Critical Release Notice

Publication number: 297-2621-390
Publication release: Standard 06.02

The content of this customer NTP supports the SN06 (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.

Attention!

Adobe® Acrobat® Reader™ 5.0 is required to view bookmarks in color.

Publication History

March 2004

Standard release 06.02 for software release SN06 (DMS).

Change of phone number from 1-800-684-2273 to 1-877-662-5669, Option 4 + 1.

297-2621-390

Digital Switching Systems

UCS DMS-250

FlexDial Framework Application Guide

UCS13 Preliminary 06.01 March 2000

NORTEL NETWORKS™
How the world shares ideas.

Digital Switching Systems

UCS DMS-250

FlexDial Framework Application Guide

Publication number: 297-2621-390
Product release: UCS13
Document release: Preliminary 06.01
Date: March 2000

Printed in the United States of America

NORTEL NETWORKS CONFIDENTIAL: The information contained herein is the property of Nortel Networks and is strictly confidential. Except as expressly authorized in writing by Nortel Networks, the holder shall keep all information contained herein confidential, shall disclose the information only to its employees with a need to know, and shall protect the information, in whole or in part, from disclosure and dissemination to third parties with the same degree of care it uses to protect its own confidential information, but with no less than reasonable care. Except as expressly authorized in writing by Nortel Networks, the holder is granted no rights to use the information contained herein.

Information is subject to change without notice. Nortel Networks reserves the right to make changes in design or components as progress in engineering and manufacturing may warrant.

Publication history

March 2000

Preliminary release 06.01 for UCS13 software release.

Technical changes based on A60007768 DDOC, SPM FlexDial Enhancements. Additions and rewrites made in Chapter 1, pages 1-11 through 1-15, and Chapter10, pages 10-1 through 10-2.

December 1999

Draft release 07.01 for UCS13 software release.

November 1999

Standard release 05.02 for UCS12 software release.

August 1999

Preliminary release 05.01 for UCS12 software release.

Changed range for fields TKEYDUR and NTKEYDUR of the REORGTYP option from 5–30 to 4–30 in Table 7-23 of Chapter 7, “Table FLEXFEAT.” Added two notes after the lead-in text to Table 7-23 to provide information about how the setting of the REORIG_SHORT_OR_LONG office parameter in table OFCVAR affects the range and the interpretation of the TKEYDUR and NTKEYDUR values by the UCS DMS-250 switch.

Global change from UCS09 to UCS12 for the CDRTMPLT option in all applicable tables.

ATTENTION

The UCS12 software release does not support Enhanced Operator Position System (EOPS) functionality. The UCS software continues to support operated-assisted calls through other platforms such as Enhanced Services Provider (ESP). Refer to Appendix A in the *UCS DMS-250 Feature Change Reference Guide* for additional information about EOPS removal.

Removed the “OPERDISP option” section from Chapter 5, “Table FLEXTYPE.”

Removed the “OPERDISP call processing application” section from Chapter 17, “FLEXTYPE features and characteristics applications.”

Removed the “AXXESS to EOPS interactions” section in Appendix D.

March 1999

Standard release 04.03 for software release UCS09.

Technical revisions made to the following:

- MODDIGS sequence collectable in Chapter 2, “Table FLEXDIAL”
- name of terminal type in Chapter 9, “Table TRKGRP”
- AGNTDATA restrictions and limitations in Chapter 16, “FlexDial collectables call processing applications”

November 1998

Standard release 04.02 for software release UCS09.

Revision for this release: added log FLEX309.

October 1998

Preliminary release 04.01 for software release UCS09. Some of the changes include:

- Added the new collectable FGDPARM in table FLEXDIAL.
- Added new SUBR addressee messages VALIDATE, FAILACT, FEATIGN to table MSGCTR.
- Added the CICSIZE option to tables TRKFEAT and TRKSIG.
- Added the CIC option to table TRKFEAT.
- Added the OUTCIC and CICBLK options to table TRKSIG.
- Added ITRANSTS and CICDELV options to table FLEXFEAT.
- Added DEFCIC option to AGNTDATA collectable in table FLEXDIAL.
- Replaced CICDELV option in table TRKFEAT with a CICBLK option in table TRKSIG.

August 1998

Standard release 03.02 for software release UCS08.

May 1998

Preliminary release 03.01 for software release UCS08. Some of the changes include:

- Added new collectables RETRIEVE, NOTIFY, TIMER, KILLTMR, and CALLCOND.
- Added new FLEXFEAT table options TRANSNUM, TRANSYS, IEXCLINX, FEATVAR, and CPACTVAL.
- Added new office parameter NUM_CPRC_EXT_BLK.
- Added new log FLEX308.
- Removed provisioning guide information. (FlexDial provisioning examples are still included in Appendix C).

Contents

Table FLEXFEAT 7-1

Comparison to existing subscriber number provisioning 8-1

Table TRKGRP 9-1

Table TRKSIG 10-1

FLEXFEAT features and characteristics applications 18-1

TRKGRP features and characteristics applications 19-1

TRKFEAT features and characteristics application 20-1

TRKSIG features and characteristics applications 21-1

About this document

This document describes the Flexdial framework application and provides information on understanding, planning, datafilling, implementing, and testing FlexDial.

In addition, the document contains information on the FlexDial conversion tool, which converts non-FlexDial trunk agencies to FlexDial-supported agencies. Also included is information on the FlexDial simulation tool, which provides the ability to simulate FlexDial call processing from a MAP terminal. The document also provides real-life FlexDial provisioning examples.

Intended audience

This application guide assists telecommunications engineers, technicians, switching system developers, operating company personnel, and anyone else who requires technical information on FlexDial framework application.

The 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 chapters in this document provide the following:

Chapter 1, FlexDial framework overview

Chapter 1 provides an overview of the FlexDial framework and includes information on originating agent terminology and interactions, trunk group type AXXESS, FlexDial collectables, and FlexDial framework provisioning. It also provides information on SPM FlexDial enhancements.

Chapter 2, Table FLEXDIAL

Chapter 2 describes the FlexDial Agent Interaction Definition (FLEXDIAL) table.

Chapter 3, Table FLEXMOD

Chapter 3 describes the FlexDial Digit Modification (FLEXMOD) table.

Chapter 4, Table MSGCTR

Chapter 4 describes the FlexDial Message Center (MSGCTR) table.

Chapter 5, Table FLEXTYPE

Chapter 5 describes the FlexDial Subscriber and Call Type Definitions (FLEXTYPE) table.

Chapter 6, Table FLEXVAL

Chapter 6 describes the FlexDial Subscriber Number Validation (FLEXVAL) table.

Chapter 7, Table FLEXFEAT

Chapter 7 describes the FlexDial Subscriber Number and Call Type Features (FLEXFEAT) table.

Chapter 8, Comparison to existing subscriber number provisioning

Chapter 8 identifies how the existing subscriber number feature and characteristic provisioning maps into the FlexDial framework provisioning scheme.

Chapter 9, Table TRKGRP

Chapter 9 describes the Trunk Group (TRKGRP) table.

Chapter 10, Table TRKSIG

Chapter 10 describes the FlexDial Trunk Group Signaling (TRKSIG) table.

Chapter 11, Table TRKFEAT

Chapter 11 describes FlexDial Trunk Group Features (TRKFEAT) table.

Chapter 12, Comparison to existing trunk group provisioning

Chapter 12 identifies how the existing trunk group feature and signaling information provisioning maps into the new FlexDial framework provisioning scheme for AXXESS agents.

Chapter 13, Office parameter provisioning

Chapter 13 describes the office parameters defined for interaction with the FlexDial framework.

Chapter 14, QFLEXVAL, QTRK, and QSUBR CI commands

Chapter 14 includes information on the QFLEXVAL command interpreter (CI) increment that provides an additional level of manipulation capability for FLEXVAL table. It also includes information on the QTRK CI command (to display provisioning information for an AXXESS trunk group), and the QSUBR CI command (to display provisioning information for a particular subscriber).

Chapter 15, FlexDial call processing applications overview

Chapter 15 describes the FlexDial call processing applications.

Chapter 16, FlexDial collectables call processing applications

Chapter 16 describes call processing applications using the FlexDial collectables.

Chapter 17, FLEXTYPE features and characteristics applications

Chapter 17 provides information on the FLEXTYPE features and characteristics applications.

Chapter 18, FLEXFEAT features and characteristics applications

Chapter 18 provides information on the FLEXFEAT features and characteristics applications.

Chapter 19, TRKGRP features and characteristics applications

Chapter 19 provides information on the TRKGRP features and characteristics applications.

Chapter 20, TRKFEAT features and characteristics applications

Chapter 20 provides information on the TRKFEAT features and characteristics applications.

Chapter 21, TRKSIG features and characteristics applications

Chapter 21 provides information on the TRKSIG features and characteristics applications.

Chapter 22, FlexDial logs and OMs

Chapter 22 provides information on the FlexDial logs and Operational Measurements (OMs).

Chapter 23, System requirements

Chapter 23 provides information on the system requirements for FlexDial framework.

Appendix A, FlexDial simulator

Appendix A describes the FlexDial simulation tool.

Appendix B, FlexDial conversion tool

Appendix B describes the FlexDial conversion tool.

Appendix C, FlexDial provisioning examples

Appendix C provides FlexDial provisioning examples.

Appendix D, FlexDial interactions

Appendix D describes FlexDial interactions with CAIN and TCAP.

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 second 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 and rereleased in the *same* software release cycle. For example, the second release of a document in the first 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 *UCS DMS-250 Master Index of Publications*, 297-2621-001.

This document is written for all 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 *UCS DMS-250 Master Index of Publications*, 297-2621-001.

References in this document

The following documents are referred to in this document:

- *UCS DMS-250 Billing Records Application Guide*, 297-2621-395

- *UCS DMS-250 Cain/FlexDial Interactions*, 297-2621-372
- *UCS DMS-250 Call Detail Record Reference Manual*, 297-2621-119
- *UCS DMS-250 Commands Reference Manual*, 297-2621-819
- *UCS DMS-250 CSP Translations Reference Manual*, 297-2621-350
- *UCS DMS-250 Data Schema Reference Manual*, 297-2621-851
- *UCS DMS-250 Feature Change Reference Guide*, 297-2621-050
- *UCS DMS-250 Feature Group D (FGD) Application Guide*, 297-2621-385
- *UCS DMS-250 General Description*, 297-2621-100
- *UCS DMS-250 Logs Reference Manual*, 297-2621-840
- *UCS DMS-250 Master Index of Publications*, 297-2621-001
- *UCS DMS-250 NetworkBuilder Application Guide*, 297-2621-370
- *UCS DMS-250 NT6X50EC Integrated Echo Canceller Application Guide*, 297-2621-365
- *UCS DMS-250 Office Parameters Reference Manual*, 297-2621-855
- *UCS DMS-250 Operational Measurements Reference Manual*, 297-2621-814
- *UCS DMS-250 Software Optionality Control (SOC) User's Manual*, 297-2621-301
- *UCS DMS-250 Spectrum Reference Manual*, 297-2621-550
- *UCS DMS-250 Transaction Capabilities Application Part (TCAP) Application Guide*, 297-2621-355

What precautionary messages mean

The types of precautionary messages used in Nortel Networks documents 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.

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.

How commands, parameters, and responses are represented

Commands, parameters, and responses in this document conform to the following conventions.

Input prompt (>)

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

```
>BSY
```

Commands and fixed parameters

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

```
>BSY CTRL
```

Variables

Variables are shown in lowercase letters:

```
>BSY  CTRL  ctrl_no
```

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:

```
FP 3 Busy CTRL 0: Command request has been submitted.
FP 3 Busy CTRL 0: Command passed.
```

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

1 Manually busy the CTRL on the inactive plane by typing

>BSY CTRL ctrl_no
and pressing the Enter key.

where

ctrl_no is the number of the CTRL (0 or 1)

Example of a MAP response:

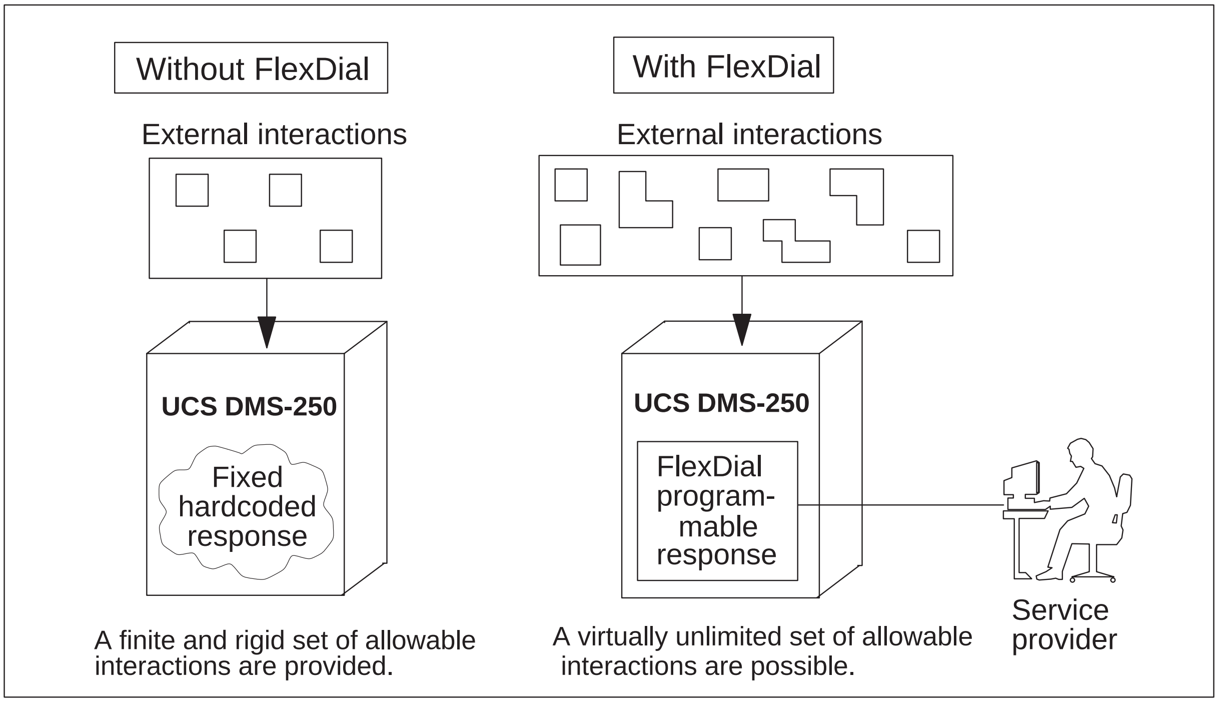
```
FP 3 Busy CTRL 0: Command request has been submitted.
FP 3 Busy CTRL 0: Command passed.
```

FlexDial framework overview

The FlexDial framework is a call processing application that defines, through provisioning, any desired interaction with the originating agent. For example, FlexDial can mimic FGA or FGD interactions, or introduce an interaction set that is unique from existing functionality.

Without the FlexDial framework, the UCS DMS-250 software architecture relies heavily on hard-coding to make critical decisions during call setup. Customizing dialplan interactions requires costly software development. With the FlexDial application, this restriction can be removed through use of the new datafill control for dialplan, agent, and subscriber features. The actual dialog with the user/interface for a particular access terminal is completely provisioned by the service provider, as shown in Figure 1-1.

Figure 1-1
FlexDial and external interactions


Without FlexDial
With FlexDial
External interactions
External interactions
UCS DMS-250
Fixed hardcoded response
UCS DMS-250
FlexDial program-mable response
Service provider
A finite and rigid set of allowable interactions are provided.
A virtually unlimited set of allowable interactions are possible.

With FlexDial, you can essentially “program” the call origination side of the call to implement customized functionality. FlexDial opens up all aspects of the interactions (or dialplan) with the originating agent through a *new provisioning data model* that is used to define the desired interaction.

The FlexDial framework identifies three areas of provisioning:

- dialplan provisioning, allowing you to mimic existing protocols or interactions (such as FGD or DAL), modify existing protocols or interactions, or create your own interactions or services.
- agent provisioning, allowing you to customize agent features independent of dialplan interactions. With few exceptions, the functionality of the FlexDial agency meets or exceeds the capability of non-FlexDial agencies.
- subscriber provisioning, allowing you to treat all subscriber numbers generically and create your own subscriber number types with specific services, such as account codes or Mechanized Calling Card Services (MCCS). With few exceptions, the capability of processing subscriber numbers and subscriber features meets or exceeds the capability of non-FlexDial subscriber provisioning.

Note: FlexDial can co-exist with existing non-FlexDial agency functionality.

Software optionality control

Software optionality control (SOC) for FlexDial framework restricts the number of provisionable entries (tuples) in table FLEXDIAL to the number purchased from Northern Telecom (Nortel). The order code for the FlexDial SOC is UTRS0002.

A standard set of datafill is delivered with the table upon initial purchase; the limit is initially set to the number of tuples present in the standard datafill. When a new tuple is added, the count is incremented provided that the limit is not reached. If the limit is reached, any attempts to add a new tuple will be denied.

For more information on SOC for the FlexDial framework, see the *UCS DMS-250 Software Optionality Control (SOC) User’s Manual*.

Billing records feature

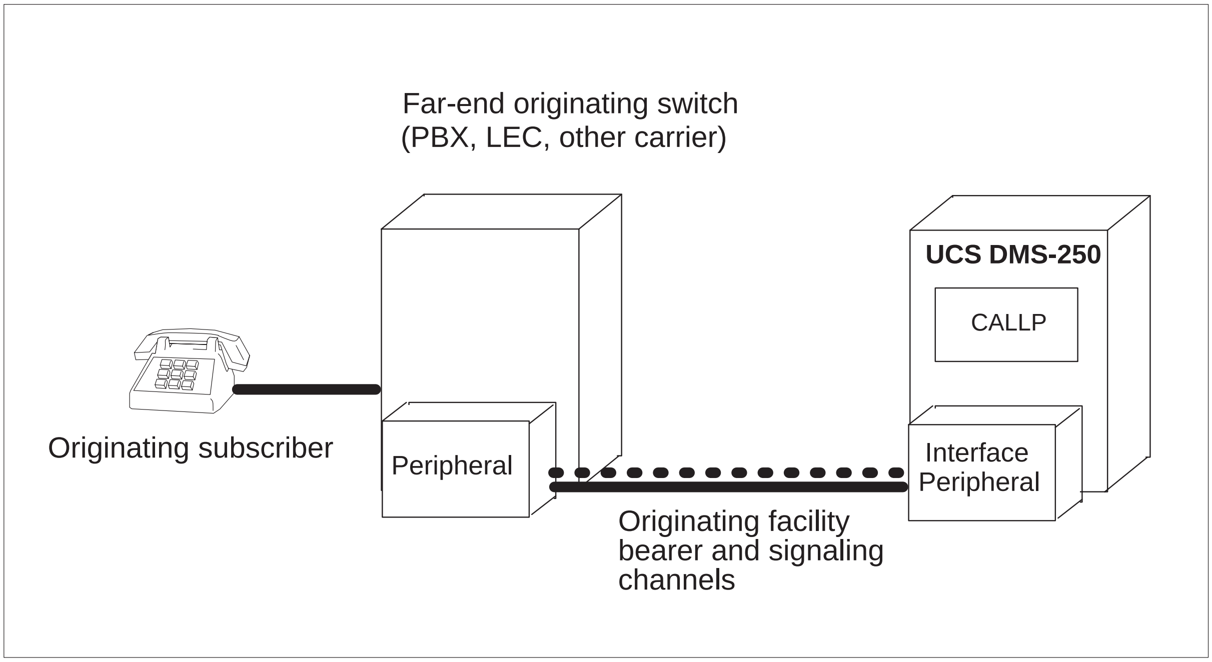
The capabilities of FlexDial are enhanced when you use the Billing Records feature. This feature allows your switch to generate customized call detail records (CDRs).

For more information, see the *UCS DMS-250 Billing Records Application Guide*.

Originating agent terminology and interactions

The term *originating agent* describes either the far-end originating switch or originating subscriber, as shown in Figure 1-2.

Figure 1-2
Originating agents


Far-end originating switch
(PBX, LEC, other carrier)
UCS DMS-250
CALLP
Originating subscriber
Peripheral
Interface Peripheral
Originating facility bearer and signaling channels

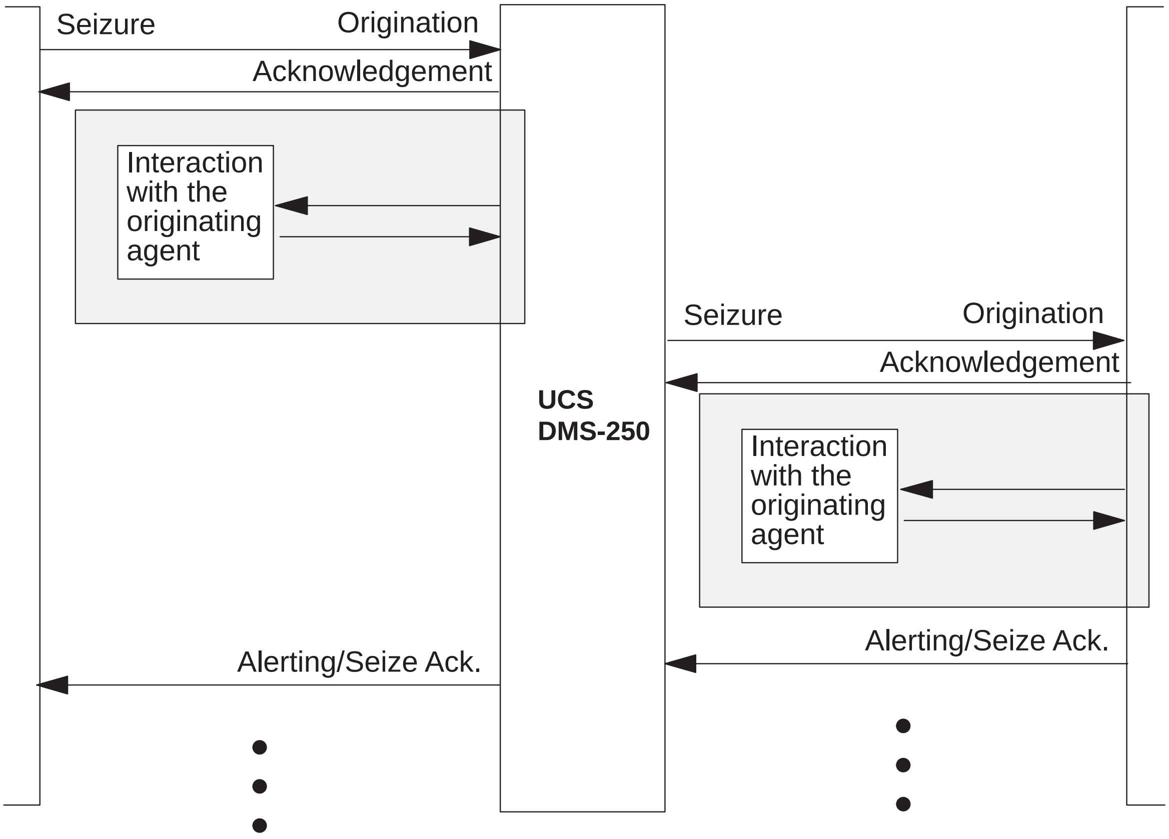
A UCS DMS-250 switch receives different types of information from the originating far-end switch. For example, a switch will use different signaling protocols, or can treat the signaling channel and bearer channel as separate entities. After a switch has established a connection with the originating far-end switch, it can receive inband digits directly from the subscriber.

There are multiple sources of digits (originating far-end switch and originating subscriber) and variants of the originating facility types that affect what digits are received and how they are received. This document uses the term *originating agent* generically to describe the entity with which the call processing is establishing a dialog or interaction.

Originating agent interaction

A UCS DMS-250 switch’s interaction with the originating agent occurs immediately after it detects origination, and typically ends when it translates and routes the call, as shown in Figure 1-3.

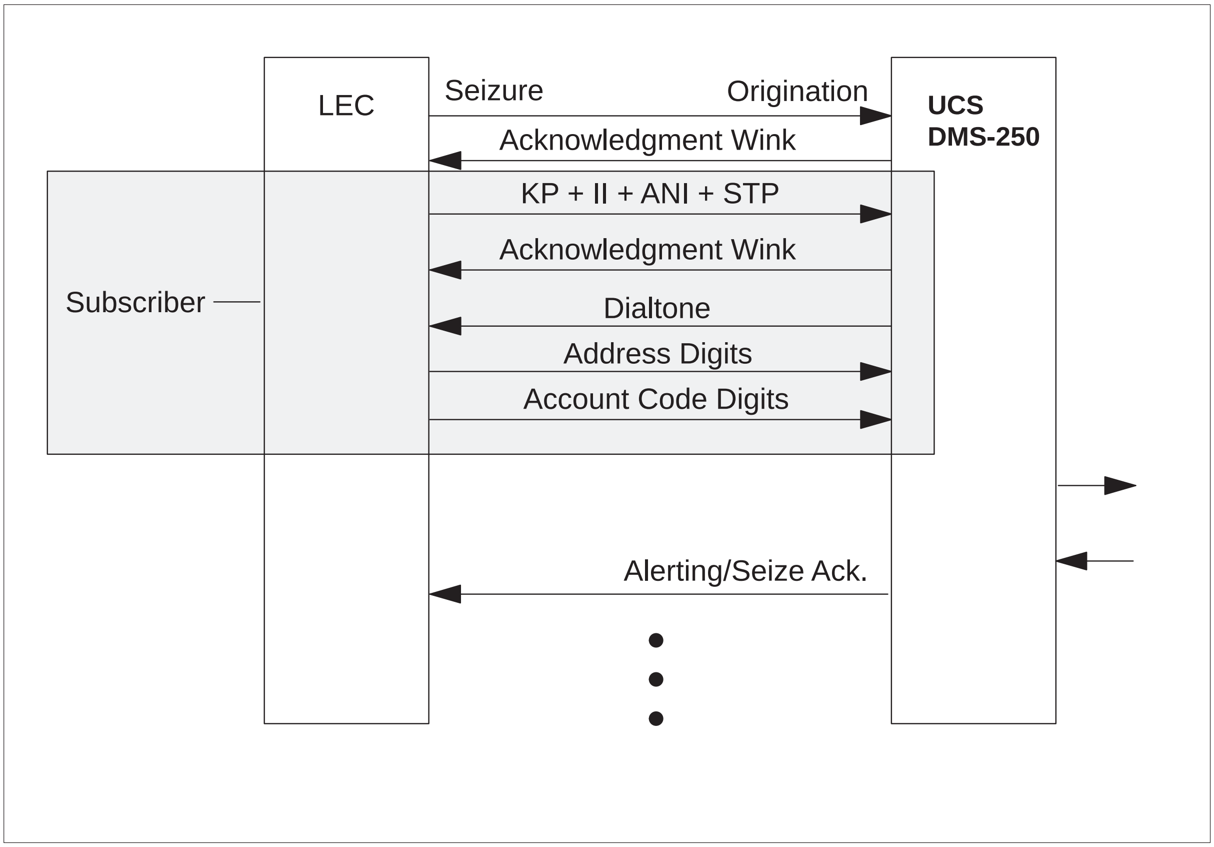
Figure 1-3
Call processing interaction with the originating agent



Originating agent interaction example

An example of an interaction with an FGD protocol originating agent is shown in Figure 1-4.

Figure 1-4
FGD protocol originating agent interaction example



Originating agent interaction modifications

Any interaction with the originating agent is initially identified by the trunk group; the definition of the trunk group includes an indication of how the switch will set up a call.

During this interaction, a number of things can occur:

- A switch can receive certain digit sequences that change the interaction. For example, receiving 1NX instead of normal information digits identifies a three-stage international call on FGD instead of a normal two-stage call.
- A switch can validate certain digit groups that change the interaction. For example, validation of an automatic number identification (ANI) number can identify that account code digits must be collected.
- A switch can receive certain nature of address (NOA) values that change the interaction. For example, the NO_NUM_CT NOA indicates a cut-through interaction and changes the defined interaction being processed.

- A switch can receive external message interactions that change the interaction. For example, N00 IN1 Transaction Capabilities Application Part (TCAP) (VER2) can alter the interaction with the originating agent.

FlexDial framework provisioning overview

The FlexDial framework enables you to datafill or provision all aspects of the interaction with the originating agent for a particular trunk group or subscriber. Therefore, FlexDial table control provisioning is the primary component of the framework application.

The FlexDial framework provides several new tables to allow for the necessary provisioning schemes, which are grouped into three logical areas:

- dialplan interaction provisioning
- subscriber and call type feature provisioning
- agent provisioning

Note: The data model for table control for FlexDial provides better reuse of datafill and better organization of the data.

Dialplan interaction provisioning

Dialplan provisioning refers to the identification of the interaction with the originating agent. Without FlexDial, dialing plan schemes are hard-coded in the UCS DMS-250 switch, and are typically based on standards or a subscriber dialing plan model. Hard-coded schemes are inflexible and difficult to expand or modify. With FlexDial, you can essentially “program” the call origination side of the call to implement new and customized functionality.

This provisioning includes tables FLEXDIAL, FLEXMOD, and MSGCTR:

- FLEXDIAL—FlexDial Agent Interaction Definition table

  Table FLEXDIAL, the main table in the framework, defines interactions with the originating agent.

- FLEXMOD—FlexDial Digit Modification table

  The FLEXMOD table provides a provisioning scheme in the FlexDial framework for incoming digit manipulation.

- MSGCTR—FlexDial Message Center table

  The MSGCTR table is an extension to the TRKFEAT and FLEXFEAT tables, and stores data for the TRKFEAT and FLEXFEAT MSGCTR option.

With the FlexDial framework, the FLEXDIAL table outlines the complete interaction with the originating agent. This enables you to match or modify existing standards or create your own interaction schemes. All required instructions are provided by the FLEXDIAL table, and can be arranged in any order to define the desired interaction sequence.

FlexDial collectables

The interaction with the originating agent provided by table FLEXDIAL is defined as a series of instructions, called *collectables.* A collectable is a specific service or instruction that is processed during the *collect information* point in the call model. This instruction executes some portion of the overall defined interaction with the originating agent. Through table FLEXDIAL collectables, a macro or script capability exists through provisioning, and the interaction is actually “programmed” by operating company personnel.

The FlexDial framework defines six classes of collectables:

- protocol collectables

  Protocol collectables deal with protocol-specific instructions.

- framework collectables

  Framework collectables provide call processing mechanisms for handling the execution of collectables.

- sequence collectables

  Sequence collectables contain instructions that deal with sequences of digits or digit-related information (such as NOA).

- digit collectables

  Digit collectables group, identify, and process received digits. There are two types of digit collectables:

  — Inband digit collectables refer to digits collected as tone pulses in the bearer channel of the originating agent.

  — Out-of-band digit collectables refer to digits received as binary data in a message received in the signaling channel of the originating agent.

- call type collectables

  Call Type collectables allow provisioning for call types based on dialing plans. A call type is defined as the subscriber’s actions to terminate the call to the desired destination.

- variable collectables

The Variable collectables perform mathematical operations involving variables defined for FLEXDIAL, and allow conditional branching of collectable processing based on these variables.

Table 1-1 provides a list of all FlexDial collectables, their classification (type), and a brief description.

Table 1-1
Summary of FlexDial collectables

Collectable type	Collectable name	Description
Protocol	SIG	Alters digit collection signaling characteristics during the setup interaction with the originating agency.
	RCVSIG	Receives an inband wink, answer, or release signal from the originating agent and report within a timeout value. The type of wink scanned for is opposite the current hook state. Provides application for inband signaled agent types only (not yet supported).
	SNDSIG	Sends an inband wink, an out-of-band alerting, or an answer or release signal to the originating agent. The answer or release signal can be sent inband or out-of-band. The type of wink sent is opposite the current hook state.
	FGDPARM	Executes the SS7 FGD protocol within one collectable.
Framework	TERMINATE	Indicates that the call should be forwarded to the terminating agent of the call (as specified by translations) at this point.
	NOTIFY	Provides the ability to play an uninterruptable tone or announcement during the interaction with the originating agent.
	TIMER	Provides an additional triggering mechanism to initiate FlexDial collectable execution for the originating agent.
	KILLTMR	Provides the ability to disable a FlexDial timer activated through execution of the TIMER collectable.
	CALLCOND	Provides an additional triggering mechanism to initiate FlexDial collectable execution for the originating agent.
	GOTO	Allows the execution of collectables to branch to a separate collectable list and continue processing.
	DO	Allows the execution of collectables to branch to a separate collectable list before processing is continued in the current list.

—continued—

Table 1-1
Summary of FlexDial collectables (continued)

Collectable type	Collectable name	Description
Sequence	INCLUDE	Allows a list of collectables to be included in the currently processing collectable list.
	APTRMT	Specifies that if a treatment has been set during previous collectable processing, then treatment should now be applied at this point in the call.
	ROUTE	Indicates that the call should terminate to the route specified, ignoring any normal translation information. This collectable supports the identification of a single route, not a route list.
	NOOP	Performs no action.
	IFTRMT	Branches within the collectable list based on the current treatment setting for the call.
	IFPRMT	Branches within the collectable list based on the last issued prompt.
	IFTOD	Branches within the collectable list, based on a range of times.
	IFDIGS	Compares digits received against the specified digit vector. Branches if equal or not equal, depending on provided options.
	IFCNT	Compares the count of the number of digits received against the identified value. Branches if equal or not equal, depending on provided options.
	IFNOA	Compares the received nature of address (NOA) against the specified NOA. For DTMF digit collection, an unknown NOA value is received. For MF digit collection, the KP and ST values received are used to determine the NOA. For ISUP, the NOA is taken from the Initial Address Message (IAM). Branches if equal or not equal, depending on provided options.
	IFPARM	Determines if the identified parameter exists in the out-of-band message and branches if it does.
	DELDIGS	Removes a number of digits from the received digit buffer.
	ADDDIGS	Adds the specified digit vector to the digit buffer.
	MODDIGS	Allows modifications of digits strings in the digit buffer.
	COPYDIGS	Copies specified digits in the digit buffer to another location within the buffer.

—continued—

Table 1-1
Summary of FlexDial collectables (continued)

Collectable type	Collectable name	Description
	MODNOA	Provides the ability to alter the calling or called party NOA with the value specified.
	AGNTDATA	Takes digit information stored in the AXXESS agent provisioning and places it into the digit buffer to be processed by other collectables.
	RETRIEVE	Provides the ability to retrieve the identified digits that have been already processed for the call and add them to the digit buffer at the desired location.
Digit	COLDIG	Collects inband (PTS) raw digit information.
	SUBR	Collects and validates inband (PTS) subscriber numbers.
	ADDR	Collects and validates inband (PTS) called party address numbers.
	OLI	Collects and validates inband (PTS) information digits.
	CIC	Collects and processes inband (PTS) carrier identification code (CIC) digits.
	REPLDIG	Collects digits to replace those in the digit buffer. Only applies to inband (PTS) digit collection.
	COLPARM	Collects out-of-band (SS7) raw digit information.
	SUBRPARM	Collects and validates out-of-band (SS7) subscriber numbers.
	ADDRPARM	Collects and validates out-of-band (SS7) called party address numbers.
	OLIPARM	Collects and validates out-of-band (SS7) information digits.
	CICPARM	Collects and processes out-of-band (SS7) CIC digits.
Call type	CALLTYPE	Provides the means to identify features, characteristics, and billing information applicable to the call. Since the CALLTYPE collectable can be provisioned anywhere within a collectable list, the feature, characteristic, and billing information can also be related to the interaction occurring with the originating agent.

—continued—

Table 1-1
Summary of FlexDial collectables (continued)

Collectable type	Collectable name	Description
	CLRFTRS	Provides the capability of resetting features and characteristics to their default value.
	SETTRANS	Provides the capability of setting the translation system to be used for the call.
	SETTRMT	Sets the identified treatment as a delayed treatment.
	OM	Increments a user-defined OM as a result of following along a particular path of FLEXDIAL execution.
	APRESET	Applies a deferred reset in the FLEXDIAL collectable processing.
Variable	VAROP	Performs mathematical operations involving a user-defined variable.
	IFVAR	Branches within the collectable list, based on the value of a pre-defined variable.

—end—

Examples of using collectables include the following:

- The SIG protocol collectable alters the digit collection signaling characteristics, such as pulse type and timer values.
- The IFDIGS sequence collectable can modify the dialing plan based on the receipt of certain digits or digit patterns.
- The MODDIGS sequence collectable uses the FLEXMOD table to modify digits in the digit buffer. Digits are removed from the buffer, and indexed into the FLEXMOD table to retrieve digits to be reinserted into the buffer.
- The FLEXDIAL table supports branching mechanisms to provide support of interaction alterations based on receipt of certain NOA values. For example, receipt of an STP digit on a FGD type protocol identifies that cut-through subscriber digit collection is to occur. Through the IFNOA sequence collectable, performing this type of dialing plan modification is possible.
- Through the FLEXFEAT DPIDX options, the dialing plan scheme can be modified based on the validation of particular subscriber numbers.

See Chapters 2 and 16 for detailed information on collectables.

See Appendix C for dialplan provisioning examples.

Agent provisioning and the AXXESS trunk group type

The immense differences that the FlexDial framework introduces with existing implementation require a “generic” trunk group type called AXXESS.

Note: FLEXCONV, the FlexDial conversion tool, enables you to convert non-AXXESS trunk members to AXXESS trunk members. See Appendix B for more information.

For AXXESS agents, the UCS DMS-250 switch provisions trunk group interactions and signaling protocols *independently* from the trunk group definition. This allows the interactions to become service-specific rather than agent-specific. Multiple trunk groups reference the same feature set indexes for better organization of data.

For Flexdial SPM AXXESS trunks, the following functionality has been added:

- PTS and ISUP signalling
- The tracking of MF and DTMF resources is performed for table TRKSIG.

Note: An error is issued when the pulsetype of DTMF AXXESS trunk is changed to MF trunk in table TRKSIG provided that this trunk has been provisioned in table TRKMEM and there are no MF resources provisioned for this SPM in table MNCKTPAK. Similar warning is issued when the pulsetype of an MF trunk is changed to DTMF trunk in table TRKSIG, if no DTMF resources are allocated.

The SPM consists of two commom equipment module (CEM) units and up to 26 resource modules (RM). The digital signaling processing (DSP) RM in the SPM consists of a number of digital signal processors which are used to provide call processing resources. SPM PTS uses AB bit resources to scan and generate AB trunk signals. AB bit resource is provided in the DSP RM. Multi–frequency (MF) and dual–tone multi–frequency (DTMF) resources are also provided in the DSP RM to support SPM PTS trunks. The number of AB bit resources, DTMF resources and MF resources that are configured on a DSP RM is provisioned in table MNCKTPAK.

Flexdial is a CALLP application, constructed of reusable components that provide the ability to define through provisioning any desired dialing plan or interaction with the originating agent for the purpose of call setup, for specific interaction services or point of presence access.

The AXXESS trunk group is used with the Flexdial framework. AXXESS trunks use the TRKFEAT, TRKSIG, and FLEXDIAL tables to identify the attributes of the trunk group. Table TRKFEAT may also utilize table MSGCTR in order to identify information needed to override specific aspects of Flexdial processing. TRKSGRP is not used to provision AXXESS trunk groups, but instead TRKSIG is used to identify the signaling and interface facility characteristics of the AXXESS agent in trunk group table. Many trunk groups that have identical signaling and interface facility characteristics can share a single TRKSIG table entry. This table replaces the use of table TRKSGRP for AXXESS agent. Table TRKSIG tuples must share the same physical store area as table TRKSGRP tuples.

For a legacy PTS trunk, when changes are made to the signaling information in table TRKSGRP, the trunk has to be busied (BSY) and then returned to service (RTS) for the changes to be dynamically updated. However for a legacy ISUP trunk when changes are made to the trunk TRKSGRP, the trunk does not need to be made busy and returned to service for the changes to be dynamically updated.

If PTS AXXESS trunks are not allocated for Table TRKMEM, then no error messages are displayed in Table TRKSIG. However if PTS AXXESS trunks allocated for Table TRKMEM, and if there are enough MF and DTMF resources for SPM in Table MNCKTPAK, then it will do the appropriate change. However if there are not enough resources in Table MNCKTPAK then it will display a warning in Table TRKSIG.

For legacy SPM PTS trunks when there are no MF or DTMF resources provisioned for this SPM in table MNCKTPAK, then trying provisioning an MF or DTMF PTS trunk for this SPM in table TRKMEM is not allowed. Same condition applies for AXXESS trunks, such as AXDAL, AXEAN, and others.

When legacy SPM PTS trunks such as MF trunk DAL220 is allocated for Table TRKMEM and MF is equal or more than one, and DTMF equals zero in Table MNCKTPAK, then in Table TRKSGRP if an attempt is made to change this MF trunk to a DTMF trunk then a warning is issued that there are no DTMF resources in Table MNCKTPAK.

This feature binds the TRKSIG data to the audit procedure. The audit procedure compares the CM data with the CEM data and updates the CEM data with the CM data in case of a data mismatch. A check is added to this procedure to see if it fuctioning properly. The dynamic update procedure that is sent to the SPM after a PTS trunk is manually busied and returned to service is also checked.

This feature does not change any current functionality for SPM. However it does ensure that the following procedures are enhanced:

- A track is kept of MF and DTMF resources for the PTS SPM AXXESS trunks.
- If an MF or DTMF trunk is allocated in Table TRKMEM, and if there are no MF or DTMF resources in Table MNCKTPAK for this SPM, then a warning is issued if user tries to change a DTMF trunk to an MF trunk or viceversa in Table TRKSIG.
- Changes to the signalling information in Table TRKSIG for ISUP AXXESS trunks and for the PTS AXXESS trunks take effect when the trunks are busied and returned to service.

Note: FLEXCONV, the FlexDial conversion tool, enables you to convert non-AXXESS trunk members to AXXESS trunk members. See Appendix B for more information.

With the FlexDial framework capability, the AXXESS trunk group type can mimic the following UCS DMS-250 trunk group types:

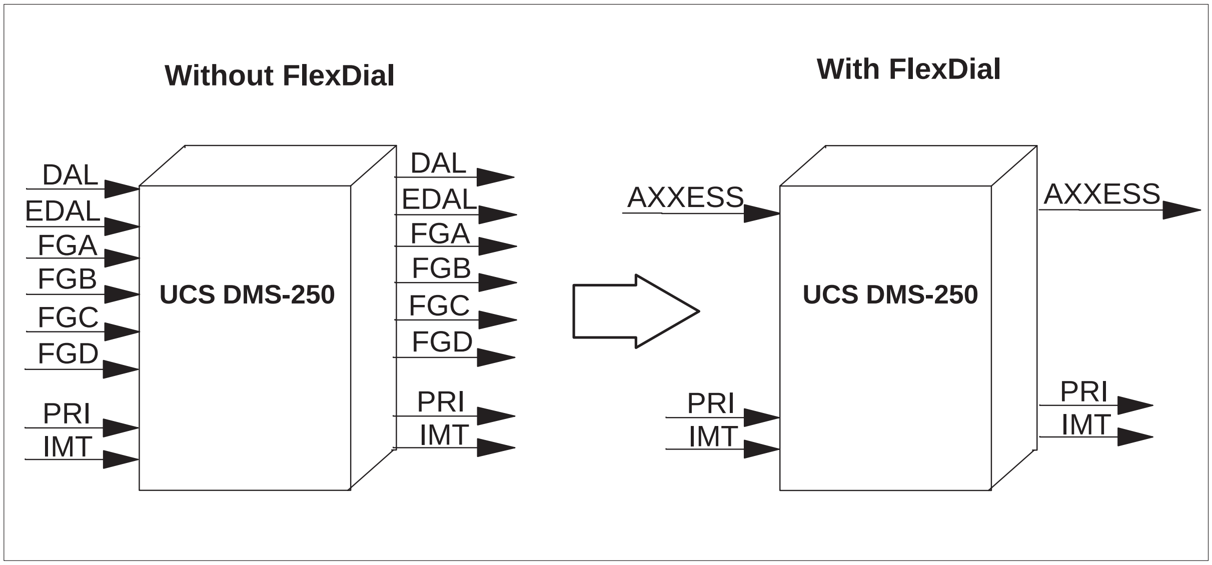
- DAL
- EDAL
- FGA (ONAL)
- FGB (ONAT)
- FGC (ONAT)
- FGD (EANT)

Note: FlexDial framework does not support intermachine trunks (IMTs) or PRI (PRA250) trunks.

In addition to supporting the protocols or agent characteristics above, FlexDial may enable you to define and implement new protocols or modify existing protocols that aren’t currently supported by the UCS DMS-250 switch.

Figure 1-5 shows how agent provisioning is simplified using FlexDial framework and AXXESS trunks.

Figure 1-5
UCS DMS-250 non-FlexDial and FlexDial agent provisioning



Agent provisioning includes tables TRKFEAT, TRKSIG, MSGCTR, and TRKGRP:

- TRKFEAT—FlexDial Trunk Group Features table

  The TRKFEAT table identifies the feature and characteristic profile associated with a particular FlexDial framework (AXXESS) trunk group.

- TRKSIG—FlexDial Trunk Group Signaling table

  The TRKSIG table replaces the Trunk Subgroup (TRKSGRP) table provisioning for AXXESS trunk group type agents.

- MSGCTR—FlexDial Message Center table

  Table MSGCTR provides an extension to table TRKFEAT, where feature and characteristic information of a particular trunk group must be delivered to the FlexDial collectables for processing.

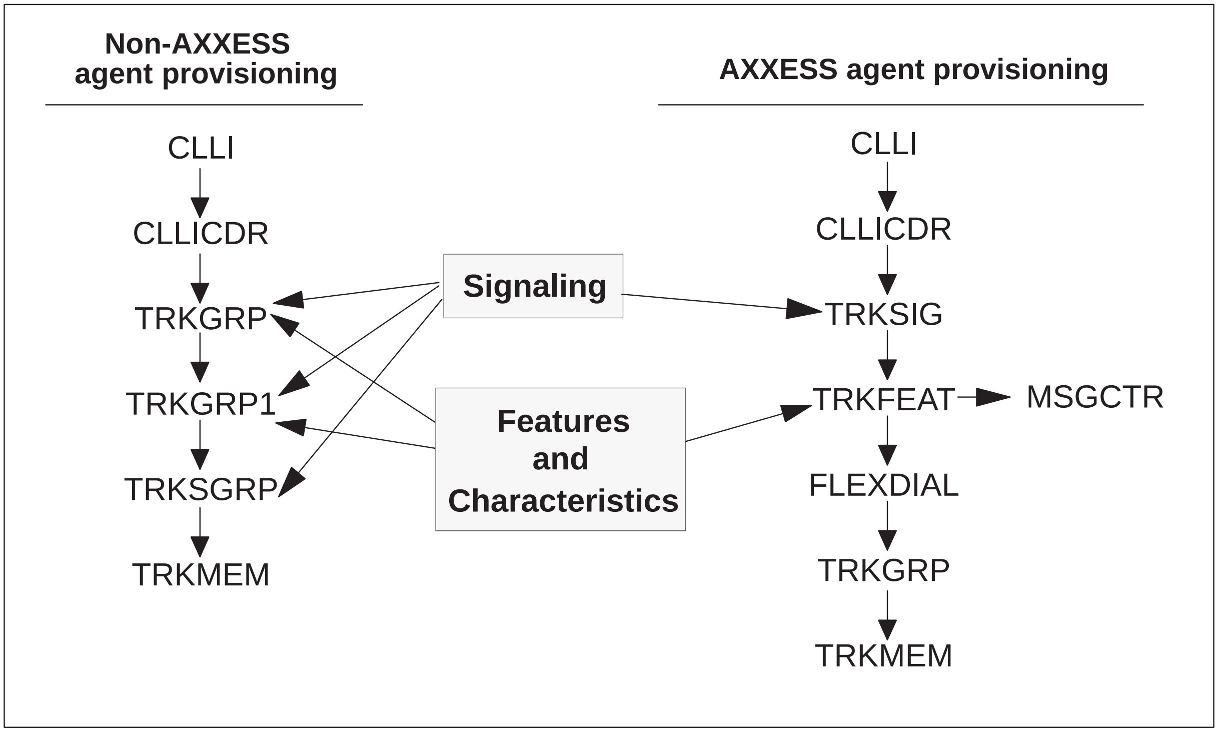
The existing table modified by the framework changes includes:

- TRKGRP—Trunk Group table

  The TRKGRP table is modified to add the AXXESS trunk group type refinement.

Introduction of the AXXESS trunk group type changes the previous trunk group provisioning model, as shown in Figure 1-6.

Figure 1-6
Comparisons of non-AXXESS to AXXESS trunk group provisioning



Non-AXXESS agent provisioning

For non-AXXESS agents:

- Features and characteristics are provisioned in table TRKGRP, and as an extension, table TRKGRP1.
- Signaling characteristics of the agent are provisioned in table TRKSGRP.

