2
= 32

CTRHoldingTime = (CTRCalls% x (CTRHoldTime x Avg#CTRsPerCall))
= (20% x (55 X 1.3))
= 20% x 71.5
= 14.3

MCHoldingTime = (MCCalls% x (MCHoldTime x Avg#MCsPerCall))
= (5% x (32 x 1.1))
= 1.8

Therefore:

HoldingTime = CTRHoldingTime + MCHoldingTime
= 14.3 + 1.8
= 16.1

ExtBlocksRequiredPerCall = 4 (extension blocks required per conversational **`Connect_To_Resource`** and **`Merge_Call`** call

---

Therefore:

NumberPORTPERMBlocksRequired = CAINcallRate x HoldingTime x ExtBlocksRequiredPerCall
= 8 x 16.1 x 4
= 515 extension blocks

Therefore, provision 515 additional extension blocks in the NUMPERMEXT OFCENG office parameter. Refer to the *UCS DMS-250 Office Parameters Reference Manual* for more information.

Note: This scenario shows a large number of PORTPERM extension blocks are required. Your switch may not require as many PORTPERM extension blocks.

The NT1X81AA Six-port Conference circuit card is utilized by this functionality. Refer to *UCS DMS-250 Conference Circuit Guide* and *DMS-100 Feature Description Manual*, for more information on the

NT1X81AA Six-port Conference circuit card. Refer to *UCS DMS-250 Data Schema Reference Manual*, for information on provisioning the CONF6PR (Six-port Conference Circuit) table.

Limitations and restrictions

CAIN limitations and restrictions include, but are not limited to the following:

- Only originating DAL, FGD, SS7 Inter-IMT, SS7 Global-IMT, PRI, and AXXESS agencies are eligible for CAIN services.
- Only terminating DAL, FGB, FGC, FGD, SS7 Inter-IMT, SS7 Global-IMT, PRI, and AXXESS agencies are eligible for CAIN services.
- Office-based subscription should be used as a default feature set for an office, for use when no other subscription applies or for features like LNP, that are used on an office-wide basis.

  Note: Refer to *UCS DMS-250 Local Number Portability Application Guide* for more information.

- Message switch and buffer 7 (MSB7) cannot be used or provisioned in a UCS DMS-250 office requiring CAIN services.
- CAIN interaction with enhanced operator services (EOPS) provides limited termination support.
- Interaction between mechanized calling card services (MCCS) and CAIN is not supported.
- Delivery of ***CallingPartyID*** and ***ChargeNumber*** is subject to in-switch feature restrictions.
- CAIN outpulsing is subject to in-switch outpulsing logic.
- Class of service (COS) override is not supported when CAIN call processing performs COS screening.
- Direct termination through a tandem switch (using table TANDMRTE) is only supported over IMTs that are datafilled to support UCS-to-UCS ISUP protocol. When any other protocol is used, retranslation occurs at the tandem switch.

  Note: This restriction does not apply to routing using table TERMRTE.

- Only the following pretranslator selectors are supported: CT, ES, UA, IP, IN, and UAX for AXXESS.

  Note: UAX is used for AXXESS agents, which are handled differently. Refer to *UCS DMS-250 CAIN/FlexDial Interactions* for more information.

- CAIN only supports offboard IN/1 validation of account codes, speed

numbers, or N00 numbers after the call leaves the **Collect_Information** and before the call triggers at the **Analyze_Information** PIC.

- Full implementation of the UCS09 CAIN software requires the CAIN_PROTOCOL_STREAM parameter to be set to UCS09 and the CAIN_PROTOCOL_VERSION parameter to be set to V4. Both parameters are provisioned in table CAINPARM.
- PRI originations do not support reorigination.
- When the call is answered, both the CAIN extension block and the CAIN framework extension block are deallocated, unless an ***O_Mid_Call*** event is armed.

Digital Switching Systems

UCS DMS-250

NetworkBuilder Application Guide,
Volume 1 of 5

Product Documentation–Dept 3423
Nortel Networks
P.O. Box 13010
RTP, NC 27709–3010
1–877-662-5669

Publication number: 297-2621–370
Product release: UCS17
Document release: Standard 10.01
Date: July 2002
Printed in the United States of America

NORTEL NETWORKS™

How the world shares ideas.