Due to the provisioning scheme, non-AXXESS trunk groups with identical provisioning are still datafilled independently, and cannot share provisioned characteristics. For example, each trunk group must provision its own signaling data, even though many trunk groups share the same signaling characteristics.

AXXESS agent provisioning

For AXXESS agents, trunk group features and characteristics and signaling data are provisioned independently from the trunk group definition, as follows:

- The AXXESS trunk group provisioning relies on the Common Language Location Identifier (CLLI) and TRKGRP tables. AXXESS is a trunk group type in table TRKGRP.

- For AXXESS agents, trunk group features and characteristics are provisioned in table TRKFEAT. Table TRKFEAT replaces most of the TRKGRP datafill and all the TRKGRP1 datafill.
- Signaling characteristics are provisioned in table TRKSIG. TRKSIG replaces table TRKSGRP for AXXESS agents.
- AXXESS trunk groups then reference an index into tables TRKFEAT and TRKSIG to identify the features and signaling characteristics of trunk group members.

Advantages of AXXESS provisioning scheme

The advantages of provisioning your switch using AXXESS trunk groups include the following:

- Less store is used to provision an identical number of AXXESS agents. Many trunk groups can share the same features or signaling characteristics (in tables TRKFEAT and TRKSIG, respectively) and this reduces datafill.
- Trunk group provisioning data is more logically organized.

TRKGRP table provisioning for AXXESS agents is simplified, as all but basic and necessary fields are removed from the table. See Figure 1-7.

Figure 1-7
TRKGRP table provisioning for AXXESS agents

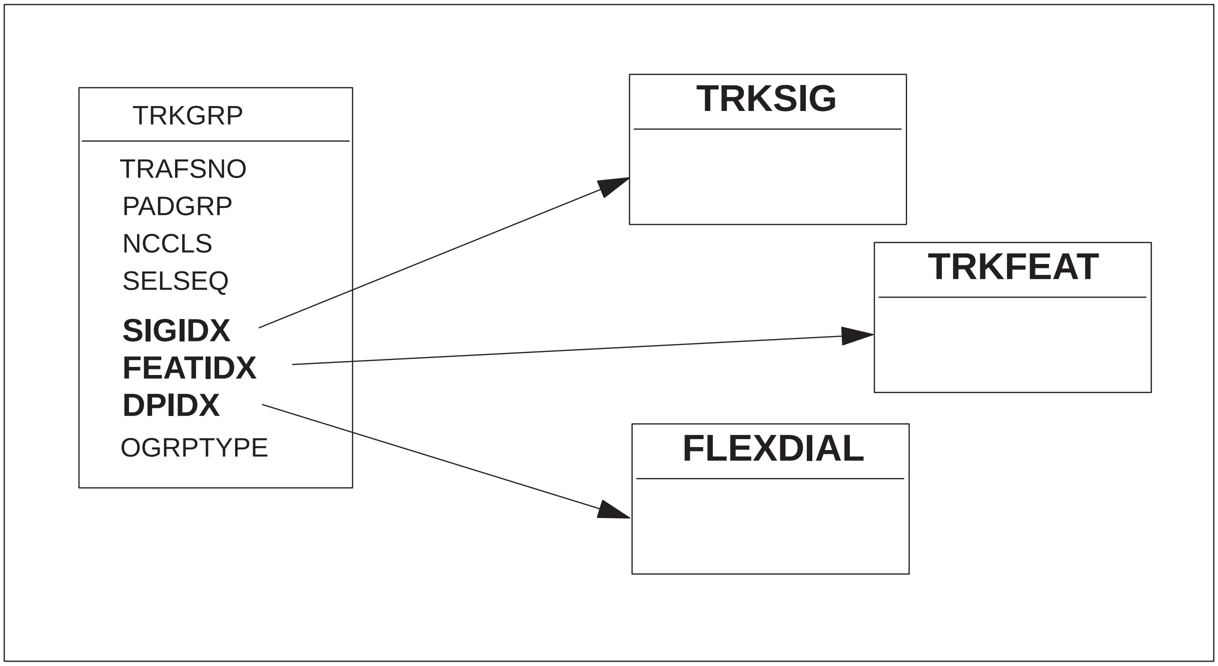


Table FLEXDIAL completely defines the interaction with the originating agent for AXXESS trunk group types (unlike non-AXXESS trunk groups). For example, for an FGD type of interaction, nothing specific defines the interaction as “FGD,” except the methodology of FlexDial collectable provisioning.

See Chapter 2 for information on the FLEXDIAL table.

See Appendix D for information on FlexDial interactions.

Because AXXESS agents are two-way, they are also used as terminating trunks. The OGRPTYPE field (outgoing group type) identifies the outpulsing scheme for the terminating AXXESS agent. Values DAL, ONAL, ONAT, and EANT are supported.

See Chapters 10 and 11 for information on the TRKSIG and TRKFEAT tables, respectively.

See Chapter 14 for a description of the CI QTRK command. QTRK is a CI command used to display provisioning information for an AXXESS trunk group.

See Appendix C for an agent provisioning example.

Subscriber and call type feature provisioning

Subscriber and call type feature provisioning allow you to identify features, characteristics, and billing information for subscriber numbers and call types.

Subscriber number provisioning

For non-AXXESS agents, there are five types of subscriber numbers that can be processed on the agent: authorization codes (AUTHs), automatic number identification numbers (ANIs), personal identification numbers (PINs), account code digits (ACCTs), and travel card numbers (TCNs).

The UCS DMS-250 switch recognizes each type of subscriber number, and performs specific actions based on the subscriber number type being processed:

- specific screening (tables AUTHCODU, AUTHCDU2, AUTHCDU3, AUTHCDU4, AUTHCDU5, ANISCUSP, ACSCRN2, and MULTIPIN)
- specific feature and characteristic handling
- capturing of the subscriber number in specific billing record field
- specific treatments on validation failures

With FlexDial, however, the UCS DMS-250 switch does not distinguish among subscriber number types; the definition of specific subscriber number types (like AUTH or ANI) is provisionable in the framework. This allows you to define your own subscriber number types and thus your own new subscriber services (for example, a voice menu service).

This generic subscriber number provisioning capability is provided by the following tables:

- FLEXTYPE—FlexDial Subscriber and Call Type Definitions table

  The FLEXTYPE table defines the subscriber number type, the location in the billing record for capture of this number, and other options regarding this subscriber number type.

- FLEXVAL—FlexDial Subscriber Number Validation table

  The FLEXVAL table screens or validates the subscriber number digits. This table retrieves a FLEXFEAT index that contains the features and characteristics for the subscriber number.

- FLEXFEAT—FlexDial Subscriber Number and Call Type Features table

  The FLEXFEAT table identifies the features and characteristics that can be assigned to a particular subscriber number.

- MSGCTR—FlexDial Message Center table

  The MSGCTR table provides an extension to table FLEXFEAT, where feature and characteristic information of a particular subscriber number must be delivered to other collectables for processing.

These tables are used in conjunction with the SUBR and SUBRPARM digit collectables provisioned in the FLEXDIAL table. See Chapters 2 and 16 for information on these collectables.

Call type feature provisioning

A call type is defined as the subscriber’s actions to terminate the call to the desired destination.

Without FlexDial, the UCS DMS-250 switch types the call through screening the called party address digits. The new call type collectables provides the UCS DMS-250 switch with greater capability in identifying features, characteristics, and billing information for the call, based on the interaction with the originating agent (not strictly based on the called party address).

See Appendix C for detailed call type provisioning examples.

Advantages of subscriber/call type number provisioning

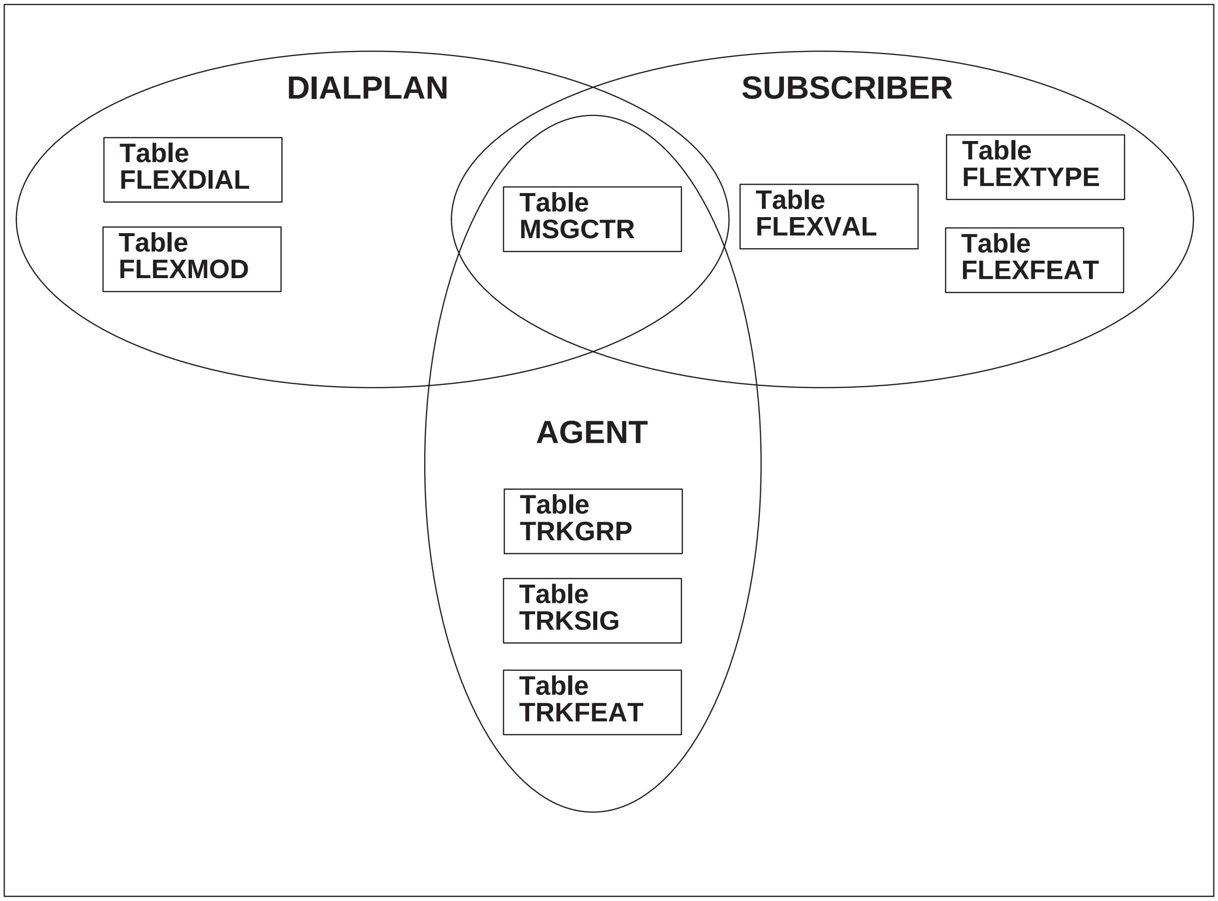
The advantages of the FlexDial framework subscriber/call type number handling capabilities include the following:

- One to 16-digit length subscriber numbers of any type and any size are supported.
- Variable length subscriber number collection is supported. The process uses provisioned MIN and MAX values.
- Subscriber number types are no longer limited to certain sizes of digits. For example, FlexDial can support international ANIs.
- Only two tables are used for all subscriber number type validation: FLEXVAL (screening) and FLEXFEAT (features and characteristics). Subscriber numbers can share FLEXFEAT indexes.
- The available set of features and characteristics can be provisioned against any subscriber number type.
- Enhanced prompting capability exists for all digit collection in the FlexDial framework. This includes the capability to use announcement voice prompts at any time, as well as numerous tones (including those defined in table TONES).
- All of the current internal switch subscriber number functionality (except for off-board validation) is supported for subscriber numbers in the FlexDial framework.

Provisioning table relationships

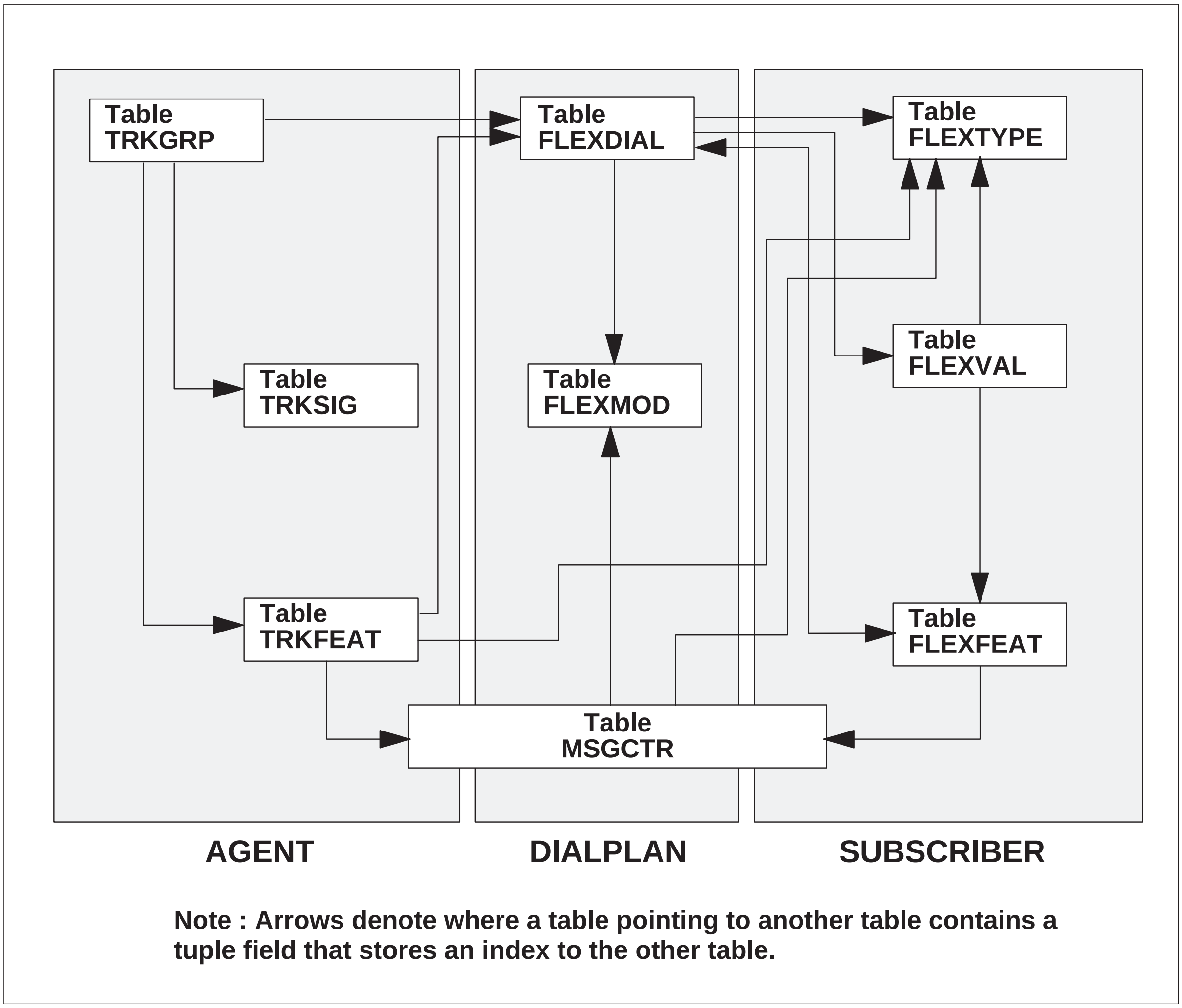
Figure 1-8 shows how the three provisioning areas interact with each other.

Figure 1-8
FlexDial framework provisioning model



These interactions or associations are listed on a per-table basis, where one table contains an index or reference to another table. Figure 1-9 identifies the complete matrix of interactions.

Figure 1-9
FlexDial provisioning framework table associations



Note : Arrows denote where a table pointing to another table contains a tuple field that stores an index to the other table.

FlexDial framework provisioning table associations

Table 1-2 provides FlexDial framework provisioning system table associations.

Table 1-2
Provisioning system table associations

Table Association	Rationale
TRKGRP –> TRKSIG	The TRKSIG table identifies the provisioned signaling information for the trunk group. This table replaces the use of TRKSGRP for trunk group signaling datafill for AXXESS trunk group types. The SIGIDX field identifies the index in table TRKSIG that must be previously provisioned. Many trunk groups can share the same TRKSIG index.
TRKGRP –> TRKFEAT	The TRKFEAT table identifies the provisioned features and characteristics of the trunk group. Features and characteristics are no longer provisioned in the TRKGRP and TRKGRP1 tables. The FEATIDX field identifies the index in table TRKFEAT that must be previously provisioned. Many trunk groups can share the same TRKFEAT index.
TRKGRP –> FLEXDIAL	The trunk group tuple DPIDX field defines the initial dialing plan defined interaction with the originating agent.
TRKFEAT –> FLEXDIAL	The COSOVE option identifies a FLEXDIAL table index to provide a new thread of execution for FLEXDIAL call processing.
TRKFEAT –> MSGCTR	Through the MSGCTR option, the TRKFEAT table contains indexes into the MSGCTR table.
TRKFEAT –> FLEXTYPE	Through the REORIGAL option, table TRKFEAT identifies subscriber number types provisioned in the FLEXTYPE table that require revalidation on reoriginated calls.
FLEXDIAL –> FLEXMOD	The MODDIGS sequence collectable contains an index into table FLEXMOD to provide a table lookup scheme for modifying digits.
FLEXDIAL –> FLEXTYPE	The SUBR and CALLTYPE collectables in the FLEXDIAL table contain a FLEXTYPE table index that identifies the type of subscriber number being collected or call type features and characteristics being set.
FLEXDIAL –> FLEXVAL	The SUBR digit collectable in the FLEXDIAL table provides a numeric index for validating the subscriber number digits in table FLEXVAL.

—continued—

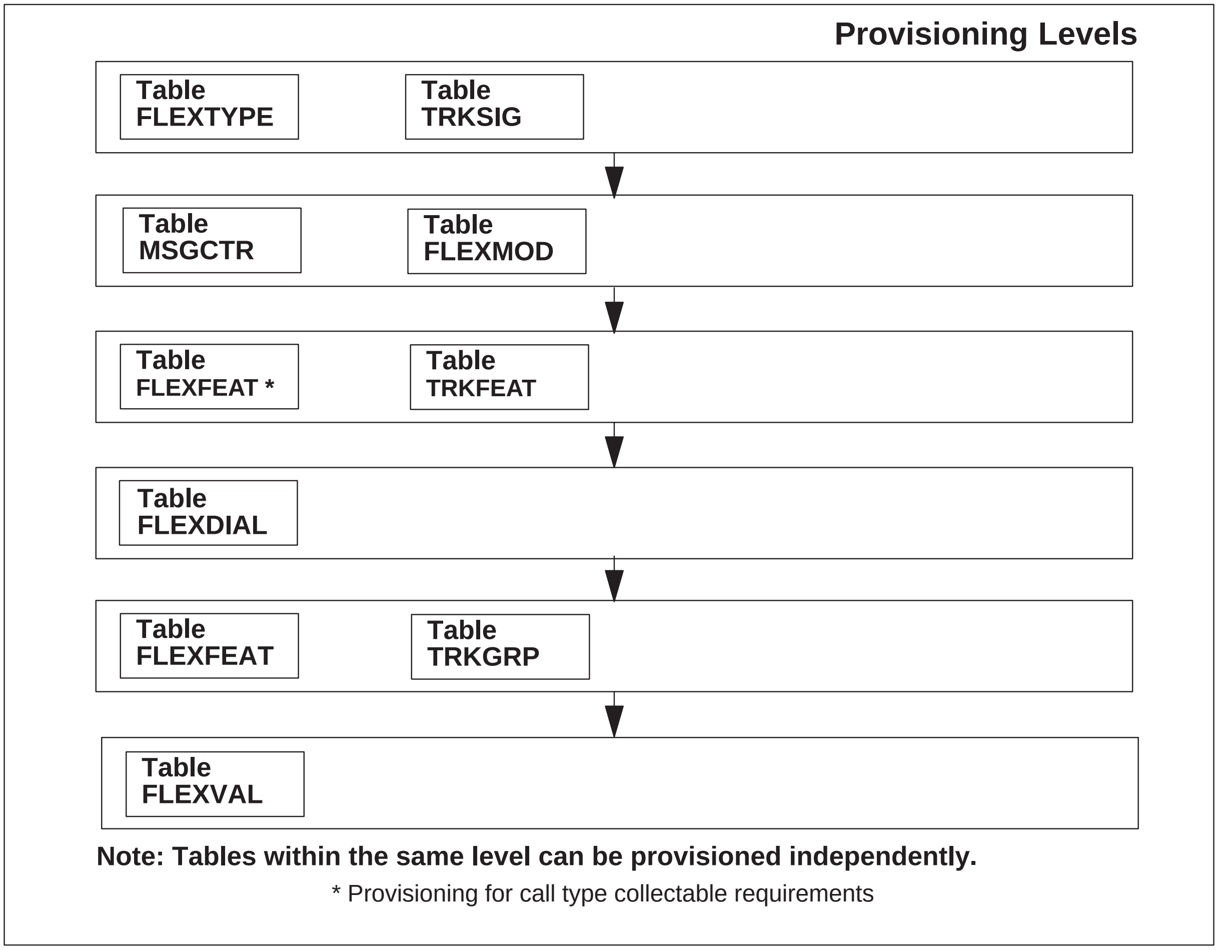
Table 1-2
Provisioning system table associations (continued)

Table Association	Rationale
FLEXDIAL –> FLEXFEAT	The CALLTYPE collectable identifies a FLEXFEAT table index outlining features and characteristics that apply to the call. The CLRFTRS collectable identifies a list of FLEXFEAT defined features and characteristics that are to be reset (to their default value).
MSGCTR –> FLEXTYPE	Tuples in the MSGCTR table can reference indexes in the FLEXTYPE table.
MSGCTR –> FLEXMOD	The MSGCTR MODDIGS message type contains an index into the FLEXMOD table.
FLEXVAL –> FLEXFEAT	The FLEXVAL table FEATIDX field identifies an index into the FLEXFEAT table. This index provides the feature and characteristic provisioning information for the subscriber number that is used to index the FLEXVAL table.
FLEXVAL –> FLEXTYPE	A part of the FLEXVAL table key consists of a valid FLEXTYPE table index.
FLEXFEAT –> FLEXDIAL	The DPIDX and REORIGACT FLEXFEAT table options identify an index into the FLEXDIAL table. The FLEXDIAL index provides a new thread of execution for FlexDial call processing.
FLEXFEAT –> MSGCTR	Through the MSGCTR option, the FLEXFEAT table contains indexes into the MSGCTR table. Additionally, the REORGACT option contains a MSGCTR table index that is applied on reoriginated calls.
—end—	

Datafill order

The datafill order of tables in the FlexDial framework is generally in the order shown in Figure 1-10, although an exception exists for FLEXFEAT/FLEXDIAL associations: provisioning the CALLTYPE collectable in the FLEXDIAL table requires that the FLEXFEAT index be already provisioned. But provisioning the DPIDX FLEXFEAT option requires that the FLEXDIAL index be already provisioned.

Figure 1-10
FlexDial provisioning order



FlexDial framework restrictions and limitations

FlexDial limitations and restrictions include, but are not limited to, the following:

- IMT and PRI trunks cannot be simulated with FlexDial.
- Translation verification (TRAVERS) tool does not work with the FlexDial trunks. (The FLEXSIM tool is provided, however; see Appendix A.)
- Universal tone receivers (UTRs) are required for FlexDial trunks; FlexDial trunks can receive digits using the existing UTR hardware in the current extended multi-processor system peripheral module (XPM) digit trunk controller (DTC) peripheral. FlexDial trunks do not support digit collection using multifrequency (MF) or dual-tone MF (DTMF) receivers on a series I (maintenance trunk module, or MTM; service trunk module, or STM) peripheral.

Note: The FlexDial agent supports a reorigination capability also utilizing UTR hardware receivers. Potential overuse of UTR resources (for call origination and reorigination) can be a potential problem for call processing. Maximum engineering of UTR hardware for peripherals as well as minimal use of UTR reorigination capability is recommended.

- The off-board validation feature is not available for FlexDial trunks. However, the N00 TCAP application is available.
- Dump and restore of non-FlexDial group types to the FlexDial AXXESS group type is not supported.

Table FLEXDIAL

This chapter describes the FlexDial Agent Interaction Definition (FLEXDIAL) table.

Overview

Purpose

The FLEXDIAL table is the main part of the provisioning foundation for the FlexDial framework and defines the individual instructions that comprise the interaction with the originating agency. The table provides a list of all possible instructions pertaining to signaling and digit collection that together can define the desired interaction. Without FlexDial, the interaction with the originating agent occurs through hard-coded “dialing plans” on the UCS DMS-250 switch.

Description

The defined interaction with the originating agent is based on agent terminal characteristics, subscriber (number) characteristics, and dialed number characteristics. Agent terminal characteristics actually define the initial dialing plan interaction scheme, such as collecting address digits, and also define signaling, protocol, and call type characteristics that apply to the interaction.

Subscriber (number) characteristics refer to subscriber information (acquired through validation of the subscriber number) that affects or defines how the interaction with the originating agent is handled or modified. For instance, one type of subscriber number may cause another subscriber number to be collected (ANI –> ACCT). Subscriber-specific information also includes indices that are used to validate collected digits. Other types of subscriber information that affect the interaction with the originating agent include filed digits for collectables to be processed (such as filed address digits on a hotline call).

Dialed number characteristics refer to how the called party number validation identifies other interactions between call processing and the originating agent. For instance, address digits that are identified through validation as a travel card access number requires the collection of the travel card number and possibly other subscriber numbers.

Provisioning within the FLEXDIAL table centrally locates all information regarding the processing of a particular instruction. For instance, the send wink instruction identifies the pre-wink duration time, the actual wink duration time, and the post-wink duration or guard time. The provisioned instruction contains all information regarding the sending of the wink. All information regarding the collection, such as prompt, filed digits, and validation of those digits is provisioned within the instruction.

The concept of containing all information within the provisioned instruction implies that the FLEXDIAL table contains agent, subscriber, and dialed–number specific information. However, a goal of the framework is to have all the subscriber information within the subscriber provisioning tables, and not require subscriber provisioning to occur within the FLEXDIAL table. (This simplifies provisioning and makes more efficient use of the FLEXDIAL table.) Therefore, the system allows subscriber number provisioning to override FLEXDIAL provisioning for particular instructions. See the FLEXFEAT MSGCTR option on page 7-24 for more information.

The framework allows for FLEXDIAL provisioning to include agent terminal characteristics. The defined interaction with the originating agent is more of an integral part in the definition of an agent terminal, and therefore, agent characteristics are more intertwined with the FLEXDIAL table.

While dialed number characteristics can alter the interaction with the originating agent, no explicitly defined dialed number characteristics are provisioned within the FLEXDIAL table. (This includes the possibility of screening an 800 number as a called party address subscriber number). Similar to subscriber number characteristics, the framework provides the capability of not provisioning dialed number characteristics directly within the FLEXDIAL table.

General layout

The FLEXDIAL table is indexed by a 24-character string (1 to 24 characters) as the key that identifies a tuple. The options available for provisioning constitute the instructions that are linked together to form the interaction being defined. You can choose any of the options in any order to provide the desired interaction with the originating agent.

These instructions or options are termed *collectables* by the FlexDial framework.

Key

The key to the FLEXDIAL table consists of a 24-character string (1 to 24 characters). This string value provides a unique index into the FLEXDIAL table through use of the data dictionary.

Because of limitations in the data dictionary for the string type key, the table size is limited to 32696 entries. Table 2-1 describes the FLEXDIAL table key.

Table 2-1
FLEXDIAL table key

Key Field	**Description**	**Values**
KEY	The table key consists of a string of up to 24 characters. 32696 possible unique entries are provided.	Vector of up to 24 characters

Fields

The FLEXDIAL table tuple does not contain a fixed set of fields, but consists of an options vector that can be provisioned with any of the defined collectables. Each collectable in turn defines its own data refinement for provisioning purposes. Because the tuple consists of an options vector, the default is to not have any options provisioned. However, when provisioning the field of the collectable, a default value for the field type can be specified by the DEFDATA table.

Table 2-2 describes the FLEXDIAL table fields.

Table 2-2
FLEXDIAL table fields

Fields	Description	Values	Default
OPTION	This field consists of an options vector that contains a list of collectables. The option field identifies the collectable type that is being provisioned. Collectable types are Protocol, Framework, Sequence, Digit, Call Type, and Variable.	NIL Protocol collectables: SIG, RCVSIG, SNDSIG, FGDPARM Framework collectables: TERMINATE, GOTO, DO, INCLUDE, APTRMT, ROUTE, NOOP, IFTRMT, IFPRMT, IFTOD, NOTIFY, TIMER, KILLTMR, CALLCOND Sequence collectables: IFDIGS, IFCNT, IFNOA, IFPARM, DELDIGS, ADDDIGS, MODDIGS, COPYDIGS, MODNOA, AGNTDATA, RETRIEVE Digit collectables : COLDIG, SUBR, ADDR, OLI, CIC, REPLDIG, COLPARM, SUBRPARM, ADDRPARM, OLIPARM, CICPARM Call Type collectables: CALLTYPE, CLRFTRS, SETTRANS, SETTRMT, OM, APRESET Variable collectables : VAROP, IFVAR	N/A
CONTINUE	This field indicates whether the list of collectables should continue with the list identified by the DPIDX field.	N or Y	N/A

—continued—

Table 2-2
FLEXDIAL table fields (continued)

Fields	Description	Values	Default
DPIDX	If the CONTINUE field is set to Y, then this field is present and identifies a FLEXDIAL table index. The list of collectables represented by the FLEXDIAL index is appended to the list identified by the option vector for call processing execution.	An existing entry in table FLEXDIAL.	N/A
—end—			

Each collectable option has its own field refinements and is presented independently.

Error reporting system

A unique error reporting system helps to identify where a particular provisioning error occurs within the options field vector for collectable provisioning. When an error occurs in provisioning a collectable within the options vector, the following traceback is provided to the terminal display:

```
<Option> OPTION AT POSITION <pos> FAILED VALIDATION
```

<Option> identifies the particular collectable, and <pos> identifies the collectable's position within the vector where the failure occurred.

For example,

```
SUBR OPTION AT POSITION 2 FAILED VALIDATION
```

For collectables, such as digit collectables, that also contain vectors of options, the error traceback also identifies the option the error occurred on. For example:

```
VALIDATE OPTION AT POSITION 1 FAILED VALIDATION
SUBR OPTION AT POSITION 2 FAILED VALIDATION
```

or

```
PROMPT OPTION AT POSITION 1 FAILED VALIDATION
RESET OPTION AT POSITION 2 FAILED VALIDATION
SUBR OPTION AT POSITION 3 FAILED VALIDATION
```

With the traceback information, the source of the provisioning error can be quickly identified.

In the case of multiple errors in the table entry attempt, the traceback displays only the first error that occurs.

FlexDial collectables

This section contains information on each of the FlexDial collectables, which are the instructions or options in table FLEXDIAL.

Note: For detailed information on the FlexDial collectables, see Chapter 16.

NIL collectable

The NIL collectable enables the option vector to be easily updated. NIL can be inserted instead of removing a collectable in the list by retyping the provisioned collectables that follow, or replacing the collectable with a NOOP collectable. Using NIL enables the inserted collectable to be effectively removed without having to retype the remaining provisioned collectables.

The NIL option does not remain visible once the update is complete.

Note: The NIL option is identical to the existing NIL option for TRKGRP options vector provisioning. See the *UCS DMS-250 Data Schema Reference Manual*.

Definition

The NIL option does not contain any specific field refinements.

Examples

An example showing the use of NIL option is:

- Editing existing tuple:

  ```
  MY_IDX (COLL1) (COLL2) (COLL3)$
  ```

- To remove COLL2, change the table entry. When prompted at COLL2, enter NIL.

  ```
  option: COLL2
  > NIL
  ```

- After editing, display of the tuple shows:

  ```
  MY_IDX (COLL1) (COLL3)$
  ```

NIL collectable restrictions and limitations

There are no identified table control restrictions for the NIL option.

SIG protocol collectable

The Signaling (SIG) protocol collectable provides a way to alter the signaling characteristics of digit collection during the process of collecting digit information from the originating agent. The pulse-type, timer values, and digit masks are information that define the signaling characteristics of digit collection.

Definition

All defined default values for fields can be specified by defining the default value for the type in the DEFDATA table. Because collectables are optional by nature, there are not default values for their fields if the collectable is not provisioned. During provisioning of the collectable, the default value for the field type can be identified by the DEFDATA table.

Table 2-3 contains the SIG protocol collectable field refinements.

Table 2-3
SIG protocol collectable fields

Fields	Description	Values	Default
PULSTYP	This field identifies the pulsetype for incoming digits.	None, MF, DTMF	N/A
PSEIZTMR	This field identifies the permanent seize timeout value.	1 to 30 seconds	N/A
PDILTMR	This field identifies the partial dial timeout value. This value identifies the maximum amount of time between received digits until the minimum amount of digits to collect are received.	1 to 30 seconds	N/A
MINRTMR	This field identifies the maximum amount of time between received digits after the minimum number of digits have been collected.	1 to 30 seconds	N/A
FDIGMASK	If the pulse type is equal to MF, then the first digit mask field is present and identifies the types of digits that are allowed to be the first digit received. All other digits received before this first digit are ignored.	Subset of {KP, KPP} or ALL or NONE. ALL includes all possible KP digits within the set, while NONE indicates no first digit mask is identified.	N/A

—continued—

Table 2-3
SIG protocol collectable fields (continued)

Fields	Description	Values	Default
LDIGMASK	If the pulse type is equal to MF, then the last digit mask field is present and identifies the types of digits that indicate when the last digit in the MF stream has been received.	Subset of {ST, STP, ST2P, ST3P} or ALL or NONE. ALL includes all possible ST digits within the set, while NONE indicates that no last digit mask is identified.	N/A
DIGMASK	If the pulse type is equal to DTMF, then the digit mask field is present and identifies the types of DTMF digits that are reported. Any digits received that are not contained within this set are ignored. The S identifies an asterisk digit, and the P identifies an octothorpe digit.	Subset of {0–9, A, B, C, D, S, P} or ALL or NONE. ALL includes all possible DTMF digits within the set, while NONE indicates that no DTMF digits are recognized. ***Note:*** The A, B, C, D digits represent 4th column DTMF digits.	N/A
TRMDIGIT	If the pulse type is equal to DTMF, then this field is present and identifies the terminating digit that when entered causes all digits collected to be immediately reported. The S identifies an asterisk digit, and the P identifies an octothorpe digit.	S or P	N/A
RESETDIGIT	If the PULSETYP is equal to DTMF, then this field is settable and defines either the octothorpe or the asterisk as the reset digit.	S, P P = octothorpe; S = asterisk	
—end—			

With these signaling parameters, digit collection is performed in the following manner:

- The terminal controller (XPM) is instructed to wait for a time equal to the PSIGTMR value before the first digit is received. If the first digit is not received within this time frame, a permanent seize timeout report is sent to the central module (CM).

- For MF digit collection, the terminal controller collects digits from the start of the first digit until the final digit is received (as identified by the final digit mask). When the final digit is received, the digits are reported to the central module (CM). Upon the receipt of each digit, a timer is started with either the PDILTMR or MINRTMR timer value. If the timer expires before the next digit is received, then a partial dial timeout is reported to the CM along with currently received digits.
- For DTMF digit collection, the CM identifies the minimum number of digits to be collected by the terminal controller (the minimum number of digits is identified by the digit collectable). The digits are reported to the CM if a timeout has occurred (PDILTMR value), if the terminating digit has been received, or if the minimum number of digits have been received.

Once the minimum number of digits have been received, the CM instructs the terminal controller to collect up to the maximum number of digits identified using the MINRTMR as the interdigit timer value. The terminal controller reports the digits if the timer expires, if the terminating digit is received, or if the requested number of digits is received.

The RESETDIGIT field allows the operating company personnel to define a common reset digit throughout the dial plan. The reset digit can also be set within table TRKSIG.

Although the reset digit can still be redefined by the collectables OLI, CIC, SUBR, SIG, REPLDIG, COLDIGS, and ADDR, performance is optimized when a single reset digit is assumed across the entire dial plan.

If the SIG collectable is used to set prompts, its setting overrides provisioning in table TRKSIG. Furthermore, a prompt set in a collectable overrides settings in both table TRKSIG and/or a SIG collectable.

Examples

The following are examples of SIG protocol collectable use:

```
SIG MF 10 4 4 (KP) $ (ST STP ST2P) $
```

This identifies that MF pulse type signaling is set for the agent, with a permanent seize timer of ten seconds, a partial dial timer of four seconds, and a minimum collected partial dial timer of four seconds. The first digit mask consists of a KP digit only, and the last digit mask identifies that the terminating digit in the MF stream can be either ST, STP, or ST2P.

```
SIG DTMF 10 15 4 (0 1 2 3 4 5 6 7 8 9 A S P) $ P
```

This example identifies that DTMF pulse type inband digit signaling is set for the agent, with a permanent seize timer of ten seconds, a partial dial timer of 15 seconds, and a minimum collected partial dial timer of four seconds. The digit mask for DTMF digits includes 0–9, A, asterisk, and octothorpe digits. An octothorpe digit may be received as the terminating digit.

SIG collectable restrictions and limitations

The following restrictions and limitations apply to the SIG collectable:

- For PTS interface type AXXESS agencies, the TRKSIG table defines the initial digit collection signaling characteristics that are used. The SIG protocol collectable alters the digit collection signaling characteristics during the interaction with the originating agency.
- For CCS7 interface type AXXESS agencies, the SIG protocol collectable is required when inband cut-through digit collection is performed.

  Because of the nature of SS7 messaging, all information required to setup the call is typically contained in the initial address message (IAM). It is only for certain cases that cut-through digit collection needs to be performed on an ISUP agent. In this case, the SIG collectable is required to define the inband digit collection signaling characteristics when cut-through digit collection needs to be performed. All requests for digits are normally fulfilled by acquiring the needed digits from the SS7 IAM. If an inband request for digits is performed without the previous execution of a SIG protocol collectable to specify the inband digit collection parameters, a failure occurs on the digit collectable attempting the request for digits.

RCVSIG protocol collectable

Note: This collectable is currently not supported. Generally, interactions with the originating agent require signals to be sent rather than received. If an off-hook type of signal is received, call processing will be initiated. If an on-hook (release) type of signal is received, the call will be torn down. However, the capability to define Receive Signal interactions may eventually exist, so information on the collectable is included.

The Receive Signal (RCVSIG) protocol collectable is required for user interaction purposes where it is necessary to adhere to a certain protocol consisting of winks and hook changes in order to complete the interaction dialog with the originating agent.

The RCVSIG protocol collectable receives an inband wink, answer, or release signal from the originating agent and report within a timeout value. If the signal is not received within the time specified, a failed report is sent and either treatment can be applied to the call or the failed report can be ignored. The type of wink scanned for is opposite of the current hook state.

If the facility is on-hook, then an off-hook wink is scanned. If the facility is off-hook, then an on-hook wink is scanned.

The RCVSIG protocol collectable only provides application for inband signaling agent types. This collectable provides no functionality for common channel signaling agent types.

Definition

Table 2-4 contains the RCVSIG protocol collectable field refinements.

Table 2-4
RCVSIG protocol collectable fields

Fields	Description	Values	Default
SIGTYPE	This field identifies the type of inband signal that is scanned for.	WINK, ANSWER, RELEASE	N/A
FLTRTMR	If the SIGTYPE is set to ANSWER or RELEASE, then this field is present and identifies the amount of time the hook change signal must remain constant before the signal is received.	1 to 255 in 10 ms increments (10 ms to 2.55 seconds)	N/A
TIMEOUT	Timeout value for wink reception	1 to 30 seconds	N/A
OPTIONS	This field contains a vector of options that may be added to the RCVSIG collectable.	NIL or SETTRMT See Table 2-5 for description of SETTRMT option field values.	N/A

Table 2-5 contains the RCVSIG protocol collectable Set Treatment (SETTRMT) option.

Table 2-5
RCVSIG protocol collectable SETTRMT option

Fields	Description	Values	Default
TRMT	If the SETTRMT option is present, and a timeout occurs during execution of the RCVSIG collectable, then the call routes to the treatment specified.	Extended treatment type range This type outlines switch defined treatments provisionable in table TMTCNTL. ***Note:*** See *UCS DMS-250 Data Schema Reference Manual* for values of the extended treatment type.	N/A

Examples

Examples of RCVSIG protocol collectable use:

```
RCVSIG WINK 5 (SETTRMT RODR)$
```

This example identifies that a wink must be received within five seconds, or reorder treatment is applied to the call.

```
RCVSIG RELEASE 80 5 (SETTRMT RODR) $
```

This example identifies that an release (on-hook) signal must be received within five seconds or the call receives reorder treatment. The on-hook signal itself must stay constant for 80 ms before the terminal controller identifies it as a valid hook state change and reports the received signal to the CM.

RCVSIG collectable restrictions and limitations

This collectable is currently not supported, but may be implemented in a future release.

SNDSIG protocol collectable

The Send Signal (SNDSIG) protocol collectable is required for user interaction purposes when it is necessary to adhere to a certain protocol consisting of winks and hook changes in order to complete the interaction dialog with the originating agent.

The SNDSIG protocol collectable sends an inband wink, or an alerting, answer, or release signal to the originating agent. The alerting, answer, or release signal may be sent inband or out-of-band. The type of wink sent is the opposite of the current hook state. If the facility is on-hook, then an off-hook wink is sent. If the facility is off-hook, then an on-hook wink is sent.

Inband signals may be sent only on inband signaling (PTS) agents. Out-of-band signals may be sent only on common channel signaling (CCS) agents.

Definition

Table 2-6 contains the SNDSIG protocol collectable field refinements.

Table 2-6
SNDSIG protocol collectable fields

Fields	Description	Values	Default
SIGTYPE	This field identifies the type of signal that is sent.	WINK, ALERTING, ANSWER, RELEASE	N/A
PRETMR	If the SIGTYPE is set to WINK, then this field is present and identifies the pre-wink time. This is the amount of time that is allowed to elapse before the leading edge of the wink pulse.	1 to 255 in 10ms ticks (10 ms to 2.55 seconds).	N/A
DURTMR	If the SIGTYPE is set to WINK, then this field is present and identifies the duration of wink time. This is the amount of time that the peripheral goes off-hook for the wink.	1 to 255 in 10ms ticks (10 ms to 2.55 seconds).	N/A
GRDTMR	If the SIGTYPE is set to WINK, then this field is present and identifies the guard time. This is the amount of time following the off-hook wink before the peripheral begins processing another task.	1 to 255 in 10ms ticks (10 ms to 2.55 seconds).	N/A

Examples

The following are examples of SNDSIG protocol collectable use:

```
SNDSIG WINK 8 30 8
```

Only valid on inband (PTS) type agents, this example sends a wink signal of duration 300 ms after a guard time of 80 ms. After the wink, the terminal controller pauses 80 ms before reporting completion to the CM.

```
SNDSIG ALERTING
```

In this example, an alerting signal is sent (audible ringing for PTS agents, ACM for CCS7 agents).

```
SNDSIG ANSWER
```

In this example, an answer (off-hook) signal is sent.

SNDSIG collectable restrictions and limitations

There are no identified table control restrictions for the SNDSIG collectable.

FGDPARM protocol collectable

The FGDPARM collectable executes the SS7 FGD protocol within one collectable. The parameter digits received in a message-based signaling protocol (for example, the parameter digits within the IAM message) are extracted in sequence and placed in the digit buffer for processing by the FGDPARM collectable.

The FGDPARM collectable reuses the functionality of the existing SS7 digit collectables, such as OLIPARM, SUBRPARM, and ADDRPARM. For example, if the message ADDR N Y PRTNM AXX is in table MSGCTR, the ADDRPARM functionality of FGDPARM consumes and processes the message.

The FGDPARM collectable is designed to be stand-alone to process the SS7 FGD protocol, but it may be provisioned with other collectables.

Fields

Table 2-7 contains the FGDPARM collectable field refinements.

Table 2-7
FGDPARM protocol collectable fields

Fields	Description	Values	Default
PROCESS_CIP	If this field is set to Y, the switch looks for and processes the CIP parameter, if the parameter is present. If this field is set to N, the switch does not look for the CIP parameter and FGDPARM skips to the OPTOLI field.	Y, N	N
SET_INTL_TRANS	If this field is set to Y, after the switch processes the TNS CIC digits, the switch sets IN or IP translations. If this field is set to N, the switch does not set IN or IP translations.	Y, N	N
OPTOLI	If this field is set to Y, the OLI digits are optionally received. If this field is set to N, the OLI digits must be received to avoid ADBF treatment.	Y, N	N
OPTANI	If this field is set to Y, the ANI digits are optionally received. If this field is set to N, the ANI digits must be received to avoid ADBF treatment.	Y, N	N
IDPRTNM	This field specifies the pretranslator name for FGDPARM's OLIPARM processing.	pretranslator_name	NPRT
PRTNM	This field specifies the pretranslator name for FGDPARM's SUBRPARM processing.	pretranslator_name	NPRT
CIC_FLEXTYPE	This field specifies the FLEXTYPE index to use for CIC validation in table FLEXVAL.	An existing entry in table FLEXTYPE	
CIC_UNIQUE_IDX	This field specifies the unique index to use for CIC validation in table FLEXVAL.	0 to 1,047,999	
ANI_FLEXTYPE	This field specifies the FLEXTYPE index to use for ANI validation in table FLEXVAL.	An existing entry in table FLEXTYPE	

—continued—

Table 2-7
FGDPARM protocol collectable fields (continued)

Fields	Description	Values	Default
ANI_UNIQUE_IDX	This field specifies the unique index to use for ANI validation in table FLEXVAL.	0 to 1,047,999	
CLGPTYNO_ FLEXTYPE	This field specifies the FLEXTYPE index to use for CLGPTYNO CDR capture.	An existing entry in table FLEXTYPE	
CUTRUIDX	This field identifies the FLEXDIAL table index used to process the remainder of the call when FGD cut-thru call is determined.	An existing entry in table FLEXDIAL ***Note:*** This entry must exist in table FLEXDIAL before you enter it in this field.	N/A
CUTRUIDX_ ACTION	This field indicates where to place the cutruidx in the collectable list.	INSERT, APPEND, REPLACE, EXEC INSERT inserts the new list into the current processing list before the next collectable. APPEND appends the new list to the current processing list. REPLACE replaces the remainder of the unprocessed collectables with those identified by the FLEXDIAL index. EXEC executes as a sublist the list identified by the FLEXDIAL table index.	N/A

—continued—

Table 2-7
FGDPARM protocol collectable fields (continued)

Fields	Description	Values	Default
TRANSIDX	This field identifies the FLEXDIAL table index used to process the remainder of the call when FGD transitional call is determined.	An existing entry in table FLEXDIAL ***Note:*** This entry must exist in table FLEXDIAL before you enter it in this field.	N/A
TRANSIDX_ ACTION	This field indicates where to place the transidx in the collectable list.	INSERT, APPEND, REPLACE, EXEC INSERT inserts the new list into the current processing list before the next collectable. APPEND appends the new list to the current processing list. REPLACE replaces the remainder of the unprocessed collectables with those identified by the FLEXDIAL index. EXEC executes as a sublist the list identified by the FLEXDIAL table index.	N/A
FGDPARM_ OPTION	This field is a vector of options reserved for future use.	NIL	N/A

—end—

Example

Example of FGDPARM collectable use:

```
FGD_SS7_INDEX (FGDPARM Y N Y N NPRT NPRT CIC 1 ANI 1 CLGPTY
FLEXDIAL_CUTTHRU_INDEX INSERT FLEXDIAL_TRANS_INDEX INSERT
NIL) $ N

FLEXDIAL_CUTTHRU_INDEX (APTRMT) (INCLUDE SIG_DTMF) (GOTO
MNOA_AUV_SD_AD) $ N

FLEXDIAL_TRANS_INDEX (APTRMT) (INCLUDE SIG_DTMF) (IFNOA IS
CALLED (950_CT) $ REPLACE MNOA_AUV_SD_AD NIL) (VAROP AVAR ASG
INTEGER 1) $ Y AD_SD
```

Error messages

If the entry in the CUTRUIDX field does not exist in table FLEXDIAL, the UCS DMS-250 switch displays the following error message:

```
***ERROR***
<entry name>

TYPE OF CUTHRUIDX IS FLEXDIAL_INDEX_TYPE
CUTTHRUIDX:
```

If the entry in the TRANSIDX field does not exist in table FLEXDIAL, the UCS DMS-250 switch displays the following error message:

```
***ERROR***
<entry name>

TYPE OF TRANSIDX IS FLEXDIAL_INDEX_TYPE
TRANSIDX:
```

TERMINATE framework collectable

For the framework to handle call scenarios that require the call to complete to a terminating agent before the interaction with the originating agent has been completed, the TERMINATE collectable provides a way to complete the call to the identified terminating agency. When the dial plan call processing framework resumes processing of the provisioned collectables, it resumes with the collectable following the TERMINATE collectable.

The TERMINATE collectable is ideally used when necessary protocol signals need to be sent to the originating agent after call setup with the terminating agent has been completed. The FGD protocol equal access acknowledgement wink is the prime example.

Definition

Table 2-8 contains the TERMINATE collectable field refinements.

Table 2-8
TERMINATE framework collectable fields

Fields	Description	Values	Default
PRCDIR	The processing direction indicates whether the remaining unprocessed collectables should be processed upon completion of the TERMINATE collectable and the immediate return to the FlexDial call processing framework.	CONTINUE or STOP CONTINUE indicates that after seizure of and connection to the identified terminating agency occurs, remaining collectables are processed. STOP indicates that after seizure of and connection to the identified terminating agency occurs, the call is suspended and remaining collectables are not processed.	N/A
OPTIONS	This field contains a vector of options that may be added to the TERMINATE collectable.	NIL or NOTIFY The NIL option is described below. See Table 2-9 for a description of the NOTIFY option field values.	N/A

NIL option

The NIL option provides an easy way to update the options vector. Instead of removing an option by retyping the provisioned options that follow, NIL can be inserted. Using NIL enables the inserted option to be effectively removed without having to retype the remaining provisioned options.

The NIL option does not remain visible once the update is complete.

An example showing the use of NIL option is:

1 Editing existing tuple:

```
OPTIONS: (option1) (option2) (option3)$
```

2 To remove option2, change the table entry. When prompted at option2, enter NIL.

```
OPTIONS: option2
> NIL
```

3 After editing, display of the tuple shows:

```
OPTIONS: (option1) (option3)$
```

NOTIFY option

The NOTIFY option identifies a tone or announcement that is played before the TERMINATE option is executed (before the call is completed to the terminating agency). This is similar to prefixing the route list with the identified tone or announcement CLLI. The tone or announcement is played in its entirety before the call completes to the terminating agency, and is not interruptible by the user.

Table 2-9 describes the TERMINATE collectable NOTIFY option.

Table 2-9
TERMINATE collectable NOTIFY option

Field	Description	Values	Default
CLLI	This field identifies the actual tone or announcement that is played.	TONE or ANNOUNCEMENT CLLI TONE CLLIs are provisioned in table TONES. Announcement CLLIs are provisioned in table ANNS.	N/A

Examples

Examples of TERMINATE collectable use:

```
TERMINATE STOP $
```

In this example, the call terminates to treatment or the agent identified by translations, pre-determined by validation screening, or by the ROUTE collectable. Call processing does not process additional collectables after the call is set up.

```
TERMINATE CONTINUE (NOTIFY MYANNC)$
```

In this example, the originator receives the announcement message identified by the MYANNC CLLI, and then the call terminates to treatment or the agent identified by translations, pre-determined by validation screening, or by the ROUTE collectable. After the call has terminated, remaining collectables in the list are processed (with the exception of collectables requiring digit collection requests from the agent terminal controller).

TERMINATE collectable restrictions and limitations

There are no identified table control restrictions for the TERMINATE collectable.

NOTIFY framework collectable

The NOTIFY collectable provides the ability to play an uninterruptable tone or announcement during the interaction with the originating agent. Prior to UCS08, the only mechanism for playing an uninterruptable announcement within FlexDial call processing was through the TERMINATE collectable NOTIFY option.

Purpose

The NOTIFY collectable provides the ability to play an uninterruptable tone or announcement during the interaction with the originating agent. The tone or announcement notification may be played at any point during the interaction.

Fields

Table 2-10 describes the NOTIFY collectable fields.

Table 2-10
NOTIFY collectable fields

Field	Description	Values	Default
USER	This field identifies the agent in the call that is the recipient of the tone or announcement notification.	ORIGINATOR	N/A
CLLI	This field identifies the tone or announcement that is played.	TONE or ANNOUNCEMENT CLLI TONE CLLIs are provisioned in table TONES. Announcement CLLIs are provisioned in table ANNS.	N/A

Example

Examples of NOTIFY collectable use are:

```
NOTIFY ORIGINATOR FLEXANNC1
```

In this example, the originator receives the announcement message identified by the FLEXANNC1 CLLI. After the announcement message has completed, the next identified collectable is executed.

Provisioning Order

The TONE or ANNC CLLI must be provisioned in table TONES or ANNS before it can be provisioned in the NOTIFY collectable.

Restrictions and limitations

The following provisioning restrictions or limitations exist for the NOTIFY collectable.

- The TONE or ANNC CLLI must be provisioned in table TONES or ANNS before it can be provisioned in the NOTIFY collectable. If an attempt to provision a CLLI other than a TONE or ANNC CLLI is performed, then the following error message is displayed:

```
An announcement (table ANNS), or a tone (table TONES) must be
provisioned as the identified CLLI.
```

TIMER framework collectable

The TIMER collectable provides an additional triggering mechanism to initiate FlexDial collectable execution for the originating agent. The triggering mechanism is based upon the expiration of the identified FlexDial timer, initiating execution of the collectable list identified by the FLEXDIAL table index.

Up to three individual FlexDial timers may be activated through use of the TIMER collectable. Additionally, when a FlexDial timer expires, full bearer channel capabilities are enabled for FlexDial collectable processing.

Fields

Table 2-11 describes the TIMER collectable fields.

Table 2-11
TIMER collectable fields

Field	Description	Values	Default
TIMERID	This field identifies which of the three available FlexDial timers is to be activated.	TIMER1, TIMER2, TIMER3	N/A
TIMERVAL	This field identifies whether the time limit value is to be specified as an integer value or taken from a FlexDial variable.	INTEGER, FLEXVAR	N/A

—continued—

Table 2-11
TIMER collectable fields (continued)

Field	Description	Values	Default
VALUE	If the TIMERVAL field is set to integer, then this field is present and identifies the specific time limit.	1 to 3600	N/A
FLEXVAR	If the TIMERVAL field is set to FLEXVAR, then this field is present and identifies the FlexDial variable which contains the time limit value.	AVAR, BVAR, CVAR, DVAR, FEATVAR	N/A
FLEXTYPE	If the FLEXVAR field is present and set to FEATVAR, then this field is also present and specifies the FLEXTYPE for proper identification of the FLEXFEAT variable being used to contain the time limit value.	A valid index to table FLEXTYPE	N/A
UNITS	This field identifies the units of the time limit value used.	SECONDS, MINUTES	N/A
DPIDX	This field identifies the FLEXDIAL table index that is used upon expiration of the FlexDial timer.	A valid index to table FLEXDIAL	N/A

—end—

Example

Examples of TIMER collectable use are:

```
TIMER TIMER1 INTEGER 40 SECONDS EXPR_IDX
```

In this example, the FlexDial timer TIMER1 is enabled with a value of 40 seconds. When the timer expires, the collectable list identified by the EXPR_IDX FLEXDIAL table index is executed for additional interaction with the originating agent.

Restrictions and limitations

No provisioning restrictions or limitations exist for the TIMER collectable.

KILLTMR framework collectable

The KILLTMR (kill timer) collectable provides the ability to disable a FlexDial timer activated through execution of the TIMER collectable.

Fields

Table 2-12 describes the KILLTMR collectable field.

Table 2-12
KILLTMR collectable field

Field	Description	Values	Default
TIMER	This field identifies which of the three available FlexDial timers is to be deactivated.	TIMER1, TIMER2, TIMER3	N/A

Example

Examples of KILLTMR collectable use are:

```
KILLTMR TIMER1
```

In this example, the FlexDial timer TIMER1 is disabled.

Restrictions and limitations

No provisioning restrictions or limitations exist for the KILLTMR collectable.

CALLCOND framework collectable

The CALLCOND (call condition) collectable provides an additional triggering mechanism to initiate FlexDial collectable execution for the originating agent. The triggering mechanism is based upon occurrence of the specified call condition for the call, initiating execution of the collectable list identified by the FLEXDIAL table index.

Currently, only two call conditions are supported by the CALLCOND collectable, answer and disconnect.

Fields

Table 2-13 describes the CALLCOND collectable fields.

Table 2-13
CALLCOND collectable fields

Field	Description	Values	Default
CONDITION	This field identifies the call event that when processed for the call triggers FlexDial collectable execution.	ANSWER, DISCONNECT	N/A
DPIDX	This field identifies the FLEXDIAL table index that is used upon occurrence of the condition.	A valid index to table FLEXDIAL	N/A

Example

Examples of CALLCOND collectable use are:

```
CALLCOND ANSWER ANSW_IDX
```

In this example, the collectable list identified by the ANSW_IDX FLEXDIAL table index is executed when answer occurs for the call.

Provisioning Order

The FLEXDIAL table index used for call condition triggering must be defined before use within the CALLCOND collectable.

Restrictions and limitations

No provisioning restrictions or limitations exist for the CALLCOND collectable.

GOTO framework collectable

To provide enough dynamic directional control over the execution of collectables defined for a particular user interaction, and because of the physical limitations of the data dictionary interface for the FLEXDIAL table, the GOTO framework collectable allows the execution of collectables to branch to a separate FLEXDIAL index. The data dictionary restrictions allow only a set number of collectables to be provisioned at each index. To allow the capacity for a non-fixed number of collectables, the GOTO framework collectable allows the thread of execution for a defined interaction to branch and continue processing.

A collectable list is a group of collectables identified by a FLEXDIAL index. Call processing handles this list through the Setup Information entity, and processes each collectable in the list sequentially. The GOTO collectable provides the expansion for the current list of collectables being processed.

Definition

Table 2-14 contains the GOTO framework collectable field refinement.

Table 2-14
GOTO framework collectable field

Field	Description	Values	Default
INDEX	The index field identifies the FLEXDIAL table index to be processed next. All remaining unprocessed collectables in the current list do not get processed for the call.	An existing entry to table FLEXDIAL	N/A

Examples

An example of GOTO framework collectable use:

```
GOTO PLAIN_FGD_ADDR
```

In this example, processing continues with the collectable list identified by index PLAIN_FGD_ADDR.

GOTO collectable restrictions and limitations

There are no identified table control restrictions for the GOTO framework collectable.

DO framework collectable

The DO framework collectable also allows the execution of collectables to branch to a separate FLEXDIAL index. The data dictionary restrictions allow only a set number of collectables to be provisioned at each index. To allow the capacity for a non-fixed number of collectables, the DO framework collectable allows the branch of execution for a defined interaction to branch and continue processing.

A collectable list is a group of collectables identified by a FLEXDIAL index. Call processing handles this list through the Setup Information entity, and processes each collectable in the list sequentially. The DO collectable provides the means to branch processing to a new list before processing is continued in the current list.

Definition

Table 2-15 contains the DO framework collectable field refinement.

Table 2-15
DO framework collectable field

Field	Description	Values	Default
INDEX	The index field identifies the FLEXDIAL table index to be processed before processing the next collectable in the list. Once the thread of collectable processing has ended from the branch by the DO collectable, the remaining unprocessed collectables in the original list are processed.	An existing entry to table FLEXDIAL	N/A

Examples

Examples of DO framework collectable use:

```
DO ADDR
```

In this example, processing branches to the collectable list identified by index ADDR. After that thread of collectables has been processed, control returns to the task of processing the remainder of the original collectable list.

```
DO PLAIN_FGD_ADDR
```

In this example, processing branches to the collectable list identified by index PLAIN_FGD_ADDR. After that thread of collectables has been processed, control returns to the task of processing the remainder of the original collectable list.

DO collectable restrictions and limitations

There are no identified table control restrictions for the DO framework collectable.

INCLUDE framework collectable

The INCLUDE framework collectable allows a list of collectables represented by the FLEXDIAL index to be included in the currently processing collectable list. The INCLUDE collectable is a place marker for all of the collectables represented by the FLEXDIAL index, and the included collectables replace the INCLUDE collectable in the current collectable list.

A collectable list is a group of collectables identified by a FLEXDIAL index. Call processing handles this list through a collectable manager, and processes each collectable in the list sequentially. The INCLUDE collectable provides the means to define subroutine-like areas of collectable processing, yet still retain the notion of executing a single collectable list.

Note: This notion is important for areas such as collectable list manipulation. For example, if the following collectable list was being processed: (DO ADDR) (DO ACCT), execution of the DO ADDR branches off into another collectable list identified by the FLEXDIAL index. If processing of the list causes a DPIDX REPLACE action to occur on the collectable list, then only the secondary list started by the DO collectable is affected. The (DO ACCT) from the initial list is not affected and is executed when the branch of execution identified by the ADDR "subroutine" is complete. In this example, using the INCLUDE collectable, the ADDR "subroutine" is included in the current list along with the (INCLUDE ACCT) collectable. Therefore any list manipulation such as DPIDX REPLACE affects the INCLUDE ACCT collectable.

Definition

Table 2-16 contains the INCLUDE framework collectable field refinement.

Table 2-16
INCLUDE framework collectable field

Field	Description	Values	Default
INDEX	The index field identifies a FLEXDIAL table index. The collectable list provisioned against the FLEXDIAL list is included in the currently processing list following the INCLUDE collectable.	An existing entry to table FLEXDIAL	N/A

Examples

An example of INCLUDE framework collectable use:

```
INCLUDE ADDR
```

In this example, the collectable list identified by index ADDR is included in the currently processing collectable list.

INCLUDE collectable restrictions and limitations

There are no identified table control restrictions for the INCLUDE framework collectable.

APTRMT framework collectable

The Apply Treatment (APTRMT) is a special case of the TERMINATE instruction. If a call has a treatment set when APTRMT is executed, the call is setup to the treatment and no further collectables are processed. If the call does not have a treatment set, then processing continues with the next collectable.

Definition

The APTRMT option does not contain any specific field refinements.

Examples

An example of the APRTMT Framework collectable is:

```
APTRMT
```

In this example, if a treatment has been identified for the call by previous digit collectable processing, then the interaction with the originating agent is terminated and the identified treatment is applied to the call.

APTRMT collectable restrictions and limitations

There are no identified table control restrictions for the APTRMT collectable.

ROUTE framework collectable

To provide an enhanced capability of routing the call based on unique trunk group, subscriber number (such as ANI or Authcode) characteristics, or Transaction Capabilities Application Part (TCAP) response information, the ROUTE framework collectable provides a way to directly define a terminating agency indicating the call destination. This framework collectable overrides any previously set route choice.

Definition

Table 2-17 contains the ROUTE framework collectable field refinements.

Table 2-17
ROUTE framework collectable fields

Fields	Description	Values	Default
SELECTOR	The valid route selector fields for the ROUTE framework collectable are the S and T selectors. Each selector has its own unique refinements.	S or T	N/A
CLLI	For the S selector, the CLLI field identifies the CLLI to route the call to.	All available tones, announcements, or trunk CLLIs as provisioned in table CLLI	N/A

EXTRTEID	For the T selector, the external route ID identifies an office route or TOPS route.	{OFRT, EXDGTRTE, TOPSAMA, TOPS, TTL4, RRTE, OFR4, OFR3, OFR2} each with a defined key value.
		TOPSAMA and TOPS key value is CALL_ORIGINATION
		EXDGTRTE key value is {0 to 1023}
		RRTE key value is {0 to 1023}
		TTL4 key value is {0 to 7}
		OFRT key value is {0 to 1023}
		OFR2 key value is {0 to 1023}
		OFR3 key value is {0 to 1023}
		OFR4 key value is {0 to 1023}
			N/A

Examples

Examples of ROUTE framework collectable use:

```
ROUTE S IMTCLLI
```

In this example, a route to the IMTCLLI trunk group is defined for the call.

```
ROUTE T OFRT 5
```

In this example, the route list defined by the OFRT 5 index is set up for the call.

ROUTE collectable restrictions and limitations

The CLLI specified must be either an announcement, tone, or trunk group CLLI. An attempt to provision another type of CLLI results in the following error message:

```
An announcement (table ANNS), a tone (table TONES), or a
trunk (Table TRKGRP) must be provisioned as the identified
CLLI.
```

NOOP framework collectable

The No Operation (NOOP) collectable identifies no operation instruction. This collectable exists in FLEXDIAL table control, but provides no call processing functionality. NOOP can be used as a placeholder for a defined index, or provides a function similar to the NIL collectable.

The NOOP collectable provides a way to create FLEXDIAL table entries, but not immediately complete the collectable provisioning for the index. When an index is created, NOOP can be used for the option list. References to the index can then be added as required, and the option list updated at a later date.

The NOOP collectable also provides a way to remove collectables within the options vector. Instead of removing a collectable by retyping the provisioned collectables that follow, NOOP can be inserted. Using NOOP enables the inserted collectable to be effectively removed without having to retype the remaining provisioned collectables.

A NOOP collectable remains visible in the table entry.

Definition

The NOOP collectable does not contain any specific field refinements.

NOOP collectable restrictions and limitations

The NOOP collectable is used in the predefined NIL FLEXDIAL table index.

IFTRMT framework collectable

The IFTRMT collectable performs a test to determine if a specific treatment has been set. If a specific treatment has been set, the collectables at the corresponding index into the table FLEXDIAL replace the remaining portion of the current list.

Definition

Table 2-18 contains the IFTRMT framework collectable field refinements.

Table 2-18
IFTRMT framework collectable fields

Field	Description	Values	Default
COMPMETH	The index may be executed whether the above condition is TRUE or FALSE.	IS or NOT	N/A
TRTMTS	A vector that includes up to four treatment types.	Vector of up to 4 extended treatments.	N/A
ACTION	An enumeration of the actions to be taken on either the TRUE branch or the ELSE branch.	INSERT, APPEND, REPLACE, EXEC INSERT inserts the new list into the current processing list before the next collectable. APPEND appends the new list to the current processing list. REPLACE replaces the remainder of the unprocessed collectables with those identified by the FLEXDIAL index. EXEC executes as a sublist the list identified by the FLEXDIAL table index.	N/A
IFTRUE	The INDEX field identifies the FLEXDIAL table index that is to be processed if the IFTRMT comparison is successful.	An existing entry to table FLEXDIAL.	N/A
IFFALSE	The INDEX field identifies the FLEXDIAL table index that is to be processed if the IFTRMT comparison is FALSE.	An existing entry to table FLEXDIAL.	N/A

Examples

Examples of the IFTRMT collectable are shown below:

```
(IFTRMT NOT (VACT) $ EXEC DO_THIS ELSE_THIS)
```

This example tests against the vacant treatment code. If the treatment is not VACT, then the collectables at DO_THIS are executed, and the ELSE_THIS index collectable is ignored. If the treatment is VACT, then the collectables at DO_THIS are ignored, and the remainder of the list is executed.

```
(IFTRMT IS (UNDT) $ EXEC DO_THIS ERROR_PROCESSING)
```

This example tests if any treatment has been set.

IFTRMT collectable restrictions and limitations

The index must exist within table FLEXDIAL.

IFPRMT framework collectable

The IFPRMT collectable branches within the collectable list, based on the last issued prompt.

Definition

Table 2-19 contains the IFPRMT framework collectable field refinements.

Table 2-19
IFPRMT framework collectable fields

Fields	Description	Values	Default
COMPMETH	Determines whether the test case passes on FAIL or SUCCESS.	IS or NOT	N/A
CONDITION	A vector that includes up to 4 prompt types.	Vector of up to 4 tone values of {PMATCH, CLLI, TONETYPE, STDTONE, TONEDUR}	N/A
—continued—			

Table 2-19
IFPRMT framework collectable fields (continued)

Fields	Description	Values	Default
ACTION	An enumeration of the actions to be taken on either the TRUE branch or the ELSE branch.	INSERT, APPEND, REPLACE, EXEC INSERT inserts the new list into the current processing list before the next collectable. APPEND appends the new list to the current processing list. REPLACE replaces the remainder of the unprocessed collectables with those identified by the FLEXDIAL index. EXEC executes as a sublist the list identified by the FLEXDIAL table index.	N/A
IFTRUE	The INDEX field identifies the FLEXDIAL table index that is to be processed if the IFPRMT comparison is successful. If successful, all remaining unprocessed collectables in the current list do not get processed for the call.	An existing entry to table FLEXDIAL.	N/A
IFFALSE	The INDEX field identifies the FLEXDIAL table index that is to be processed if the IFPRMT comparison is FALSE.	An existing entry to table FLEXDIAL.	N/A
—end—			

Table 2-20 contains the IFPRMT collectable CONDITION field values.

Table 2-20
IFPRMT framework collectable condition fields

Field	Description	Value	Default
PMATCH	Identifies the logic used to match on the previous prompt. The previous prompt can be either an announcement, tone, no prompt, or any prompt.	ANY, NOPROMPT, ANNC, TONE	N/A
CLLI	This field identifies the tone or announcement prompt that is played for the user as a request for digits indication. This field is visible if one of the following conditions is met: The PMATCH is set to ANNC. The PMATCH is set to TONE and the TONETYPE is set to TONES.	TONE or ANNOUNCEMENT CLLI Announcement CLLIs are provisioned in table ANNS. TONE CLLIs are provisioned in table TONES.	N/A
TONETYPE	This field must include the tone type if PMATCH is TONE.	TONES, STD	N/A

—continued—

Table 2-20
IFPRMT framework collectable condition fields (continued)

Field	Description	Value	Default
STDTONE	If the PMATCH type is set to TONE and the TONETYPE is set to STD, then this field is present and identifies the type of tone that is used.	CD, SD, SSD, H, L, 0–9, A, B, C, D, S, P CD identifies carrier dial tone (400 Hz). SD identifies standard dial tone (350 + 440 Hz). SSD identifies a tone consisting of a short burst of stuttered dial tone, followed by standard dial tone. H identifies a high tone (480 Hz). L identifies a low tone (480 + 620 Hz). 0 through 9 represent the associated DTMF digit tones. S refers to the DTMF asterisk digit. P refers to the DTMF octothorpe digit. ***Note:*** STDTONE values are pending a resolution as to allowable MF digits.	N/A
TONEDUR	If the PMATCH type is set to TONE and the TONETYPE is set to STD, then this field is present and identifies the duration of the tone that is being played.	{0 to 255} in ten millisecond ticks. The value of 255 indicates that a continuous tone is to be applied. The value of 0 indicates that any tone duration is to be matched.	N/A

—end—

Examples

Examples of the IFPRMT collectable are shown below:

```
(IFPRMT IS (ANY) $ EXEC DO_THIS ELSE_THIS)
```

This example tests true if any type of prompts are issued.

```
(IFPRMT IS (NOPROMPT) $ INSERT DO_THIS ELSE_THIS)
```

This example demonstrates how to detect if no prompt was applied.

```
(IFPRMT IS (ANNC OPS_1) $ EXEC DO_THIS ELSE_THIS)
```

This example checks to determine if the prompt is ANNC_OPS_1.

```
(IFPRMT IS (TONE STD P 10) $ EXEC DO_THIS ELSE_THIS)
```

This example checks to determine if the prompt is a standard tone ”P” lasting 100 milliseconds.

For all cases, if the test is true, DO_THIS is executed and ELSE_THIS is ignored. If the test is false, DO_THIS is ignored, and ELSE_THIS is executed.

IFPRMT collectable restrictions and limitations

The indexes must exist within tables CLLI, TONES, ANNS, and FLEXDIAL, as appropriate.

IFTOD framework collectable

The IFTOD collectable performs dynamic branches within the collectable list, based on the current time. Because this collectable accepts a range of times, the dial plan can behave differently based on either current time of day or day of the week.

Definition

Table 2-21 contains the IFTOD framework collectable field refinements.

Table 2-21
IFTOD framework collectable fields

Field	Description	Values	Default
COMPMETH	Determines whether the test case passes on FAIL or SUCCESS.	IS or NOT	N/A
CONDITION	A vector that includes up to 4 time range boundaries. The index is executed if the current time is within any of these time ranges	Vector of up to 4 time range values of {LDAY, LTIMEHRS, LTIMEMIN, UDAY, UTIMEHRS, UTIMEMIN}	N/A
ACTION	An enumeration of the actions to be taken on either the TRUE branch or the ELSE branch.	INSERT, APPEND, REPLACE, EXEC INSERT inserts the new list into the current processing list before the next collectable. APPEND appends the new list to the current processing list. REPLACE replaces the remainder of the unprocessed collectables with those identified by the FLEXDIAL index. EXEC executes as a sublist the list identified by the FLEXDIAL table index.	N/A
IFTRUE	The INDEX field identifies the FLEXDIAL table index that is to be processed if the IFTOD comparison is successful.	An existing entry to table FLEXDIAL.	N/A
IFFALSE	The INDEX field identifies the FLEXDIAL table index that is to be processed if the IFTOD comparison is FALSE.	An existing entry to table FLEXDIAL.	N/A

Table 2-22 contains the IFTOD collectable CONDITION field values.

Table 2-22
IFTOD framework collectable condition fields

Field	Description	Value	Default
LDAY	An enumeration for the day of week which is the lower boundary for the test. The keyword ANY indicates ANY day of the week.	SUN, MON, TUE, WED, THU, FRI, SAT, ANY	N/A
LTIMEHRS	Hours for the lower boundary of the test. This is a 24–hour clock with 00:00 equal to 12AM.	[0–23]	N/A
LTIMEMIN	Minutes for the lower boundary.	[0–59]	N/A
UDAY	Enumeration for the day of week which is the upper boundary for the test. The keyword ANY indicates ANY day of the week.	SUN, MON, TUE, WED, THU, FRI, SAT, ANY	N/A
UTIMEHRS	Hours for the upper boundary of the test. This is a 24–hour clock with 00:00 equal to 12AM.	[0–23]	N/A
UTIMEMIN	Minutes for the upper boundary of the test.	[0–59]	N/A

IFDIGS sequence collectable

The If Digits (IFDIGS) sequence collectable allows the execution of collectables to branch to a separate FLEXDIAL index based on a successful comparison against incoming digits. This collectable does not request digits from the originating agent, but performs the comparison against what is currently in the digit buffer. (Therefore, the COLDIG or COLPARM collectable would likely precede use of the IFDIGS collectable when collecting raw digit information for the comparison.) This collectable does not remove digits from the digit buffer.

A collectable list is a group of collectables identified by a FLEXDIAL index. Call processing handles this list through the Setup Information entity, and processes each collectable in the list sequentially. The IFDIGS collectable provides the means to conditionally branch to a new list to continue collectable processing based on the incoming digits. The remainder of the current list is not processed.

Definition

Table 2-23 contains the IFDIGS framework collectable field refinements.

Table 2-23
IFDIGS framework collectable fields

Field	Description	Value	Default
COMPMETH	This field identifies the comparison method used for the digits comparison.	ARE or NOT ARE means that if the digits being compared match any of the patterns provisioned in the DIGITS vector, then branching occurs. NOT means that if the digits being compared do not match any of the patterns provisioned in the DIGITS vector, then branching occurs.	N/A
DIGVEC	This field identifies the digit patterns to compare against. Up to four patterns may be specified.	Vector of up to 4 of {vector of up to 15 of {0 -9, A, B, C, D, S, P, N, W, X, Z}} The A, B, C, D digits represent the 4th column DTMF digits. The S identifies an asterisk digit, and the P identifies an octothorpe digit. This digit vector also supports N, which represents digits 2 to 9; W, which represents digits 7 to 9; X, which represents digits 0 to 9; and Z, which represents any digit.	N/A
INDEX	The index field identifies the FLEXDIAL table index to be processed if the IFDIGS comparison is successful. All remaining unprocessed collectables in the current list do not get processed for the call.	An existing entry to table FLEXDIAL	N/A

—continued—

Table 2-23
IFDIGS framework collectable fields (continued)

Field	Description	Value	Default
ACTION	An enumeration of the actions to be taken on either the TRUE branch or the ELSE branch.	INSERT, APPEND, REPLACE, EXEC INSERT inserts the new list into the current processing list before the next collectable. APPEND appends the new list to the current processing list. REPLACE replaces the remainder of the unprocessed collectables with those identified by the FLEXDIAL index. EXEC executes as a sublist the list identified by the FLEXDIAL table index.	N/A
IFTRUE	The INDEX field is the index to branch in the true case.	An existing FLEXDIAL INDEX field	N/A
IFFALSE	The INDEX field identifies the FLEXDIAL table index that is to be processed if the IFDIGS comparison is FALSE.	An existing entry to table FLEXDIAL	NIL
—end—			

Up to four patterns of digits may be identified in the DIGITS vector. When multiple patterns are specified, the logic for the comparisons is as shown in Figure 2-1.

Figure 2-1
Logic for IFDIGS comparison

ARE Comparison Method:

Result = (digits = A) or (digits = B) or (digits = C) or (digits = D)

NOT Comparison Method:

Result = (digits ^= A) and (digits ^= B) and (digits ^= C) and (digits ^= D)

Note: ^= means ‘not equal to’

If the result is true, then branching occurs.

Examples

Examples of IFDIGS sequence collectable use:

```
IFDIGS ARE (1NX)$ REPLACE FGD_INTL_IDX NIL
```

In this example, if the first three available digits in the digit buffer match the 1NX pattern, then collectable processing branches to the list represented by the FGD_INTL_IDX FLEXDIAL index, otherwise collectable processing continues on to the next collectable in the list.

```
IFDIGS ARE (800)(00Y)$ REPLACE N00_IDX NIL
```

In this example, if the first three available digits in the digit buffer are 800 or 00Y, then collectable processing branches to the list represented by the N00_IDX FLEXDIAL index, otherwise collectable processing continues on to the next collectable in the list.

```
IFDIGS ARE (123) $ EXEC THIS ELSE_THIS
```

In this example, if the first three available digits buffer are 123, the collectables at THIS are executed within a sublist. Otherwise, the collectables at ELSE_THIS are executed within a sublist.

IFDIGS collectable restrictions and limitations

The FLEXDIAL index must exist first.

IFCNT sequence collectable

The If Count (IFCNT) sequence collectable allows the execution of collectables to branch to a separate FLEXDIAL index if the number of reported digits is equal to the value identified. This collectable does not request digits from the originating agent, but performs the comparison against what is currently in the digit buffer. (Therefore the COLDIG or COLPARM collectable would likely precede use of the IFCNT collectable in order to collect raw digit information for the comparison.)

A collectable list is a group of collectables identified by a FLEXDIAL index. Call processing handles this list through the Setup Information entity, and processes each collectable in the list sequentially. The IFCNT collectable provides a way to conditionally branch to a new list to continue collectable processing based on the number of incoming digits. The remainder of the current list is not processed.

Definition

Table 2-24 contains the IFCNT sequence collectable field refinements.

Table 2-24
IFCNT sequence collectable fields

Field	Description	Values	Default
COMPMETH	This field identifies the comparison method used for the digits comparison.	IS or NOT	N/A
MIN	This count field identifies the minimum count value used for the comparison. The number of available digits in the digit buffer must be greater than or equal to this number.	0 to 16	N/A
MAX	This count field identifies the maximum count value used for the comparison. The number of available digits in the digit buffer must be less than or equal to this number.	0 to 16	N/A

—continued—

Table 2-24
IFCNT sequence collectable fields (continued)

Field	Description	Values	Default
INDEX	The index field identifies the FLEXDIAL table index to be processed if the IFCNT comparison is successful. All remaining unprocessed collectables in the current list do not get processed for the call.	An existing entry to table FLEXDIAL	N/A
ACTION	An enumeration of the actions to be taken on either the TRUE branch or the ELSE branch.	INSERT, APPEND, REPLACE, EXEC INSERT inserts the new list into the current processing list before the next collectable. APPEND appends the new list to the current processing list. REPLACE replaces the remainder of the unprocessed collectables with those identified by the FLEXDIAL index. EXEC executes as a sublist the list identified by the FLEXDIAL table index.	N/A
IFTRUE	The INDEX field is the index to branch in the true case.	An existing FLEXDIAL INDEX field.	N/A
IFFALSE	The INDEX field identifies the FLEXDIAL table index that is to be processed if the IFCNT comparison is FALSE.	An existing entry to table FLEXDIAL.	N/A

—end—

Examples

Examples of IFCNT sequence collectable use:

```
IFCNT IS 2 4 REPLACE SPEED_ADDR_IDX NIL
```

In this example, if the number of available digits in the digit buffer is greater than or equal to two and less than or equal to four, then collectable processing branches to the list represented by the SPEED_ADDR_IDX FLEXDIAL index, otherwise collectable processing continues on to the next collectable in the list.

```
IFCNT IS 2 5 REPLACE THIS NIL
```

In this example, IFCNT checks the current count of digits. If the current count of digits are between 2 and 5, then the collectables at the THIS replace the remainder of the list.

IFCNT collectable restrictions and limitations

The FLEXDIAL index must exist first.

IFNOA sequence collectable

The If Nature Of Address (IFNOA) sequence collectable allows the execution of collectables to branch to a separate FLEXDIAL index based on a successful comparison against an incoming nature of address (NOA) value.

A collectable list is a group of collectables identified by a FLEXDIAL index. Call processing handles this list through the Setup Information entity, and processes each collectable in the list sequentially. The IFNOA collectable provides a way to conditionally branch to a new list to continue collectable processing. The remainder of the current list is not processed.

For DTMF PTS type agents, no NOA values are received with the digit streams and therefore the NOA values are equal to “unknown.” For MF PTS type agents, the NOA for the calling and called party digits is indicated by the type of ST digit received (ST, STP, ST2P, ST3P). FGD type signaling also provides an international stage that indicates an international called party address NOA. For CCS7 type agents, the NOA values for the calling and called party are contained in the initial address message (IAM). See “Result of digit collection” on pages 16-86 and 16-97.

For CCS7 type agents, the NOA value is taken from the parameter that provides the calling or called party digits. For example, this covers having a charge number and calling party address parameter in the IAM. The NOA uses whichever parameter that provides the digits.

Definition

Table 2-25 contains the IFNOA sequence collectable field refinements.

Table 2-25
IFNOA sequence collectable fields

Field	Description	Values	Default
PARTY	The party field indicates whether the NOA compared is the calling party NOA, the called party NOA or the DIALED NOA.	CALLING, CALLED or DIALED ***Note:*** The FLEXTYPE CALLING option is used to identify the SUBSCRIBER number type to identify the calling party address for the call. (See “Calling option” on page 5-8). Currently the calling party address NOA value is not used for call processing purposes.	N/A
COMPMETH	This field identifies the comparison method for the IFNOA collectable.	IS or NOT IS means that the calling or called nature of address is equal to one of the values in the NATOFADD vector. NOT means that the calling or called nature of address is not equal to any of the values in the NATOFADD vector.	N/A
NATOFADD	The NOA field contains a vector of up to four nature of address values used for the IFNOA comparison.	Vector of of up to 4 of {SUBR, VPN, NATL, INTL, TREAT_OP, SUBR_OP, NATL_OP, INTL_OP, NO_NUM_OP, NO_NUM_CT, 950_CT, TEST, PANI, INTL_INB_OP, INTL_OP_WZ1}	
INDEX	The index field identifies the FLEXDIAL table index to be processed if the IFNOA comparison is successful. All remaining unprocessed collectables in the current list do not get processed for the call.	An existing entry to table FLEXDIAL	N/A

—continued—

Table 2-25
IFNOA sequence collectable fields (continued)

Field	Description	Values	Default
ACTION	An enumeration of the actions to be taken on either the TRUE branch or the ELSE branch.	INSERT, APPEND, REPLACE, EXEC INSERT inserts the new list into the current processing list before the next collectable. APPEND appends the new list to the current processing list. REPLACE replaces the remainder of the unprocessed collectables with those identified by the FLEXDIAL index. EXEC executes the list identified by the FLEXDIAL table index as a subroutine, returning to the next collectable in the original list once execution is completed.	N/A
IFTRUE	The INDEX field is the index to branch in the true case.	An existing FLEXDIAL INDEX field	N/A
IFFALSE	The INDEX field identifies the FLEXDIAL table index that is to be processed if the IFNOA comparison is FALSE.	An existing entry to table FLEXDIAL	N/A

—end—

Up to four NOA values may be identified in the NATOFADD vector. When multiple values are specified, the logic for the comparisons is as as shown in Figure 2-2.

Figure 2-2
Logic for IFNOA comparison

IS Comparison Method:

Result = (NOA = A) or (NOA = B) or (NOA = C) or (NOA = D)

NOT Comparison Method:

Result = (NOA ^= A) and (NOA ^= B) and (NOA ^= C) and (NOA ^= D)

Note: ^= means 'not equal to'

If the result is true, then branching occurs.

Examples

Examples of IFNOA sequence collectable use:

```
IFNOA CALLED IS (NO_NUM_CT)$ REPLACE FGD_CUTTHRU_IDX NIL
```

In this example, if the received NOA value is equal to NO_NUM_CT, then collectable processing branches to the list represented by the FGD_CUTTHRU_IDX FLEXDIAL index.

```
IFNOA CALLED IS (NO_NUM_OP)(NATL_OP)(INTL_OP)$ REPLACE
OPER_RTE_IDX
```

In this example, if the received NOA value is equal to NO_NUM_OP, NATL_OP, or INTL_OP, then collectable processing branches to the list represented by the OPER_RTE_IDX FLEXDIAL index.

```
IFNOA CALLED NOT (INTL)$ REPLACE NATL_IDX NIL
```

In this example, if the received NOA value is not equal to INTL (international), then collectable processing branches to the list represented by the NATL_IDX FLEXDIAL index.

```
IFNOA CALLED IS (INTL) $ INSERT THIS ELSE_THIS
```

In this example, IFNOA inserts the collectables at THIS when the CALLED number is INTL, and inserts the collectables at ELSE_THIS when it is not INTL.

```
IFNOA IS DIALED (NO_NUM_CT)$ INSERT SIG_DTMF NIL
```

In this example, if the dialed number nature of address value is equal to no number cut-thru, then the collectable list identified by the FLEXDIAL table index "SIG_DTMF" is inserted into the current collectable list.

IFNOA collectable restrictions and limitations

The FLEXDIAL index must exist first.

IFPARM sequence collectable

The If Parameter (IFPARM) sequence collectable is designed to allow the execution of collectables to branch to a separate FLEXDIAL index if the identified parameter exists within the incoming out-of-band call setup message (the IAM for CCS7). This collectable does not request any information from the message, but simply checks if an identified parameter is included in the received out-of-band message.

This collectable performs no operation if an out-of-band message has not been received for the call.

A collectable list is a group of collectables identified by a FLEXDIAL index. Call processing handles this list through the Setup Information entity, and processes each collectable in the list one-at-a-time. The IFPARM collectable provides the means to conditionally branch to a new list to continue collectable processing, based on the receipt of specific IAM parameters. The remainder of the current list is not processed.

Definition

Table 2-26 contains the IFPARM framework collectable field refinements.

Table 2-26
IFPARM framework collectable fields

Field	Description	Values	Default
COMPMETH	This field identifies the comparison method for the IFPARM collectable.	IS or NOT	N/A

—continued—

Table 2-26
IFPARM framework collectable fields (continued)

Field	Description	Values	Default
NAME	This field identifies up to two parameter names that are scanned for in the out-of-band message. If just one name is found in the message, then processing branches to the FLEXDIAL index identified by the INDEX field.	Vector of up to 2 of Parameter Type. (See Table 2-50 on page 2-107 for a description of Parameter Type)	N/A
INDEX	The index field identifies the FLEXDIAL table index that is to be processed if the IFPARM comparison is successful. All remaining unprocessed collectables in the current list do not get processed for the call.	An existing entry to table FLEXDIAL	N/A
ACTION	An enumeration of the actions to be taken on either the TRUE branch or the ELSE branch.	INSERT, APPEND, REPLACE, EXEC INSERT inserts the new list into the current processing list before the next collectable. APPEND appends the new list to the current processing list. REPLACE replaces the remainder of the unprocessed collectables with those identified by the FLEXDIAL index. EXEC executes the list identified by the FLEXDIAL table index as a subroutine, returning to the next collectable in the original list once execution is completed.	N/A
IFTRUE	The INDEX field is the index to branch in the true case.	An existing FLEXDIAL INDEX field	N/A

—continued—

Table 2-26
IFPARM framework collectable fields (continued)

Field	Description	Values	Default
IFFALSE	The INDEX field identifies the FLEXDIAL table index that is to be processed if the IFPARM comparison is FALSE.	An existing entry to table FLEXDIAL	NIL
—end—			

Examples

An example of IFPARM sequence collectable use:

```
IFPARM IS (TNS)$ REPLACE S7_TNS_IDX NIL
```

In this example, if the TNS parameter is present in the out-of-band message, then collectable processing branches to the list represented by the S7_TNS_IDX FLEXDIAL index.

```
(IFPARM IS (CIP) (TNS) $ INSERT TRUE_PATH FALSE_PATH)$ N
```

In this example, if the CIP or TNS parameter (or both) appear in the CCS7 IAM, then execution branches to TRUE_PATH. Otherwise, the FALSE_PATH index is executed.

IFPARM collectable restrictions and limitations

The FLEXDIAL index must exist first.

DELDIGS sequence collectable

The Delete Digits (DELDIGS) sequence collectable provides the capability to manipulate the incoming digits before they are processed by the digit collectables The DELDIGS sequence collectable effectively erases digits from the incoming buffer of received digits.

When digits are received from the facility terminal controller (XPM), they are placed in an incoming digits buffer. This buffer holds the digits until they are requested by digit collectables. Digits that are removed from the digit buffer by the DELDIGS collectable cannot be retrieved and therefore are not processed by sequence or digit collectables.

ATTENTION
Operating company personnel who use this manipulation collectable must have intimate knowledge of the defined interaction with the originating agent at this defined point in the process.

Definition

Table 2-27 contains the DELDIGS sequence collectable field refinements.

Table 2-27
DELDIGS sequence collectable fields

Fields	Description	Values	Default
BUFLOC	The buffer location field identifies which digits from the incoming buffer are deleted.	ALLDIGS, PREFIXED, SUFFIXED, POS ALLDIGS identifies that all digits are removed from the digit buffer. PREFIXED identifies that the digits are removed from the start of the digit buffer (POS = 1; digits removed are from POS <= i < POS + DIGCNT where i <= buffer digit count) SUFFIXED indicates that the digits are removed backwards from the end of the received digits in the digit buffer. (POS = buffer digit count; digits removed are from POS - DIGCNT < i <= POS where i > 0). POS identifies a specific position in the buffer from where the digits are removed. Digits are removed in a forward manner from the digit buffer in a similar fashion as the PREFIXED variation.	N/A
BUFPOS	If the BUFLOC field is set to POS, then this field is present and identifies the specific location within the digit buffer from which the digits are deleted. A position value of 1 identifies the first position in the buffer.	1 to 64	N/A
DIGCNT	If the BUFLOC field is not equal to ALLDIGS, then this field is present and identifies the number of incoming digits that are deleted from the digit buffer.	1 to 16	N/A

Examples

Examples of DELDIGS sequence collectable use:

```
DELDIGS PREFIXED 3
```

In this example, the first three digits in the digit buffer are removed from the buffer, provided there are at least three digits in the buffer. A buffer that contained digits 2145555511 now contains 5555511.

```
DELDIGS POS 4 3
```

In this example, three digits starting at position 4 are removed from the digit buffer, provided there are at least six digits in the buffer. A buffer that contained digits 2145555511 now contains 2145511.

```
DELDIGS SUFFIXED 4
```

In this example, four digits at the end of the digit buffer are removed from the buffer, provided there are at least four digits in the buffer. A buffer that contained digits 2145555511 now contains 214555.

DELDIGS collectable restrictions and limitations

There are no identified table control restrictions for the DELDIGS sequence collectable.

Note: Provisioning does not catch incorrect use of the DELDIGS collectable, such as trying to delete digits that aren’t there. The DELDIGS collectable does not make a request for digits from the originating agent, but works with what is already in the digit buffer. Therefore, if a specific amount of digits are required in the buffer to perform manipulation, the COLDIG digit collectable must be used to first collect the digits from the originating agent.

ADDDIGS sequence collectable

The Add Digits (ADDDIGS) sequence collectable provides the capability to manipulate the incoming digits before they are processed by the digit collectables. The ADDDIGS sequence collectable adds digits to the incoming buffer of received digits.

When digits are received from the facility terminal controller (XPM), they are placed in an incoming digits buffer. This buffer holds the digits until they are requested by sequence or digit collectables. When digits are added to the digit buffer by the ADDDIGS collectable, it’s as if they were collected from the facility terminal controller.

ATTENTION

Operating company personnel who use this manipulation collectable must have intimate knowledge of the defined interaction with the originating agent at this defined point in the process.

Definition

Table 2-28 contains the ADDDIGS sequence collectable field refinements.

Table 2-28
ADDDIGS sequence collectable fields

Field	Description	Values	Default
BUFLOC	The buffer location field identifies where in the buffer the digits are placed.	PREFIX, SUFFIX, POS PREFIX identifies that the digits are inserted before the first digit at the start of the digit buffer (POS = 1) SUFFIX indicates that the digits are appended to the end of the received digits in the digit buffer. (POS = buffer digit count). POS identifies a specific position in the buffer where the digits are inserted into the digit buffer (inserted before digit at POS location).	N/A
BUFPOS	The buffer position identifies an actual location within the digit buffer where the digits are placed. This field is only present when the BUFLOC field equals POS. A position value of 1 identifies the first position in the buffer.	1 to 64	N/A

—continued—

Table 2-28
ADDDIGS sequence collectable fields (continued)

Field	Description	Values	Default
DIGITS	This is a digit vector which contains the digits that are added to the digit buffer to be processed by the digit collectables.	Vector of up to 15 of {0–9, A,B,C,D, S, P} Digits A,B,C, and D represent 4th column DTMF digits. The S identifies an asterisk digit, and the P identifies an octothorpe digit.	N/A
—end—			

Examples

Examples of ADDDIGS sequence collectable use:

```
ADDDIGS PREFIX 214
```

In this example, the digits 214 are inserted to the beginning of the digit buffer. A buffer that contained 5555511 now contains 2145555511.

```
ADDDIGS POS 4 555
```

In this example, the digits 555 are inserted at position 4 of the digit buffer. A buffer that contained 2145511 now contains 2145555511.

```
ADDDIGS SUFFIX 5511
```

In this example, the digits 5511 are appended to the digits in the digit buffer. A buffer that contained 214555 now contains 2145555511.

ADDDIGS collectable restrictions and limitations

There are no identified table control restrictions for the ADDDIGS sequence collectable.

Note: Provisioning does not catch incorrect use of the ADDDIGS collectable, such as trying to add digits at position 10 when only three digits are actually in the digit buffer (creating a hole). The ADDDIGS collectable does not make a request for digits from the originating agent, but works with what is already in the digit buffer. Therefore, if a specific amount of digits are required in the buffer to perform manipulation, the COLDIG digit collectable must be used to first collect the digits from the originating agent.

MODDIGS sequence collectable

The Modify Digits (MODDIGS) sequence collectable provides the capability to manipulate the incoming digits before they are processed by the digit collectables. The MODDIGS sequence collectable replaces digits in the incoming buffer of received digits.

The MODDIGS sequence collectable is designed to replace digits in the incoming buffer of received digits with other provisioned digits via a table control lookup algorithm in the FLEXDIAL table. The key to the FLEXDIAL table is a unique index along with the digits that are being modified.

When digits are received from the facility terminal controller (XPM), they are placed in a digit buffer. This buffer holds the digits until they are requested by digit collectables.

For digit manipulation, MODDIGS is a two-step process. First the digits are removed from the digit buffer according to the rules of the DELDIGS collectable. Then digits are added to the buffer at the proper position according to the rules of the ADDDIGS collectable. This allows for cases where the number of digits being added to the buffer is not equal to the number of digits being removed.

ATTENTION

Operating company personnel who use this manipulation collectable must have intimate knowledge of the defined interaction with the originating agent at this defined point in the process.

Definition

Table 2-29 contains the MODDIGS sequence collectable field refinements.

Table 2-29
MODDIGS sequence collectable fields

Fields	Description	Values	Default
BUFLOC	The buffer location field identifies within the digit buffer of the digits to be modified.	PREFIXED, SUFFIXED, POS PREFIXED identifies that the digits that are modified are from the start of the digit buffer (POS = 1; digits being modified are from POS <= i < POS + DIGCNT where i <= buffer digit count) SUFFIXED indicates that the digits that are modified are from the end of the received digits in the digit buffer and counting backwards. (POS = buffer digit count; digits being modified are from POS - DIGCNT > i >=POS where i > 0). POS identifies a specific position in the buffer where the digits that are modified are from.	N/A
BUFPOS	If the buffer location field is set to POS, then this buffer position field is present and identifies an actual location within the digit buffer where the digits that are to be modified are taken from.	1 to 64	N/A
DIGCNT	The digit count field identifies the number of incoming digits that are modified through use of the FLEXDIAL table before the digits are actually processed by digit collectables.	1 to 16	N/A
MODIDX	This field identifies the index into the FLEXMOD table for digit modifications.	An existing entry in table FLEXMOD	N/A

Examples

An example of MODDIGS sequence collectable use:

```
MODDIGS PREFIX 3 NPA_IDX
```

In this example, the first three digits in the digit buffer are removed and replaced by digits retrieved from table FLEXDIAL.

MODDIGS collectable restrictions and limitations

There are no identified table control restrictions for the MODDIGS sequence collectable.

Note: Provisioning does not catch incorrect use of the MODDIGS collectable, such as trying to modify digits that aren’t there. The MODDIGS collectable does not make a request for digits from the originating agent, but works with what is already in the digit buffer. Therefore, if a specific amount of digits are required in the buffer to perform manipulation, the COLDIG digit collectable must be used to first collect the digits from the originating agent.

COPYDIGS sequence collectable

The Copy Digits (COPYDIGS) sequence collectable provides the capability to manipulate the incoming digits before they are processed by the digit collectables by duplicating digits in the incoming buffer of received digits.

The COPYDIGS sequence collectable is designed to duplicate digits in the incoming buffer of received digits at another location within the buffer.

When digits are received from the facility terminal controller (XPM), they are placed in an incoming digits buffer. This buffer holds the digits until they are requested by sequence or digit collectables. When digits are copied within the digit buffer by the COPYDIGS collectable, it appears they are collected from the facility terminal controller.

> **ATTENTION**
>
> Operating company personnel who use this manipulation collectable must have intimate knowledge of the defined interaction with the originating agent at this defined point in the process.

Definition

Table 2-30 contains the COPYDIGS sequence collectable field refinements.

Table 2-30
COPYDIGS sequence collectable fields

Fields	Description	Values	Default
FBUFLOC	The “from” buffer location field identifies within the digit buffer of the digits to be copied.	PREFIXED, SUFFIXED, POS PREFIXED identifies that the digits that are copied are from the start of the digit buffer (POS = 1; digits being modified are from POS <= i < POS + DIGCNT where i <= buffer digit count) SUFFIXED indicates that the digits that are copied are from the end of the received digits in the digit buffer and counting backwards. (POS = buffer digit count; digits being modified are from POS - DIGCNT > i >= POS where i > 0). POS identifies a specific position in the buffer where the digits that are copied are from.	N/A
FBUFPOS	If the “from” buffer location field FBUFLOC is set to POS, then this buffer position field is present and identifies an actual location within the digit buffer where the digits that are to be copied are taken from.	1 to 64	N/A
DIGCNT	The digit count field identifies the number of incoming digits to be copied before the digits are actually processed by digit collectables.	1 to 16	N/A

—continued—

Table 2-30
COPYDIGS sequence collectable fields (continued)

Fields	Description	Values	Default
TBUFLOC	The “to” buffer location field identifies where in the buffer the copied digits are placed.	PREFIX, SUFFIX, POS PREFIX identifies that the digits are inserted before the first digit at the start of the digit buffer (POS = 1) SUFFIX indicates that the digits are appended to the end of the received digits in the digit buffer. (POS = buffer digit count). POS identifies a specific position in the buffer where the digits are inserted into the digit buffer (inserted before digit at POS location).	N/A
TBUFPOS	The “to” buffer position identifies an actual location within the digit buffer where the copied digits are placed. This field is only present when the TBUFLOC field equals POS. A position value of 1 identifies the first position in the buffer.	1 to 64	N/A
—end—			

Examples

Examples of COPYDIGS sequence collectable use:

```
COPYDIGS PREFIXED 3 PREFIX
```

In this example, the first three digits in the buffer are copied and inserted at the beginning of the digit buffer. A buffer that contained 5555511 now contains 5555555511.

```
COPYDIGS POS 4 4 SUFFIX
```

In this example, four digits starting at position four of the digit buffer are appended to the end of the buffer. A buffer that contained 5555511 now contains 55555115511.

COPYDIGS collectable restrictions and limitations

There are no identified restrictions for the COPYDIGS sequence collectable.

Note: Provisioning does not catch incorrect use of the COPYDIGS collectable, such as trying to copy digits to position 10 when only three digits are actually in the digit buffer (creating a hole). The COPYDIGS collectable does not make a request for digits from the originating agent, but works with what is already in the digit buffer. Therefore, if a specific amount of digits are required in the buffer to perform manipulation, the COLDIG digit collectable must be used to first collect the digits from the originating agent.

MODNOA sequence collectable

The Modify Nature Of Address (MODNOA) sequence collectable provides the ability to change the value of the incoming nature of address (NOA).

Definition

Table 2-31 contains the MODNOA sequence collectable field refinements.

Table 2-31
MODNOA sequence collectable fields

Fields	Description	Values	Default
PARTY	This field identifies whether the calling, called, or dialed party nature of address is modified before it is processed by any other collectables.	CALLING, CALLED, or DIALED	N/A
NATOFADD	This field identifies the nature of address value that replaces the current nature of address value.	SUBR, VPN, NATL, INTL, TREAT_OP, SUBR_OP, NATL_OP, INTL_OP, NO_NUM_OP, NO_NUM_CT, 950_CT, TEST, PANI, INTL_INB_OP, INTL_OP_WZ1	N/A

Examples

An example of MODNOA sequence collectable use:

```
MODNOA CALLED NATL
```

In this example, the called party address NOA value is being set to the national (NATL) value.

```
MODNOA DIALED UNKNOWN
```

In this example, the dialed number nature of address value is set to the value of “UNKNOWN”.

MODNOA collectable restrictions and limitations

There are no identified table control restrictions for the MODNOA sequence collectable.

AGNTDATA sequence collectable

The Agent Data Sequence (AGNTDATA) collectable puts digits provisioned against the AXXESS agent into the digit buffer.

Definition

Table 2-32 contains the AGNTDATA sequence collectable field refinements.

Table 2-32
AGNTDATA sequence collectable fields

Field	Description	Values	Default
BUFLOC	The buffer location field identifies where the digits are placed in the digit buffer.	PREFIX, SUFFIX, POS PREFIX identifies that the digits are inserted before the first digit at the start of the digit buffer (POS = 1) SUFFIX indicates that the digits are appended to the end of the received digits in the digit buffer. (POS = buffer digit count). POS identifies a specific position where the digits are inserted into the digit buffer (inserted before digit at POS location).	N/A
BUFPOS	If the buffer location field is set to POS, then this buffer position field is present and identifies an actual location within the digit buffer where the agent data digits are put.	1 to 64	N/A

—continued—

Table 2-32
AGNTDATA sequence collectable fields (continued)

Field	Description	Values	Default
DIGTYPE	This field identifies which provisioned AXXESS agent field of digits is put into the digit buffer.	SNPA, SNXX, CITYCODE SNPA indicates that the TRKFEAT SNPA option defined digits are put into the incoming digits buffer. SNXX indicates that the TRKFEAT SNXX option defined digits are put into the incoming digits buffer. CITYCODE indicates that the TRKFEAT CITYCODE defined digits are put into the incoming digits buffer. DEFCIC indicates that the TRKFEAT DEFCIC digits are put into the incoming digits buffer.	N/A
—end—			

The SNPA and SNXX values are retrieved from the Originating Options (ORIGOPTS) vector of the TRKFEAT table entry.

Examples

Examples of AGNTDATA sequence collectable include:

```
AGNTDATA PREFIX SNPA
```

In this example, the SNPA provisioned information for the agent is inserted at the beginning of the digit buffer. If the digits 5555511 were in the digit buffer and the SNPA was provisioned with a value of 214, then the digit buffer would contain 2145555511.

```
AGNTDATA SUFFIX SNXX
```

In this example, the SNXX provisioned information for the agent is appended to the end of the digit buffer. If the digits 214 were in the digit buffer and the SNXX was provisioned with a value of 555, then the digit buffer would contain 214555.

```
AGNTDATA PREFIX DEFCIC
```

In this example, the DEFCIC from table TRKFEAT is inserted at the beginning of the digit buffer.

AGNTDATA collectable restrictions and limitations

The AGNTDATA collectable will only place a DEFCIC in the digit buffer if a DEFCIC option is present in table TRKFEAT and if the DEFCIC contains the AGNTDATA suboption in table TRKFEAT. If a DEFCIC option containing an AGNTDATA option does not exist, the AGNTDATA collectable does not place any digits into the digit buffer.

RETRIEVE sequence collectable

The RETRIEVE collectable provides the ability to retrieve the identified digits that have been already processed for the call and add them to the digit buffer at the desired location. Digit manipulation or alternative processing may then be performed on the buffered digits.

Fields

Table 2-33 describes the RETRIEVE collectable fields.

Table 2-33
RETRIEVE collectable fields

Field	Description	Values	Default
RETDIGS	This field identifies the source of the digits being retrieved.	DIALED, ADDR, SUBR, OLI, CIC	N/A
FLEXTYPE	If the RETDIGS field is set to SUBR, then this field is present and identifies the type of subscriber number digits being retrieved.	A valid entry to table FLEXTYPE	N/A
OCCUR–RENCE	If the RETDIGS field is set to SUBR, then this field is present and identifies if either the first or last occurrence of the processed subscriber number type is to be retrieved. If only a single subscriber number of this type was processed for the call, then the value of this field is irrelevant to execution of the RETRIEVE collectable.	FIRST, LAST	N/A

—continued—

Table 2-33
RETRIEVE collectable fields

Field	Description	Values	Default
DIGLEN	This field identifies the maximum number of digits from the digit source that are inserted into the digit buffer. If the digit source contains equal or fewer digits than the length identified by this field, then all the digits identified by the source are added to the digit buffer. If the digit source contains more digits than identified by the length of this field, then only the first DIGLEN number of digits are added to the digit buffer.	0 to 18	N/A
BUFLOC	The buffer location field identifies where in the buffer the digits are placed.	PREFIX, SUFFIX, POS PREFIX identifies that the digits are inserted before the first digit at the start of the digit buffer (POS = 1) SUFFIX indicates that the digits are appended to the end of the received digits in the digit buffer. (POS = buffer digit count). POS identifies a specific position in the buffer where the digits are inserted into the digit buffer (inserted before digit at POS location).	N/A
BUFPOS	The buffer position identifies an actual location within the digit buffer where the digits are placed. This field is only present when the BUFLOC field equals POS. A position value of 1 identifies the first position in the buffer.	1 to 64	N/A

—end—

Provisioning Example

Examples of RETRIEVE collectable use are:

```
RETRIEVE SUBR AUTH FIRST PREFIX
```

In this example, the first occurrence of processed AUTH subscriber number digits are retrieved and are prefixed to the beginning of the digit buffer. If AUTH subscriber number type digits have not yet been processed for the call, then nothing is inserted to the digit buffer.

RETRIEVE sequence collectable restrictions and limitations

No provisioning restrictions or limitations exist for the RETRIEVE collectable.

COLDIG digit collectable

The Collect Digits (COLDIG) digit collectable is needed to collect digits from the originating agent through MF or DTMF inband tone signals. Digits collected are placed in the digit buffer for processing by other sequence and digit collectables. This collectable does not perform any validation or processing of the received digits.

Definition

Table 2-34 contains the COLDIG digit collectable field refinements.

Table 2-34
COLDIG digit collectable fields

Field	Description	Values	Default
MIN	This field identifies the minimum number of digits to collect.	0 to 32	N/A
MAX	This field identifies the maximum number of digits to collect.	1 to 32	N/A

—continued—

Table 2-34
COLDIG digit collectable fields (continued)

Field	Description	Values	Default
PROCNOA	This field identifies how the nature of address for the COLDIG application is to be processed.	IGNORE, CALLED, CALLING IGNORE identifies that no nature of address value is generated with the COLDIG application. CALLED identifies that the nature of address for the called party number is generated by the COLDIG application. CALLING identifies that the nature of address for the calling party number is generated by the COLDIG application.	N/A
OPTION1	This field contains a vector of options that may be added to the COLDIG digit collectable definition.	Vector of up to 3 of {NIL, PROMPT, SIGFEAT} The NIL option is described below. For PROMPT option field values, see Table 2-35. For SIGFEAT option field values see Table 2-36.	N/A
OPTION2	This field contains a vector of options that may be added to the COLDIG digit collectable definition.	Vector of up to 3 of {NIL, RESET, FILED} The NIL option is described below. For RESET option field values, see Table 2-37. For FILED option field values, see Table 2-38. ***Note:*** With the exception of the NIL option, options cannot be duplicated.	N/A
—end—			

The options defined for COLDIG are common to all digit collectable definitions.

No options are provisioned by entering $ when prompted for the first option in the vector field.

NIL option

The NIL option provides an easy way to update the options vector. Instead of removing an option by retying the provisioned options that follow, NIL can be inserted. Using NIL enables the inserted option to be effectively removed without having to retype the remaining provisioned options.

The NIL option does not remain visible once the update is complete.

An example showing the use of NIL option is:

1 Editing existing tuple:

```
OPTIONS: (option1) (option2) (option3)$
```

2 To remove option2, change the table entry. When prompted at option2, enter NIL.

```
OPTIONS: option2
> NIL
```

3 After editing, display of the tuple shows:

```
OPTIONS: (option1) (option3)$
```

PROMPT option

The PROMPT option identifies the tone or announcement to be played as a prompt to begin entering digits. Upon reception of the first digit, the prompt tone or announcement is terminated.

Table 2-35 contains the COLDIG digit collectable PROMPT option fields.

Table 2-35
COLDIG digit collectable PROMPT option fields

Field	Description	Values	Default
PRMTTYPE	This field identifies the prompt type.	TONE or ANNC	N/A
TONETYPE	If the prompt type is set to TONE, then this field is present and identifies the type of tone provisioning that defines the tone.	TONES or STD	N/A

—continued—

Table 2-35
COLDIG digit collectable PROMPT option fields (continued)

Field	Description	Values	Default
CLLI	This field identifies the tone or announcement prompt played to request digits. This field is visible if one of the following conditions is met: The prompt type is set to ANNC. The prompt type is set to TONE and the TONETYPE is set to TONES.	TONE or ANNOUNCEMENT CLLI (as provisioned in table TONES or table ANNS)	N/A
STDTONE	If the prompt type is set to TONE and the TONETYPE is set to STD, then this field is present and identifies the type of tone that is used.	CD, SD, SSD, H, L, 0 - 9, A, B, C, D, S, P CD identifies carrier dial tone (400Hz). SD identifies standard dial tone (350 + 440 Hz). SSD identifies a tone consisting of a short burst of stuttered dial tone followed by standard dial tone. H identifies a high tone (480 Hz). L identifies a low tone (480 + 620 Hz). 0 through 9 represent the associated DTMF digit tones. A, B, C, D refer to the 4th column DTMF digits. S refers to the DTMF asterisk digit. P refers to the DTMF octothorpe digit.	N/A

—continued—

Table 2-35
COLDIG digit collectable PROMPT option fields (continued)

Field	Description	Values	Default
TONEDUR	If the prompt type is set to TONE and the TONETYPE is set to STD, then this field is present and identifies the duration of the tone that is being played.	{1 to 255} in ten millisecond ticks. The value of 255 indicates a continuous tone is to be applied.	N/A
PREDISTN	The tone to pulse converter disable tone field identifies if the special tone pulse is sent before actually applying the prompt tone or announcement.	N or Y	N/A
—end—			

SIGFEAT option

The Signaling Feature (SIGFEAT) option contains information for special signaling characteristics of digit collection. This option would typically be used when collecting raw called party address information.

Table 2-36 contains the COLDIG digit collectable SIGFEAT option fields.

Table 2-36
COLDIG digit collectable SIGFEAT option fields

Fields	Description	Values	Default
PDIL234	This field defines a special short partial dial timer used with DTMF collection. When this option is present, this timer is in effect after collection of the second, third, and fourth digits. The partial dial timer is reset to the partial dial timer value (either the PDILTMR or MINRTMR identified by table TRKSIG or set by the SIG collectable) after collection of the other digits. A value of zero indicates that this signaling feature is not in effect.	0 to 15 seconds A value of 0 indicates that PDIL234 timing is not active.	0
SPLFDIG	This is a special option for DTMF digit collection that causes the identified SPLFTMR timer value to be in effect after the first digit is received if it matches the digit identified. A, B, C, and D identify 4th column DTMF digits. S represents the asterisk digit, and P represents the octothorpe digit. The special first digit is automatically included in the digit mask used for DTMF digit collection for the address digits.	Set of {S, 1–9, 0, P, A, B, C, D} or ALL or NONE ALL indicates that all digits are included in the special first digit set. NONE identifies no special first digit is identified. Any other digit identifies that digit as the special first digit. ***Note:*** Although the provisioning is developed to support a set of special first digits, currently table control restrictions limit provisioning more than one digit, and the value of ALL may not be used.	N/A

—continued—

Fields	Description	Values	Default
SPLFTMR	This field identifies the timer value used after reception of the first digit if the first digit received is the digit identified by the SPLFDIG field as the special first digit. After receipt of more digits, use of the PDILTMR or MINRTMR is restored. Additionally, a value greater than zero identifies that special first digit handling is also in effect for the asterisk and octothorpe digits, separate from their possible use as reset and/or terminating digits.	0 to 15 seconds A value of zero indicates that special first digit processing is not active, and can only be used when SPLFDIG = NONE.	0
—end—			

The SIGFEAT options apply only to DTMF pulse type digit collection. If the pulse type is set to MF or NP, then these options are not in effect.

RESET option

The RESET option identifies that reset can occur when collecting the digits for this collectable, meaning the subscriber can start over at the beginning after pressing the defined reset digit.

Table 2-37 contains the COLDIG digit collectable RESET option fields.

Table 2-37
COLDIG digit collectable RESET option fields

Field	Description	Values	Default
LIMIT	This field identifies the number of resets that may occur on the particular call.	0 to 255	N/A
TRMT	The TRMT field identifies the treatment applied to the call if an attempt is made to exceed the maximum number of resets allowed.	Extended treatment type range ***Note:*** See *UCS DMS-250 Data Schema Reference Manual* for values of the extended treatment type.	N/A
WDIGACT	The with digits action identifies whether the prompt option specified is replayed when a reset occurs after digits have been collected for the collectable.	NOPROMPT, PROMPT, PREVIOUS, PREPRMPT NOPROMPT identifies that an identified prompt is not reapplied for the digit collectable. PROMPT identifies that either the RESET PROMPT or the original PROMPT is reapplied for the digit collectable. PREVIOUS identifies that the call is reset to the previous digit collectable executed. The history of executed collectables is searched to identify the previous digit collectable executed. PREPRMPT identifies that the call is reset to the previous digit collectable executed which contains a prompt or reset prompt option. The history of executed collectables is searched to identify the most previous digit collectable executed with the prompt or reset prompt option.	N/A

—continued—

Table 2-37
COLDIG digit collectable RESET option fields (continued)

Field	Description	Values	Default
WODIGACT	The without digits action identifies the action to occur if a reset digit is received and no digits have been collected for the digit collectable.	SAME, PREVIOUS, PREPRMPT, DPIDX SAME: Process the same collectable again. If a PROMPT is identified, the prompt is replayed. PREVIOUS identifies that the call is reset to the previous digit collectable executed. The history of executed collectables is searched to identify the most previous digit collectable executed. PREPRMPT identifies that the call is reset to the previous digit collectable executed which contains a prompt or reset prompt option. The history of executed collectables is searched to identify the previous digit collectable executed with the prompt or reset prompt option. DPIDX: Jump to the index identified and begin collectable processing as the reset action.	N/A

—continued—

Table 2-37
COLDIG digit collectable RESET option fields (continued)

Field	Description	Values	Default
INDEX	If the WODIGACT field is set to DPIDX, then this field is present and identifies the FLEXDIAL index that is processed for the without digits reset action.	An existing entry in table FLEXDIAL	N/A
OPTIONS	This field consists of a vector of options which may be added to the RESET option.	NIL, PROMPT The NIL option is identical in use and function as that defined in Table 2-34. The RESET PROMPT option identifies a unique reset prompt that is played for the collectable in place of the defined PROMPT option. If a reset action occurs after some digits have been collected, and the WDIGACT field is set to PROMPT, then the RESET PROMPT is played instead of the defined PROMPT option. The RESET PROMPT option fields have the same definition as the PROMPT option fields defined in Table 2-35.	N/A

—end—

FILED option

The FILED option identifies filed digits for the digit collectable. These digits are used as the collected digits instead of collecting some or all of the digits from the originating agent.

Table 2-38 contains the COLDIG digit collectable FILED option fields.

Table 2-38
COLDIG digit collectable FILED option fields

Fields	Description	Values	Default
BUFLOC	This field identifies the location for the digit collectable digits where the filed digits are placed. The amount of digits remaining to be collected from the originating agent is MIN - count of digits filed.	PREFIX or SUFFIX PREFIX indicates that the FILED digits precede any digits collected from the originating agent. SUFFIX indicates that the FILED digits are appended to any digits collected from the originating agent.	N/A
DIGVEC	This is a digit vector which contains the digits that are added to the digit collectable digits for processing by the digit collectable.	Vector of up to 15 of {0–9, A,B,C,D, S, P} The A, B, C, and D digits represent 4th column DTMF digits. The S identifies an asterisk digit, and the P identifies an octothorpe digit.	N/A

Examples

Examples of COLDIG digit collectable use:

```
COLDIG 7 7 IGNORE (PROMPT ANNC MYANNC) $ (FILED PREFIX 555) $
```

In this example, an announcement prompt identified by the CLLI MYANNC is provided for the subscriber. Three filed digits are inserted into the digit buffer and when the collectable parses, it collects four more digits from the subscriber. The NOA value associated with the received digits is ignored.

```
COLDIG 10 11 CALLED (PROMPT TONE STD H 255 N) $ $
```

In this example, a continuous high tone is applied to the bearer channel as a prompt for the subscriber. A minimum of 10 and maximum of 11 digits are collected from the subscriber. The NOA value associated with the received digits is handled as the called party NOA.

```
COLDIG 6 7 IGNORE (PROMPT ANNC MY_ANNC) $ (RESET S 10 RODR
PROMPT PREPRMPT PROMPT ANNC MY_RESET_ANNC) $
```

In this example, an announcement prompt identified by the CLLI MY_ANNC is provided for the subscriber to collect a minimum of six and maximum of seven digits from the originating agent. If the reset digit is pressed, and digits have been entered, then the announcement identified by the MY_RESET_ANNC CLLI is played as an interruptible prompt for the customer to re-enter the digits. If no digits have been entered prior to the reset digit, then the previously processed digit collectable which contained a prompt is reset and processed again. The NOA value associated with the received digits is ignored.

COLDIG collectable restrictions and limitations

The following restrictions and limitations apply for the COLDIG collectable:

- Options may not be duplicated within the OPTIONS1 or OPTIONS2 vectors. An attempt to duplicate options results in the following error message:

  ```
  Cannot duplicate OPTIONS.
  ```

- The number of filed digits must not be greater than the MAX value for a particular digit collectable. An attempt to do this results in the following error message:

  ```
  Filed digit count may not exceed MAX value.
  ```

- If a PROMPT option is not specified for the collectable (either the PROMPT option or RESET PROMPT option), then the WDIGACT RESET option field cannot be set to PROMPT for a provisioned RESET option. An attempt to do this results in the following error message:

  ```
  PROMPT option must be specified for RESET PROMPT action.
  ```

- A tone (table TONES) or an announcement (table ANNS) CLLI must be provisioned for the associated PRMTTYPE/TONETYPE values. An attempt to provision a non-announcement CLLI for an announcement prompt type results in the following error message:

  ```
  An announcement CLLI (table ANNS) must be provisioned for
  the ANNC PRMTTYPE value.
  ```

  An attempt to provision a non-tone CLLI for a TONE prompt type or TONES tone type results in the following error message:

  ```
  A tone CLLI (table TONES) must be provisioned for the TONE
  PRMTTYPE / TONES TONETYPE values.
  ```

- MIN value specified must be less than or equal to the MAX value specified. An attempt to do this results in the following error message:

  ```
  MIN value must be less than or equal to MAX value.
  ```

- When the SPLFDIG field of the SIGFEAT option is set to NONE, the SPLFTMR field must be set to zero. An attempt to provision a non-zero value under these circumstances results in the following error message:

  ```
  SPLFTMR must be set to zero.
  ```

- When the SPLFDIG field of the SIGFEAT option is set to a value other than NONE, the SPLFTMR field must be set to a value greater than zero. An attempt to provision a value of zero under these circumstances results in the following error message:

  ```
  SPLFTMR must be greater than zero.
  ```

- Only one digit can be included in the SPLFDIG set, and the value of ALL may not be used. An attempt to provision more than one digit in the set or the value ALL results in the following error message:

  ```
  SPLFDIG field may only contain one digit.
  ```

- Only the digit zero (0) is supported as the SPLFDIG due to current implementation restrictions. If an attempt to provision a value other than zero is performed, then the following error message is displayed:

  ```
  Only the digit zero (0) is supported for SPLFDIG.
  ```

All errors result in the failure of requested table control changes.

SUBR digit collectable

The Subscriber Number (SUBR) digit collectable collects and validates subscriber number digits received through MF or DTMF inband tone signals. This collectable requests and optionally removes required digits from the incoming digits buffer for processing. If the minimum amount of digits required is not available in the buffer, this collectable requests more digits from the originating agent.

Definition

Table 2-39 contains the SUBR digit collectable field refinements.

Table 2-39
SUBR digit collectable fields

Fields	Description	Values	Default
MIN	This field identifies the minimum number of digits to collect.	1 to 16	N/A
MAX	This field identifies the maximum number of digits to collect.	1 to 16	N/A

—continued—

Table 2-39
SUBR digit collectable fields (continued)

Fields	Description	Values	Default
FLEXTYPE	This field identifies the type of subscriber number being validated. This is required so that the number can be properly recorded in the billing record and also to properly index the FLEXVAL table to perform the screening of the digits (if FLEXVAL INSWITCH validation is used).	An existing entry in table FLEXTYPE	N/A
BILLCAPT	This field identifies whether to capture the collected number in the billing record. (The billing record field is identified by the FLEXTYPE table).	N or Y	N/A
REPBDIGS	This field identifies whether the digits removed from the incoming digits buffer for processing by this SUBR digit collectable should be placed back into the buffer after validation has occurred. This allows these digits to be reprocessed by another digit collectable. ***Note:*** The immediate application of this capability is pseudo-ANI, where the validation of an authcode subscriber number retrieves a new FLEXDIAL table index for further processing. The index retrieved contains an ANI type subscriber number digit collectable.	N or Y	N/A

—continued—

Table 2-39
SUBR digit collectable fields (continued)

Fields	Description	Values	Default
OPTION1	This field contains a vector of options that may be added to the SUBR digit collectable definition.	Vector of up to 3 of {NIL, PROMPT, CITYVAL, CASUBLK} NIL and PROMPT options are identical to the options for the COLDIG collectable. See "COLDIG digit collectable" information on page 2-67. CITIVAL option field values are discussed in Table 2-41. CASUBLK option field values are discussed in Table 2-42.	N/A
OPTION2	This field contains a vector of options that may be added to the SUBR digit collectable definition.	Vector of up to 3 of {NIL, RESET, FILED, VALIDATE} ***Note:*** With the exception of the NIL option, options cannot be duplicated. NIL, RESET and FILED options are identical to the options for the COLDIG collectable. See "COLDIG digit collectable" information on page 2-67.. VALIDATE option field values are discussed in Table 2-40.	N/A
—end—			

SUBR VALIDATE option

The Subscriber Validate (SUBR VALIDATE) option contains information for screening the subscriber number digits collected. Some of this information (such as the indexes in the FLEXVAL table to use for validation) can be subscriber number specific and can be overridden by FLEXFEAT provisioning. See "SUBR addressee messages" on page 4-9 for more information.

Table 2-40 contains the SUBR VALIDATE digit collectable option fields.

Table 2-40
SUBR VALIDATE digit collectable option fields

Fields	Description	Values	Default
VALTYPE	The validation type field identifies the type of validation for the collected authcode digits.	INSWITCH	N/A
IVALTYPE	If the validation type is set to INSWITCH, then the INSWITCH validation type field is present and identifies the type of INSWITCH validation that occurs.	MATCH or FLEXVAL MATCH indicates that the received digits must match the digits identified in the DIGITS field. FLEXVAL indicates that the digits are screened in the FLEXVAL table using the index provided by the INDEX field below.	N/A
DIGITS	If the INSWITCH validation type is set to MATCH, then this field is present and identifies the digits that must be received for successful validation.	Vector of up to 16 of {0–9, A, B, C, D} The A, B, C, and D digits represent 4th column DTMF digits.	N/A

—continued—

Table 2-40
SUBR VALIDATE digit collectable option fields (continued)

Fields	Description	Values	Default
FLEXIDXS	If the INSWITCH validation type is set to FLEXVAL, then this index vector is present and indicates the number of subscriber digits to validate at each defined index. Each element in the vector contains 2 fields: 1) A numeric index within the range of the key-index in the FLEXVAL table. 2) The count value of the number of digits to validate at this index. If 2 indexes are identified, then the first index validates the first x digits of the subscriber number using the associated FLEXVAL table index, and the second index validates the next y digits of the subscriber number using the second FLEXVAL table index.	Vector of up to 2 of {INDEX, DIGCNT} INDEX is a value {0 to 10479999} that identifies the index into table FLEXVAL for validation lookup. ***Note:*** This is a value within the valid range of the table defined key. You do not have to actually provision the FLEXVAL table index. DIGCNT is a value {1 to 16} that identifies the number of subscriber number digits that are validated at the defined FLEXVAL index.	N/A

—continued—

Table 2-40
SUBR VALIDATE digit collectable option fields (continued)

Fields	Description	Values	Default
FAILACT	The fail action field identifies the action to take if a validation attempt fails. (That is, the subscriber number is not found or MATCH failed.) If a validation failure occurs, then the proper FLEX trouble log is generated and the proper operational measurements are pegged.	TRMT, RESET, IGNORE, DPIDX TRMT indicates that if the validation attempt fails, then treatment is applied to the call. ***Note:*** See *UCS DMS-250 Data Schema Reference Manual* for values of the extended treatment type. RESET indicates that if the validation attempt fails, then the digit collectable resets according to the RESET option specifications. IGNORE indicates that a failed validation attempt is ignored by call processing and the call proceeds as if validation passed. DPIDX indicates that the interaction identified by the FLEXDIAL index is performed.	N/A
DEFERED	The DEFERED field is enabled when the FAILACT's reset should be deferred for later processing. This is only prompted when FAILACT is RESET.	N or Y	N
OVERRIDE	If the fail action field is set to TRMT, this field is present and identifies whether the treatment set overrides any previous treatment set. If this field is set to N, then any existing treatment value is not overridden.	N or Y	N/A
TRMT	If the fail action field is set to TRMT, this field is present and identifies the treatment that is applied.	Extended treatment type range ***Note:*** See *UCS DMS-250 Data Schema Reference Manual* for values of the extended treatment type.	N/A

—continued—

Table 2-40
SUBR VALIDATE digit collectable option fields (continued)

Fields	Description	Values	Default
ACTION	If the fail action field (FAILACT) is set to DPIDX, this field is present and identifies how call processing handles the collectable list specified by the FLEXDIAL index.	INSERT, APPEND, REPLACE, EXEC INSERT inserts the new list into the current processing list before the next collectable. APPEND appends the new list to the current processing list. REPLACE replaces the remainder of the unprocessed collectables with those identified by the FLEXDIAL index. EXEC executes as a sublist the list identified by the FLEXDIAL table index,	N/A
INDEX	If the fail action field (FAILACT) is set to DPIDX, this field is present and identifies the FLEXDIAL index that is processed as a result of validation failure.	An existing entry in table FLEXDIAL	N/A

—end—

SUBR CITYVAL option

The Subscriber City Code Validation (SUBR CITYVAL) option identifies that city code validation occurs for the subscriber number processed. City code validation compares the CITYCODE of the subscriber number to the CITYCODE value set for the call. If the CITYCODE values are identical, then validation is successful. Otherwise the validation attempt fails.

City code validation can only occur if the CITYCODE option is provisioned for the subscriber number in table FLEXFEAT and another CITYCODE value has been previously identified for the call (such as provisioning the CITYCODE option for the originating agent in table TRKFEAT).

If the two required CITYCODE values are not available, then the city code validation does not occur and the validation attempt is successful by default.

City code validation may alternatively be triggered using the FLEXFEAT table CITYVAL option (see “FLEXFEAT CITYVAL option” on page 7-7).

In order to provision the CITYVAL option, the SUBR collectable VALIDATE option must also be provisioned, specifying the FLEXVAL type of INSWITCH validation.

Table 2-41 contains the SUBR CITYVAL digit collectable option fields.

Table 2-41
SUBR CITYVAL digit collectable option fields

Fields	Description	Values	Default
FAILACT	This field identifies the action to take if the city code validation fails.	VALIDATE, DPIDX, TRMT VALIDATE identifies that the action specified by the VALIDATE option FAILACT field is taken. DPIDX specifies the interaction identified by the FLEXDIAL index is performed. TRMT identifies that the treatment specified is to be set for the call.	N/A
ACTION	If the FAILACT field is set to DPIDX, then this field is present and identifies how the collectable list specified by the FLEXDIAL index is handled call processing.	INSERT, APPEND, REPLACE, EXEC INSERT inserts the new list into the current processing list before the next collectable. APPEND appends the new list to the current processing list. REPLACE replaces the remainder of the unprocessed collectables with those identified by the FLEXDIAL index. EXEC executes as a sublist the list identified by the FLEXDIAL table index.	N/A

—continued—

Table 2-41
SUBR CITYVAL digit collectable option fields (continued)

Fields	Description	Values	Default
INDEX	If the FAILACT field is set to DPIDX, then this field is present and identifies the index that is processed as a result of validation failure.	An existing entry in table FLEXDIAL	N/A
APPLYFTR	If the FAILACT field is set to TRMT, then this field is present and identifies whether or not other FLEXFEAT features and characteristics provisioned are to be applied for the subscriber number in the event of CITYCODE validation failure. Typically features and characteristics are not applied on validation failures.	N or Y	N/A
OVERRIDE	If the FAILACT field is set to TRMT, then this field is present and indicates whether the treatment identified overrides a treatment that may already be set for the call.	N or Y	N/A
TRMT	If the FAILACT field is set to TRMT, then this field is present and identifies the treatment to set for the call.	Extended treatment type range ***Note:*** See *UCS DMS-250 Data Schema Reference Manual* for values of the extended treatment type.	N/A
	—end—		

SUBR CASUBLK option

The Subscriber Casual Block (SUBR CASUBLK) option can be included only if the VALIDATE option is present with an INSWITCH validation type of FLEXVAL. This option identifies that casual subscriber numbers (as identified by the CASUAL characteristic) are not allowed, and the validation is considered to fail.

Table 2-42 contains the SUBR CASUBLK digit collectable option fields.

Table 2-42
SUBR CASUBLK digit collectable option fields

Fields	Description	Values	Default
FAILACT	This field identifies the validation failed action to take if a casual number as identified by the FLEXFEAT CASUAL option is being screened.	VALIDATE or TRMT VALIDATE identifies that the action specified by the VALIDATE option FAILACT field is taken. TRMT indicates that if the validation attempt fails, then treatment is applied to the call.	N/A
OVERRIDE	If the FAILACT field is set to TRMT, then this field is present and identifies whether or not the treatment identified overrides a treatment that may already be set for the call.	N or Y	N/A
TRMT	If the FAILACT field is set to TRMT, then this field is present and identifies the treatment to set for the call.	Extended treatment type range ***Note:*** See *UCS DMS-250 Data Schema Reference Manual* for values of the extended treatment type.	N/A

Examples

Examples of the SUBR digit collectable use:

```
SUBR 4 4 ACCT Y N $ (VALIDATE INSWITCH FLEXVAL (101 2)$ TRMT
Y INAC) $
```

In this example, a four-digit subscriber number is being collected, recorded in billing, and validated. The subscriber number type is identified as an ACCT type, and only two of the four digits collected are validated in the FLEXVAL table using index 101. If validation fails for this subscriber number, then INAC treatment is set for the call.

```
SUBR 3 10 ANI Y N (CASUBLK TRMT Y AARD) $ (VALIDATE INSWITCH
FLEXVAL (53 10)$ TRMT Y ADBF) $
```

In this example, a three- to ten-digit subscriber number is being collected, recorded in billing, and validated. The subscriber number type is identified as an ANI type, and the ten digits are validated in the FLEXVAL table using index 53. If validation fails for this subscriber number, then ADBF treatment is set for the call.

If the subscriber number is defined as a CASUAL subscriber number, then validation is considered to fail and AARD treatment is set for the call.

```
SUBR 6 7 AUTH Y N (PROMPT ANNC MY_ANNC) (CITYVAL DPIDX INSERT
PIN_4) $ (VALIDATE INSWITCH FLEXVAL (3 6)$ RESET) (RESET S 10
RODR PROMPT SAME PROMPT ANNC MY_RESET_ANNC) $
```

In this example, an announcement prompt identified by the CLLI MY_ANNC is provided for the subscriber to collect a minimum of six and maximum of seven digits from the originating agent, recorded in the billing record, and validated.

If the reset digit is pressed, then the announcement identified by the MY_RESET_ANNC CLLI is played as an interruptible prompt for the customer to re-enter the digits. This action is taken whether or not digits are received prior to the receipt of the reset digit.

Two forms of validation occur for this collectable. First validation occurs through the FLEXVAL table where six digits are validated using index 3. If validation fails, a reset is performed. The second validation is a city code validation which is performed if the first validation passes. The validation attempt matches the digits from the FLEXFEAT city code option with digits from the TRKFEAT city code option. If the city code validation fails, then the interaction identified by the PIN_4 FLEXDIAL table index is inserted into the currently processing collectable list.

```
SUBR 4 4 PIN Y N $ (VALIDATE INSWITCH MATCH 1234 IGNORE)$
```

In this example, a four-digit subscriber number is being collected, recorded in billing, and validated. The subscriber number type is identified as a PIN type, and the four digits collected must be equal to 1234. No action is taken if validation of the digits fails.

SUBR collectable restrictions and limitations

The following restrictions and limitations apply for the SUBR collectable:

- Options may not be duplicated within the OPTIONS1 or OPTIONS2 vectors. An attempt to duplicate options results in the following error message:

  ```
  Cannot duplicate OPTIONS.
  ```

- The number of filed digits must not be greater than the MAX value for a particular digit collectable. An attempt to do this results in the following error message:

```
Filed digit count may not exceed MAX value.
```

- If a PROMPT option is not specified for the collectable (either the PROMPT option or RESET PROMPT option), then the WDIGACT RESET option field cannot be set to PROMPT for a provisioned RESET option. An attempt to do this results in the following error message:

  ```
  PROMPT option must be specified for RESET PROMPT action.
  ```

- A tone (table TONES) or an announcement (table ANNS) CLLI must be provisioned for the associated PRMTTYPE/TONETYPE values. An attempt to provision a non-announcement CLLI for an announcement prompt type results in the following error message:

  ```
  An announcement CLLI (table ANNS) must be provisioned for
  the ANNC PRMTTYPE value.
  ```

  An attempt to provision a non-tone CLLI for a TONE prompt type or TONES tone type results in the following error message:

  ```
  A tone CLLI (table TONES) must be provisioned for the TONE
  PRMTTYPE / TONES TONETYPE values.
  ```

- MIN value specified must be less than or equal to the MAX value specified. An attempt to do this results in the following error message:

  ```
  MIN value must be less than or equal to MAX value.
  ```

- If a fail action (FAILACT) of RESET is provisioned in the VALIDATE option, then the RESET option must also be present in the provisioning. An attempt to provision the reset fail action without the RESET option results in the following error message:

  ```
  Validate FAILACT action requires RESET option.
  ```

- If the INSWITCH validation type of MATCH is used, then the count of the digits to be matched against in the provisioned VALIDATE option cannot be greater than the MIN specified value. An attempt to do this results in the following error message:

  ```
  Validate MATCH digit count exceeds MIN value.
  ```

- In order to provision the CITYVAL option, the VALIDATE option must be provisioned with the FLEXVAL type of INSWITCH validation. An attempt to do this results in the following error message:

  ```
  INSWITCH validation required for CITYVAL option.
  ```

- In order to provision the CASUBLK option, the VALIDATE option must be provisioned with the FLEXVAL type of INSWITCH validation. An attempt to do this results in the following error message:

```
INSWITCH validation required for CASUBLK option.
```

All errors result in the failure of requested table control changes.

ADDR digit collectable

The Address (ADDR) digit collectable collects and validates address digits received through MF or DTMF inband tone signals. This collectable requests and removes required digits from the incoming digits buffer for processing. If the minimum amount of digits required is not available in the buffer, this collectable requests more digits from the originating agent.

Definition

Table 2-43 contains the ADDR digit collectable field refinements.

Table 2-43
ADDR digit collectable fields

Fields	Description	Values	Default
MIN	This field identifies the minimum number of digits to collect.	1 to 18	N/A
MAX	This field identifies the maximum number of digits to collect.	1 to 18	N/A

—continued—

Table 2-43
ADDR digit collectable fields (continued)

Fields	Description	Values	Default
OPTION1	This field contains a vector of options which may be added to the ADDR digit collectable definition.	Vector of up to 3 of {NIL, PROMPT, SIGFEAT} NIL, PROMPT, and SIGFEAT options are identical to the options for the COLDIG collectable.; see “COLDIG digit collectable” information on page 2-67.	N/A
OPTION2	This field contains a vector of options which may be added to the ADDR digit collectable definition.	Vector of up to 3 of {NIL, RESET, FILED, VALIDATE} ***Note:*** With the exception of the NIL option, options cannot be duplicated. NIL, RESET and FILED options are identical to the options for the COLDIG collectable.; see “COLDIG digit collectable” information on page 2-67. VALIDATE option field values are discussed in Table 2-44.	N/A

—end—

ADDR VALIDATE option

The Address Validate (ADDR VALIDATE) option contains information needed to screen the address digits within table STDPRTCT. Some of this information can be associated with a subscriber number, and therefore overridden by FLEXFEAT table provisioning.

Note: See “ADDR addressee messages” on page 4-21.

Table 2-44 contains the ADDR VALIDATE digit collectable option fields.

Table 2-44
ADDR VALIDATE digit collectable option fields

Fields	Description	Values	Default
VALTYPE	The validation type field identifies the type of validation for the collected digits.	INSWITCH ***Note:*** Address validation is performed in the switch using the STDPRTCT table. Using the ES SACREMOT feature, the address can be validated at the SCP using the N00 TCAP application.	N/A
PRTNM	If the validation type is equal to INSWITCH, then this pretranslator name field is present and identifies the index into the STDPRTCT table for screening the address digits.	An existing entry in table STDPRTCT.	N/A
FAILACT	The fail action field identifies the action to take if a validation attempt fails.	TRMT, RESET, IGNORE, DPIDX ***Note:*** For a description of the FAILACT options, see Table 2-40.	N/A
DEFERED	The DEFERED field is enabled when the FAILACT's reset should be deferred for later processing. This is only prompted when FAILACT is RESET.	N or Y	N
OVERRIDE	If the fail action field is set to TRMT, this field is present and identifies whether the treatment set overrides any previous treatment set. If this field is set to N, then any existing treatment value is not overridden.	N or Y	N/A
TRMT	If the fail action field is set to TRMT, this field is present and identifies the treatment that is applied.	Extended treatment type range ***Note:*** See *UCS DMS-250 Data Schema Reference Manual* for values of the extended treatment type.	N/A

—continued—

Table 2-44
ADDR VALIDATE digit collectable option fields (continued)

Fields	Description	Values	Default
ACTION	If the fail action field (FAILACT) is set to DPIDX, this field is present and identifies how call processing handles the collectable list specified by the FLEXDIAL index.	INSERT, APPEND, REPLACE, EXEC INSERT inserts the new list into the current processing list before the next collectable. APPEND appends the new list to the current processing list. REPLACE replaces the remainder of the unprocessed collectables with those identified by the FLEXDIAL index. EXEC executes as a sublist the list identified by the FLEXDIAL table index.	N/A
INDEX	If the fail action field (FAILACT) is set to DPIDX, this field is present and identifies the FLEXDIAL index that is processed as a result of validation failure.	An existing entry in table FLEXDIAL	N/A
	—end—		

Examples

Examples of ADDR digit collectable use:

```
ADDR 10 10 $ (VALIDATE INSWITCH UNI TRMT Y VACT) $
```

In this example, ten address digits are collected and validated. The pretranslator name used for INSWITCH validation is the UNI pretranslator table. If validation fails for the address digits, then VACT treatment is set for the call.

```
ADDR 2 4 (PROMPT ANNC MY_ANNC) (SIGFEAT 2 (0)$ 2) $ (RESET S
10 RODR PROMPT SAME PROMPT ANNC MY_RESET_ANNC) (VALIDATE
INSWITCH SPD1 RESET) $
```

In this example, an announcement prompt identified by the CLLI MY_ANNC is provided for the subscriber to collect and validate a minimum of two and maximum of four digits from the originating agent.

If the reset digit is pressed, then the announcement identified by the MY_RESET_ANNC CLLI is played as an interruptible prompt for the customer to reenter the digits. This action is taken whether or not digits are received prior to the receipt of the reset digit.

The digits are validated using the SPD1 pretranslator table. If validation fails, a reset occurs.

A special partial dial timer (PDILTMR) of two seconds occurs after reception of the second, third, and fourth digits. Also the digit zero is recognized immediately as a special first digit and reported by the terminal controller (XPM) to the central module (CM) after a special first digit timer of two seconds has expired and no other digits have been received.

ADDR collectable restrictions and limitations

The following restrictions and limitations apply for the ADDR collectable:

- Options may not be duplicated within the OPTIONS1 or OPTIONS2 vectors. An attempt to duplicate options results in the following error message:

  ```
  Cannot duplicate OPTIONS.
  ```

- The number of filed digits must not be greater than the MAX value for a particular digit collectable. An attempt to do this results in the following error message:

  ```
  Filed digit count may not exceed MAX value.
  ```

- If a PROMPT option is not specified for the collectable (either the PROMPT option or RESET PROMPT option), then the WDIGACT RESET option field cannot be set to PROMPT for a provisioned RESET option. An attempt to do this results in the following error message:

  ```
  PROMPT option must be specified for RESET PROMPT action.
  ```

- A tone (table TONES) or an announcement (table ANNS) CLLI must be provisioned for the associated PRMTTYPE/TONETYPE values. An attempt to provision a non-announcement CLLI for an announcement prompt type results in the following error message:

  ```
  An announcement CLLI (table ANNS) must be provisioned for
  the ANNC PRMTTYPE value.
  ```

  An attempt to provision a non-tone CLLI for a TONE prompt type/TONES tone type results in the following error message:

  ```
  A tone CLLI (table TONES) must be provisioned for the TONE
  PRMTTYPE / TONES TONETYPE values.
  ```

- MIN value specified must be less than or equal to the MAX value specified. An attempt to do this results in the following error message:

  ```
  MIN value must be less than or equal to MAX value.
  ```

- If a fail action (FAILACT) of RESET is provisioned in the VALIDATE option, then the RESET option must also be present in the provisioning. An attempt to provision the reset fail action without the RESET option results in the following error message:

  ```
  Validate FAILACT action requires RESET option.
  ```

All errors result in the failure of requested table control changes.

OLI digit collectable

The Originating Line Information (OLI) digit collectable is required to collect and validate information digits received through MF or DTMF inband tone signals. Typically information digits identify a characteristic of the originating agent (for example, coin phone origination) or identify a characteristic of the calling party subscriber number (for example, ANI fail). This collectable requests and removes required digits from the incoming digits buffer for processing. If the minimum amount of digits required is not available in the buffer, this collectable requests more digits from the originating agent.

Definition

Table 2-45 contains the OLI digit collectable field refinements.

Table 2-45
OLI digit collectable fields

Field	Description	Values	Default
MIN	This field identifies the minimum number of digits to collect.	1 to 3	
MAX	This field identifies the maximum number of digits to collect.	1 to 3	

—continued—

Table 2-45
OLI digit collectable fields (continued)

Field	Description	Values	Default
OPTION1	This field contains a vector of options which may be added to the OLI digit collectable definition.	Vector of up to 3 of {NIL or PROMPT} NIL and PROMPT options are identical to the definition of the same options for the. “COLDIG digit collectable” information on page 2-67.	N/A
OPTION2	This field contains a vector of options which may be added to the OLI digit collectable definition.	Vector of up to 3 of {NIL, RESET, FILED, VALIDATE} NIL, RESET and FILED options are identical to the definition of the same options for the. “COLDIG digit collectable” information on page 2-67. VALIDATE option fields are discussed in Table 2-46. ***Note:*** With the exception of the NIL option, options cannot be duplicated.	N/A
—end—			

OLI VALIDATE option

The OLI VALIDATE option contains information needed to screen the address digits within table STDPRTCT. Some of this information can be associated with a subscriber number, and therefore be overridden by FLEXFEAT table provisioning. See “OLI addressee message” on page 4-26.

Table 2-46 contains the VALIDATE OLI digit collectable option fields.

Table 2-46
VALIDATE OLI digit collectable option fields

Fields	Description	Values	Default
VALTYPE	The validation type field identifies the type of validation for the collected digits.	INSWITCH	N/A
PRTNM	If the validation type is equal to INSWITCH, then this pretranslator name field is present and identifies the index into the STDPRTCT table for screening the information digits.	An existing entry in table STDPRTCT	N/A
FAILACT	The fail action field identifies the action to take if a validation attempt fails.	TRMT, RESET, IGNORE, DPIDX ***Note:*** For a description of the FAILACT options, see Table 2-40**.**	N/A
DEFERED	The DEFERED field is enabled when the FAILACT's reset should be deferred for later processing. This is only prompted when FAILACT is RESET.	N or Y	N
OVERRIDE	If the fail action field is set to TRMT, this field is present and identifies whether the treatment set overrides any previous treatment set. If this field is set to 'N', then any existing treatment value is not overridden.	N or Y	N/A
TRMT	If the fail action field is set to TRMT, this field is present and identifies the treatment that is applied.	Extended treatment type range ***Note:*** See *UCS DMS-250 Data Schema Reference Manual* for values of the extended treatment type.	N/A

—continued—

Table 2-46
VALIDATE OLI digit collectable option fields (continued)

Fields	Description	Values	Default
ACTION	If the fail action field (FAILACT) is set to DPIDX, this field is present and identifies how call processing the collectable list specified by the FLEXDIAL index is handled.	INSERT, APPEND, REPLACE, EXEC INSERT inserts the new list into the current processing list before the next collectable. APPEND appends the new list to the current processing list. REPLACE replaces the remainder of the unprocessed collectables with those identified by the FLEXDIAL index. EXEC executes as a sublist the list identified by the FLEXDIAL table index.	N/A
INDEX	If the fail action field (FAILACT) is set to DPIDX, this field is present and identifies the FLEXDIAL index that is processed as a result of validation failure.	An existing entry in table FLEXDIAL	N/A
—end—			

Examples

Examples of OLI digit collectable use:

```
OLI 2 2 $ (VALIDATE INSWITCH EAPT TRMT Y ADBF) $
```

In this example, two information digits are collected from the originating agent and validated using the EAPT pretranslator table. If validation fails, then ADBF treatment is set for the call.

```
OLI 1 2 $ (VALIDATE INSWITCH IDPT IGNORE) $
```

In this example, a minimum of one and maximum of two digits are collected from the originating agent and validated using the IDPT pretranslator table. No action is taken if validation of the digits fails.

OLI collectable restrictions and limitations

The following restrictions and limitations apply for the OLI collectable:

- Options may not be duplicated within the OPTIONS1 or OPTIONS2 vectors. An attempt to duplicate options results in the following error message:

  ```
  Cannot duplicate OPTIONS.
  ```

- The number of filed digits must not be greater than the MAX value for a particular digit collectable. An attempt to do this results in the following error message:

  ```
  Filed digit count may not exceed MAX value.
  ```

- If a PROMPT option is not specified for the collectable (either the PROMPT option or RESET PROMPT option), then the WDIGACT RESET option field cannot be set to PROMPT for a provisioned RESET option. An attempt to do this results in the following error message:

  ```
  PROMPT option must be specified for RESET PROMPT action.
  ```

- A tone (table TONES) or an announcement (table ANNS) CLLI must be provisioned for the associated PRMTTYPE/TONETYPE values. An attempt to provision a non-announcement CLLI for an announcement prompt type results in the following error message:

  ```
  An announcement CLLI (table ANNS) must be provisioned for
  the ANNC PRMTTYPE value.
  ```

- An attempt to provision a non-tone CLLI for a TONE prompt type or TONES tone type results in the following error message:

  ```
  A tone CLLI (table TONES) must be provisioned for the TONE
  PRMTTYPE / TONES TONETYPE values.
  ```

- MIN value specified must be less than or equal to the MAX value specified. An attempt to do this results in the following error message:

  ```
  MIN value must be less than or equal to MAX value.
  ```

- If a fail action (FAILACT) of RESET is provisioned in the VALIDATE option, then the RESET option must also be present in the provisioning. An attempt to provision the reset fail action without the RESET option results in the following error message:

  ```
  Validate FAILACT action requires RESET option.
  ```

  All errors result in the failure of requested table control changes.

CIC digit collectable

The Carrier Identification Code (CIC) digit collectable collects and processes CIC digits received through MF or DTMF inband tone signals. This collectable requests and removes required digits from the incoming digits buffer for processing. If the minimum amount of digits required is not available in the buffer, this collectable requests more digits from the originating agent.

Definition

Table 2-47 contains the CIC collectable field refinements.

Table 2-47
CIC collectable fields

Fields	Description	Values	Default
MIN	This field identifies the minimum number of digits to collect.	3 to 4	N/A
MAX	This field identifies the maximum number of digits to collect.	3 to 4	N/A
OPTION1	This field contains a vector of options which may be added to the CIC collectable definition.	Vector of up to 3 of {NIL or, PROMPT} NIL and PROMPT options are identical to the definition of the same options for the. “COLDIG digit collectable” information on page 2-67.	N/A
OPTION2	This field contains a vector of options which may be added to the CIC collectable definition.	Vector of up to 3 of {NIL, RESET, FILED} NIL, RESET and FILED options are identical to the definition of the same options for the. “COLDIG digit collectable” information on page 2-67. ***Note:*** With the exception of the NIL option, options cannot be duplicated.	N/A

CIC digits are not screened for validation purposes in the same manner as other digit collectables. These digits are used to index applicable tables to identify provisioned features based on the CIC.

Examples

An example of CIC collectable use:

```
CIC 3 3 $ $
```

In this example, three CIC digits are received and processed.

CIC collectable restrictions and limitations

The following restrictions and limitations apply for the CIC collectable:

- Options may not be duplicated within the OPTIONS1 or OPTIONS2 vectors. An attempt to duplicate options results in the following error message:

  ```
  Cannot duplicate OPTIONS.
  ```

- The number of filed digits must not be greater than the MAX value for a particular digit collectable. An attempt to do this results in the following error message:

  ```
  Filed digit count may not exceed MAX value.
  ```

- If a PROMPT option is not specified for the collectable (either the PROMPT option or RESET PROMPT option), then the WDIGACT RESET option field cannot be set to PROMPT for a provisioned RESET option. An attempt to do this results in the following error message:

  ```
  PROMPT option must be specified for RESET PROMPT action.
  ```

- A tone (table TONES) or an announcement (table ANNS) CLLI must be provisioned for the associated PRMTTYPE/TONETYPE values. An attempt to provision a non-announcement CLLI for an announcement prompt type results in the following error message:

  ```
  An announcement CLLI (table ANNS) must be provisioned for
  the ANNC PRMTTYPE value.
  ```

  An attempt to provision a non-tone CLLI for a TONE prompt type or TONES tone type results in the following error message:

  ```
  A tone CLLI (table TONES) must be provisioned for the TONE
  PRMTTYPE / TONES TONETYPE values.
  ```

- MIN value specified must be less than or equal to the MAX value specified. An attempt to do this results in the following error message:

  ```
  MIN value must be less than or equal to MAX value.
  ```

All errors result in the failure of requested table control changes.

REPLDIG digit collectable

The Replace Digits (REPLDIG) digit collectable collects digits from the originating agent through MF or DTMF inband tone signals, and replaces the identified digits in the digit buffer with those collected. This collectable does not perform any validation or processing of the received digits.

Definition

Table 2-48 contains the REPLDIG digit collectable field refinements.

Table 2-48
REPLDIG digit collectable fields

Fields	Description	Values	Default
MIN	This field identifies the minimum number of digits to collect.	1 to 32	N/A
MAX	This field identifies the maximum number of digits to collect.	1 to 32	N/A
BUFLOC	This buffer location field identifies within the digit buffer where the digits that are to be replaced are located.	PREFIXED, SUFFIXED, POS ***Note:*** For a description, see Table 2-30.	N/A
BUFPOS	If the buffer location field is set to POS, then this buffer position field is present and identifies an actual location within the digit buffer where the digits that are to be replaced are located.	1 to 64	N/A
DIGCNT	This digit count field identifies the number of digits in the digit buffer that are to be replaced.	1 to 16	N/A

—continued—

Table 2-48
REPLDIG digit collectable fields (continued)

Fields	Description	Values	Default
OPTION1	This field contains a vector of options which may be added to the COLDIG digit collectable definition.	Vector of up to 3 of {NIL, PROMPT, SIGFEAT} NIL PROMPT, and SIGFEAT options are identical to the definition of the same options for the. “COLDIG digit collectable” information on page 2-67.	N/A
OPTION2	This field contains a vector of options which may be added to the COLDIG digit collectable definition.	Vector of up to 3 of {NIL, RESET, FILED} NIL, RESET and FILED options are identical to the definition of the same options for the. “COLDIG digit collectable” information on page 2-67. ***Note:*** With the exception of the NIL option, options cannot be duplicated.	N/A
—end—			

The NIL, PROMPT, SIGFEAT, RESET, and FILED options are identical to the definition of the same options for the COLDIG collectable. See “COLDIG digit collectable” information on page 2-67.

Examples

Examples of REPLDIG digit collectable use:

```
REPLDIG 3 3 PREFIXED 3 (PROMPT ANNC MY_ANNC) $ (RESET S 10
RODR PROMPT SAME PROMPT ANNC MY_RESET_ANNC) $
```

In this example, three digits are collected and replace the first three digits in the digit buffer. The announcement prompt identified by the CLLI MY_ANNC is applied as a proceed to send signal for the subscriber.

If the reset digit is pressed, then the announcement identified by the MY_RESET_ANNC CLLI is played as an interruptible prompt for the customer to reenter the digits. This action is taken whether or not digits are received prior to the receipt of the reset digit.

REPLDIG collectable restrictions and limitations

The following restrictions and limitations apply for the REPLDIG collectable:

- Options may not be duplicated within the OPTIONS1 or OPTIONS2 vectors. An attempt to duplicate options results in the following error message:

  ```
  Cannot duplicate OPTIONS.
  ```

- The number of filed digits must not be greater than the MAX value for a particular digit collectable. An attempt to do this results in the following error message:

  ```
  Filed digit count may not exceed MAX value.
  ```

- If a PROMPT option is not specified for the collectable (either the PROMPT option or RESET PROMPT option), then the WDIGACT RESET option field cannot be set to PROMPT for a provisioned RESET option. An attempt to do this results in the following error message:

  ```
  PROMPT option must be specified for RESET PROMPT action.
  ```

- A tone (table TONES) or an announcement (table ANNS) CLLI must be provisioned for the associated PRMTTYPE/TONETYPE values. An attempt to provision a non-announcement CLLI for an announcement prompt type results in the following error message:

  ```
  An announcement CLLI (table ANNS) must be provisioned for
  the ANNC PRMTTYPE value.
  ```

  An attempt to provision a non-tone CLLI for a TONE prompt type or TONES tone type results in the following error message:

  ```
  A tone CLLI (table TONES) must be provisioned for the TONE
  PRMTTYPE / TONES TONETYPE values.
  ```

- MIN value specified must be less than or equal to the MAX value specified. An attempt to do this results in the following error message:

  ```
  MIN value must be less than or equal to MAX value.
  ```

All errors result in the failure of requested table control changes.

COLPARM digit collectable

The Collect Parameter Digits (COLPARM) digit collectable extracts digits received in an out-of-band message based signaling protocol (for example, PRI or CSS7). Digits extracted are placed in the digit buffer for processing by other sequence and digit collectables. This collectable does not perform any validation or processing of the received digits.

The COLPARM digit collectable has the capability to identify a primary and secondary parameter from which to extract the digits. The secondary parameter is used only if the primary parameter is not included in the received signaling message. There is no permanent signaling or partial dial timers used as there is no involved interaction required to get the digits. If the parameters are not included in the received message, then no digits are placed in the digit buffer.

Definition

Table 2-49 contains the COLPARM digit collectable field refinements.

Table 2-49
COLPARM digit collectable fields

Fields	Description	Values	Default
PARMS	This field identifies the primary and optionally secondary parameters within which the digits are received.	Vector of up to 2 of {NAME, SUBTYPE} For PARMS definition of vector fields, see Table 2-50.	N/A
PROCNOA	This field identifies how the Nature of Address for the COLPARM application is to be processed.	IGNORE, CALLED, CALLING IGNORE identifies that no nature of address value is generated with the COLDIG application. CALLED identifies that the nature of address for the called party number is generated by the COLPARM application. CALLING identifies that the nature of address for the calling party number is generated by the COLPARM application.	N/A

—continued—

Table 2-49
COLPARM digit collectable fields (continued)

Fields	Description	Values	Default
OPTION1	This field contains a vector of options which may be added to the COLPARM digit collectable definition.	Vector of up to 3 of {NIL} The NIL option is identical to the definition of the same options for the. "COLDIG digit collectable" information on page 2-67.	N/A
OPTION2	This field contains a vector of OPTIONS which may be added to the COLPARM digit collectable definition.	Vector of up to 3 of {NIL or FILED} NIL and FILED options are identical to the definition of the same options for the. "COLDIG digit collectable" information on page 2-67. ***Note:*** With the exception of the NIL option, options cannot be duplicated.	N/A
	—end—		

No options are provisioned by entering $ when prompted for the first option.

Table 2-50 contains the PARMS vector fields.

Table 2-50
PARMS vector fields

Fields	Description	Values	Default
NAME	This field identifies the name of the message based parameter that contains the required digits.	CLD, CLG, CHG, OLI, GADDR, GDIGS, TNS, CIP, TNSCKT ***Note:*** These parameters names are listed as identified by the Q.764 protocol (CCS7), but map to their counterparts in the Q.931 protocol (PRI).	N/A
SUBTYPE	If the parameter name is generic address (GADDR) or generic digits (GDIGS), then this field is present and identifies the parameter subtype that identifies the specific parameter.	0 to 255	N/A

Examples

Examples of COLPARM digit collectable use:

```
COLPARM (GDIGS 5)$ IGNORE $ $
```

In this example, digits are retrieved from the Generic Digits parameter with a subtype value of 5. If the parameter is not present in the message, no digits are placed in the digit buffer. The NOA value associated with the received digits is ignored.

```
COLPARM (GADDR 20)(CLD)$ CALLED $ $
```

In this example, the message is first searched for a Generic Address parameter to extract the required digits. If no Generic Address parameter is found, then the Called Party Address parameter is used for the required digits. The NOA value associated with the received digits is handled as the called party NOA.

```
COLPARM (CLD)$ CALLED $ (FILED PREFIX 214)$
```

In this example, the digits in the called party address parameter of the message are extracted and placed into the digit buffer. Then the digits 214 are placed into the beginning of the digit buffer.

COLPARM collectable restrictions and limitations

At least one parameter type must be specified for the PARMS field. An attempt to provision an empty PARMS vector results in the following error message:

```
Must specify a parameter for the PARMS field.
```

SUBRPARM digit collectable

The Subscriber Number Parameter (SUBR) digit collectable is required in order to extract subscriber number digits received in a message-based signaling protocol (for example, PRI or SS7). Digits extracted are placed in the digit buffer for processing by this collectable and other sequence and digit collectables. This collectable requests and optionally removes required digits from the incoming digits buffer for processing.

The SUBRPARM digit collectable has the capability to identify a primary and secondary parameter from which to extract the digits. The secondary parameter is used only if the primary parameter is not included in the received signaling message. There are no permanent signaling or partial dial timers used as there is no involved interaction required to get the digits. If the parameters are not included in the received message, then no digits are placed in the digit buffer.

Definition

Table 2-51 contains the SUBRPARM digit collectable field refinements.

Table 2-51
SUBRPARM digit collectable fields

Fields	Description	Values	Default
PARMS	This field identifies the primary and optional secondary parameters within which the digits are received.	Vector of up to 2 of Parameter Type The PARMS field is identical to the parameter vector defined for the COLPARM digit collectable on Table 2-49.	N/A
FLEXTYPE	This field identifies the type of subscriber number being validated. This is required so that the number can be properly recorded in the billing record and also to properly index the FLEXVAL table to perform the screening of the digits (if FLEXVAL INSWITCH validation is used).	An existing entry in table FLEXTYPE	N/A
BILLCAPT	This field identifies whether to capture the collected subscriber number in the billing record. (The billing record field is identified by the FLEXTYPE table)	N or Y	N/A
REPBDIGS	This field identifies if the digits that were pulled (removed) from the incoming digits buffer for processing by this SUBRPARM digit collectable should be placed back into the buffer after processing or validation has occurred. This allows these digits to be reprocessed by another digit collectable. ***Note:*** The immediate application of this capability is PANI.	N or Y	N/A

—continued—

Table 2-51
SUBRPARM digit collectable fields (continued)

Fields	Description	Values	Default
OPTION1	This field contains a vector of options which may be added to the SUBRPARM digit collectable definition.	Vector of up to 3 of {NIL, CITYVAL, CASUBLK} The NIL option is identical to the definition of the same options for the. “COLDIG digit collectable” information on page 2-67. CITYVAL and CASUBLK options are identical to the definition of the same options for the. “SUBR collectable” information on Table 2-39.	N/A
OPTION2	This field contains a vector of options which may be added to the SUBRPARM digit collectable definition.	Vector of up to 3 of {NIL, FILED, VALIDATE} NIL and FILED options are identical to the definition of the same options for the. “COLDIG digit collectable” information on page 2-67. The VALIDATE option is identical to the definition of the same options for the. “SUBR collectable” information on Table 2-39.. ***Note:*** With the exception of the NIL option, options cannot be duplicated.	N/A
—end—			

No options are provisioned by entering $ when prompted for the first option of each vector.

Examples

Examples of the SUBRPARM digit collectable use:

```
SUBRPARM (GDIGS 10)$ ACCT Y N $ (VALIDATE INSWITCH FLEXVAL
(101 2)$ TRMT Y INAC) $
```

In this example, the subscriber number is retrieved from the Generic Digits parameter, recorded in billing, and validated. The subscriber number type is identified as an ACCT type, and only two of the received digits are validated in the FLEXVAL table using index 101. If validation fails for this subscriber number, then INAC treatment is set for the call.

```
SUBRPARM (CHG)(CLG)$ ANI Y N $ (VALIDATE INSWITCH FLEXVAL (53
10)$ TRMT Y ADBF) $
```

In this example, the subscriber number is retrieved from either the Charge Number parameter (primary), or the Calling Party Address parameter (secondary), recorded in billing, and validated. The subscriber number type is identified as an ANI type, and ten digits are validated in the FLEXVAL table using index 53. If validation fails for this subscriber number, then ADBF treatment is set for the call.

```
SUBRPARM (GDIGS 9)$ AUTH Y N (CITYVAL FAILACT) $ (VALIDATE
INSWITCH FLEXVAL (3 6)$ TRMT Y INAU) $
```

In this example, the subscriber number is retrieved from the Generic Digits parameter, recorded in the billing record, and validated.

Two forms of validation occur for this collectable. First validation occurs through the FLEXVAL table where six digits are validated using index 3. If validation fails, INAU treatment is set for the call. The second validation is a city code validation which is performed if the first validation passes. The validation attempt matches the digits from the FLEXFEAT CITYCODE option with digits from the TRKFEAT CITYCODE option. If the city code validation fails, then INAU treatment is set for the call.

SUBRPARM collectable restrictions and limitations

The following restrictions and limitations apply for the SUBRPARM collectable:

- At least one parameter type must be specified for the PARMS field. An attempt to provision an empty PARMS vector results in the following error message:

  ```
  Must specify a parameter for the PARMS field.
  ```

- A fail action (FAILACT) of RESET cannot be provisioned in the VALIDATE option. An attempt to provision the reset fail action results in the following error message:

  ```
  Cannot use RESET value for VALIDATE FAILACT action.
  ```

- In order to provision the CITYVAL option, the VALIDATE option must be provisioned with the FLEXVAL type of INSWITCH validation. An attempt to do this results in the following error message:

  ```
  INSWITCH validation required for CITYVAL option.
  ```

- In order to provision the CASUBLK option, the VALIDATE option must be provisioned with the FLEXVAL type of INSWITCH validation. An attempt to do this results in the following error message:

  ```
  INSWITCH validation required for CASUBLK option.
  ```

All errors result in the failure of requested table control changes.

ADDRPARM digit collectable

The Address Parameter (ADDRPARM) digit collectable extracts address digits received in a message-based signaling protocol (for example, PRI or CSS7). Digits extracted are placed in the digit buffer for processing by this collectable and other sequence and digit collectables. This collectable requests and optionally removes required digits from the incoming digits buffer for processing.

The ADDRPARM digit collectable has the capability to identify a primary and secondary parameter from which to extract the digits. The secondary parameter is used only if the primary parameter is not included in the received signaling message. There are no permanent signaling or partial dial timers used as there are no involved interaction required to get the digits. If the parameters are not included in the received message, then no digits are placed in the digit buffer.

Definition

Table 2-52 contains the ADDRPARM digit collectable field refinements.

Table 2-52
ADDRPARM digit collectable fields

Fields	Description	Values	Default
PARMS	This field identifies the primary and optional secondary parameters within which the digits are received.	Vector of up to 2 of Parameter Type The PARMS field is identical to the parameter vector defined for the COLPARM digit collectable on Table 2-49.	N/A
OPTION1	This field contains a vector of options which may be added to the ADDRPARM digit collectable definition.	Vector of up to 3 of {NIL} The NIL option is identical to the definition of the same options for the. “COLDIG digit collectable” information on page 2-67.	N/A
OPTION2	This field contains a vector of options which may be added to the ADDRPARM digit collectable definition.	Vector of up to 3 of {NIL, FILED, VALIDATE} NIL and FILED options are identical to the definition of the same options for the. “COLDIG digit collectable” information on page 2-67. The VALIDATE option is identical to the definition of the same options for the. “SUBR collectable” information on Table 2-39. ***Note:*** With the exception of the NIL option, options cannot be duplicated.	N/A

No options are provisioned by entering $ when prompted for the first option.

Examples

Examples of the ADDRPARM digit collectable use:

```
ADDRPARM (GADDR 20)(CLD)$ $ (VALIDATE INSWITCH EANT TRMT Y
RODR) $
```

In this example, the address digits are retrieved from either the Generic Address parameter (primary), or the Called Party Address parameter (secondary), recorded in billing, and validated. The address digits are validated in the STDPRTCT table using the EANT pretranslator name. If validation fails for this subscriber number, then RODR treatment is set for the call.

ADDRPARM collectable restrictions and limitations

The following restrictions and limitations apply for the ADDRPARM collectable:

- At least one parameter type must be specified for the PARMS field. An attempt to provision an empty PARMS vector results in the following error message:

  ```
  Must specify a parameter for the PARMS field.
  ```

- A fail action (FAILACT) of RESET cannot be provisioned in the VALIDATE option. An attempt to provision the reset fail action results in the following error message:

  ```
  Cannot use RESET value for VALIDATE FAILACT action.
  ```

All errors result in the failure of requested table control changes.

OLIPARM digit collectable

The Originating Line Information (OLIPARM) Parameter digit collectable extracts information digits received in a message-based signaling protocol (for example, PRI or SS7). Digits extracted are placed in the digit buffer for processing by this collectable and other sequence and digit collectables. This collectable requests and optionally removes required digits from the incoming digits buffer for processing.

The OLIPARM digit collectable has the capability to identify a primary and secondary parameter from which to extract the digits. The secondary parameter is used only if the primary parameter is not included in the received signaling message. There are no permanent signaling or partial dial timers used as there are no involved interaction required to get the digits. If the parameters are not included in the received message, then no digits are placed in the digit buffer.

Definition

Table 2-53 contains the OLIPARM digit collectable field refinements.

Table 2-53
OLIPARM digit collectable fields

Fields	Description	Values	Default
PARMS	This field identifies the primary and optional secondary parameters within which the digits are received.	Vector of up to 2 of Parameter Type. The PARMS field is identical to the parameter vector defined for the COLPARM digit collectable on Table 2-49.	N/A
OPTION1	This field contains a vector of options which may be added to the OLIPARM digit collectable definition.	Vector of up to 3 of {NIL} The NIL option is identical to the definition of the same options for the. “COLDIG digit collectable” information on page 2-67.	N/A
OPTION2	This field contains a vector of options which may be added to the OLIPARM digit collectable definition.	Vector of up to 3 of {NIL, FILED, VALIDATE} NIL and FILED options are identical to the definition of the same options for the. “COLDIG digit collectable” information on page 2-67. The VALIDATE option is identical to the definition of the same options for the. “SUBR collectable” information on Table 2-39. ***Note:*** With the exception of the NIL option, options cannot be duplicated.	N/A

No options are provisioned by entering $ when prompted for the first option.

Examples

Examples of the OLIPARM digit collectable use:

```
OLIPARM (OLI)$ $ (VALIDATE INSWITCH EAPT TRMT Y ADBF) $
```

In this example, the OLI digits are retrieved from the Originating Line Information parameter, recorded in billing, and validated. The information digits are validated in the STDPRTCT table using the EAPT pretranslator name. If validation fails for this subscriber number, then ADBF treatment is set for the call.

OLIPARM collectable restrictions and limitations

The following restrictions and limitations apply for the OLIPARM collectable:

- At least one parameter type must be specified for the PARMS field. An attempt to provision an empty PARMS vector results in the following error message:

  ```
  Must specify a parameter for the PARMS field.
  ```

- A fail action (FAILACT) of RESET cannot be provisioned in the VALIDATE option. An attempt to provision the reset fail action results in the following error message:

  ```
  Cannot use RESET value for VALIDATE FAILACT action.
  ```

All errors result in the failure of requested table control changes.

CICPARM digit collectable

The Carrier Identification Code Parameter (CICPARM) digit collectable extracts CIC digits received in a message-based signaling protocol (for example, PRI or CSS7). Digits extracted are placed in the digit buffer for processing by this collectable and other sequence and digit collectables. This collectable requests and optionally removes required digits from the incoming digits buffer for processing.

The CICPARM digit collectable has the capability to identify a primary and secondary parameter from which to extract the digits. The secondary parameter is used only if the primary parameter is not included in the received signaling message. There are no permanent signaling or partial dial timers used as there are no involved interaction required to get the digits. If the parameters are not included in the received message, then no digits are placed in the digit buffer.

Definition

Table 2-54 contains the CICPARM digit collectable field refinements.

Table 2-54
CICPARM digit collectable fields

Fields	Description	Values	Default
PARMS	This field identifies the primary and optional secondary parameters within which the digits are received.	Vector of up to 2 of Parameter Type The PARMS field is identical to the parameter vector defined for the COLPARM digit collectable on Table 2-49.	N/A
OPTION1	This field contains a vector of options which may be added to the CICPARM digit collectable definition.	Vector of up to 3 of {NIL} The NIL option is identical to the definition of the same options for the. “COLDIG digit collectable” information on page 2-67.	N/A
OPTION2	This field contains a vector of options which may be added to the CICPARM digit collectable definition.	Vector of up to 3 of {NIL or FILED} NIL and FILED options are identical to the definition of the same options for the. “COLDIG digit collectable” information on page 2-67. ***Note:*** With the exception of the NIL option, options cannot be duplicated.	N/A

No options are provisioned by entering the “$” when prompted for the first option.

CIC digits are not screened for validation purposes in the same manner as other digit collectables. These digits are used to index applicable tables to identify provisioned features based on the CIC.

Examples

An example of the CICPARM digit collectable use:

```
CICPARM (TNS)$ $ $
```

In this example, the CIC digits are retrieved from the transit network selector parameter and processed.

CICPARM collectable restrictions and limitations

At least one parameter type must be specified for the PARM. An attempt to provision an empty PARMS vector results in the following error message:

```
Must specify a parameter for the PARMS field.
```

All errors result in the failure of requested table control changes.

CALLTYPE call type collectable

The CALLTYPE collectable identifies features, characteristics, and billing information applicable to the call. This information is saved by call processing and affects call processing for the call as applicable.

Because the dialing plan is initially defined by the agent (via TRKGRP DPIDX field), this information can be access related. Since the CALLTYPE collectable can be provisioned anywhere within a collectable list, the feature, characteristic, and billing information can also be related to the interaction occurring with the originating agent.

Definition

Table 2-55 contains the CALLTYPE collectable field refinements.

Table 2-55
CALLTYPE collectable fields

Fields	Description	Values	Default
FLEXTYPE	This field identifies the call type name that indexes the FLEXTYPE table and identifies features, characteristics, and billing information applicable to this call type.	An existing entry in table FLEXTYPE	N/A
FLEXFEAT	This field identifies the index into the FLEXFEAT table which identifies features and characteristics applicable to this call type.	An existing entry in table FLEXFEAT	N/A

CALLTYPE collectable example

An example of the CALLTYPE collectable use are:

```
CALLTYPE CNAME1 245
```

In this example, the features, characteristics, and billing information identified by the FLEXTYPE CNAME1 and FLEXFEAT index 245 apply to the call.

CALLTYPE collectable restrictions and limitations

The FLEXFEAT index must be provisioned in the FLEXFEAT table. Otherwise the following error message results:

```
FLEXFEAT index is not datafilled in table FLEXFEAT.
```

CLRFTRS call type collectable

The Clear Features (CLRFTRS) collectable resets the features and characteristics provisioned in table FLEXFEAT (and overlapping TRKFEAT options). These features and characteristics may have been set through TRKFEAT table specifications for the originating agent, by previous validation of a subscriber number and subsequent processing of the FLEXFEAT table index, or by the CALLTYPE collectable and subsequent processing of the identified FLEXFEAT table index.

Definition

Table 2-56 contains the CLRFTRS collectable field refinement.

Table 2-56
CLRFTRS call type collectable fields

Fields	Description	Values	Default
FEATLIST	This field consists of an options vector that contains a list of the features and characteristics available in the FLEXFEAT table that are to be reset to their default values or no longer apply to the call.	NIL, ANSCDR, BCCOMPAT, CAINGRP, CASUAL, CDRTMPLT, CITYCODE, CITYVAL, CLDPBILL, DELIVER, DPIDX, FAILVAL, FLDONLY, GENLOG, MLTCOSID, MSGCTR, NOANSDUR, ONNET, PVSPDIDX, REORGACT, REORGTYP, REVALIDATE, SPLASHBK, TCAPANNC, TRANSTS, TRANSNUM, TRANSYS, IEXCLINX, FEATVAR, CPACTVAL ***Note:*** Only one of REORGACT or REORGTYP options needs to be provisioned in order to disable re-origination for the call.	N/A

The FEATLIST vector identifies all available FLEXFEAT options (and overlapping TRKFEAT options). The CLRFTRS collectable sets the identified features into an “off” state, or default value as applicable. The DPIDX, MSGCTR, and REVALIDATE options have special meaning for the CLRFTRS collectable:

- DPIDX

  Use of the DPIDX option in the CLRFTRS collectable identifies that all call processing collectable execution history maintained for the call is cleared, that is, all identification of collectables executed before the CLRFTRS collectable is erased.

  Normally this action has no effect on the call in progress, as execution of the remaining collectable list continues. This action does affect processing resets, as the collectable list would now appear to begin with the CLRFTRS collectable.

- MSGCTR

  Use of the MSGCTR option in the CLRFTRS collectable identifies that all messages posted at the call processing message center entity originating at a provisioned source (table MSGCTR) are removed, or unposted. Messages posted with a call processing source are not removed.

- REVALIDATE

  Use of the REVALIDATE option in the CLRFTRS collectable identifies that revalidation does not occur for subscriber numbers identified to be revalidated on a reoriginated call.

The NIL option can be used on tuple updates to remove options provisioned in the list. During the tuple change, replace the option with NIL to remove it from the defined list.

Example

Examples of the CLRFTRS collectable use are:

```
CLRFTRS ( REORGACT COSIDX PVSPDIDX NETSEC MSGCTR ) $
```

In this example, the four features and characteristics identified by the FEATLIST vector are reset to their default values or no longer apply to the call. In addition, all messages stored at the message center are erased.

```
CLRFTRS (ANSCDR) (CAINGRP) (MLTCOSID) (IEXCLINX)$
```

In this example, if the identified features ANSCDR, CAINGRP, MLTCOSID, and IEXCLINX are reset or cleared for the call.

CLRFTRS collectable restrictions and limitations

Duplicated options are ignored when provisioning the FEATLIST vector, and the following warning message is displayed:

```
Duplicate feature datafilled. Duplicates will be removed.
```

Only one copy of the option remains in the added or modified table entry.

SETTRANS call type collectable

The Set Translation System (SETTRANS) collectable provides a means of setting the translation system to be used for the call. Typically this would be used when pre–translation of called party address digits is not performed, and the default setting of “national” translations is inappropriate for the call.

Performing pre–translation of called party address digits also typically sets the translation system used for the call, and may overwrite the value set through the SETTRANS collectable. Likewise, the use of SETTRANS may overwrite the value set during pre–translations.

Definition

Table 2-57 contains the SETTRANS collectable field refinement.

Table 2-57
SETTRANS call type collectable field

Field	Description	Values	Default

TRANSYS	This field the translation system now applicable for the call.	NA, IN, NO, IP
		NA = National (use tables HNPACONT / FNPACONT)
		IN = International (use table CCTR)
		NO = Nil translation system
		IP = International Partitioned (use tables CTHEAD/CTCODE/CTRTE)
			N/A

Examples

An example of the SETTRANS collectable use:

```
SETTRANS IN
```

In this example, the translation system for the call is set to the international translation system.

SETTRANS collectable restrictions and limitations

There are no identified table control restrictions for the SETTRANS collectable.

SETTRMT call type collectable

The SETTRMT collectable sets the identified treatment as a delayed treatment. The APTRMT collectable must follow the SETTRMT collectable if the call should be terminated immediately.

Definition

Table 2-58 contains the SETTRMT collectable field refinements.

Table 2-58
SETTRMT call type collectable fields

Field	Description	Value	Default
TRMT	Identifies the treatment that is applied to the call.	Extended Treatment type range ***Note:*** See the *UCS DMS-250 Data Schema Reference Manual* for values of the extended treatment type.	N/A
OVERRIDE	Override the current treatment, even if it is already set.	Y or N	N/A

Examples

The following are examples of the SETTRMT collectable:

```
(SETTRMT VACT Y)
```

This example sets the vacant code treatment. However, the vacant code may still be overridden later in the call.

```
(SETTRMT PDIL N) (APTRMT)
```

This example sets the partial dial treatment and immediately terminates the call.

```
(SETTRMT UNDT Y)
```

This example clears any treatment previously set.

SETTRMT collectable restrictions and limitations

There are no identified table control restrictions for the SETTRMT collectable.

OM call type collectable

The purpose of the OM collectable is to increment a count whenever it executes. This counter can be used by system administrators to measure the frequency of execution of a dial plan.

This collectable is integrated with the standard OM facility. All tuples are in the same FLEXDIAL OM group. For information on the FLEXDIAL OM group, see Chapter 22, ”FlexDial Logs and OMs.”

Definition

Table 2-59 contains the OM call type collectable field refinement.

Table 2-59
OM call type collectable fields

Field	Description	Value	Default
DIALPLAN_ OM	Identifies an OM tuple that is initialized to 0 when first datafilled. A maximum of 1024 OM types are available.	Vector of up to 24 characters	N/A

Examples

An example of an OM collectable is shown below:

```
OM AUTH_CODE_FAIL
```

In this example the registers in the AUTH_CODE_FAIL tuple are incremented.

OM collectable restrictions and limitations

The OMGROUP associated with OM collectable only supports 1024 OM tuples. Each of these tuples may count up to 2,147,483,647 before the register is reset to 0. The tuple is implicitly deleted when no other OM collectable references it anymore. A limit of 32,767 OM collectables may reference the same tuple.

APRESET call type collectable

The purpose of the APRESET collectable is to apply the reset if it has been previously deferred. A field in the VALIDATE option’s FAILACT field (in the OLI, SUBR, and ADDR collectables) indicates whether the RESET should be deferred at the time of validation failure.

Definition

The APRESET call type collectable has no fields.

Examples

An example of the APRESET collectable is shown below:

```
(APRESET)
```

In this example, if a reset has been deferred, it is applied at this point.

APRESET collectable restrictions and limitations

A deferred reset is applied in the following cases, aside from the APRESET:

- While executing the TERMINATE and ROUTE collectables
- Before triggering a CAIN event
- While terminating the collectable list

VAROP variable collectable

The VAROP collectable is designed to provide multiple methods to assign specific or relational values to either the four FlexDial variables (AVAR, BVAR, CVAR, or DVAR), or to FLEXTYPE FEATVAR defined variables. The VAROP collectable is constructed of three components: 1) the left side operator, 2) the operation, and 3) the right side operator. For VAROP collectable application, the left side operator is modified according to the operation which incorporates the right side operator, the operation updating the value of the left side operator.

Definition

Table 2-60 contains the VAROP variable collectable field refinements.

Table 2-60
VAROP variable collectable fields

Fields	Description	Values	Default
FLEXVAR	This field identifies the FlexDial variable being updated, and identifies the left side operator for the VAROP operation.	AVAR, BVAR, CVAR, DVAR, FEATVAR AVAR, BVAR, CVAR, and DVAR represent the four generic FlexDial variables. FEATVAR represents us of the FLEXTYPE FEATVAR table option.	
FLEXTYPE	If the FLEXVAR field is set to FEATVAR, then this field is present and identifies the FLEXTYPE associated with the desired FEATVAR.	A valid entry to table FLEXTYPE.	
OPER	This field identifies the operation taking place on the left side operator.	ASG, INCR, DECR, MULT, DIV, MOD ASG indicates that the value of the right operator is assigned to the left operator. INCR indicates that the left operator is incremented by the amount of the right operator. DECR indicates that the left operator is decremented by the amount of the right operator. MULT indicates that the value of the left operator is multiplied by the value of the right operator. DIV indicates that the value of the left operator is divided by the value of the right operator. MOD indicates that the value of the left operator is MOD’d by the value of the right operator.	

—continued—

Table 2-60
VAROP variable collectable fields (continued)

Fields	Description	Values	Default
RHS_TYPE	This field identifies the type of the right side operator for the VAROP operation.	VARIABLE, INTEGER, CALLDUR	
INTVAL	If the RHS_TYPE field is set to INTEGER, then this field is present and identifies the integer value of the right side operator.	–1073741824 to 1073741823	
FLEXVAR	If the RHS_TYPE field is set to VARIABLE, then this field is present and identifies a FlexDial variable or FLEXTYPE FEATVAR variable as the right side operator.	AVAR, BVAR, CVAR, DVAR, FEATVAR	
FLEXTYPE	If the RHS_TYPE field is set to variable and the FLEXVAR field is set to FEATVAR, then this field is present and identifies the FLEXTYPE associated with the desired FEATVAR.	A valid entry to table FLEXTYPE.	
UNITS	If the RHS_TYPE field is set to CALLDUR, then this field is present and identifies what units for the call duration value are to be used, defining the value for the right side operator.	MINUTES, SECONDS	
—end—			

Examples

The following are examples of the VAROP collectables:

```
(VAROP AVAR ASG INTEGER 2)
```

This example sets AVAR to a value of 2.

```
(VAROP AVAR INCR INTEGER 2)
```

This example adds 2 to AVAR.

```
(VAROP BVAR ASG VARIABLE AVAR)
```

This example assigns AVAR to BVAR.

```
(VAROP CVAR MULT INTEGER 2
```

This example multiplies CVAR by a value of 2.

```
(VAROP CVAR DIV INTEGER 2)
```

This example divides CVAR by a value of 2.

```
VAROP FEATVAR ANI ASG INTEGER 400
```

In this example, if the FEATVAR option provisioned in table FLEXFEAT for the ANI FLEXTYPE is assigned a value of 400.

```
VAROP AVAR DECR CALLDUR SECONDS
```

In this example, if the AVAR variable is decremented by the call duration in seconds.

VAROP collectable restrictions and limitations

The VAROP variable collectable has the following restrictions and limitations:

- References to uninitialized variables or overflow for INCR, MULT, DECR, DIV, and MOD operations cause a FNAL treatment and generate the FLEX 307 log.
- The current value can be overridden by a message in table MSGCTR.

IFVAR variable collectable

The IFVAR collectable tests the current value for one of the program variables. If the comparison passes the test, the provisioned index replaces the list. If the comparison fails the test, the remainder of the collectable list is executed. References to uninitialized variables cause a FNAL treatment.

Definition

Table 2-61 contains the IFVAR variable collectable field refinements.

Table 2-61
IFVAR variable collectable fields

Fields	Description	Values	Default
COMPMETH	Determines whether the test case passes on FAIL or SUCCESS.	IS or NOT	N/A
CONDITION	Vector that includes up to 4 conditionals.	Vector up to 4 conditions of {VAR, RELATION, VALUE} See table 2-62 for CONDITION field values.	N/A
ACTION	An enumeration of the actions to be taken on either the TRUE branch or the ELSE branch.	INSERT, APPEND, REPLACE, EXEC INSERT inserts the new list into the current processing list before the next collectable. APPEND appends the new list to the current processing list. REPLACE replaces the remainder of the unprocessed collectables with those identified by the FLEXDIAL index. EXEC executes as a sublist the list identified by the FLEXDIAL table index.	N/A
IFTRUE	The INDEX field identifies the FLEXDIAL table index that is to be processed if the IFVAR comparison is successful.	An existing FLEXDIAL table INDEX field.	N/A
IFFALSE	The index field identifies the FLEXDIAL table index that is to be processed if the IFVAR comparison is FALSE.	An existing entry to table FLEXDIAL.	N/A

Table 2-62 contains the IFVAR collectable CONDITION fields.

Table 2-62
IFVAR variable collectable condition fields

Field	Description	Value	Default
VAR	An enumeration for one of the four variables.	AVAR, BVAR, CVAR, DVAR, FEATVAR AVAR, BVAR, CVAR, and DVAR represent the four generic FlexDial variables. FEATVAR represents use of the FLEXFEAT FEATVAR table option.	N/A
FLEXTYPE	If the VAR field is set to FEATVAR, then this field is present and identifies the FLEXTYPE associated with the desired FEATVAR.	A valid entry to table FLEXTYPE.	N/A
RELATION	Indicates how the fields are compared.	EQL, NEQ, LSS, LEQ, GTR, GEQ	N/A
VALUE	Tests the value for the variable.	–1,073,741,824 to 1,073,741,823	N/A

Up to four IFVAR expressions may be identified in the VAR vector. When multiple expressions are specified, the logic for the comparisons is as shown in 16-3:

Figure 16-3
Logic For IFVAR Expression Comparisons

IS Comparison Method:

Result = (ExprA = True) or (ExprB = True) or (ExprC = True) or (ExprD = True)

NOT Comparison Method:

Result = (ExprA ^= True) and (ExprB ^= True) and (ExprC ^= True) and (ExprD ^= True)

Note: ^= means ‘not equal to’

If the result of the expression evaluation according to the comparison method is true, then the IFTRUE FLEXDIAL table index is used. Otherwise, the IFFALSE FLEXDIAL table index is used.

Examples

The following are examples of the IFVAR collectable:

```
IFVAR IS (FEATVAR ANI LSS 60)$ INSERT L60IDX GE60IDX
```

In this example, if the FEATVAR option provisioned in table FLEXFEAT for the ANI FLEXTYPE contains a value less than 60, then the collectables provisioned against FLEXDIAL index L60IDX are inserted into the current collectable list. Otherwise if the value is equal to or greater than 60, the collectables provisioned against FLEXDIAL index GE60IDX are inserted into the current collectable list.

```
(IFVAR (AVAR LSS 10) $ EXEC AVAR_LSS_10 NIL)
```

In this example, if the AVAR variable is less than 10, then this will branch to AVAR_LSS_10.

IFVAR collectable restrictions and limitations

The following restrictions and limitations apply:

- The FLEXDIAL index must exist first.
- Comparisons involving uninitialized variables will result in FNAL treatment and generation of the FLEX 307 log.

System requirements

The FLEXDIAL table allocates store on an as needed basis for up to 32696 table entries. Store is allocated in blocks of up to 20480 bytes, where each block represents between 200 and 5120 provisioned collectables. A maximum of 18.0 Mbytes of store can be allocated for all 32696 table entries. At initialization, 3520 bytes are allocated for table store.

Datafill order

Collectables in the FLEXDIAL table are dependent on provisioning in the following tables:

- MODDIGS collectable
- STDPRTCT (ADDR, ADDRPARM, OLI, OLIPARM collectables)
- CLLI (digit collectables SHROPTS PROMPT option)
- FLEXTYPE (SUBR, SUBRPARM, CALLTYPE collectables)
- FLEXVAL (SUBR, SUBRPARM collectables)

Restrictions and limitations

The following restrictions and limitations apply for table FLEXDIAL:

- The FLEXDIAL table provides a maximum of 32,696 entries. An attempt to provision more than 32,696 entries results in the following error message:

  ```
  Table FLEXDIAL is full. No more entries possible.
  ```

- All interactions with the originating agent are defined through the FLEXDIAL table. Interactions with the originating agent are not defined outside of the FLEXDIAL table.
- All entries in the FLEXDIAL table defined prior to and including the index defined by the FLEXDIAL_ACCESS_INDEX are restricted and cannot be modified or deleted unless the read/write protected office parameter FLEXDIAL_ACCESS_CONTROL is set. An attempt to do so results in the following error message:

  ```
  FLEXDIAL entries before the index identified by the
  FLEXDIAL_ACCESS_INDEX office parameter may not be modified
  unless the FLEXDIAL_ACCESS_CONTROL office parameter is set
  to N.
  ```

- Indexes in the FLEXDIAL table may not be deleted if they are referenced in other FLEXDIAL entries or other tables. An attempt to delete a used index results in the following error message:

  ```
  TUPLE REFERENCED BY ANOTHER TABLE - USE TABREF TO GET
  POTENTIAL TABLE LIST.
  ```

- At least one collectable must be provisioned against a FLEXDIAL table index. An attempt to provision an empty index results in the following error message:

  ```
  Must specify a collectable for the optionS field.
  ```

- It is not possible to delete or alter the NIL table index. An attempt to delete or alter the NIL table index results in the following error message:

  ```
  Cannot modify the NIL tuple.
  ```

- It is not possible to solely provision the NIL option for a table entry. An attempt to datafill the option vector with only NIL options (for example, KEY (NIL) (NIL) (NIL)$ N) results in the following error message:

  ```
  Write Error - Cannot datafill only NIL Options.
  ```

All errors result in the failure of requested table control changes.

Table FLEXMOD

This chapter describes the FlexDial Digit Modification (FLEXMOD) table.

Purpose

The FLEXMOD table provides a provisioning scheme in the FlexDial framework for incoming digit manipulation. The MODDIGS sequence collectable accesses this table to change received digits before they are processed by digit collectables.

General layout

The table is indexed to retrieve a digit vector that replaces the digits defined by the MODDIGS sequence collectable. This manipulation scheme meets the operating company requirements simply and effectively.

The FLEXMOD table is indexed by a 16-character string (1 to 16 characters), and a digit vector containing the digits to be replaced. This key retrieves a tuple that consists of a second digit vector that provides the replacement digits. The two digit vectors do not have to be identical in length.

Key

The key for indexing the FLEXMOD table consists of a two-part key:

- 16-character string

  This string value provides a unique index into the FLEXDIAL table through use of the data dictionary. Due to limitations of the data dictionary, this part of the key is limited to 32K variations.

- Digit vector containing up to 16 of {0–9, A, B, C, D, S, P}

  This digit vector contains up to 16 digits used to index the table. A maximum of 100,000 unique digit vectors are supported by each 16-character string.

Entries in table FLEXMOD are ordered according to the following rules:

- Entries are first listed by order of the 16-character strings in the data dictionary.
- Entries for a particular 16-character string are listed by the following order of digits:

  P, S, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D

The maximum table size is 12 million entries, although the actual storage capability of the table is limited by system memory.

Table 3-1 provides the FLEXMOD table key.

Table 3-1
FLEXMOD table key

Key field	Description	Values
NAME	The first part of the table key consists of a string of up to 16 characters. 32K possible unique entries are provided.	Vector of up to 16 characters
DIGITS	The second part of the key consists of a vector of up to 16 digits. Theoretically there can be an unlimited number of digit vectors provisioned against a unique NAME, but the overall table size is limited to 12 million entries. A, B, C, D represent 4th column DTMF digits. S represents the asterisk digit, and P represents the octothorpe digit.	Vector of up to 16 of {S, P, 0–9, A, B, C, D}

Fields

The FLEXMOD table tuples contain one field as outlined in Table 3-2.

Table 3-2
FLEXMOD table fields

Field	Description	Values	Default
DIGITS	This field identifies the digits that replace those identified by the MODDIGS sequence collectable. A, B, C, D represent 4th column DTMF digits. S represents the asterisk digit, and P represents the octothorpe digit.	Vector of up to 16 of {S, P, 0–9, A, B, C, D}	N/A

System requirements

The FLEXMOD table allocates data store on an as needed basis for a maximum of 12 million table entries. Store is allocated in blocks of 47,014 bytes, where each block represents 2048 provisioned entries. A maximum of 450 Mbytes of store can be allocated for all 12 million table entries. No store is allocated unless an entry is provisioned in the table.

Datafill order

The FLEXMOD table does not require the presence of any other table datafill.

The FLEXMOD table must be datafilled with necessary indexes for use by the MODDIGS FlexDial collectable and the TRKFEAT/FLEXFEAT MSGCTR option.

Dump and restore

This table does not require a dump and restore from any other tables.

Restrictions and limitations

The following restrictions and limitations apply for table FLEXMOD:

- The 16-character NAME index, which is the first half of the FLEXDIAL table key, provides a maximum of 32K table entries. An attempt to provision more than 32K NAME indexes results in the following error message:

  ```
  Table FLEXMOD limit reached. No more unique string indexes
  possible.
  ```

- There can be a maximum of 100,000 digit vector key indexes against a single NAME key index. An attempt to datafill more than this maximum results in the following error message:

  ```
  Table FLEXMOD limit reached. No more digit entries against
  the string index possible.
  ```

- Size of the FLEXMOD table is limited to 12 million entries. An attempt to provision more than 12 million entries results in the following error message:

  ```
  Table FLEXMOD is full. No more entries are possible.
  ```

- Indexes in the FLEXMOD table may not be deleted if they are referenced in other tables. An attempt to delete a referenced index results in the following error message:

  ```
  Cannot remove referenced FLEXMOD tuple.
  ```

All errors result in the failure of requested table control changes.

Table MSGCTR

This chapter describes the FlexDial Message Center (MSGCTR) table.

Purpose

The MSGCTR table provides a provisioning scheme in the FlexDial framework for identifying the transfer of information from one collectable to another. This table is an extension to the provisioning schemes of tables TRKFEAT and FLEXFEAT, and can identify specific information associated with the originating trunk group, call type, or subscriber number used for collectable processing.

The MSGCTR table identifies processing information for the specified collectable type, and relates necessary information to the collectable that it needs to properly process the call.

The information being related to the collectable takes precedence over information provisioned with the collectable (in the FLEXDIAL table). At the time of processing the subscriber number collectable or trunk group features, it is unknown if the related collectable is to be processed for the call. (For example, we have information for an ADDR collectable, but do not know if an ADDR collectable exists or will exist in the collectable list to be processed.) Therefore, this information is contained in a “posted message” that is left with the message center call processing entity for the identified collectable. When a collectable begins processing, it first checks the message center for applicable posted messages.

Messages are posted at the message center in the order that they are provisioned.

This option identifies messages that a collectable recognizes when querying the message center for messages.

General layout

The MSGCTR table is indexed by a numeric value to retrieve a vector of messages that are to be posted in call processing due to use of the trunk group, call type, or particular subscriber number. This allows a general layout of dialing plan schemes in table FLEXDIAL, with specific details being filled in at run-time by the posted messages.

Key

Table 4-1 provides the key to the table, which is a simple, 24-bit numeric value providing approximately 16.8 million unique values (although this upper boundary is limited by system memory).

Table 4-1
MSGCTR table key

Key field	Description	Values
INDEX	The key consists of a numeric value providing 16.8 million unique indexes.	0 to 16,777,215

Fields

The MSGCTR table tuples contain one field as outlined in Table 4-2.

Table 4-2
Table MSGCTR field

Field	Description	Values	Default
ADDRESS	This field consists of a vector of messages to post. A message is first identified by the address of the message, which identifies the collectable type that has pre-defined messages that are recognized.	COLDIG, SUBR , ADDR, OLI, MODDIGS, REPLDIG, VAROP or NIL	N/A

Each collectable type identifies its own set of defined messages. All the messages follow a basic outline:

- address (the ADDRESS field above)
- message type—the reason for the message
- data—the actual information being transferred

Each addressee specifies unique, provisionable messages.

NIL option

The NIL option provides a way to update the message vector. NIL can be inserted instead of removing a message by retyping the provisioned messages that follow. Using NIL allows you to remove the inserted message without having to retype the remaining provisioned messages.

The NIL option does not remain visible once the update is complete.

Note: This NIL option is identical to the existing NIL option for TRKGRP OPTIONS vector provisioning (see the *UCS DMS-250 Data Schema Reference Manual).*

Definition

The NIL option does not contain any specific field refinements.

Examples

An example showing the use of NIL option follows:

1 Observe the example of an existing tuple that is to be edited:

```
MY_IDX (MESSAGE1) (MESSAGE2) (MESSAGE3)$
```

2 To remove MESSAGE2, change the table entry. When prompted at MESSAGE2, enter NIL:

```
MESSAGE: MESSAGE2
```

>NIL

3 After editing, display of the tuple shows:

```
MY_IDX (MESSAGE1) (MESSAGE3)$
```

COLDIG addressee messages

The COLDIG addressee defines three types of messages for the ADDR and ADDRPARM digit collectables—FILED, MINMAX, and PROMPT— and sends information to the COLDIG or COLPARM digit collectable.

Note: A MINMAX message is not processed by a COLPARM collectable. Only the FILED message can be processed by a COLPARM collectable.

The FILED message type allows a variation on the use of filed digits instead of collecting the raw information from the originating agent.

The MINMAX message type identifies the minimum and maximum number of digits for COLDIG collectable processing, overriding the information provisioned in the COLDIG collectable. This message is ignored by COLPARM digit collectables.

The PROMPT message type is used to modify the prompt played by a COLDIG digit collectable. The MSGCTR PROMPT option fields have a similar definition to the PROMPT option for the digit collectables.

Fields

Table 4-3 contains the field refinements in the COLDIG message type data fields.

Table 4-3
Table MSGCTR COLDIG message data fields

FIelds	Description	Values	Default
DOMINANT	This field indicates that the message provisioned is the last message of this specific message type that can be processed by an individual COLDIG/COLPARM collectable.	N or Y	N/A
REPOST	This field indicates that after the message is processed by the collectable, it must be reposted at the message center for the next COLDIG/COLPARM collectable. A message is always reposted as a non-dominant message.	N or Y	N/A
MSGTYPE	If the ADDRESS is set to COLDIG, then this field identifies the pre-defined message types that the COLDIG and COLPARM digit collectables recognize.	FILED, MINMAX, PROMPT	N/A

FILED message

If the ADDRESS is set to COLDIG and the message type is set to FILED, then the fields in Table 4-4 identify the data for the FILED message.

Table 4-4
Table MSGCTR COLDIG addressee FILED message data fields

Fields	Description	Values	Default
FILEDTYP	This field identifies how the filed digits for the run-time message are determined. Either the digits are provisioned in the message being provisioned, or they are determined at run-time.	DIGS, VALDIGS, CALLING DIGS identifies that the digits for the run-time message are provisioned in the DIGITS field. VALDIGS identifies that the digits for the run-time message are the digits that are validated by the subscriber number collectable. CALLING identifies that the digits for the run-time message are taken from the received calling party address digits.	N/A
DIGITS	When the FILEDTYP field is set to DIGS, this field is present and identifies the filed digits to be used for the received address digits of the ADDR digit collectable. A, B, C, and D represents fourth column DTMF digits. S represents the asterisk digit, and P represents the octothorpe (#)digit.	Vector of up to 18 of (0–9, A, B, C, D, S, P)	N/A
BUFLOC	This field identifies the location for the digit collectable digits where the filed digits are placed, and is used for all values of the FILEDTYP field.	PREFIX or SUFFIX	N/A

The VALDIGS FILED message type only applies to FILED messages being processed through execution of a subscriber number collectable. A message being processed through TRKFEAT or the CALLTYPE collectable does not contain any digits at run-time.

When subscriber number digits are being validated at multiple indexes, then the digits validated by both indexes are included in the FILED message.

A message processed using the CALLING message type contains digits at run-time only if a subscriber number collectable was previously processed, where the subscriber number type used is provisioned with the CALLING FLEXTYPE table option.

The digits for a VALDIGS FILED message must be available during run-time execution at the time the message is being posted at the message center. The digits for a CALLING FILED message must be available during run-time execution at the time the message is being retrieved from the message center.

MINMAX message

If the ADDRESS is set to COLDIG and the message type is set to MINMAX, then the fields in Table 4-5 identify the data for the MINMAX message.

Table 4-5
Table MSGCTR COLDIG addressee MINMAX message data fields

Fields	Description	Values	Default
MIN	This field identifies the minimum number of digits that the COLDIG collectable must collect.	1 to 18	N/A
MAX	This field identifies the maximum number of digits that he COLDIG collectable must collect.	1 to 18	N/A

PROMPT message

If the ADDRESS is set to COLDIG and the message type is set to PROMPT, then the fields in Table 4-6 identify the data for the PROMPT message.

Table 4-6
Table MSGCTR COLDIG addressee PROMPT message data fields

Fields	Description	Values	Default
HASPROMPT	Indicates whether a prompt should be used.	Y or N If HASPROMPT is set to Y, then the prompt type in the MSGCTR message overrides any previously set prompt type for the collectable. When HASPROMPT is set to N, the digit collectable ignores any datafilled PROMPT option and does not play a prompt.	N
PREDISTN	The tone to pulse converter disable tone field identifies if the special tone pulse is sent before actually applying the prompt tone or announcement.	N or Y	N/A
PRMTTYPE	This field is present when HASPROMPT is Y. This field identifies how the type of prompt is set.	TONE or ANNC	N/A
CLLI	This field identifies the tone or announcement prompt that is played for the user as a request for digits indication. This field is visible if one of the following conditions are met: The PRMTYTYPE is set to ANNC. The PRMTTYPE s set to TONE and TONETYPE is set to TONES.	TONE or ANNOUNCEMENT CLLI	N/A
TONETYPE	If PRMTTYPE is TONE, then this field is present and is used to indicate the type of tone prompt to be used.	TONES, STD	N/A

—continued—

Table 4-6
Table MSGCTR COLDIG addressee PROMPT message data fields (continued)

Fields	Description	Values	Default
STDTONE	If PRMTTYPE is TONE and TONETYPE is STD, then this field is present and identifies the type of standard tone that is used.	CD, SD, SSD, H, L, 0–9, A, B, C, D, S, P CD identifies carrier dial tone (400 Hz). SD identifies standard dial tone (350 + 440 Hz). SSD identifies a tone consisting of a short burst of stuttered dial tone followed by standard dial tone. H identifies a high tone (480 Hz). L identifies a low tone (480 + 620 Hz). 0–9 represent the associated DTMF digit tones. A, B, C, D refer to the 4th column DTMF digits. S refers to the DTMF asterisk digit. P refers to the DTMF octothorpe digit.	N/A
TONEDUR	If PRMTTYPE Is TONE and TONETYPE is STD, then this field is present and identifies the duration of the tone that is being played.	(1 to 25) in 10 millisecond ticks. The value of 255 indicates that a continuous tone is to be applied.	N/A
—end—			

Examples

Examples of the COLDIG message include:

```
COLDIG N N MINMAX 10 10
```

In this example, the minimum and maximum number of digits to process for the COLDIG collectable is ten. This message is not dominant, so another MINMAX message type can be processed by the same COLDIG collectable. This message is also not reposted after it is retrieved and used.

```
COLDIG Y N FILED DIGS 2145555511 PREFIX
```

In this example, filed digits of 2145555511 are identified for the next processing COLDIG or COLPARM digit collectable. This message is dominant, so this is the last FILED message that can be processed by the particular COLDIG or COLPARM digit collectable. This message is also not reposted after it is retrieved and used. This message is also not reposted after it is retrieved and used.

```
COLDIG Y N PROMPT Y N TONE STD H 50
```

This is an example of a dominant message instructing a COLDIG collectable to play a standard high-tone prompt of 500 milliseconds duration.

SUBR addressee messages

The SUBR Addressee supports eight types of messages—MINMAX, FILED, PROMPT, VALIDATE, MATCH, INDEXES, FAILACT, and FEATIGN—to send information to the SUBR and SUBRPARM digit collectables.

The MINMAX message identifies the minimum and maximum number of digits for the SUBR collectable processing the identified subscriber number type, overriding the information provisioned in the SUBR collectable. This message is ignored by the SUBRPARM digit collectables.

Note 1: If the REPOST indicator is set for the MINMAX message, the message is still reposted by a SUBRPARM collectable.

Note 2: The MINMAX and PROMPT messages are not processed by the SUBRPARM collectable.

The FILED message allows for the use of filed digits, instead of collecting raw information from the originating agent.

The PROMPT message is used to modify the prompt played by a SUBR digit collectable. The MSGCTR PROMPT option fields have a similar definition to the PROMPT option for the digit collectables. This message is ignored by the SUBRPARM digit collectables.

Note: If the REPOST indicator is set for the PROMPT message, the message is still reposted by a SUBRPARM collectable.

The VALIDATE message, in conjunction with the FAILACT message and either the MATCH or INDEXES message, allows call processing events to determine how subscriber digits are validated. The VALIDATE message overrides the VALIDATE option provisioned for the SUBR/SUBRPARM collectables in table FLEXDIAL.

The MATCH message identifies the subscriber number digits to be used for validation.

The INDEXES message identifies the table FLEXVAL indexes to be used for validation of the identified subscriber number type.

The FAILACT message identifies a fail action to replace the FAILACT provisioned for the SUBR/SUBRPARM collectable in table FLEXDIAL.

The FEATIGN message identifies a list of table FLEXFEAT features that the SUBR or SUBRPARM collectable processing must ignore.

Fields

Table 4-7 contains the field refinements in the SUBR message.

Table 4-7
Table MSGCTR SUBR message data fields

Fields	Description	Values	Default
SUBRTYPE	If the ADDRESS is set to SUBR, then this field is present and identifies the subscriber number type that this message is for. This field is an extension to the address for the SUBR or SUBRPARM collectable type.	An existing entry in table FLEXTYPE	N/A
DOMINANT	This field identifies that the message provisioned is the last message of this specific message type that can be processed by an individual SUBR/SUBRPARM <SUBRTYPE> collectable.	N or Y	N/A

—continued—

Table 4-7
Table MSGCTR SUBR message data fields (continued)

Fields	Description	Values	Default
REPOST	This field identifies that after the message is processed by the collectable, it must be reposted at the message center for the next SUBR/SUBRPARM <SUBRTYPE> collectable. A message is always reposted as a non-dominant message.	N or Y	N/A
MSGTYPE	If the ADDRESS is set to SUBR, then this field identifies the pre-defined message types that the SUBR and SUBRPARM digit collectables recognize.	MINMAX, FILED, PROMPT, VALIDATE, MATCH, INDEXES, FAILACT, FEATIGN	N/A
—end—			

MINMAX message

If the ADDRESS is set to SUBR and the message type is set to MINMAX, then the fields in Table 4-8 identify the data for the MINMAX message.

Table 4-8
Table MSGCTR SUBR MINMAX message data fields

Fields	Description	Values	Default
MIN	If the ADDRESS is set to SUBR and the MSGTYPE is set to MINMAX, then this field is present and identifies the minimum number of digits that must be collected by the SUBR collectable.	1 to 16	N/A
MAX	If the ADDRESS is set to SUBR and the MSGTYPE is set to MINMAX, then this field is present and identifies the maximum number of digits that the SUBR collectable must collect.	1 to 16	N/A

FILED message

The SUBR FILED message is identical to the FILED message definition for the COLDIG address. See “FILED message” on page 4-4 for more information.

PROMPT message

The SUBR PROMPT message is identical to the PROMPT message definition for the COLDIG address. See Table 4-6 for more information.

VALIDATE message

If the ADDRESS is set to SUBR and the message type is set to VALIDATE, then the fields in Table 4-9 identify the data for the VALIDATE message.

Table 4-9
Table MSGCTR SUBR VALIDATE message data fields

Fields	Description	Values	Default
ENFORCE	This field identifies whether to perform validation on the subscriber digits. If set to ‘N’, then no validation is performed.	N, Y	N/A
VALTYPE	If the ENFORCE field is set to ‘Y’, this field is present and identifies the type of validation that is to occur for the collected digits.	INSWITCH	N/A

The VALIDATE message allows call processing events to determine how the subscriber digits are validated. It can activate or deactivate subscriber number validation for a given FLEXTYPE. However, the VALIDATE message alone is unable to activate subscriber number validation. It must be accompanied by a FAILACT message and either an INDEXES or MATCH message. But, the VALIDATE message alone can deactivate subscriber number validation. Hence, a VALIDATE N message alone is valid and is always processed. See “VALIDATE message restrictions and limitations.”

VALIDATE message restrictions and limitations

- If validation is turned on through the VALIDATE message, it must be accompanied by a FAILACT message to specify the fail action to be executed and either an INDEXES or a MATCH message to specify required validation information.
- If a SUBR or SUBRPARM collectable that does not have a VALIDATE option datafilled in table FLEXDIAL receives a VALIDATE Y message without also receiving a FAILACT message and either an INDEXES or MATCH message, the VALIDATE Y message will be consumed but not used and a FLEX301 trouble log is generated.
- If both VALIDATE Y and INDEXES messages are received by a SUBR or SUBRPARM collectable that did not have a VALIDATE option provisioned in table FLEXDIAL, the number of digits validated in table FLEXVAL will be the number of digits received by the SUBR or SUBRPARM collectable.

MATCH message

If the ADDRESS is set to SUBR and the message type is set to MATCH, then the field in Table 4-10 identifies the data for the MATCH message.

Table 4-10
Table MSGCTR SUBR addressee MATCH message data field

Field	Description	Values	Default
DIGITS	If the ADDRESS is set to SUBR and the MSGTYPE is set to MATCH, then this field identifies the subscriber number digits to be used for validation purposes as identified by the SUBR or SUBRPARM collectable. A, B, C, and D represent fourth column DTMF digits. S represents the asterisk digit, and P represents the octothorpe (#) digit.	Vector of up to 16 of {0–9, A, B, C, D, S, P}	N/A

Note: The presence of the MATCH message does not cause validation to occur. See “MATCH message restrictions and limitations.”

MATCH message restrictions and limitations

- The MATCH message is only valid when validation is turned on. If the subscriber number collectable does not identify a VALIDATE option and no VALIDATE message is received, then the MATCH message is consumed, but not used by the collectable.
- Both the MATCH and INDEXES messages override the validation type provisioned in table FLEXDIAL. If both messages are received by the SUBR or SUBRPARM collectable, the last message received is the validation type used.

INDEXES message

If the ADDRESS is set to SUBR and the message type is set to INDEXES, then the field in Table 4-11 identifies the data for the INDEXES message.

Table 4-11
Table MSGCTR SUBR addressee INDEXES message data field

Field	Description	Values	Default
INDEXES	If the ADDRESS is set to SUBR and the MSGTYPE is set to INDEXES, then this field is present and contains a vector of up to two indexes that are used for validation of the identified subscriber number type.	Vector of up to 2 of valid index range of the FLEXVAL table ***Note:*** This is a value within the valid range of the table defined key. You do not have to actually provision the FLEXVAL table index.	N/A

Note 1: The datafill trigger for the Early Answer Supervision feature is the “(SUBR UAX N N INDEXES (4) $)” MSGCTR datafill setup for the trunk group. Without this added message, universal access (UA) will occur without early answer. With the message, UA occurs with early answer using the indexes datafilled in tables FLEXDIAL, FLEXFEAT and FLEXVAL. For a data provisioning example of an Early Answer Supervision feature, refer to AX0300 in the *UCS DMS-250 Software Release Document for UCS08*. For more information on the Early Answer Supervision feature, see the *Feature Group D (FGD) Application Guide*.

Note 2: The presence of the INDEXES message does not cause validation to occur. See “INDEXES message restrictions and limitations.”

INDEXES message restrictions and limitations

- The INDEXES message is only valid when validation is turned on. If the subscriber number collectable does not identify a VALIDATE option and no VALIDATE message is received, then the INDEX message is consumed, but not used by the collectable.
- The actual number of indexes used for validation depends on the provisioning of the VALIDATE option for the SUBR or SUBRPARM digit collectable. The length of the SUBRIDXS vector (the number of digits to validate at each index) identifies how many indexes are used for validation purposes.
- If the SUBR or SUBRPARM digit collectable identifies validation at two separate indexes, and only one index is defined in the INDEXES message, the index is used for both table FLEXVAL screening attempts. If the collectable identifies that validation against a single index is to occur, and two indexes are defined in this message type, then only the first index is used for the validation attempt.
- Both INDEXES and MATCH messages override the validation type provisioned in table FLEXDIAL. If both messages are received by the SUBR or SUBRPARM collectable, the last message received is the validation type used.
- If both VALIDATE Y and INDEXES messages are received by a SUBR or SUBRPARM collectable that did not have a VALIDATE option provisioned in table FLEXDIAL, the number of digits validated in table FLEXVAL will be the number of digits received by the SUBR or SUBRPARM collectable.

FAILACT message

If the ADDRESS is set to SUBR and the message type is set to FAILACT, then the fields in Table 4-12 identify the data for the FAILACT message.

Table 4-12
Table MSGCTR SUBR FAILACT message data fields

Fields	Description	Values	Default
FAILACT	If the ADDRESS is set to SUBR and the MSGTYPE is set to FAILACT, then this field is present and identifies the action to take if a validation attempt fails (the subscriber number is not found or the MATCH failed). If a validation failure occurs, the proper FLEX trouble log is generated and the proper OMs are pegged.	TRMT, RESET, IGNORE, DPIDX TRMT indicates that if the validation attempt fails, then treatment is applied to the call. RESET indicates that if the validation attempt fails, then the digit collectable resets according to the RESET option specifications. IGNORE indicates that a failed validation attempt is ignored by call processing and the call proceeds as if validation passed. DPIDX indicates that the interaction identified by the FLEXDIAL index is performed.	N/A
OVERRIDE	If the FAILACT field is set to TRMT, this field is present and identifies whether the treatment set overrides any previous treatment set.	N, Y If this field is set to N, the existing treatment value is not overridden.	N/A
TRMT	If the FAILACT field is set to TRMT, this field is present and identifies the treatment that is applied.	Extended treatment type range.	N/A

—continued—

Fields	**Description**	**Values**	**Default**
ACTION	If the FAILACT field is set to DPIDX, this field is present and identifies how the collectable list specified by the FLEXDIAL index is handled by call processing.	INSERT, APPEND, REPLACE INSERT inserts the new list into the current processing list before the next collectable. APPEND appends the new list to the current processing list. REPLACE replaces the remainder of the unprocessed collectables with those identified by the FLEXDIAL index.	N/A
INDEX	If the FAILACT field is set to DPIDX, this field is present and identifies the FLEXDIAL index that is processed as a result of validation failure.	An existing entry in table FLEXDIAL	N/A
DEFERED	If the FAILACT field is set to RESET, this field is present and identifies whether the RESET is deferred for later processing.	N, Y	N/A
CDRFLG	If the FAILACT field is set to IGNORE, this field is present and identifies the CDR field that may be set if there is a failure that is ignored.	NONE, CICCASU NONE indicates that no additional CDR field is set when there is a validation failure for the subscriber digits. CICCASU indicates that if the subscriber digits fail validation, the CICCASU field will be set.	NONE

—end—

FAILACT message restrictions and limitations

- The FAILACT message is only valid when validation is turned on. If the subscriber number collectable does not identify a VALIDATE option and no VALIDATE message is received, then the FAILACT message is consumed, but not used by the collectable.
- If a FAILACT message is retrieved by a SUBR or SUBRPARM collectable that does not have a VALIDATE option provisioned in table FLEXDIAL and no VALIDATE message is retrieved, the FAILACT message is consumed but not used.
- If a FAILACT RESET message is received by a SUBR or SUBRPARM collectable that does not have a RESET option provisioned in table FLEXDIAL, the message is consumed but not used and a FLEX301 trouble log is generated.

FEATIGN message

If the ADDRESS is set to SUBR and the message type is set to FEATIGN, then the fields in Table 4-13 identify the data for the FEATIGN message.

Table 4-13
Table MSGCTR SUBR FEATIGN message data fields

Fields	Description	Values	Default
FEATLIST	This field consists of a vector of table FLEXFEAT features that are ignored by the current subscriber number collectable.	See the FLEXFEAT option on page 7-3.	N/A
METHOD	This field defines how this message will interact with any previous FEATIGN messages that may have been processed by this SUBR or SUBRPARM collectable.	{ REPLACE, UNION, INTRSECT, YIELD } REPLACE – The features included with this message replace any features included in a previous FEATIGN message. UNION – A union is formed with the features included with this message and any features that were included in previous FEATIGN messages. The result of the union becomes the current list of features to ignore. INTRSECT – An intersection is formed with the features included with this message and any features that were included in previous FEATIGN messages. The result of the intersection becomes the current list of features to ignore. YIELD – This message is only processed if no features have been specified through a previous FEATIGN message.	N/A

All features that can be provisioned in the FEATIGN message have a 1:1 correlation with an option in table FLEXFEAT. Each SUBR and SUBRPARM collectable maintains a set of FLEXFEAT ‘features to be ignored’. Before any messages have been processed, the set of ‘features to be ignored’ is empty. As FEATIGN messages are retrieved, the set is modified by the features in the FEATIGN message.

FEATIGN message restrictions and limitations

- At least one feature must be provisioned with the FEATIGN message type. If an attempt to provision the FEATIGN option without any features is made, the following error message is displayed:

```
At least one feature must be provided with the FEATIGN
option.
```

- FEATIGN features may not be duplicated. If an attempt to provision the same feature more than once is made, the duplicate is removed and the following warning is displayed:

```
Warning: Duplicate feature will be removed.
```

- Since the FLEXFEAT features REORGACT and REORGTYP must be provisioned as a pair, the existence of either REORGACT or REORGTYP in a FEATIGN message implies that both of the associated FLEXFEAT features should be ignored.

Examples

Examples of the SUBR message include:

```
SUBR PIN N N MATCH 23
```

In this example, the digits 23 are identified for a match validation to occur for the next SUBR or SUBRPARM digit collectable processing a PIN type of subscriber number. This message is not dominant, so another MATCH message can be processed by the same SUBR or SUBRPARM collectable. This message is not reposted after it is received and used.

```
SUBR ACCT Y N INDEXES (100 101)$
```

In this example, the indexes 100 and 101 are identified for inswitch FLEXVAL validation to occur for the next SUBR or SUBRPARM digit collectable processing an ACCT type of subscriber number. This message is dominant, so this is the last INDEXES message that can be processed by the particular SUBR or SUBRPARM ACCT collectable. This message is not reposted after it is received and used.

```
1003 (SUBR AUTH N N PROMPT Y N TONE STD H 255) $
```

In this example, the SUBR or SUBRPARM AUTH collectable receiving the message will play a standard high-tone prompt of infinite duration–potentially overriding any previously set prompt.

```
(SUBR ANI N N VALIDATE N)
```

In this example, the SUBR or SUBRPARM ANI collectable is instructed not to validate the subscriber digits.

```
(SUBR PIN Y N VALIDATE Y INSWITCH) (SUBR PIN Y N INDEXES (5)
$) (SUBR PIN Y N FAILACT TRMT Y INAC) $
```

In this example, the SUBR or SUBRPARM PIN collectable is instructed to validate the incoming digits using table FLEXVAL. Table FLEXVAL index 5 will be used to validate all subscriber digits. If validation fails, then apply the INAC treatment—overriding any previous treatment that was set.

```
(SUBR ANI N Y FEATIGN (TRANSTS) (ITRANSTS) $ UNION)
```

In this example, the SUBR or SUBRPARM ANI collectable is instructed to ignore the TRANSTS and ITRANSTS options from table FLEXFEAT. The UNION method instructs the collectable processing the message to form a union of the TRANSTS and ITRANSTS features with any previously defined FEATIGN features.

ADDR addressee messages

The ADDR addressee defines five types of messages for the ADDR and ADDRPARM digit collectables—OPER, FILED, PRTNM, MINMAX, and PROMPT—and sends information to the ADDR or ADDRPARM digit collectable.

The OPER message redefines the information that the ADDR or ADDRPARM collectable would normally use when screening or setting up a position route for an operator call type. If the OPER message is not waiting for the collectable when it begins processing, then special handling for operator calls is not supported.

The FILED message type allows a variation on the use of a filed address where the subscriber number identifies the address digits to use for the call. In this manner, the FLEXDIAL table need not be provisioned with subscriber specific information.

The PRTNM message type defines a pretranslator name for the ADDR or ADDRPARM collectable to use in place of its own defined VALIDATE option pretranslator name.

If the address collectable does not identify a VALIDATE option, then the PRTNM message is consumed but not used by the collectable. The presence of the PRTNM message does not cause validation to occur if it is not provisioned with the collectable.

The MINMAX message type identifies the minimum and maximum number of digits for the ADDR collectable processing, overriding the information provisioned in the ADDR collectable. This message is consumed but ignored by ADDRPARM digit collectables.

The PROMPT message type is used to modify the prompt played by an ADDR digit collectable. The MSGCTR PROMPT option fields have a similar definition to the PROMPT option for the digit collectables.

Fields

Table 4-14 contain the field refinements for the ADDR message.

Table 4-14
Table MSGCTR ADDR addressee message data fields

Fields	Description	Values	Default
DOMINANT	This field identifies that the message provisioned is the last message of this specific message type that can be processed by an individual ADDR or ADDRPARM collectable.	N or Y	N/A
REPOST	This field identifies that after the message is processed by the collectable, it must be reposted at the message center for the next ADDR/ADDRPARM collectable. A message is always reposted as a non-dominant message.	N or Y	N/A
MSGTYPE	If the ADDRESS is set to ADDR, then this field identifies the pre-defined message types that the ADDR and ADDRPARM digit collectables recognize.	OPER, FILED, PRTNM, MINMAX, PROMPT	N/A

OPER message

If the ADDRESS is set to ADDR and the message type for the ADDR addressee is set to OPER, then the fields in Table 4-15 identify the data for the OPER message.

Table 4-15
Table MSGCTR ADDR addressee OPER message data fields

Fields	Description	Values	Default
RTETYPE	This field identifies if the routing method the operator call use is NORMAL (use of an office route table and table STDPRTCT), or via the OPERRTE or OPCHOICE table.	NORMAL, OPERRTE, OPCHOICE	N/A
ZEROM	If the route method is set to NORMAL, then this field is present and identifies the office route used for routing zero minus calls.	NONE, OFRT, OFR2, OFR3, OFR4 each with a defined key value	N/A
RTEIDX	If the ZEROM route is set to OFRT, OFR2, OFR3, or OFR4, then this field is present and identifies the office route index.	0 to 1023	N/A
ZEROP	If the route method is set to NORMAL, then this field is present and identifies the pretranslator name for the STDPRTCT table that screens zero-plus (0+) and zero-one-plus (01+) traffic.	An existing entry in table STDPRTCT	N/A
INTOA	If the route method is set to NORMAL, then this field is present and identifies the office route used for routing zero-one-plus (01+) international operator-assisted calls identified by OLI collectable processing.	NONE, OFRT, OFR2, OFR3, OFR4 each with a defined key value	N/A

—continued—

Table 4-15
Table MSGCTR ADDR addressee OPER message data fields (continued)

Fields	Description	Values	Default
RTEIDX	If the INTOA route is set to OFRT, OFR2, OFR3, or OFR4, then this field is present and identifies the office route index.	0 to 1023	N/A
CARRNUM	If the route method is set to OPERRTE, then this field is present and identifies the carrier number index into table OPERRTE to retrieve the route list.	1 to 9999	N/A
OPCHIDX	If the route method is set to OPCHOICE, this field is present and identifies the index into table OPCHOICE to retrieve the route information.	1 to 255	N/A

—end—

ADDR OPER message restrictions and limitations

The OPCHIDX need not be provisioned in table OPCHOICE. If no data exists at the specified index when the ADDR or ADDRPARM collectable processes a 0– or INTOA call, the ADDR or ADDRPARM collectable will apply a delayed VACT treatment to the call. For 0+ calls the ADDR or ADDRPARM collectable will proceed with validation using table STDPRTCT and the pretranslator name it would have used for non–operator calls.

FILED message

The ADDR FILED message is identical to the FILED message definition for the COLDIG address. See “FILED message” on page 4-12.

PRTNM message

If the ADDRESS is set to ADDR and the message type for the ADDR addressee is set to PRTNM, then the field in Table 4-16 identifies the data for the PRTNM message.

Table 4-16
Table MSGCTR ADDR addressee PRTNM message data field

Field	Description	Values	Default
PRTNM	This field identifies the pretranslator name that screens the address in the STDPRTCT table.	An existing entry to table STDPRTCT	N/A

MINMAX message

The ADDR MINMAX message is identical to the MINMAX message definition for the COLDIG address. See “COLDIG addressee messages” on page 4-3 for more information.

PROMPT message

The ADDR PROMPT message is identical to the PROMPT message definition for the COLDIG address. See Table 4-6 for more information.

Examples

Examples of the ADDR message include:

```
ADDR N Y OPER OPERRTE 283
```

In this example, the carrier number of 283 is identified for the next processing ADDR or ADDRPARM digit collectable. If the call is an operator call type, this carrier number is used to index the OPERRTE table and identify a route for the call. This message is not dominant, so another OPER message can be processed by the same ADDR or ADDRPARM collectable. After being retrieved and processed by the address collectable, this message is reposted at the call processing MSGCTR for subsequent address collectable use.

```
ADDR Y N FILED DIGS 2145555511
```

In this example, filed digits of 2145555511 are identified for the next processing ADDR or ADDRPARM digit collectable. This message is dominant, so this is the last FILED message that can be processed by the particular ADDR or ADDRPARM collectable. This message is not reposted after it is retrieved and used.

```
1004 (ADDR N N PROMPT Y N ANNC ENTER_DIGITS) $
```

In this example, the ADDR collectable receiving the message will play an announcement datafilled in table ANNS called ENTER_DIGITS before collecting digits. As with all FLEXDIAL announcement prompts, the announcement will be interrupted by the first digit entered by the subscriber.

```
ADDR N Y OPER OPCHOICE 12
```

In this example, table OPCHOICE index 12 is identified to be used to process 0–, 0+, and INTOA calls.

OLI addressee message

The OLI addressee defines three types of messages for the OLI and OLIPARM digit collectables: FILED, PRTNM, and PROMPT.

The FILED message indicates that the identified digits should be used as the received originating line information digits for the OLI or OLIPARM digit collectable.

The PRTNM message type defines a pretranslator name for the OLI or OLIPARM collectable to use in place of its own defined VALIDATE option pretranslator name.

If the OLI digit collectable does not identify a VALIDATE option, then the PRTNM message is consumed, but not used by the collectable. The presence of the PRTNM message does not cause validation to occur if it is not provisioned with the collectable.

The PROMPT message type is used to modify the prompt played by an OLI digit collectable. The MSGCTR PROMPT option fields have a similar definition to the PROMPT option for the digit collectables.

Fields

Table 4-17 contains the field refinements in the OLI message.

Table 4-17
Table MSGCTR OLI addressee message data fields

Fields	Description	Values	Default
DOMINANT	This field identifies that the message provisioned is the last message of this specific message type that can be processed by an individual OLI or OLIPARM collectable.	N or Y	N/A
REPOST	This field identifies that after the message is processed by the collectable, it must be reposted at the message center for the next OLI/OLIPARM collectable. A message is always reposted as a non-dominant message.	N or Y	N/A
MSGTYPE	If the ADDRESS is set to OLI, then this field identifies the pre-defined message types that the OLI and OLIPARM digit collectables recognize.	FILED, PRTNM, PROMPT	N/A

FILED message

The OLI FILED message is identical to the FILED message definition for the COLDIG address. See “COLDIG addressee messages” on page 4-3 for more information.

PRTNM message

The OLI PRTNM message is identical to the PRTNM message definition for the ADDR address. See “PRTNM message” on page 4-24 for more information.

PROMPT message

The OLI PROMPT message is identical to the PROMPT message definition for the COLDIG address. See Table 4-6 for more information.

Examples

Examples of the OLI message include:

```
OLI N Y PRTNM IDPT
```

In this example, the pretranslator IDPT is used to validate the identified originating line information digits. This message is not dominant, so another PRTNM message can be processed by the same OLI or OLIPARM collectable. After being retrieved and processed by the OLI or OLIPARM collectable, this message is reposted at the call processing MSGCTR for subsequent originating line information collectable use.

```
OLI Y N FILED DIGS 63
```

In this example, filed digits of 63 are identified for the next processing OLI or OLIPARM digit collectable. This message is dominant, so this is the last FILED message that can be processed by the particular OLI or OLIPARM collectable. This message is not reposted after it is retrieved and used.

```
OLI Y N PROMPT N
```

This is an example of a dominant message indicating that OLI should not play a prompt, even if it has a PROMPT option provisioned.

MODDIGS addressee message

The MODDIGS addressee defines one type of message for the MODDIGS sequence collectables, the INDEX message type.

The INDEX message identifies the FLEXMOD table index for the MODDIGS collectable.

Fields

Table 4-18 contains the field refinements for the MODDIGS message.

Table 4-18
Table MSGCTR MODDIGS addressee message data fields

Fields	Description	Values	Default
DOMINANT	This field identifies that the message provisioned is the last message of this specific Message type that can be processed by an individual MODDIGS collectable.	N or Y	N/A
REPOST	This field identifies that after the message is processed by the collectable, it must be reposted at the message center for the next MODDIGS collectable. A message is always reposted as a non-dominant message.	N or Y	N/A
MSGTYPE	If the ADDRESS is set to MODDIGS, then this field identifies the pre-defined message types that the MODDIGS collectable recognizes.	INDEX	N/A

INDEX message

If the ADDRESS is set to MODDIGS and the message type for the MODDIGS addressee is set to INDEX, then the field in Table 4-19 identifies the data for the INDEX message.

Table 4-19
Table MSGCTR MODDIGS addressee INDEX message data field

Field	Description	Values	Default
INDEX	If the MESSAGE is set to MODDIGS and Message type is set to INDEX, then this field is present and identifies the index into the FLEXMOD table.	An existing entry in the FLEXMOD table	N/A

Examples

Examples of the MODDIGS message include:

```
MODDIGS N N INDEX IFGDTRANS
```

In this example, the FLEXMOD table index IFGDTRANS is used for the next processing MODDIGS sequence collectable. This message is not absolute, so another INDEX message can be processed by the same MODDIGS collectable. This message is not reposted after it is retrieved and used.

REPLDIG addressee message

The REPLDIG addressee defines three types of messages for the REPLDIG digit collectable: FILED, MINMAX, and PROMPT.

The FILED message indicates that the identified digits should be used in the replace action performed by the REPLDIG digit collectable.

The MINMAX message type identifies the minimum and maximum number of digits for REPLDIG digit collectable processing, overriding the information provisioned with the REPLDIG collectable.

The PROMPT message type is used to modify the prompt played by a REPLDIG digit collectable. The MSGCTR PROMPT option fields have a similar definition to the PROMPT option for the digit collectables.

Fields

Table 4-20 contains the field refinements for the REPLDIG message.

Table 4-20
Table MSGCTR REPLDIG addressee message data fields

Fields	Description	Values	Default
DOMINANT	This field identifies that the message provisioned is the last message of this specific Message type that can be processed by an individual REPLDIG digit collectable.	N or Y	N/A
REPOST	This field identifies that after the message is processed by the collectable, it must be reposted at the message center for the next REPLDIG collectable. A message is always reposted as a non-dominant message.	N or Y	N/A
MSGTYPE	If the ADDRESS is set to REPLDIG, then this field identifies the pre-defined message types that the REPLDIG digit collectable recognize.	FILED, MINMAX, PROMPT	N/A

FILED message

The REPLDIG FILED message is identical to the FILED message definition for the COLDIG address. See “COLDIG addressee messages” on page 4-3 for more information.

MINMAX message

The REPLDIG MINMAX message is identical to the MINMAX message definition for the COLDIG address. See “COLDIG addressee messages” for more information.

PROMPT message

The REPLDIG PROMPT message is identical to the PROMPT message definition for the COLDIG address. See Table 4-6 for more information.

Examples

Examples of the REPLDIG message include:

```
REPLDIG N N MINMAX 3 3
```

In this example, the MIN and MAX values that the next REPLDIG collectable executed are both set to 3, overriding information provisioned in the collectable. This message is not absolute, so another MINMAX message can be processed by the same REPLDIG collectable. This message is not reposted after it is retrieved and used.

```
REPLDIG Y N FILED CALLING
```

In this example, the identified filed digits for the next REPLDIG collectable executed come from the received calling party subscriber number for the call. This message is dominant, so this is the last FILED message that can be processed by the particular REPLDIG collectable. This message is not reposted after it is retrieved and used.

```
1002 (REPLDIG Y N PROMPT N) $
```

In this example, the REPLDIG collectable will not play a prompt, regardless of whether the digit collectable has a PROMPT option datafilled with it in table FLEXDIAL.

VAROP addressee message

The VAROP addressee message is able to set the value of one of the four FLEXDIAL variables (AVAR, BVAR, CVAR, DVAR) through the application of a TRKFEAT or FLEXFEAT MSGCTR index.

Each tuple in table MSGCTR references a vector up to 12 messages. The VAROP message is posted to the VAROP mailbox; however, it is only retrieved by a VAROP collectable with the same variable referenced in the message.

The VAROP message is received by the VAROP collectable and sets the specified variable’s value before the VAROP collectable takes any other actions.

The VAROP addressee defines one type of message for the VAROP variable collectable, the VALUE message type.

Fields

Table 4-21 contains the field refinements for the VAROP message.

Table 4-21
Table MSGCTR VAROP addressee message data fields

Fields	Description	Values	Default
DOMINANT	This field identifies that the message provisioned is the last message of this specific message type that can be processed by an individual VAROP variable collectable.	N or Y	N/A
REPOST	This field identifies that after the message is processed by the collectable, it must be reposted at the message center for the next VAROP collectable. A message is always reposted as a non-dominant message.	N or Y	N/A
MSGTYPE	If the ADDRESS is set to VAROP, then this field identifies the pre-defined message types that the VAROP variable collectable recognizes.	VALUE	N/A
VARID	This is an enumeration of all the variables to which the value will be sent.	AVAR, BVAR, CVAR, DVAR	N/A

VALUE message

If the ADDRESS is set to VAROP and the message type for the VAROP addressee is set to VALUE, then the field in Table 4-22 identifies the data for the VALUE message.

Table 4-22
Table MSGCTR VAROP addressee VALUE message data field

Field	Description	Values	Default
VALUE	Actual value to be stored in the variable.	–1,073,741,824 to 1,073,741,823	N/A

Examples

Examples of the VAROP message include:

```
1001 (VAROP AVAR Y N VALUE 100) $
```

In this example, a dominating message sets the value of the AVAR variable to 100.

System requirements

The MSGCTR table allocates store on an as needed basis for a maximum of 16.8 million table entries (although this upper boundary is limited by system memory). Store is allocated dynamically per entry, based on storage requirements of the number of messages provisioned. A minimum of 24 bytes or a maximum of 404 bytes can be allocated per table entry.

Datafill order

The MSGCTR table depends on provisioning in the STDPRTCT, FLEXMOD, and FLEXTYPE tables.

The MSGCTR table must be provisioned with necessary indexes for use by the TRKFEAT and FLEXFEAT table MSGCTR option.

Dump and restore

This table does not require a dump and restore from any other tables.

Restrictions and limitations

The following restrictions and limitations apply for table MSGCTR:

- The MSGCTR table is limited to 16.8 million unique entries. An attempt to provision more than 16.8 million entries results in the following error message:

  ```
  Table MSGCTR is full. No more entries are possible.
  ```

- Indexes in the MSGCTR table may not be deleted if they are referenced in other tables. An attempt to delete a referenced index results in the following error message:

  ```
  Tuple is still referenced.
  ```

- A maximum of 12 messages can be provisioned for a table entry.
- The indexes must exist within tables CLLI, TONES, ANNS, and FLEXDIAL, as appropriate.
- References made to uninitialized FLEXDIAL variables (AVAR, BVAR, CVAR, and DVAR) will generate a FLEX log indicating the failure to properly initialize the variable.
- In call processing, use of the MSGCTR VAROP message and/or the VAROP collectable ASG (assign) functionality is required before any other operations are made on a FLEXDIAL variable (such as an IFVAR comparison or a VAROP collectable INCR operation). This is not enforced when the table is datafilled; only when the tuple is executed.

All errors result in the failure of requested table control changes.

Table FLEXTYPE

This chapter describes the FlexDial Call Type Definitions (FLEXTYPE) table.

Purpose

The FLEXTYPE table provides the specification of subscriber number types (such as ANI or AUTH) and call type definitions (for use by the CALLTYPE collectable) within the FlexDial framework. The names for subscriber number types provisioned in this table are used to associate digits collected by the SUBR and SUBRPARM digit collectables with validation of those digits in the FLEXVAL table along with the defined features and characteristics in the FLEXFEAT table. This table also identifies the billing capture information in the billing record for the subscriber number type or call type definition.

The subscriber number type also identifies a unique class of operational measurement (OM) registers that peg the usage of this subscriber number type.

Note: Pegged information includes validation attempts and validation failures.

General layout

The FLEXTYPE table identifies the subscriber number type or call type definition. The indexed tuple contains a list of options that identify features and characteristics of the type definition.

Key

Table 5-1 contains the key to the table, which is a 16 character string (1 to 16 characters). A maximum of 1024 indexes are provided by the FLEXTYPE table key.

Table 5-1
FLEXTYPE table key

Key field	Description	Values
KEY	The table key consists of a string of up to 16 characters. 1024 possible unique entries are provided.	Vector of up to 16 characters

Fields

The FLEXTYPE table does not contain a fixed set of fields, but consists of an options vector that is provisioned with any of the defined features and characteristics that apply generally to the subscriber number type or call type definition. Each option in turn defines its own data refinement for datafill purposes. Therefore each option is listed independently.

The default field for the options identifies the value that is assumed for the feature options if this option is not explicitly provisioned. Normally, when a tuple consists of an options vector, the default is to not have any options provisioned. Due to the nature of feature specifications, some feature options require a default value if the option is not provisioned in the list.

When provisioning the field of an option, a default value for the field type can be specified by the DEFDATA table. Table 5-2 contains the FLEXTYPE field.

Table 5-2
Table FLEXTYPE field

Field	Description	Values	Default
OPTIONS	This field consists of an options vector which contains a list of the features and characteristics generally applicable to a subscriber number type.	NIL, ANSCDR, BILLFLD, BILLFLGS, CALLING, CDRTMPLT, CIC, EMPTYIDX, FLEXLOG, OPERDISP, CAINFLG, REVALIDATE ***Note:*** The listed options can only be provisioned once within the vector.	N/A

NIL option

The NIL option provides a way to update the options vector. NIL can be inserted, instead of removing an option by retyping the provisioned options which follow. Using NIL enables the inserted option to be effectively removed without having to retype the remaining provisioned options.

Note: This NIL option is identical to the existing NIL option for TRKGRP OPTIONS vector provisioning (see the *UCS DMS-250 Data Schema Reference Manual).*

The NIL option does not remain visible once the update is complete.

Definition

The NIL option does not contain any specific field refinements.

Examples

An example showing the use of NIL option is:

- Editing existing tuple:

  ```
  MY_IDX (OPTION1) (OPTION2) (OPTION3)$
  ```

- To remove OPTION2, change the table entry. When prompted at OPTION2, enter NIL:

  ```
  OPTION: OPTION2
  > NIL
  ```

- After editing, display of the tuple shows:

  ```
  MY_IDX (OPTION1) (OPTION3)$
  ```

NIL option restrictions and limitations

There are no identified table control restrictions for the NIL option.

ANSCDR option

The CDR on Call Answer (ANSCDR) option identifies that this type of subscriber number or call type definition requires a CDR to be generated upon answer of the call.

Fields

Table 5-3 contains the field refinements for the ANSCDR option.

Table 5-3
Table FLEXTYPE ANSCDR option fields

Fields	Description	Values	Default
TMPLTIDX	This field identifies the index into the CDRTMPLT table that is used for formatting the CDR record generated.	An existing entry in the CDRTMPLT table	N/A
USEEDIT	This field identifies that the active version of the template in the CDRTMPLT table is not used. Instead, the version of the template at the editing index is used for formatting the CDR record for the call. ***Note:*** The USEEDIT field exists for testing purposes only. It should always set to N for billable call traffic on a live network switch.	N or Y	N/A

Examples

An example of the ANSCDR option provisioning is:

```
ANSCDR ANSCDR_IDX N
```

This example identifies that the CDR on answer feature is active and that the template used to generate the CDR is the active version identified by the ANSCDR_IDX CDRTMPLT index.

ANSCDR option restrictions and limitations

The ANSCDR option does not have any identified table control restrictions.

BILLFLD option

The Billing Field (BILLFLD) option identifies the field in the billing record where the subscriber number digits are captured. If this option is not present, the defined subscriber number type is never captured for billing.

If the defined field is too small to completely capture all of the subscriber number digits collected for the defined subscriber number type, then the collected digits are truncated to fit within the field boundaries.

If two or more subscriber number types identify the same field for capturing the subscriber number digits in the billing record, and the APPEND action is used, then each subscriber number is appended to the numbers already captured in the field, until the field is full.

Fields

Table 5-4 contains the field refinement for the BILLFLD option.

Table 5-4
Table FLEXTYPE BILLFLD option fields

Field	Description	Values	Default
RECFIELD	This field identifies the field in the billing record where the subscriber number type digits are captured.	ANISP, BILLNUM, PINDIGS, ACCTCD, CLGPTYNO, CIC SUBRNUM1, SUBRNUM2, UNIVACC	N/A
ACTION	This field identifies whether or not the processed subscriber number digits are appended to or replace possible digits already stored in the billing record field.	APPEND or REPLACE	N/A

Examples

Example of the BILLFLD option provisioning are:

```
BILLFLD SUBRNUM1 REPLACE
```

This example identifies that the subscriber number being defined is captured in the SUBRNUM1 billing record field, and replaces any digits that may have already been captured.

```
BILLFLD CIC REPLACE
```

This example identifies the subscriber number being defined is captured in the CIC billing record field, and replaces any digits which may have already been captured.

BILLFLD option restrictions and limitations

The following restriction and limitation applies to the BILLFLD CIC option:

- The CIC field in the billing record is only populated with the subscriber digits if the SOC UTRS0001 is turned ON.

The following restrictions and limitations only apply to CAIN/FlexDial interactions. For more information see *UCS DMS-250 CAIN/FlexDial Interactions*.

- CAIN assigns specific meanings to the various fields of the CDR. Therefore, the BILLFLD option in table FLEXTYPE should be provisioned for use with CAIN-capable AXXESS agents in such a way as to be consistent with the CAIN use of the CDR fields.
- For non-ANI billing numbers, the BILLNUM value of the BILLFLD option must be used in table FLEXTYPE for CAIN to recognize the number as a billing number.

BILLFLGS option

The Billing Flags (BILLFLGS) option identifies billing information that is captured in the CALLTYPE field of the CDR when the subscriber number type or call type definition is used in call processing.

With the more generic processing in the FlexDial framework, the switch software contains less information about features and the meaning of events that are occurring. This option enables you to identify certain access types, service types, and call types for the billing record that apply to the use of the subscriber number type or call type definition.

This option does not affect call processing. The information identified here is captured in the billing record.

Definition

Table 5-5 contains the field refinements of the BILLFLGS option.

Table 5-5
Table FLEXTYPE BILLFLGS option fields

Fields	Description	Values	Default
ACCESS	This field identifies if the access value captured in the CDR CALLTYPE field is to be changed.	NOCHANGE or CHANGE	N/A
VALUE	If the ACCESS field is set to CHANGE, then this field is present and identifies the new access value for the CDR CALLTYPE field.	0 to 31	N/A
SERVICE	This field identifies if the specific service type value captured in the CDR CALLTYPE field is to be changed.	NOCHANGE or CHANGE	N/A
VALUE	If the SERVICE field is set to CHANGE, then this field is present and identifies the new service type value for the CDR CALLTYPE field.	0 to 63	N/A
CALLTYPE	This field identifies if the call type value captured in the CDR CALLTYPE field is to be changed.	NOCHANGE or CHANGE	N/A
VALUE	If the CALLTYPE field is set to CHANGE, then this field is present and identifies the new call type value for the CDR CALLTYPE field.	0 to 31	N/A

Examples

An example of the BILLFLGS option use:

```
BILLFLGS CHANGE 1 CHANGE 2 CHANGE 3
```

In this example, an access value of 1, a service value of 2, and a call type value of 3 are captured in the billing record CALLTYPE field.

BILLFLGS option restrictions and limitations

There are no itemized restrictions for the BILLFLGS option.

CALLING option

The Calling Party Number (CALLING) option identifies that the subscriber number type represents the calling party number for call processing. There are specific call processing actions and features dealing with received calling party address digits that requires the calling party subscriber number to be uniquely identified for call processing.

When interacting with CAIN, FlexDial identifies the ANI for a call through the use of the CALLING option. ANIs do not have their own CAINFLG parameter because a separate CAINFLG is redundant. The FlexDial collected ANIs are used in both trigger criteria checking and outgoing message parameter population. For more information see *UCS DMS-250 CAIN/FlexDial Interactions*.

Fields

The CALLING option does not contain any field refinements. The presence of the option indicates that this particular subscriber number type represents the calling party number for call processing. The absence of the option indicates that this subscriber number type does not represent the calling party number.

Examples

Examples of the CALLING option provisioning are:

```
CALLING
```

The presence of this option identifies that this particular subscriber number type represents the calling party number for call processing.

```
ANI (BILLFLD ANISP REPLACE) (CALLING) $
```

In this example, CAIN interprets the calling party number as a billing number.

CALLING option restrictions and limitations

The CALLING option does not have any identified table control restrictions.

CDRTMPLT option

The CDR Template field identifies the table CDRTMPLT defined template to be used to format the CDR for the call.

Fields

Table 5-6 contains the field refinements for the CDRTMPLT option.

Table 5-6
Table FLEXTYPE CDRTMPLT option fields

Fields	Description	Values	Default
BILLACT	This field indicates whether billing is active for the call.	N or Y	Y
TMPLTIDX	This field identifies the index into the CDRTMPLT table used for formatting the CDR record for the call. This field is only present when the BILLACT field is set to Y.	An existing entry in the CDRTMPLT table	N/A
USEEDIT	This field identifies that the active version of the template in the CDRTMPLT table is not used. Instead, the version of the template at the editing index is used for formatting the CDR record for the call. This field is only present when the BILLACT field is set to Y. ***Note:*** The USEEDIT field exists for testing purposes only. It should always set to N for billable call traffic on a live network switch.	N or Y	N/A

Examples

An example of the CDRTMPLT option provisioning is:

```
CDRTMPLT Y UCS12 N
```

This example identifies that the template used is the active template identified by the UCS12 template index.

CDRTMPLT option restrictions and limitations

The CDRTMPLT option does not have any identified table control restrictions.

CIC option

The CIC option identifies that the subscriber number type represents a Carrier Identification Code (CIC) for call processing. When SUBR/SUBRPARM encounters the CIC option in table FLEXTYPE, it will process the incoming digits as a CIC if the SOC UTRS0001 is turned ON, otherwise the SUBR/SUBRPARM will ignore the CIC option.

Fields

The CIC option does not contain any field refinements. The presence of the option indicates that this particular subscriber number type represents the Carrier Identification Code for call processing. The absence of the option indicates that this subscriber number type does not represent the CIC.

Example

An example of the CIC option is:

```
CIC
```

CIC option restrictions and limitations

The following limitations and restrictions apply to the CIC option:

- The CIC option may not coexist with the CALLING option in a FLEXTYPE tuple. If an attempt to provision both the CALLING and CIC options is made, the following error message is displayed:

```
CALLING and CIC options are mutually exclusive.
```

- The CIC option will only be processed by the SUBR/SUBRPARM collectable if the SOC UTRS0001 is turned ON.

EMPTYIDX option

The Empty Index (EMPTYIDX) option identifies that validation of the subscriber number is successful if there are no subscriber numbers of this type provisioned against a particular index in the FLEXVAL table. The default is that the subscriber number validation fails.

Fields

The EMPTYIDX option does not contain any field refinements. The presence of the option indicates that validation against an empty index is successful for the subscriber number type. The absence of the option indicates that validation against an empty index is not successful for the subscriber number type.

Examples

An example of the EMPTYIDX option provisioning is:

```
EMPTYIDX
```

The presence of this option identifies that empty index screening in the FLEXVAL table is enabled for the particular subscriber number type.

EMPTYIDX option restrictions and limitations

The EMPTYIDX option does not have any identified table control restrictions.

FLEXLOG option

The Subscriber Number Type Definition log (FLEXLOG) option identifies that a FLEX 302 validation failure log is generated for the subscriber number type when the validation attempt for a subscriber number identified by the subscriber number type fails.

Fields

The FLEXLOG option does not contain any field refinements. The presence of the option indicates that a FLEX 302 validation failure log is generated on unsuccessful validation attempts of the received subscriber number digits for the specified subscriber number type. The absence of the option indicates that a log is not generated.

Examples

An example of the FLEXLOG option provisioning is:

```
FLEXLOG
```

The presence of this option identifies that a failed subscriber number validation attempt for the specified subscriber number type generates a FLEX 302 validation failure log.

FLEXLOG option restrictions and limitations

The following applies only to CAIN/FlexDial interactions. On CAIN-capable AXXESS agents, any FLEX 302 logs generated during the execution of FLEXDIAL collectables are suppressed until CAIN has determined that it will not trigger at the relevant TDPs (O_Feature_Requested and Information_Collected), or it triggers with the trigger action of Ignore. If CAIN does not trigger, only the last FLEX 302 log generated through FLEXDIAL is displayed; previous ones are lost. For more information see *UCS DMS-250 CAIN/FlexDial Interactions*.

CAINFLG option

The CAIN flag (CAINFLG) option is provided for CAIN/FlexDial interactions. CAINFLG allows CAIN to interpret the collect number in ways that CAIN understands. CAIN subsequently uses these numbers for its operations, such as SCP query parameter population.

The CAINFLG option appends a parameter that maps the FlexDial subscriber number to the CAIN digit type. The parameters used are ACCT, AUTH, PIN, MCCS, and CLGTYADD.

For more information about FlexDial and CAIN interactions, see *UCS DMS-250 CAIN/FlexDial Interactions* or ”FlexDial Interactions” appendix.

Fields

The CAINFLG option does not contain any field refinements.

Examples

Examples of the CAINFLG option provisioning are:

```
ACCT (BILLFLD ACCTCD) (CAINFLG ACCT) $
```

In this example, CAIN interprets the collected digits as an account code.

```
AUTH (BILLFLD BILLNUM REPLACE) (FLEXLOG) (CAINFLG AUTH)
```

In this example, CAIN interprets the collected digits as an authorization code.

```
PIN (BILLFLD PINDIGS REPLACE) (CAINFLG PIN) $
```

In this example, CAIN interprets the collected digits as PIN digits.

```
MCCS (BILLFLD BILLNUM REPLACE) (CAINFLG MCCS) $
```

In this example, CAIN interprets the collected digits as MCCS/TNS digits.

```
SS7CPTYADD (BILLFLD CLGPTYNO REPLACE) (CAINFLG CLGPTYADD) $
```

In this example, CAIN interprets the collected digits as a calling_party_address parameter of an incoming ISUP IAM message.

CAINFLG option restrictions and limitations

The CAINFLG option does not have any identified table control restrictions.

REVALIDATE option

The Subscriber Number Type Revalidate (REVALIDATE) option identifies that the subscriber number is revalidated on reoriginated calls. If the call reoriginates, the subscriber number is revalidated according to the VALIDATE option provisioned against the SUBR or SUBRPARM collectable. Identified FLEXTYPE and FLEXFEAT table features and characteristics are not re-applied to the call, nor is the subscriber number re-captured in the call detail record (CDR) as identified by the BILLFLD option.

This option provides no call processing application when used with the CALLTYPE collectable. An attempt to apply the REVALIDATE option through the CALLTYPE collectable results in a FLEX 301 trouble log with the following trouble value:

```
CALLTYPE cannot use REVALIDATE
```

Fields

The REVALIDATE option does not contain any field refinements. The presence of the option indicates that the subscriber number is revalidated on reoriginated calls. The absence of the option indicates that the subscriber number is not revalidated.

Examples

An example of the REVALIDATE option is:

```
REVALIDATE
```

This example identifies that the subscriber number received is revalidated on reoriginated calls.

REVALIDATE option restrictions and limitations

For CAIN/FlexDial interactions, if any subscriber numbers (ANI, authcodes, etc.) fail validation on a given execution of FLEXDIAL, all re-validation of subscriber numbers is turned off on all re-originations from that point forward for that AXXESS origination. For more information, see *UCS DMS-250 CAIN/FlexDial Interactions*.

System requirements

The total store requirements for the FLEXTYPE table amount to 4Kbytes for a maximum of 1024 table entries. No store is allocated unless an entry is provisioned in the table.

Dump and restore

This table does not require a dump and restore from any other tables.

Datafill order

The FLEXTYPE table provisioning does not depend on any other tables.

The FLEXDIAL, FLEXVAL, FLEXFEAT, and TRKFEAT tables all require the FLEXTYPE table to be provisioned before they can be properly provisioned.

Restrictions and limitations

The following restrictions and limitations apply to table FLEXTYPE:

- The 16-character string key for the FLEXTYPE table is limited to 1024 unique values. An attempt to provision more than 1024 entries results in the following error message:

  ```
  TABLE IS FULL.
  ```

- Individual features and characteristics cannot be duplicated for a FLEXTYPE table index. An attempt to do this results in the following error message:

  ```
  Cannot duplicate option <option>.
  ```

  <option> identifies the option that is duplicated.

- Indexes in the FLEXTYPE table may not be deleted if they are referenced in other tables. An attempt to remove a referenced index results in the following error message:

  ```
  TUPLE REFERENCED BY ANOTHER TABLE – USE TABREF TO GET
  POTENTIAL TABLE LIST.
  ```

- For a complete list of restrictions and limitations for CAIN/FlexDial interactions, see, *UCS DMS-250 CAIN/FlexDial Interactions*.

All errors result in the failure of requested table control changes.

Table FLEXVAL

This chapter describes the FlexDial Subscriber Number Validation (FLEXVAL) table.

Purpose

The FLEXVAL table provides the means of provisioning all subscriber assigned numbers in the FlexDial framework for validation purposes. This table validates all subscriber numbers collected by the SUBR and SUBRPARM digit collectables that are identified by the subscriber number type as defined by the FLEXTYPE table.

The FLEXVAL table also provides a new provisioning scheme for datafilling these digits. This scheme does not conflict with the existing subscriber number provisioning schemes.

An additional switch command (QFLEXVAL) supplements existing table control ability for the FLEXVAL table. See chapter 14, “QFLEXVAL CI Increment” for more information.

General layout

The FLEXVAL table contains a three-part key consisting of a subscriber number type index, a numeric index, and a digit vector index value that retrieves a tuple containing the index into the FLEXFEAT table. The FLEXVAL table is limited to 33.5 million subscriber number entries, although the actual storage capability of the table is limited by system memory.

By providing an index into the FLEXFEAT table instead of directly listing the fields that identify the features of the subscriber numbers, it becomes possible for multiple subscriber numbers to share the same subscriber number features and characteristics.

The FLEXVAL table control verifies that the features listed for the FLEXFEAT table index are applicable to the provisioned key. This is performed by screening the features and characteristics provisioned for the index against the subscriber number type provisioning in the FLEXTYPE table.

Key

The key for indexing the FLEXVAL table consists of a three-part index:

- subscriber number type index

  The subscriber number type index is a valid entry in the FLEXTYPE table.

- number index with range of {0 to 1,047,999}

  This index provides a value to identify a unique feature set for a particular subscriber number, where this subscriber number may need to be duplicated.

- digit vector containing up to 16 digits of {0 to 9, A, B, C, D}.

  This digit vector of up to 16 digits identifies the actual subscriber number that is being screened in the FLEXVAL table. The A, B, C, D digits represent 4th column DTMF digits.

The numeric index to the key provides for 1 million unique indexes. The actual storage capability of the table is limited by system memory.

The entries in the FLEXVAL table are ordered according to the following rules:

- Entries are first listed by the order of the FLEXTYPE values in the data dictionary.
- Per FLEXTYPE value, entries are then listed in sequential order of the NUMINDEX field.
- Per FLEXTYPE value and NUMINDEX value, entries are listed by the following order of digits:

  0, 1, 2, 3, 4, 5, 6, 7, 8, 9

Table 6-1 contains the FLEXVAL table key.

Table 6-1
FLEXVAL table key

Key Field	Description	Values
FLEXTYPE_ IDX	The first part of the key identifies the type of subscriber number.	Valid FLEXTYPE table index (1024 unique values)
UNIQUE_ID	This part of the index identifies the unique numeric index of the entry.	0 to 1,047,999
SUBSCRIBER_ NUMBER	The third part of the index consists of the actual subscriber number digits.	Vector of up to16 of {0 to 9, A, B, C, D} ***Note:*** The A, B, C, D digits represent 4th column DTMF digits.

The entries in the FLEXVAL table are ordered according to the following rules:

- Entries are first listed by the order of the FLEXTYPE values in table FLEXTYPE.
- Per FLEXTYPE value, entries are then listed in sequential order of the Unique ID field.
- Per FLEXTYPE value and Unique ID value, entries are listed by the following sequential ordering of digits:

  { 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, S, P, A, B, C, D }

While it is possible to process all 4th column DTMF digits for call processing, it is not possible to capture a 4th column DTMF digit “D” into a digit field of the CDR.

Fields

Table 6-2 contains the FLEXVAL field.

Table 6-2
FLEXVAL table field

Field	Description	Values	Default
FLEXFEAT	This field identifies the index value into the FLEXFEAT table. The FLEXFEAT table identifies all the features associated with the subscriber number.	Valid FLEXFEAT index	N/A

System requirements

The FLEXVAL table allocates store on an as needed basis for a maximum of 33.5 million table entries, although the actual storage capability of the table is limited by system memory. Store is allocated in blocks of 128Kbytes, where each block represents 4K provisioned entries (30 bytes per entry). A maximum of 1072 Mbytes of store can be allocated for the maximum number of table entries. No store is allocated unless an entry is provisioned in the table.

Datafill order

The FLEXVAL table requires that the FLEXFEAT index stored in the FEATIDX field be previously provisioned before provisioning the subscriber number in the FLEXVAL table.

The FLEXVAL table requires that the subscriber number type used in the three-part key is previously provisioned in the FLEXTYPE table.

No other table depends on the provisioning of the FLEXVAL table.

Dump and restore

This table does not require a dump and restore from any other tables.

The one night process (ONP) of the table is performed through a physical data move.

Restrictions and limitations

The following restrictions and limitations apply for table FLEXVAL:

- The numeric index of the three-part key is limited to 1,048,000 unique values. This limitation is controlled through the type definition.
- The digit vector index of the three-part key is limited to 16 digits, where each digit can have the value of 0 to 9. This limitation is controlled through the type definition.
- The FLEXVAL table is limited to 33 million table entries. An attempt to provision more than 33 million entries results in the following error message:

  ```
  Table FLEXVAL is full. No more entries are possible.
  ```

All errors result in the failure of requested table control changes.

Table FLEXFEAT

This chapter describes the FlexDial Subscriber Number and Call Type Features (FLEXFEAT) table.

Purpose

The FLEXFEAT table provides the provisioning scheme to identify features and characteristics of subscriber numbers. This table is used in conjunction with the FLEXVAL table that identifies the index into the FLEXFEAT table for the subscriber number. This provides a two-tiered approach to validating subscriber numbers on the UCS DMS-250 switch.

General layout

The FLEXFEAT table is indexed by a numeric value as the key to identify a tuple that consists of a list of feature and character options. The options available for provisioning constitute all of the available subscriber number and call type defined features and characteristics.

The features and characteristics of table FLEXFEAT allows provisioning against any type of subscriber number or call type definition that table FLEXTYPE identifes.

The FLEXFEAT table controls mutual exclusivity between options and verifies that options are not duplicated in the provisioning. The FLEXVAL table verifies that the options provisioned for a FLEXFEAT index are allowed for the subscriber number type as defined by the FLEXTYPE table. The CALLTYPE collectable identifies that the options provisioned for a FLEXFEAT index are allowed for the identified call type defined by the FLEXTYPE table.

Key

The key to the table is a simple 24-bit numeric value providing approximately 16.8 million unique values, although the actual storage capability of the table is limited by system memory.

Table 7-1 contains the FLEXFEAT table key.

Table 7-1
FLEXFEAT table key

Key Field	Description	Values
INDEX	The key consists of a numeric value providing 16.8 million unique indexes.	0 to 16, 777, 215

Fields

The FLEXFEAT table does not contain a fixed set of fields, but consists of an options vector that is provisioned with any of the defined features and characteristics. Each option may define its own data refinement as needed for datafill purposes. Therefore each option is listed independently.

The default field for the options identifies the values that are assumed for feature options if the option is not present for the feature index. Normally when a tuple consists of an options vector, the default is to not have any options provisioned. Due to the nature of feature specifications though, each feature option is required to have a default value if its option is not present in the list.

When provisioning the field of an option, a default value for the field type can be specified by the DEFDATA table.

Table 7-2 contains the FLEXFEAT table field.

Table 7-2
Table FLEXFEAT field

Field	Description	Values	Default
OPTION	This field consists of an options vector that contains a list of the features and characteristics applicable to a subscriber number.	NIL, ANSCDR, BCCOMPAT, CAINGRP, CASUAL, CDRTMPLT, CICDELV, CITYCODE, CITYVAL, CLDPBILL, CPACTVAL, DELIVER, DPIDX, FAILVAL, FEATVAR, FLDONLY, GENLOG, IEXCLINX, ITRANSTS, MLTCOSID, MSGCTR, NOANSDUR, ONNET, PVSPDIDX, REORGACT, REORGTYP, REVALIDATE, SPLASHBK, TCAPANNC, TRANSNUM, TRANSTS, TRANSYS Up to 33 options can be provisioned in the vector. ***Note:*** The listed options may only be provisioned once within the vector, with the exception of MSGCTR which can be provisioned up to eight times, and the REORGTYP option that may be provisioned up to three times within the vector.	N/A

NIL option

The FLEXFEAT table NIL option as applicable for the subscriber number or call type provisioning is identical to the definition of the option for table FLEXTYPE. See “NIL option” on page 5-2 for more information.

ANSCDR option

The FLEXFEAT table CDR on Call Answer (ANSCDR) option as applicable for the subscriber number or call type definition is identical to the definition of the option for table FLEXTYPE. See “ANSCDR option” on page 5-3 for more information on ANSCDR option syntax.

BCCOMPAT option

The Bearer Capability Compatibility (BCCOMPAT) option identifies the bearer capability provisioned against the subscriber number or call type definition. The BCCOMPAT option is used for bearer capability screening through the BCCOMPAT table.

For a data path defined bearer capability, the ability to perform inband DTMF digit collection is not allowed.

Fields

Table 7-3 contains the field refinement for the FLEXFEAT BCCOMPAT option.

Table 7-3
Table FLEXFEAT BCCOMPAT option field

Field	Description	Values	Default
BCNAME	This field contains the bearer capability value for the subscriber number.	NILBC, SPEECH, 64KDATA, 64KX25, 56KDATA, DATAUNIT, 64KRES, 3_1KHZ, 7_KHZ, VOICE_DATA, 64K_RATE_AD_DATA	3_1KHZ
Note 1: The default value is only applicable if the BCCOMPAT value has not yet been set for the call. The default value does not overwrite a value currently set.			

Examples

An example of the BCCOMPAT option is:

```
BCCOMPAT VOICE_DATA
```

This example identifies that a bearer channel has the capability of a voice or data channel.

BCCOMPAT option restrictions and limitations

There are no identified table control restrictions for the BCCOMPAT option.

CAINGRP option

The Carrier AIN Group (CAINGRP) option identifies an index into the CAINGRP table for AIN processing. For more information on the CAIN feature, see the *UCS DMS-250 NetworkBuilder Application Guide*. For more information on the CAIN and FlexDial interactions, see *UCS DMS-250 CAIN/FlexDial Interactions* or ”FlexDial Interactions” appendix.

Fields

Table 7-4 contains the field refinement for the CAINGRP option.

Table 7-4
Table FLEXFEAT CAINGRP option field

Field	Description	Values	Default
CAINGRP	This field identifies the index into the CAINGRP table that is used for AIN processing.	An existing entry in the CAINGRP table	N/A

Examples

An example of the CAINGRP option provisioning is:

```
CAINGRP VPN1
```

This example identifies AIN processing should occur using CAINGRP index VPN1.

CAINGRP option restrictions and limitations

There are no identified table control restrictions for the CAINGRP option.

CASUAL option

The Casual (CASUAL) option identifies that the subscriber number is considered a casual number. In order for validation to pass for the subscriber number, the CASUBLK SUBR and SUBRPARM option must not be used with the collectable that is processing the subscriber number.

Note: The term “casual” indicates that there is not a 1-to-1 relationship with the subscriber number to a subscriber. A casual subscriber is a number that is more generic (such as an area code) and is shared by many subscribers. With casual numbers, it is not possible to explicitly identify the actual subscriber making the call.

This option provides no call processing functionality when used with the CALLTYPE collectable.

Fields

The CASUAL option does not contain any field refinements. The presence of the option indicates the subscriber number is a casual number. The absence of the option indicates that the subscriber number is not considered a casual number.

Examples

An example of the CASUAL option is:

```
CASUAL
```

This example identifies that the subscriber number is considered a casual number.

CASUAL option restrictions and limitations

There are no identified table control restrictions for the CASUAL option.

CDRTMPLT option

The FLEXFEAT Table CDR Template (CDRTMPLT) option as applicable for the subscriber number or call type definition is identical to the definition of the option for table FLEXTYPE. See “CDRTMPLT option” section on page 5-9 for more information.

CICDELV option

The CICDELV option allows subscriber number processing to specify whether a CIC should be outpulsed to the terminating agency. If the CICDELV option is not provisioned, the CIC delivery status for the call is not modified.

Fields

Table 7-5 contains the field refinement for the CICDELV option.

Table 7-5
Table FLEXFEAT CICDELV option field refinement

Field	Description	Values	Default
DELIVER	This field identifies whether the CIC is outpulsed to the terminating agency	ALWAYS, NEVER	Delivery status not modified.

Example

An example of the CICDELV option is:

```
CICDELV NEVER
```

This example specifies that an outgoing CIC is not outpulsed to the terminating agency.

CICDELV option restrictions and limitations

The following restrictions and limitations apply to the CICDELV option:

- CIC delivery is only possible on CCS7 or DS1 multistage terminators. The CICDELV option is meaningless when the terminating agency is neither CCS7 nor DS1 multistage.
- The CICBLK option, in tables TRKSIG for AXXESS terminators and TRKGRP for non–AXXESS terminators, will have precedence over the CICDELV option in table FLEXFEAT.

CITYCODE option

The City Code (CITYCODE) option identifies a city code value that relates the subscriber number with a particular service city. This value is used in city code screening as identified by the SUBR and SUBRPARM digit collectable CITYVAL option or the FLEXFEAT table CITYVAL option. The CITYCODE value is also used to index table CITYCODE to provide a public speed dial number index.

After being used on a subscriber number basis for successful citycode validation, the CITYCODE value then becomes the identified city code value for the call.

For call type collectables, the application of city code screening does not apply.

Fields

Table 7-6 contains the field refinement for the CITYCODE option.

Table 7-6
Table FLEXFEAT CITYCODE option field refinement

Field	Description	Values	Default
CITYCODE	This field contains the citycode value for the subscriber number	Vector of 3 of (0 to 9)	N/A

If this option is not present for a subscriber number, then CITYCODE screening cannot be performed.

Examples

An example of the CITYCODE option is:

```
CITYCODE 283
```

This example identifies that the citycode value to use for citycode validation purposes is 283.

CITYCODE option restrictions and limitations

The CITYCODE option does not have any identified table control restrictions.

CITYVAL option

The City Code Validation (CITYVAL) option identifies on a per-subscriber number basis that citycode validation is to occur for the subscriber number. Citycode validation compares the CITYCODE of the subscriber number to the CITYCODE value set for the call. If the CITYCODE values are identical, then validation is successful. Otherwise the validation attempt fails.

To provision the CITYVAL option, the FLEXFEAT CITYCODE option must also be provisioned in the FLEXFEAT table entry. City code validation cannot occur unless a CITYCODE value has been previously identified for the call (such as provisioning the CITYCODE option for the originating agent in table TRKFEAT).

If a CITYCODE value for the call is not available, then citycode validation does not occur and the validation attempt is successful by default.

Citycode validation may alternatively be triggered through use of the table FLEXDIAL SUBR or SUBRPARM collectable CITYVAL option (see “SUBR CITYVAL option” on page 2-85).

For call type collectables, the application of city code validation does not apply.

Fields

Table 7-7 contains the field refinement for the CITYVAL option.

Table 7-7
Table FLEXFEAT CITYVAL option field refinement

Field	Description	Values	Default
APPLYFTR	If the FAILACT field is set to TRMT, then this field is present and identifies whether or not other FLEXFEAT features and characteristics provisioned are to be applied for the subscriber number in the event of CITYCODE validation failure. Typically features and characteristics are not applied on validation failures.	N or Y	N/A
OVERRIDE	If the FAILACT field is set to TRMT, then this field is present and identifies whether or not the treatment identified overrides a treatment that may already be set for the call.	N or Y	N/A
TRMT	If the FAILACT field is set to TRMT, then this field is present and identifies the treatment to set for the call.	Range of extended treatment type	N/A

Examples

An example of the CITYVAL option is:

```
CITYVAL N Y INCC
```

This example identifies that citycode validation is to occur for the subscriber number. If validation fails, then the other features and characteristics defined in the FLEXFEAT table entry and not applied to the call, and the INCC treatment is set for the call.

CITYVAL option restrictions and limitations

The CITYCODE option must be provisioned in the table entry when attempting to provision the CITYVAL option. If an attempt to provision the CITYVAL option without the CITYCODE option is performed, then the following error message is displayed:

```
Cannot provision CITYVAL without CITYCODE.
```

All errors result in the failure of requested table control changes.

CLDPBILL option

The Called Party Billed (CLDPBILL) option identifies that billing for the call is based on the called party number, instead of the default being subscriber number billed. This option allows call processing to ignore any treatments that may have been set due to failed validation of subscriber numbers in the call, since it is no longer important that subscriber numbers be valid for the call.

Note: When processed by a SUBR or SUBRPARM collectable, the validated subscriber number digits are automatically captured into the BILLNUM field of the CDR, overwriting information that may have been previously captured. Because the CLDPBILL option was activated for the validated subscriber number, the digits processed are treated as a “called party billing number”, and hence automatically captured in the BILLNUM field. If billing capturing is also active through the FLEXTYPE BILLFLD option, the subscriber number digits are also captured in the identified billing record field.

Fields

The CLDPBILL option does not contain any field refinements. The presence of the option indicates that the call is called party billed. The absence of the option indicates that the call is subscriber number billed.

Examples

An example of the CLDPBILL option is:

```
CLDPBILL
```

CLDPBILL option restrictions and limitations

The CLDPBILL option does not have any identified table control restrictions.

CPACTVAL Option

The CPACTVAL (call processing active validation) option introduces another form of validation for subscriber number or calltype being processed. Call processing active validation identifies that a particular subscriber number or calltype may only be used for a set limit of active calls in progress.

For calltype use, a valid FLEXTYPE must also be identified by the CALLTYPE collectable. If a valid non-NIL FLEXTYPE is not specified in the CALLTYPE collectable provisioned, then the CPACTVAL option does not apply for CALLTYPE collectable processing and is subsequently ignored.

Fields

Table 7-8 contains the field refinements for the CPACTVAL option.

Table 7-8
Table FLEXFEAT CPACTVAL option field

Field	Description	Values	Default
LIMIT	This field identifies the upper boundary limit for the number of active calls that the subscriber number or calltype may be involved in.	0 to 255	N/A
CALLPS	This field is updated by call processing and identifies the current number of active calls for the particular subscriber number or calltype. This value is **NOT** modifiable through table control mechanisms.	0 to 255 When initially provisioning the CPACTVAL option, use a value of zero for this field. When modifying the CPACTVAL option through table control, maintain the current value for this field.	0
FAILACT	For call processing active validation failures, this field identifies the failure action that is to be taken. FAILACT indicates that the FAILACT provisioning for the SUBR or SUBRPARM collectable VALIDATE option is to be used. TRMT indicates that the following delayed treatment is to be set for the call.	FAILACT or TRMT	N/A

—continued—

Field	Description	Values	Default
OVERRIDE	If the FAILACT field is set to TRMT, then this field is present and identifies whether or not the treatment identified overrides a treatment that may already be set for the call.	N or Y	N/A
TRMT	If the FAILACT field is set to TRMT, then this field is present and identifies the treatment to set for the call.	Range of Extended_Treatment	N/A

—end—

Examples

Examples of the CPACTVAL option are:

```
CPACTVAL 2 0 TRMT Y ANIA
```

This example identifies that call processing active validation is to occur with a limit of two active calls at any one time. If an attempt is made to use the subscriber number or calltype for more than two active calls, then the treatment of ANIA is set for that call.

Dump and Restore

For the UCS08 and beyond dump and restore of CPACTVAL options, the CALLPS counter for a CPACTVAL option is reset to zero as the FLEXFEAT tuple is restored on the inactive CM node.

Provisioning restrictions and limitations

The following provisioning restrictions or limitations exist for CPACTVAL option use:

- For subscriber number processing, call processing active validation is only possible when performing successful FLEXVAL validation.
- For calltype processing, call processing active validation is only possible when the CALLTYPE collectable is provisioned with a valid and non-NIL FLEXTYPE table index.
- A CPACTVAL option is applied directly for the particular FLEXFEAT profile or tuple that it is defined within. The relationship that exists between tables FLEXVAL and FLEXFEAT and the CALLTYPE collectable and table FLEXFEAT enables CPACTVAL use for subscriber number or calltype processing. Note that multiple subscriber numbers and/or calltypes may reference the same FLEXFEAT profile, subjecting all of them to potential CPACTVAL validation limits.

- Only a single CPACTVAL option may be provisioned for the FLEXFEAT table entry. If an attempt to provision more than one CPACTVAL option in the tuple is performed, the following error message is displayed:

  ```
  Duplicate CPACTVAL, only one allowed.
  ```

- When the CPACTVAL CALLPS field is greater than zero, then the CPACTVAL option may not be removed from the tuple definition, nor may the tuple be removed from table FLEXFEAT. If an attempt to remove the CPACTVAL option with CALLPS value greater than zero from the tuple is performed, then the following error message is displayed:

  ```
  Error – Cannot remove in use CPACTVAL option.
  ```

  If an attempt to delete a FLEXFEAT tuple containing an active CPACTVAL option is performed, then the following error message is displayed:

  ```
  Error – CPACTVAL option active. Cannot remove entry.
  ```

- The CALLPS field for a CPACTVAL option must be initially provisioned to a value of zero. If an attempt to initially provision a CPACTVAL option with a CALLPS field greater than zero is performed, then the following warning message is displayed:

  ```
  Warning – Cannot add CPACTVAL with CALLPS greater than
  zero (0). A value of zero (0) is being enforced for this
  entry.
  ```

- The CALLPS field for a CPACTVAL option may not be modified through table control changes, and must maintain its current value through table control modifications. If an attempt to modify the CALLPS field through table control is performed, then the following warning message is displayed:

  ```
  Warning – Cannot alter CPACTVAL active CALLPS value.
  Current value is maintained.
  ```

All errors result in the failure of requested table control changes.

DELIVER option

The Deliver (DELIVER) option identifies when the calling party subscriber number is to be delivered (or outpulsed) to the next network switch (translated to destination).

This option is used in conjunction with current outpulsing capability, and the identification of a received subscriber number that is the CALLING party subscriber number. No new outpulsing capability is being developed at this time to support the generic outpulsing of any subscriber number type.

Note: See “CALLING option” on page 5-8 for more information.

Fields

Table 7-9 contains the field refinement for the DELIVER option.

Table 7-9
Table FLEXFEAT DELIVER option field

Field	Description	Values	Default
DELIVER	This field identifies the status of delivering the calling party subscriber number to the terminating switch.	ALWAYS, CPNONLY, CGNONLY, NEVER	Delivery status not modified

If the DELIVER option is not provisioned, then the calling party subscriber number delivery status for the call is not modified.

Examples

An example of the DELIVER option is:

```
DELIVER ALWAYS
```

This example identifies that the calling party address delivery status as always delivered to the terminating switch.

DELIVER option restrictions and limitations

The DELIVER option does not have any identified table control restrictions.

DPIDX option

The Dialplan Index (DPIDX) option identifies an index into the FLEXDIAL table. This index contains provisioned collectables to be processed for the call in order to perform required interaction with the originating agent.

If this option is not present, then additional or revised interaction with the originating agent is not provided.

Fields

Table 7-10 contains the field refinements for the DPIDX option.

Table 7-10
Table FLEXFEAT DPIDX option fields

Fields	Description	Values	Default
INDEX	This field identifies the index into the FLEXDIAL table.	An existing entry in table FLEXDIAL	N/A
ACTION	This field identifies how the call processing handles the collectable list specified by the index.	INSERT, APPEND, REPLACE , EXEC INSERT inserts the new list into the current processing list before the next collectable. APPEND appends the new list to the current processing list. REPLACE replaces the remainder of the unprocessed collectables with those identified by the FLEXDIAL index. EXEC executes as a sublist the list identified by the FLEXDIAL index.	N/A

Examples

An example of the DPIDX option provisioning is:

```
DPIDX PIN_VACCT REPLACE
```

In this example, collectable processing continues with the collectable list identified by the PIN_VACCT index when the current SUBR or SUBRPARM collectable has been processed. Remaining collectables in the current list are not processed.

DPIDX option restrictions and limitations

There are no identified table control restrictions for the DPIDX option.

FAILVAL option

The Fail Validation (FAILVAL) option identifies that use of this subscriber number results in a validation failure for the SUBR or SUBRPARM digit collectable. This can occur when the subscriber is being restricted by the service provider. If this option is not present, then validation of the subscriber number is successful.

When this option is used, the subscriber number validation fails. The appropriate FLEX 302 validation failure log is generated and proper operational measurements (OMs) are incremented. Any treatment defined by this option it takes precedence over information provisioned in the FAILACT field of the SUBR/SUBRPARM collectable VALIDATE option.

When this option is used, other options may be provisioned, but their associated call processing features are not in effect. For example, it is not useful to perform a feature such as COSUS screening when a treatment is already set from authcode processing.

For CALLTYPE collectable processing, the action must always be set to TRMT. This option then identifies that the specified treatment is to be applied to the call. (Note that it is more efficient to use the ROUTE collectable via an office route than the CALLTYPE collectable to route to treatment.) If the FAILACT option is processed by a CALLTYPE collectable, no call processing action is performed and a FLEX 301 trouble log is generated with the following trouble value:

```
FAILACT must be TRMT for CALLTYPE
```

Fields

Table 7-11 contains the field refinements for the FAILVAL option.

Table 7-11
Table FLEXFEAT FAILVAL option fields

Fields	Description	Values	Default
ACTION	This identifies the action to take for failure.	FAILACT or TRMT FAILACT identifies that the fail action specified by the SUBR or SUBRPARM collectable handles the validation failure. TRMT identifies that the treatment identified by the TRMT field is set for the call.	N/A
OVERRIDE	If the action field is set to TRMT, then this field is present and identifies if the treatment being set overrides any previous treatment set.	N or Y	N/A
TRMT	If the action field is set to TRMT, then his field is present and identifies the treatment that is applied to the call.	Range of extended_treatment type ***Note:*** See *UCS DMS-250 Data Schema Reference Manual* for values of the extended treatment type.	N/A

Examples

An example of the FAILVAL option provisioning is:

```
FAILVAL TRMT Y ANIA
```

In this example, validation of the subscriber number causes the ANIA treatment to be set for the call. This treatment overrides any previous treatment value set.

FAILVAL option restrictions and limitations

There are no identified table control restrictions for the FAILVAL option.

FEATVAR Option

The FEATVAR (FLEXFEAT variable) option provides a powerful call processing to table control feedback mechanism whereby actions occurring in call processing may modify the provisioned information for a subscriber number or calltype.

For calltype use, a valid FLEXTYPE must also be identified by the CALLTYPE collectable. If a valid FLEXTYPE is not specified in the CALLTYPE collectable provisioned, then the FEATVAR option does not apply for CALLTYPE collectable processing and is subsequently ignored.

FEATVAR options are associated and identified by their link to the FLEXTYPE of the subscriber number or calltype collectable.

Fields

Table 7-12 contains the field refinements for the FEATVAR option.

Table 7-12
Table FLEXFEAT FEATVAR option fields

Fields	Description	Values	Default
VALUE	This field identifies the value of the FLEXFEAT variable stored for the subscriber number or calltype.	–2147483647 to 2147483648	
CALLPS	This field is updated by call processing and identifies the current number of active calls which have active control over the FEATVAR option for the particular subscriber number or calltype. This counter holds at 255 and does not roll over in the situation where more than 255 active calls are referencing the same FEATVAR option. This value is **NOT** modifiable through table control mechanisms.	0 to 255 When initially provisioning the FEATVAR option, use a value of zero for this field. When modifying the FEATVAR option through table control, maintain the current value for this field.	

Example

Examples of the FEATVAR option are:

```
FEATVAR 400
```

This example identifies that a value of 400 is set for the FLEXFEAT variable.

Dump and Restore

There are no pre-UCS08 dump and restore requirements for the FEATVAR option.

For UCS08 and beyond dump and restore of FEATVAR options, the CALLPS counter for a FEATVAR option is reset to zero as the FLEXFEAT tuple is restored on the inactive CM node.

Provisioning restrictions and limitations

The following provisioning restrictions or limitations exist for FEATVAR option use:

- For subscriber number processing, FEATVAR activation is only possible when performing successful FLEXVAL validation.
- For calltype processing, FEATVAR activation is only possible when the CALLTYPE collectable is provisioned with a valid and non–NIL FLEXTYPE table index.
- A FEATVAR option is applied directly for the particular FLEXFEAT profile or tuple that it is defined within. The relationship that exists between tables FLEXVAL and FLEXFEAT and the CALLTYPE collectable and table FLEXFEAT enables FEATVAR activation for subscriber number or calltype processing. Note that multiple subscriber numbers and/or calltypes may reference the same FLEXFEAT profile, enabling FEATVAR use of the singular provisioned option.
- Only a single FEATVAR option may be provisioned for the FLEXFEAT table entry. If an attempt to provision more than one FEATVAR option in the tuple is performed, the following error message is displayed:

  ```
  Duplicate FEATVAR, only one allowed.
  ```

- When the FEATVAR CALLPS field is greater than zero, then the FEATVAR option may not be removed from the tuple definition, nor may the tuple be removed from table FLEXFEAT. If an attempt to remove the FEATVAR option with CALLPS value greater than zero from the tuple is performed, then the following error message is displayed:

  ```
  Error – Cannot remove in use FEATVAR option.
  ```

- If an attempt to delete a FLEXFEAT tuple containing an active FEATVAR option is performed, then the following error message is displayed:

  ```
  Error – FEATVAR option active. Cannot remove entry.
  ```

- The CALLPS field for a FEATVAR option must be initially provisioned to a value of zero. If an attempt to initially provision a FEATVAR option with a CALLPS field greater than zero is performed, then the following warning message is displayed:

  ```
  Warning – Cannot add FEATVAR with CALLPS greater than zero
  (0). A value of zero (0) is being enforced for this entry.
  ```

- The CALLPS field for a FEATVAR option may not be modified through table control changes, and must maintain its current value through table control modifications. If an attempt to modify the CALLPS field through table control is performed, then the following warning message is displayed:

  ```
  Warning – Cannot alter FEATVAR active CALLPS value.
  Current value is maintained.
  ```

All errors result in the failure of requested table control changes.

FLDONLY option

The Filed Only (FLDONLY) option identifies that the subscriber number must be filed (provisioned) in the system and not dialed by the originating agency. That is, all digits for the SUBR collectable being processed must have come from the FILED option.

This option has no call processing application when used with the CALLTYPE collectable.

Fields

The FLDONLY option does not contain any field refinements. The presence of the option indicates that the number must be filed. The absence of the option indicates that the number can be filed or dialed.

Examples

An example of the FLDONLY option is:

```
FLDONLY
```

This example identifies that only a filed subscriber number can be successfully validated.

FLDONLY option restrictions and limitations

The FLDONLY option does not have any identified table control restrictions.

GENLOG option

The Generate Log On Use (GENLOG) option identifies that a log is generated due to the use of the subscriber number or call type. This option generates a FLEX 601 information log for the call. The FLEX 601 log contains the “FLEXTYPE Processed” report indicator along with the actual FLEXTYPE table index and digits processed by the collectable.

Fields

The GENLOG option does not contain any field refinements. The presence of the option indicates a log is to be generated. The absence of the option indicates that a log is not to be generated.

Examples

An example of the GENLOG option is

```
GENLOG
```

This example identifies that a FLEX 601 information log is generated from the use of this subscriber number.

GENLOG option restrictions and limitations

The GENLOG option does not have any identified table control restrictions.

IEXCLINX Option

The IEXCLINX (incoming exclusion screening index) option is used to identify an index for incoming exclusion screening purposes for the subscriber number or calltype processed.

Fields

Table 7-13 contains the field refinement for the IEXCLINX option.

Table 7-13
Table FLEXFEAT IEXCLINX option field

Field	Description	Values	Default
INDEX	This field identifies the incoming exclusion screening index to be used for the call.	0 to 255	N/A

Example

Examples of the IEXCLINX option are:

```
IEXCLINX 12
```

This example identifies that index 12 is to be used for incoming exclusion screening for the call.

Provisioning Order

There are no provisioning order requirements for the IEXCLINX option.

IEXCLINX option restrictions and limitations

The following provisioning restrictions or limitations exist for IEXCLINX option use.

- Only a single IEXCLINX option may be provisioned for the FLEXFEAT table entry. If an attempt to provision more than one IEXCLINX option in the tuple is performed, the following error message is displayed:

  ```
  Duplicate IEXCLINX, only one allowed.
  ```

All errors result in the failure of requested table control changes.

ITRANSTS option

The ITRANSTS option allows subscriber number processing to specify an alternate STS to use for international calls. The international STS specified will only be used if the current translations system (the translations system at the time the ITRANSTS option is processed) for the call is set to international or international partitioned.

Fields

Table 7-14 contains the field refinements for the ITRANSTS option.

Table 7-14
Table FLEXFEAT TRANSTS option fields

Fields	Description	Values	Default
STSTYP	This field identifies whether the STS is directly provisioned or is acquired by looking up the provisioned partition digits in the PARTOSTS table.	STS or PARTITION	N/A
STS	If the STSTYPE is equal to STS, then this field is present and contains the value of the STS provisioned.	An existing entry in table HNPACONT	N/A

—continued—

Fields	Description	Values	Default
OPART	If the STSTYPE is equal to PARTITION, then this field is present and contains the value of the Originating Partition number.	0 to 999	N/A
TPART	If the STSTYPE is equal to PARTITION, then this field is present and contains the value of the Terminating Partition number.	0 to 31	N/A
—end—			

Examples

An example of the ITRANSTS option is:

```
ITRANSTS STS 611
```

This example identifies the STS value of 611 to be used for international translations only.

ITRANSTS option restrictions and limitations

The following restrictions and limitations apply to the ITRANSTS option:

- If a TRANSTS option is datafilled, but no ITRANSTS option, the STS derived from the TRANSTS option will be used regardless of the current translation system.
- If an ITRANSTS option is datafilled, but no TRANSTS option, the STS derived from the ITRANSTS option will only be used if the current translation system is set to international or international partitioned.

MLTCOSID option

The Multiple Class Of Service Screening Index (MLTCOSID) option identifies the index value for class of service screening.

Fields

Table 7-15 contains the field refinement for the MLTCOSID option.

Table 7-15
Table FLEXFEAT MLTCOSID option field

Field	Description	Values	Default
INDEX	This field contains the index used in the MULTICOS table for class of service screening. If the index is zero, screening is not performed. ***Note:*** The MULTICOS table provides a vector of COSUS indexes that are used to screen the address digits for validation purposes.	1 to 2047	0

Examples

An example of the MLTCOSID option is:

```
MLTCOSID 15
```

This example identifies that the MULTICOS table index of 15 is used when performing COSUS table screening.

MLTCOSID option restrictions and limitations

The MLTCOSID option does not have any identified table control restrictions.

MSGCTR option

The Message Center (MSGCTR) option contains an index into table MSGCTR. This MSGCTR table index contains a vector of messages that identify processing information for specified collectables, and relates necessary information to the collectable that it needs to properly process the call.

The information being related to the collectable takes precedence over information provisioned with the collectable (in table FLEXDIAL). At the time of processing the subscriber number collectable or trunk group features though, it is technically unknown if the collectable the information relates to is going to be processed for the call. (For example, we have information about the ADDR collectable, but do not know if an ADDR collectable exists or will exist in the collectable list to be processed.) Therefore this information is contained in a posted message that is left for the identified collectable. When a collectable begins processing, it first checks for applicable posted messages.

Fields

Table 7-16 contains the field refinement for the MSGCTR option.

Table 7-16
Table FLEXFEAT MSGCTR option field

Field	Description	Values	Default
MSGCTRID	This field contains an index into table MSGCTR that identifies applicable messages that must be posted.	An existing entry in table MSGCTR	N/A

Examples

An example of the MSGCTR option is:

```
MSGCTR 15
```

In this example, the MSGCTR table index 15 identifies messages that must be posted when using the particular call type or subscriber number.

MSGCTR option restrictions and limitations

A maximum of eight MSGCTR options may be provisioned within a FLEXFEAT table entry. An attempt to provision more than eight MSGCTR options within a single table entry results in the following error message:

```
Maximum MSGCTR exceeded, only 8 allowed.
```

All errors result in the failure of requested table control changes.

NOANSDUR option

The No Answer Duration (NOANSDUR) option identifies the amount of time the terminating agent can be seized before answer must be received by the UCS DMS-250 switch. If answer is not received within the time indicated by this option, then the connection is taken down and the action identified is applied to the call.

Fields

Table 7-17 contains the field refinements for the NOANSDUR option.

Table 7-17
Table FLEXFEAT NOANSDUR option fields

Fields	Description	Values	Default
DURATION	This field identifies the amount of time that the trunk can be seized before answer is reported.	0 to 3600 seconds	0 (indicates that feature is not active)
ACTION	This field identifies the action to take if the no answer duration timer expires before answer is received for the call.	NOANSTRMT, RTEADV, RECOL, DPIDX NOANSTRMT identifies that specified treatment is to be applied to the call due to expiration of the no answer duration timer. RTEADV identifies that a route advance is to occur. RECOL identifies that processing returns to the collect information point in the call where FlexDial collectable processing resumes. DPIDX identifies a FLEXDIAL table index containing a list of collectables to be processed for the call. The collectable list replaces any collectables remaining to be processed, and processing returns to the collect information point in the call so that the identified collectable list may be executed.	N/A

—continued—

Table 7-17
Table FLEXFEAT NOANSDUR option fields (continued)

Fields	Description	Values	Default
TRMT	If the ACTION field is set to NOANSTRMT, then this field is present and identifies the treatment value set and subsequently applied to the call.	Range of extended_treatment type ***Note:*** See *UCS DMS-250 Data Schema Reference Manual* for values of the extended treatment type.	N/A
INDEX	If the ACTION is set to DPIDX, then this field is present and identifies the FLEXDIAL table index that is applied for the call. A collectable list action of REPLACE is used for the collectable list represented by the FLEXDIAL table index.	An existing entry into table FLEXDIAL	N/A
	—end—		

The absence of this option for a provisioned index indicates that the no answer duration feature is not active.

Examples

An example of the NOANSDUR option provisioning is:

```
NOANSDUR 30 RTEADV
```

In this example, the no answer duration option is activated with a timer value of thirty seconds. If the timer expires before answer is received, then a route advance action occurs for the call.

NOANSDUR option restrictions and limitations

This option does not have any identified table control restrictions.

ONNET option

The On Network (ONNET) option identifies that the call is a network call, rather than an off-network call.

Fields

The ONNET option does not contain any field refinements. The presence of the option indicates that the call is an on-network call. The absence of the option indicates that the call is considered an off-network call.

Examples

An example of the ONNET option is:

```
ONNET
```

ONNET option restrictions and limitations

The ONNET option does not have any identified table control restrictions.

PVSPDIDX option

The Private Speed Index (PVSPDIDX) option identifies the index used for private speed number screening. Speed dial numbers are received during address collectable processing, and private speed dial numbers are identified using the CT PRVSPD selector in table STDPRTCT.

Fields

Table 7-18 contains the field refinement for the PVSPDIDX option.

Table 7-18
Table FLEXFEAT PVSPDIDX option field

Field	Description	Values	Default
INDEX	This field contains the private speed list index for the SPEEDTAB table for private speed number screening.	1 to 262140	0

Examples

An example of the PVSPDIDX option is:

```
PVSPDIDX 214
```

In this example, the private speed number index of 214 is used for private speed numbers received on this call.

PVSPDIDX option restrictions and limitations

The PVSPDIDX option does not have any identified table control restrictions.

REORGACT option

The Reorigination Process (REORGACT) option identifies the availability of reorigination for the subscriber and provides the details required to process the reorigination. The type of reorigination allowed is defined in the REORGTYP option.

Reorigination on a call requires the REORGACT and REORGTYP options in table FLEXFEAT. Reorigination also requires the REORIGAL option in table TRKFEAT.

Note: See “REORIGAL option” on page 11-15.

The CALLTYPE collectable allows reorigination to be identified on a per call type basis instead of solely on a subscriber number basis.

Fields

Table 7-19 contains the field refinements for the REORGACT option.

Table 7-19
Table FLEXFEAT REORGACT option fields

Fields	Description	Values	Default
INDEX	This field identifies the index into the FLEXDIAL table that identifies the collectables that are to be processed upon active reorigination of the call.	An existing entry in table FLEXDIAL	N/A
MSGCTRID	This field contains an index into table MSGCTR and identifies messages that are to be posted upon active reorigination of the call.	An existing entry in table MSGCTR	N/A
DISCTMR	This field performs a dual role for the ONKEY and ONDISC reorigination types provisioned in the REORGTYPE option. For ONKEY reorigination type, this field identifies the amount of time in seconds the user has to enter the reorigination digit after called party disconnect occurs. For ONDISC reorigination type, if reorigination is not immediate, then this field identifies the amount of time to delay after called party disconnect before reorigination automatically occurs. If both ONKEY and ONDISC reorigination types are provisioned in the REORGTYP vector of the REORGTYP option, then this timer value marks the transition from ONKEY to ONDISC reorigination support.	1 to 30 seconds	N/A

Examples

An example of the REORGACT option is:

```
REORGACT REORIG_IDX 14 15
```

In this example, reorigination is processed with the following information:

- Fifteen seconds after disconnect, if the reorigination digit as identified by an ONKEY type of reorigination is not pressed, then reorigination automatically occurs if an ONDISC type of reorigination is enabled. If ONDISC is not enabled, the call is taken down.
- When the call reoriginates, the FlexDial collectable list identified by the REORIG_IDX is processed.
- When the call reoriginates, messages identified by the MSGCTR table index 14 are posted.

REORGACT option restrictions and limitations

The REORGACT option must be provisioned in combination with the REORGTYP option. An attempt to provision the REORGACT option without the REORGTYP option results in the following error message:

```
Must provision REORGTYP with REORGACT.
```

All errors result in the failure of requested table control changes.

REORGTYP option

The Reorigination Type (REORGTYP) option identifies how reorigination is executed by the subscriber, and is used in conjunction with the REORGACT option, which identifies how the reorigination is processed. To enable reorigination on a call, the FLEXFEAT REORGACT and REORGTYP options must be used as well as the TRKFEAT REORIGAL option.

Note: See “REORIGAL option” on page 11-15.

Call processing has three types of reorigination:

- ONDISC identifies that reorigination occurs at disconnect, either immediately or after a certain period of time.
- ONKEY STR specifies that reorigination occurs through STR hardware monitoring of the identified reorigination digit. This may occur before answer, during the talking state of the call, and after called party disconnect.
- ONKEY UTR specifies that reorigination occurs through universal tone receiver (UTR) hardware monitoring of the identified reorigination digits This may occur before answer and after called party disconnect.

> ***Note:*** Currently, a maximum of 60 UTR channels can be provisioned on an individual XPM. Use of UTR reorigination capability should be limited in order that UTR resources are not exhausted for call processing (call origination) use.
>
> UTR reorigination cannot occur during the talking state of the call due to limitations in available UTR resources in the XPM peripheral.

The REORGTYP option may be provisioned up to three times in a table entry, each with one of the different types of reorigination available (ONDISC, ONKEY STR, or ONKEY UTR). It is not possible to provision the same type of reorigination within two or more REORGTYP options.

Through use of the CALLTYPE collectable, reorigination can be identified on a per call type basis instead of solely on a subscriber number basis.

Fields

Table 7-20 contains the field refinement for the REORGTYP option.

Table 7-20
Table FLEXFEAT REORGTYP option field

Field	Description	Values	Default
REORGTYP	This field consists of a reorigination activation type that outlines how reorigination is to be setup for the call	ONDISC or ONKEY ONDISC identifies that reorigination occurs after disconnect of the called party following expiration of the delay timer identified by the DISCTMR field. ONKEY identifies that reorigination occurs when the reorigination key is pressed during the ringing state of the call, the talking state (for STR only), or within the time specified by the DISCTMR timer value after called party disconnect occurs. ***Note:*** The UTR KEYRCVR option has precedence over STR ONKEY processing for reorigination during the ringing state of the call and after called party disconnect if the same reorigination digit is specified by both options. The ONDISC reorigination type may only be used once in a table entry, while ONKEY may be used in two different REORGTYP options when both the STR and UTR KEYRCVR types are used.	N/A

ONDISC reorigination type

Table 7-21 contains the field that requires provisioning with the ONDISC reorigination.

Table 7-21
Table FLEXFEAT ONDISC option field

Field	Description	Values	Default
IMMED	If the REORGTYP is set to ONDISC, then this field is present and identifies if reorigination occurs immediately upon called party disconnect, or if the DISCTMR timer is used to delay automatic reorigination.	N or Y	N/A

ONKEY Reorigination Type

Table 7-22 contains the field that requires provisioning with the ONKEY reorigination.

Table 7-22
Table FLEXFEAT ONKEY option field

Field	Description	Values	Default
KEYRCVR	If the REORGTYP is set to ONKEY, then this field is present and identifies the type of resource used to scan for reorigination key presses.	STR or UTR	N/A

STR key receiver type

Table 7-23 contains the fields that require provisioning with the STR ONKEY receiver type.

Note 1: The REORIG_SHORT_OR_LONG office parameter in table OFCVAR affects the range and the interpretation of the TKEYDUR and NTKEYDUR values. When this office parameter is set to LONG, the UCS DMS-250 switch interprets the option range of 4 to 30 as a range of 500 ms to 3000 ms in 100 ms increments. When this office parameter is set to SHORT, the UCS DMS-250 switch interprets the option range of 4 to 30 as a range of 40 milliseconds to 300 milliseconds in 10 ms increments.

Note 2: When the REORIG_SHORT_OR_LONG office parameter is set to LONG, both the 4 and 5 values represent 500 milliseconds.

Table 7-23
Table FLEXFEAT STR ONKEY REORGTYP option fields

Field	Description	Values	Default
STRDIGIT	If the REORGTYP is set to ONKEY and the KEYRCVR field is set to STR, then this field is present and identifies the digit that is dialed to reoriginate the call. S represents the asterisk digit, and P represents the octothorpe digit.	S or P	N/A
TKEYDUR	If the REORGTYP is set to ONKEY and the KEYRCVR field is set to STR, then this field is present and identifies the filter time for receipt of the reorigination digit during the talking state of the call.	4 to 30	N/A
NTKEYDUR	If the REORGTYP is set to ONKEY and the KEYRCVR field is set to STR, then this field is present and identifies the filter time for receipt of the reorigination digit during the ringing state of the call and after disconnect of the called party occurs.	4 to 30	N/A

UTR key type

Table 7-24 contains the field that requires provisioning with the UTR ONKEY receiver type.

Table 7-24
Table FLEXFEAT UTR ONKEY REORGTYP option field

Field	Description	Values	Default
UTRDIGIT	If the REORGTYP is set to ONKEY and the KEYRCVR field is set to UTR, then this field is present and identifies the digit that is dialed to reoriginate the call. S represents the asterisk digit, SS represents two asterisk digits, and P represents the octothorpe digit.	S, SS, P	N/A

The minimum duration for the reorigination digits for UTR ONKEY reorigination type is set at 24 ms.

Examples

An example of the REORGTYP option is:

```
(REORGTYP ONDISC N) (REORGTYP ONKEY STR P 20 5) (REORGTYPE
ONKEY UTR SS)$
```

In this example, all three reorigination types are provisioned for the table entry, and subscriber reorigination can occur in multiple ways:

- Two asterisk digits of normal duration can be pressed during the ringing state of the call or after called party disconnect before expiration of the DISCTMR value provisioned in the REORGACT option.
- An octothorpe digit of a 500 ms duration can be pressed during the ringing state of the call or after called party disconnect before expiration of the DISCTMR value, or an octothorpe digit with a two second duration may be pressed during the talking state of the call. This assumes the office parameter REORIG_SHORT_OR_LONG is set to long.
- In the amount of time after disconnect as identified by the DISCTMR value, if the reorigination digit is not pressed, then reorigination automatically occurs.

REORGTYP option restrictions and limitations

The following restrictions and limitations apply to the REORGTYP option:

- The REORGTYP option must be provisioned in combination with the REORGACT option. An attempt to provision the REORGTYP option without the REORGACT option results in the following error message:

  ```
  Must provision REORGACT with REORGTYP.
  ```

- The ONKEY reorigination type value may only be provisioned in up to two REORGTYP options for a table entry when both STR and UTR KEYRCVR values are used. An attempt to provision duplicate REORGTYP options with either the ONKEY STR KEYRCVR or ONKEY UTR KEYRCVR results in the following error message:

  ```
  KEYRCVR types or ONDISC may not be duplicated in REORGTYP
  options.
  ```

- The ONDISC reorigination type value may not be used more than once in REORGTYP options provisioned in a table entry. An attempt to provision duplicate REORGTYP options with the ONDISC reorigination type results in the following error message:

  ```
  KEYRCVR types or ONDISC may not be duplicated in REORGTYP
  options.
  ```

All errors result in the failure of requested table control changes.

REVALIDATE option

The FLEXFEAT Table REVALIDATE option as applicable for the subscriber number or call type definition is identical to the definition of the option for table FLEXTYPE. See “Revalidate option” on page 5-13 for more information on the REVALIDATE option.

SPLASHBK option

The Splash Back (SPLASHBK) option identifies an index into the SPLASHID table to provide a special splashback tone before a treatment tone is applied.

Fields

Table 7-25 contains the field refinement for the SPLASHBK option.

Table 7-25
Table FLEXFEAT SPLASHBK option field

Field	Description	Values	Default
INDEX	This field contains the index into the SPLASHID table for this feature. A value of zero means this feature is inactive.	1 to 4	0

Examples

An example of the SPLASHBK option is:

```
SPLASHBK 1
```

This example identifies that the splash back index of one provides special splash back tones before treatment is applied to the call.

SPLASHBK option restrictions and limitations

The SPLASHBK option does not have any identified table control restrictions.

TRANSNUM Option

The TRANSNUM (translated number) option is used to identify a translated number for the subscriber number being validated through tables FLEXVAL and FLEXFEAT. After validation has occurred, the translated number replaces the validated subscriber number digits for the remainder of call processing.

This option does not apply for CALLTYPE collectable processing and is subsequently ignored.

Fields

Table 7-26 contains the field refinement for the TRANSNUM option.

Table 7-26
Table FLEXFEAT TRANSNUM option field

Field	Description	Values	Default
DIGITS	This field identifies the translated number for the “dialed” subscriber number being processed.	Vector of up to 18 of { 0 TO 9, A, B, C, D} { A, B, C, D } represent the 4th column DTMF digits.	N/A

Examples

Examples of the TRANSNUM option are:

```
TRANSNUM 2146841000
```

This example identifies a translated number of 2146841000 for the processed subscriber number.

Provisioning Order

There are no provisioning order requirements for the TRANSNUM option.

TRANSNUM option restrictions and limitations

The following provisioning restrictions or limitations exist for TRANSNUM option use.

- Only a single TRANSNUM option may be provisioned for the FLEXFEAT table entry. If an attempt to provision more than one TRANSNUM option in the tuple is performed, the following error message is displayed:

  ```
  Duplicate TRANSNUM, only one allowed.
  ```

All errors result in the failure of requested table control changes.

TCAPANNC option

The TCAP Announcement (TCAPANNC) option identifies the custom announcement index into the TCAPANNC table for custom announcement values received in the N00 TCAP application.

Fields

Table 7-27 contains the field refinement for the TCAPANNC option.

Table 7-27
Table FLEXFEAT TCAPANNC option field

Field	Description	Values	Default
INDEX	This field contains the index that is used in the TCAPANNC table to provide a mapping of the custom announcement parameter value to a defined action.	An available index into table TCAPANNC	0

Examples

An example of the TCAPANNC option is:

```
TCAPANNC 15
```

This example identifies that the TCAPANNC table index of 15 maps the received custom announcement value into an action for call processing.

TCAPANNC option restrictions and limitations

The TCAPANNC option does not have any identified table control restrictions.

Note: For more information on the interaction of FlexDial AXXESS agents with the TCAP feature, see the *UCS DMS-250 Transaction Capabilities Application Part (TCAP) Application Guide*. For information on table FEATBYTE, which identifies call types and interaction requirements related to information contained in a TCAP response message, see the *UCS DMS-250 Data Schema Reference Guide*.

TRANSTS option

The Translation Serving Translation Scheme (TRANSTS) option identifies the STS used for translations.

Fields

Table 7-28 contains the field refinements for the TRANSTS option.

Table 7-28
Table FLEXFEAT TRANSTS option fields

Fields	Description	Values	Default
STSTYP	This field identifies whether the STS is directly provisioned or is acquired by looking up the provisioned partition digits in the PARTOSTS table.	STS or PARTITION	N/A
STS	If the STSTYPE is equal to STS, then this field is present and contains the value of the STS provisioned.	An existing entry in table HNPACONT	N/A

—continued—

Fields	Description	Values	Default
OPART	If the STSTYPE is equal to PARTITION, then this field is present and contains the value of the Originating Partition number.	0 to 999	N/A
TPART	If the STSTYPE is equal to PARTITION, then this field is present and contains the value of the Terminating Partition number.	0 to 31	N/A
—end—			

If this option is not present, then the DEFAULT_STS office parameter defines the STS used for translations.

Examples

Examples of the TRANSTS option are:

```
TRANSTS STS 214
```

This example identifies that the STS value of 214 is to be used for translations purposes.

```
TRANSTS PARTITION 611 25
```

This example identifies that the OPART and TPART values of 611 and 25 is to be used to determine the STS value for translations purposes.

TRANSTS option restrictions and limitations

There are no identified table control restrictions for the TRANSTS option.

TRANSYS Option

The TRANSYS (translations system) option is used to identify the translations system for the call according to the subscriber number or calltype processed. The translations system may either be national (NA), international (IN), or international partitioned (IP). Each translations system refers to a specific set of tables that are to be used to translate the call.

Fields

Table 7-29 contains the field refinements for the TRANSYS option.

Table 7-29
Table FLEXFEAT TRANSYS option fields

Fields	Description	Values	Default
TRANSYS	This field identifies the translations system to use for the call.	NP, NA, IN, IP NP indicates that the translations system is not identified. NA identifies that the national translations system is to be used. IN identifies that the international translations system is to by used. IP identifies that the international partitioned translations system is to be used.	NA

Example

Examples of the TRANSYS option are:

```
TRANSYS NA
```

This example identifies that the translations system is set to national for the call.

Provisioning Order

There are no provisioning order requirements for the TRANSYS option.

TRANSYS restrictions and limitations

The following provisioning restrictions or limitations exist for TRANSYS option use.

- Only a single TRANSYS option may be provisioned for the FLEXFEAT table entry. If an attempt to provision more than one TRANSYS option in the tuple is performed, the following error message is displayed:

  ```
  Duplicate TRANSYS, only one allowed.
  ```

All errors result in the failure of requested table control changes.

System requirements

The UCS DMS-250 switch allocates memory for table FLEXFEAT on an as needed basis for a maximum of 16.8 million table entries. Memory is allocated dynamically per entry, based on storage requirements of options provisioned. A minimum of 24 bytes or a maximum of 264 bytes can be allocated per table entry.

Datafill order

The FLEXFEAT table depends on provisioning in the FLEXDIAL, CDRTMPLT, CAINGRP, HNPACONT, and MSGCTR tables for defined options.

Indexes in table FLEXFEAT must be provisioned before they can be referenced by the table FLEXVAL entries or table FLEXDIAL CALLTYPE collectables.

Dump and restore

This table does not require a dump and restore from any other tables.

The one night process (ONP) of the table is performed through a physical data move.

Requirements and limitations

The following requirements and limitations apply for table FLEXFEAT:

- The FLEXFEAT table is limited to 16.8 million unique entries. An attempt to provision more than 16.8 million entries results in the following error message:

  ```
  Table FLEXFEAT is full. No more entries are possible.
  ```

- Indexes in the FLEXFEAT table may not be deleted if they are referenced in other tables. An attempt to remove a referenced index results in the following error message:

  ```
  Cannot remove referenced FLEXFEAT index.
  ```

- Options may not be duplicated within the FLEXFEAT table, with the exception of the MSGCTR option. An attempt to provision a duplicated option results in the following error message:

  ```
  Cannot duplicate option <option>.
  ```

  <option> identifies the FLEXFEAT option duplicated.

- A maximum of eight MSGCTR options may be provisioned within a FLEXFEAT table entry. An attempt to provision more than eight MSGCTR options within a single table entry results in the following error message:

  ```
  Maximum MSGCTR exceeded, only 8 allowed.
  ```

All errors result in the failure of requested table control changes.

Comparison to existing subscriber number provisioning

Without FlexDial, the following tables handle the provisioning scheme for subscriber numbers:

- ANISCUSP
- AUTHCODU, AUTHCDU2, AUTHCDU3, AUTHCDU4, AUTHCDU5
- MULTIPIN
- ACSCRN2

Tables 8-1 and 8-2 identify how the non-FlexDial subscriber number feature and characteristic provisioning maps into the FlexDial framework provisioning scheme.

ANISCUSP table mapping

Table 8-1 contains the ANISCUSP table to FlexDial framework mapping.

Table 8-1
ANISCUSP table to FlexDial framework mapping

Old field	New table	New field	Notes
STATUS	FLEXFEAT	FAILVAL option CASUAL option SUBR digit collectable CASUBLK option	An invalid status defines a treatment to not let the call continue.
ACCTLEN	FLEXFEAT FLEXDIAL	DPIDX option –> SUBR digit collectable	Using the DPIDX option, a FLEXDIAL index is specified that contains the properly provisioned SUBR digit collectable that collects ACCT digits. The SUBR digit collectable identifies the MIN and MAX values.
	FLEXFEAT	MSGCTR option	SUBR message, MINMAX message type

Table 8-1
ANISCUSP table to FlexDial framework mapping (continued)

Old field	New table	New field	Notes
ACCTVAL	FLEXFEAT FLEXDIAL	DPIDX option –> SUBR digit collectable	Use of the INSWITCH FLEXVAL values for the SUBR VALIDATE option
BCNAME	FLEXFEAT	BCCOMPAT option	
OPART	FLEXFEAT	TRANSTS option	
TERMPART	FLEXFEAT	TRANSTS option	
SATRES			The FlexDial framework does not use this characteristic.
PINLEN	FLEXFEAT FLEXDIAL	DPIDX option –> SUBR digit collectable	Using the DPIDX option, a FLEXDIAL index is specified that contains the properly provisioned SUBR digit collectable that collects PIN type digits.
	FLEXFEAT	MSGCTR option	SUBR message, MINMAX message type
PININDEX	FLEXFEAT FLEXDIAL	DPIDX option –> SUBR digit collectable	Use of the INSWITCH FLEXVAL values for the SUBR VALIDATE option
	FLEXFEAT	MSGCTR option	SUBR message, INDEXES message type
PINDIGS	FLEXFEAT FLEXDIAL	DPIDX option –> SUBR digit collectable	Use of the INSWITCH MATCH values for the SUBR VALIDATE option
	FLEXFEAT	MSGCTR option	SUBR message, MATCH message type
COSINDEX	FLEXFEAT	MLTCOSID option	
ANIDELV	FLEXFEAT	DELIVER option	
ACCTIDX	FLEXFEAT FLEXDIAL	DPIDX option –> SUBR digit collectable	Use of the INSWITCH FLEXVAL values for the SUBR VALIDATE option
	FLEXFEAT	MSGCTR option	SUBR message, INDEXES message type
OPCHOICE	FLEXFEAT	MSGCTR option	ADDR message, OPER message type

—continued—

Table 8-1
ANISCUSP table to FlexDial framework mapping

Old field	New table	New field	Notes
CAINGRP	FLEXFEAT	CAINGRP option	
PASSTHRU	FLEXFEAT FLEXDIAL	DPIDX option –> ROUTE collectable	Using the ROUTE collectable, immediately specify a PASSTHRU route.

—end—

AUTHCODU table mapping

Table 8-2 contains the AUTHCODU table to FlexDial framework mapping.

Table 8-2
AUTHCODU table to FlexDial framework mapping

Old field	New table	New field or option	Notes
STATUS	FLEXFEAT	FAILVAL option CASUAL option	An invalid status defines a treatment to not let the call continue.
ACCTLEN	FLEXFEAT FLEXDIAL	DPIDX option –> SUBR digit collectable	Using the DPIDX option, a FLEXDIAL index is specified that contains the properly provisioned SUBR digit collectable that collects ACCT digits. The SUBR digit collectable identifies the MIN and MAX values.
	FLEXFEAT	MSGCTR option	SUBR message, MINMAX message type
ACSCRIDX	FLEXFEAT FLEXDIAL	DPIDX option –> SUBR digit collectable	Use of the INSWITCH FLEXVAL values for the SUBR VALIDATE option
	FLEXFEAT	MSGCTR option	SUBR message, INDEXES message type
OPART	FLEXFEAT	TRANSTS option	
TPART	FLEXFEAT	TRANSTS option	
PINDIGS	FLEXFEAT FLEXDIAL	DPIDX option –> SUBR digit collectable	Use the VALIDATE option for the SUBR collectable
	FLEXFEAT	MSGCTR option	SUBR message, MATCH message type

—continued—

Table 8-2
AUTHCODU table to FlexDial framework mapping (continued)

Old field	New table	New field or option	Notes
COSINDEX	FLEXFEAT	MLTCOSID option	
HOTLINE	FLEXFEAT	MSGCTR option	ADDR message, FILED message type
	FLEXDIAL	ADDR digit collectable	Use of the FILED option for the ADDR collectable
PVSINDEX	FLEXFEAT	PVSPDIDX option	
SATRES			The FlexDial framework does not use this characteristic.
FLDONLY	FLEXFEAT	FLDONLY option	
AUTHTRAP	FLEXFEAT	GENLOG option	Generates a FLEX log instead of a TRK log.
ACCTVAL	FLEXFEAT FLEXDIAL	DPIDX option –> SUBR digit collectable	Use of the INSWITCH FLEXVAL values for the SUBR VALIDATE option
SPLASHBK	FLEXFEAT	SPLASHBK option	
TRVALLOW	FLEXDIAL	SUBR digit collectable	CITYVAL validation on a trunk group basis, used in conjunction with the FLEXFEAT CITYCODE option
	FLEXFEAT	CITYCODE option CITYVAL option	CITYVAL validation on a subscriber number basis
PININDEX	FLEXFEAT FLEXDIAL	DPIDX option –> SUBR digit collectable	Use the VALIDATE option for the SUBR collectable
	FLEXFEAT	MSGCTR option	SUBR message, INDEXES message type
PINLEN	FLEXFEAT FLEXDIAL	DPIDX option –> SUBR digit collectable	Using the DPIDX option, a FLEXDIAL index is specified that contains the properly provisioned SUBR digit collectable that collects PIN type digits.
	FLEXFEAT	MSGCTR option	SUBR message, MINMAX message type

—continued—

Table 8-2
AUTHCODU table to FlexDial framework mapping (continued)

Old field	New table	New field or option	Notes
OPCHOICE	FLEXFEAT	MSGCTR option	ADDR message, OPER message type
CAINGRP	FLEXFEAT	CAINGRP option	

—end—

ACSCRN2 table mapping

The ACSCRN2 table does not contain any fields. If the ACCT digits are found in the table lookup, then the ACCT number has been successfully validated.

MULTIPIN table mapping

The MULTIPIN table does not contain any fields. If the PIN digits are found in the table lookup, then the PIN number has been successfully validated.

Table TRKGRP

This chapter describes the Trunk Group (TRKGRP) table.

Purpose

Table TRKGRP identifies trunk groups and their related features (both AXXESS and non-AXXESS). For the FlexDial framework, the TRKGRP table provisions AXXESS trunk groups. These AXXESS trunk groups represent the only agent that can be used with the FlexDial framework.

Note: FLEXCONV, the FlexDial conversion tool, enables you to convert non-AXXESS trunk members to AXXESS trunk members. See Appendix B for more information.

General layout

The general layout of the trunk group table is not altered. A trunk group type and a refinement area for that trunk group type are included for the AXXESS agency.

Key

The TRKGRP table is indexed by the common language location identifier (CLLI) that is the defined name of the trunk group, and must be provisioned in table CLLI.

Table 9-1 contains the TRKGRP table key.

Table 9-1
TRKGRP table key

Key Field	Description	Values
CLLI	The table key consists of a valid CLLI as defined in table CLLI.	Valid index to table CLLI

Fields

The existing TRKGRP table field layout is not altered, but a new value for the GRPTYP field is specified for the table provisioning. The new AXXESS value is used to define AXXESS trunk groups and increases the range of the GRPTYP field as outlined in tables 9-2 and 9-3 below.

Note: For a complete description of the TRKGRP table fields, see the *UCS DMS-250 Data Schema Reference Manual*.

Table 9-2
TRKGRP table fields with AXXESS GRPTYP

Field	Description	Values	Default
GRPTYP	This field identifies the trunk group type for the provisioned trunk group.	IT, MAINT, E911, T101, DS0, NU, TI, T2, T0, TTL2, LOOPA, SOCKT, TOPS, T250, ONAT, ONAL, IMT, EANT, DAL, TL, ATR, C101, PRA250, EDAL, IT250, **AXXESS, NILGRPTYP**	N/A
TRAFSNO	This field identifies the traffic separation number for use with the traffic separation feature.	0 to 127	N/A
PADGRP	This field identifies the pad group for the trunk group.	Valid pad group as defined by the PADDATA table.	N/A
NCCLS	This field identifies the no circuit class type for the trunk group.	NCRT, NCTC, NCLT, NOSC, NCBN, NCID, NCOT, NCIT, NCIM, NCON, NCOF	N/A
GRPINFO	This field identifies the area refinements that are defined by trunk group type.	Existing refinement, plus the new refinement applicable to the **AXXESS GRPTYP**	N/A

Table 9-3 provides the area refinement fields for the AXXESS trunk.

Table 9-3
AXXESS GRPTYP trunk group refinement area fields

Field	Description	Values	Default
SELSEQ	This field identifies the trunk member selection algorithm for selecting the next idle member from the trunk group.	MIDL, LIDL, ASEQ, DSEQ, WIDEBAND	N/A
WBSELSEQ	If the SELSEQ field is set to WIDEBAND, then this field is present and identifies the selector algorithm for the wideband trunk group.	ASEQ or SEQ	N/A
WBGRPING	If the SELSEQ field is set to WIDEBAND, then this field is present and identifies the DS1 grouping algorithm for selecting DS1 members of the wideband group.	FIXED, FLOATING, FLEXIBLE	N/A
WBSEARCH	If the SELSEQ field is set to WIDEBAND, then this field is present and identifies the search algorithm for finding idle members for the wideband group.	BESTFIT or FIRSTFIT	N/A
SIGIDX	This field identifies the index into the TRKSIG table that contains the signaling characteristics of members of this trunk group.	A vector of up to two of an existing entry in table TRKSIG	N/A
FEATIDX	This field identifies the index into the TRKFEAT table that contains the feature characteristics of the trunk group.	An existing entry in table TRKFEAT	N/A

—continued—

Table 9-3
AXXESS GRPTYP trunk group refinement area fields (continued)

Field	Description	Values	Default
DPIDX	This field consists of a vector of FLEXDIAL table indexes that identify the initial interaction with the originating agency for call setup.	Vector of up to 4 of an existing entry in table FLEXDIAL	N/A
OGRPTYP	This field identifies the type of outpulsing used for the AXXESS trunk group. The range of values identifies the different trunk groups that the AXXESS trunk group mimics when outpulsing.	DAL, ONAL, ONAT, EANT	N/A
—end—			

Provisioning the AXXESS agent with a WIDEBAND SELSEQ value is subject to the same conditions as existing inter-machine trunk (IMT) and Feature Group D (FGD) agents. The UDWS0001 software optionality control (SOC) must be in the ON state in order to provision the WIDEBAND SELSEQ value. When transitioning from ON to IDLE, the transition will fail if there are any AXXESS agents provisioned with a WIDEBAND SELSEQ value.

Note: For information on software optionality control (SOC), refer to the *UCS DMS-250 Software Optionality Control (SOC) User's Manual*.

System requirements

The data store requirements for the TRKGRP table are not changed.

Datafill order

To minimize the impact to existing agents, the AXXESS trunk uses a terminal type defined as ABAXX_TERM, whereas existing agents use the AB250 terminal type.

In table LTCINV, the ABAXX_TERM terminal type must be provisioned using the AXX250 exec lineup, as this is the only exec lineup supported for per-trunk signaling (PTS) AXXESS trunk group members. Existing agents continue using the DTC250 or UTR250 exec lineups.

The amount of store in the XPM for exec lineups is limited to 12151 bytes. The sizes of the exec lineups used for the UCS DMS-250 switch are:

- UTR250—5225 bytes (used with AB250 terminal type)
- DTC250—4754 bytes (used with AB250 terminal type)
- AXX250—4366 bytes (used with ABAXX_TERM terminal type)
- DTCEX—2941 bytes (used with ABTRK terminal type)

Additionally, the XPM on which members of the AXXESS trunk group are provisioned must contain UTR hardware. If the XPM does not contain UTR hardware, then members of the AXXESS agent provisioned will fail to return to service (RTS).

In order to provision an AXXESS trunk group tuple in the TRKGRP table, the CLLI, CLLICDR, TRKSIG, TRKFEAT, and FLEXDIAL tables must be previously datafilled with the necessary tuples.

Table TRKGRP with multiple TRKSIG indexes must be provisioned before tables ISUPDEST and TRKMEM. When a single TRKSIG table entry is provisioned in the SIGIDX vector, then a SGRP value of 0 is used for tables ISUPDEST and TRKMEM (and other tables requiring a subgroup value as part of the provisioning; for example, table BBTKSGRP). When multiple TRKSIG table entries are provisioned in the SIGIDX vector, then the first TRKSIG table index provisioned in the vector is represented when using a SGRP value of 0, and the second TRKSIG table index provisioned in the vector is represented when using the SGRP value of 1.

Dump and restore

No dump and restore of non-FlexDial group types to the AXXESS group type is supported. The current dump and restore capability of the TRKGRP table is not modified.

For UCS08, AXXESS trunk groups are restored with a single TRKSIG table index in the SIGIDX vector.

Restrictions and limitations

The following restrictions and limitations apply for table TRKGRP:

- At least one valid TRKSIG table index must be provisioned in the SIGIDX vector. The NIL TRKSIG table index cannot be used as a valid index. If an attempt to provision an empty SIGIDX vector is performed, then the following error message is displayed:

  ```
  Must provision a TRKSIG index.
  ```

  If an attempt to provision only NIL TRKSIG indices in the SIGIDX vector is performed, then the following error message is displayed:

```
Must provision a non–NIL TRKSIG index.
```

- When removing a TRKSIG table index from the vector, all members utilizing that TRKSIG index must be removed from table TRKMEM before the TRKSIG index can be removed from the SIGIDX vector. If an attempt to remove the TRKSIG index is performed prior to removing all trunk group members referencing the TRKSIG index is performed, then the following error message is displayed:

  ```
  Cannot remove TRKSIG index referenced by trunk members.
  ```

- All members of a trunk group utilizing a particular TRKSIG table index must be off-line before the SIGIDX vector can be changed. If an attempt to change a TRKSIG index within the SIGIDX vector is performed prior to off–lining all trunk group members referencing that TRKSIG table index is performed, then the following error message is displayed:

  ```
  Cannot change TRKSIG index <n> until all members using
  index <n> in table TRKMEM are set to INB.
  ```

- Incompatibilities between signalling types is not allowed between TRKSIG table indices provisioned in the SIGIDX vector. If CCS7 is used for one TRKSIG profile, then CCS7 must also be used for the second. Likewise if PTS signalling (FXS, FXO, DS1) is used for one TRKSIG profile, then PTS must also be used for the second. If an attempt to mix signalling types for a TRKGRP tuple is performed, then the following error message is displayed:

  ```
  Cannot mix signalling types in multiple TRKSIG indices.
  ```

- Existing checks which are currently performed to verify compatibility between the TRKSIG, TRKFEAT, and TRKGRP provisioning for an Axxess trunk group occurs for both TRKSIG indices when provisioned.
- With the current limitations on exec lineup store, the UTR250, AXX250 and DTCEX exec lineups cannot be included on a single DTC. Combinations of exec lineups with associated terminal types may be provisioned providing the store limitation is not exceeded.
- PTS AXXESS trunk group members do not properly RTS unless the AXX250 exec lineup is provisioned for the XPM in table LTCINV, and UTR hardware is available for the XPM.
- All defined AXXESS trunk types are two-way trunks.
- The value of zero must be used when provisioning AXXESS agents in tables requiring a subgroup value to be entered, such as tables TRKMEM and ISUPDEST.
- AXXESS trunk groups mimic the outpulsing scheme of the trunk group type identified by the OGRPTYP field.

- The WIDEBAND SELSEQ value may not be provisioned for AXXESS agents unless the UDWS0001 SOC is in an ON state. When transitioning from ON to IDLE for this SOC, the transition will fail if there are any AXXESS agents provisioned with a WIDEBAND SELSEQ value.
- The selection sequence value of WIDEBAND may only be used with CCS7 type signaling. An attempt to use other signaling results in the following error message:

  ```
  WIDEBAND SELSEQ value is only supported for CCS7 type
  signaling.
  ```

- The WIDEBAND search algorithm may not be changed via a tuple update. An attempt to do so results in the following error message:

  ```
  Cannot change <current> to <new>.
  ```

  where <current> is the selection sequence before the initiated change and <new> is the value attempting to change to.

- Existing wideband trunk group member (table TRKMEM) validation is performed for AXXESS trunk groups.
- The TRKSIG index specified for an AXXESS trunk group cannot be changed until all the group members defined in table TRKMEM are set to an INB state. An attempt to change the TRKSIG index while trunk group members are not all INB results in the following error message:

  ```
  Cannot change TRKSIG index until all members in table
  TRKMEM are set to INB.
  ```

TRKSIG and TRKFEAT indexes used are validated against the trunk group data. Individual datafill errors for TRKSIG include the following. (See Chapter 10 for information on table TRKSIG.)

- The DELIVER option is only applicable for DS1 and CCS7 type signaling agents. An attempt to provision the DELIVER option for non-DS1 or SS7 signaling results in the following error message:

  ```
  DELIVER is only applicable for DS1 and CCS7 sigtypes.
  ```

- The CPIALLOW option is only applicable for DS1 signaling type and the DAL outgoing group type as identified by the TRKGRP OGRPTYP field. An attempt to datafill the CPIALLOW option against other signaling types or outgoing group types results in the following error message:

  ```
  Option CPIALLOW is only applicable to DS1 signaling type
  and OGRPTYP DAL.
  ```

- The DETDIAL option is only applicable to DAL and ONAL outgoing trunk group types identified by the TRKGRP OGRPTYP field. An attempt to datafill DETDIAL against other OGRPTYP values results in the following error message:

  ```
  DETDIAL is only applicable for DAL and ONAL OGRPTYPs.
  ```

- The DIGSOUTP option is only applicable for the DAL outgoing group type as identified by the TRKGRP OGRPTYP field. An attempt to datafill the DIGSOUTP option against other outgoing group types results in the following error message:

  ```
  DIGSOUTP is only applicable for DAL OGRPTYP.
  ```

- The MLTSTAGE option is only applicable for DS1 signaling type and the EANT outgoing group types as identified by the TRKGRP OGRPTYP field. An attempt to datafill the MLTSTAGE option against other signaling types or outgoing group types results in the following error message:

  ```
  MLTSTAGE is only applicable for DS1 SIGTYPE and EANT
  OGRPTYP.
  ```

- The CICDELV option is only applicable for CCS7 and DS1 Multi-stage signaling agents. An attempt to provision the CICDELV option for other agents results in the following error message:

  ```
  CICDELV only allowed on CCS7 and DS1 MLTSTAGE agents.
  ```

All errors result in the failure of requested table control changes.

Table TRKSIG

This chapter describes the FlexDial Trunk Group Signaling (TRKSIG) table.

Purpose

The TRKSIG table identifies the signaling and interface facility characteristics of the AXXESS trunk group. For more information on AXXESS trunks, refer to the Intro chapter of this document. This table replaces the use of the TRKSGRP table for AXXESS agents.

The design of the TRKSIG table with relation to the TRKGRP table reverses the order of the trunk group to signaling provisioning. This enables many trunk groups that typically have identical signaling and interface facility characteristics to share a single TRKSIG table index.

General layout

The TRKSIG table is indexed by a 16-character string (1 to 16 characters) as the key that identifies the tuple that contains the signaling information for the trunk group.

Key

The key for indexing the TRKSIG table is a 16-character string. This index provides for 8192 unique table entries.

Table 10-1 provides the TRKSIG table key.

Table 10-1
TRKSIG table key

Key field	Description	Values
KEY	The table key consists of a string of up to 16 characters. 8192 possible unique entries are provided.	Vector of up to 16 characters

Fields

The information in the TRKSIG table identifies the signaling information for the trunk group, including the initial digit collection signaling parameter values for PTS AXXESS trunk groups. The SIG protocol collectable modifies the digit collection signaling information during the interaction with the originating agent. Table 10-2 provides the table TRKSIG fields.

Table 10-2
TRKSIG table fields

Field	Description	Values	Default
SIGTYPE	This field identifies the signaling type of the trunk group (represents the card code). The remaining TRKSIG fields are refined based on the SIGTYPE value.	DS1, FXS, FXO (for PTS); CCS7	N/A
RESETDIGIT	This field defines either the octothorpe or the asterisk as the reset digit.	P or S P=octothorpe, S=asterisk	N/A

In comparison to existing TRKSGRP datafill, the SIGTYPE values map accordingly:

- DS1—maps to CARDCODE = DS1SIG and SIGDATA = STD.
- FXS—maps to CARDCODE = FXSLS or FXSGS and SIGDATA = STD.
- FXO—maps to CARDCODE = FXOLS or FXOGS and SIGDATA = STD.
- CCS7—maps to CARDCODE = DS1SIG and SIGDATA = C7UP.

The remaining TRKSIG fields are refined based on the signaling type indicator.

Since most trunks need to enforce a common reset digit throughout the dialplan, the RESETDIGIT field allows the craftsperson to define one reset digit throughout the dialplan within table TRKSIG. The craftsperson can also modify the reset digit on a per–call basis, as required.

The reset digit can still be redefined by the collectables OLI, CIC, SUBR, SIG, REPLDIG, COLDIGS, and ADDR. This feature enhances this capability, while optimizing performance.

Interactions

The reset digit may be the same digit as the identified terminating digit, in which case special rules of processing are engaged. See the “General Application of Digit Collection” in this document.

The reset digit is no longer provisionable in the RESET option of FlexDial digit collectables, and must be identified through either table TRKSIG or through the FlexDial SIG collectable for DTMF reset capability.

Call Processing Restrictions / Limitations

The following restrictions / limitations apply to reset digit call processing:

- Identifying the reset digit does not guarantee reset digit capability during DTMF interaction digit collection. The digit collectable performing the DTMF digit collection request must contain the RESET option in order to enable reset digit capability.
- The identified reset digit in table TRKSIG may be overridden through use of the SIG collectable.

PTS signaling type TRKSIG data refinements

The per-trunk signaling types consist of the DS1, FXS, and FXO SIGTYPE values. These signaling types share the same data refinement for additional datafill.

Table 10-3 contains the PTS signaling types refinement fields.

Table 10-3
PTS signaling type refinement fields

Field	Description	Values	Default

ISTARTSG	This field identifies the start signal sent to acknowledge receipt of the originating seizure signal	SIGTYPE=DS1:
		IM, DD, WK, S Z
		SIGTYPE=FXS, FXO:
		GS, L S
			N/A

IPULSTYP	This field identifies the pulse type for incoming digits.	NP, MF, DTMF	N/A
PSEIZTMR	If the incoming pulse type is not equal to NP, then this field is present and identifies the permanent signal timeout value.	1 to 30 seconds	N/A

—continued—

Table 10-3
PTS signaling type refinement fields (continued)

Field	Description	Values	Default
PDILTMR	If the incoming pulse type is not equal to NP, then this field is present and identifies the partial dial timeout value. This value identifies the maximum amount of time between received digits until the minimum amount of digits to collect are received.	1 to 30 seconds	N/A
MINRTMR	This field identifies the maximum amount of time between received digits after the minimum number of digits have been collected.	1 to 30 seconds	N/A
FDIGMASK	If the incoming pulse type is equal to MF, then the First Digit Mask field is present and identifies the types of digits that are allowed to be the first digit received. All other digits received before this first digit are ignored.	Subset of : {KP, KPP} or ALL or NONE ALL includes all possible KP digits within the set, while NONE indicates no first digit mask is identified.	N/A
LDIGMASK	If the incoming pulse type is equal to MF, then the Last Digit Mask field is present and identifies the types of digits that identify when the last digit in the MF stream has been received.	Subset of {ST, STP, ST2P, ST3P} or ALL or NONE ALL includes all possible ST digits within the set, while NONE indicates that no last digit mask is identified.	N/A
DIGMASK	If the incoming pulse type is equal to DTMF, then the Digit Mask field is present and identifies the types of DTMF digits that are reported. Any digits received that are not contained within this set are ignored. A, B, C, and D refer to 4th column DTMF digits. S represents the asterisk digit and P represents the octothorpe digit.	Subset of {0 to 9, A, B, C, D, S, P}	N/A

—continued—

Table 10-3
PTS signaling type refinement fields (continued)

Field	Description	Values	Default
TRMDIGIT	If the incoming pulse type is equal to DTMF, then this field is present and identifies the terminating digit that when entered causes all digits collected to be immediately reported. This digit value is automatically included in the DIGMASK set. S represents the asterisk digit and P represents the octothorpe digit.	S or P	N/A
DIALMODE	This field identifies if the received digits are customer dialed or machine produced.	C or M	N/A

OSTARTSG	This field identifies the start signal that is received as an acknowledgment in response to an outgoing seizure signal.	SIGTYPE=DS1:
		IM, DD, WK, SZ
		SIGTYPE=FXS, FXO:
		GS, LS
			N/A

OPULSTYP	This field identifies the pulse type for outpulsing inband digits.	NP, MF, DTMF ***Note:*** DP pulsing is not supported for AXXESS agencies.	N/A
OIDGTMR	This field identifies the outpulsing inter-digit timer value. This is the amount of pause time between outpulsed digits.	0 to 100 in ten-millisecond intervals.	N/A

—continued—

Table 10-3
PTS signaling type refinement fields (continued)

Field	Description	Values	Default
TRKGRDTM	This field identifies the amount of time that the trunk remains idle after disconnect before it can be seized for another call.	For subgroups with IM, DD, and WK start signals, the guard time range is {7 to 225} in ten-millisecond intervals. (70 ms to 2.25 seconds). For ground start and loop start sig {7 to 225} in 160 millisecond intervals. (1.12 seconds to 36 seconds).	N/A
OPTIONS	This field contains a vector of optional fields that can be added to the TRKSIG tuple for the DS1, FXS, and FXO signaling types, except where specifically noted.	NIL, ACKWINK, ALTSEIZ, ANSWFLTR, ATDANS, BCCOMPAT, CICBLK, CICSIZE, CPIALLOW, DELIVER, DETDIAL, DIGSOUTP, ECSTAT, ESUPR, GLAREYD, IRINGCHK, MLTSTAGE, ODSCFLTR, ORIGFLTR, OUTCIC, REMBSY, RETOFFHK, TDSCFLTR, SPMECIDX Up to 23 options can be provisioned in the OPTIONS vector.	N/A
—end—			

Each option may define fields for provisioning purposes.

PTS signaling type NIL option

The TRKSIG table NIL option as applicable for trunk groups is identical to the definition of the option for table FLEXTYPE. See “NIL option” on page 5-2 for more information.

PTS signaling type ACKWINK Option

The Acknowledgment Wink (ACKWINK) option for DS1 signaling agents indicates that an acknowledgment wink is to be transmitted upon completion of the interaction with the originating agent. The acknowledgment wink may be transmitted only once through one of the following call processing triggers:

- during execution of a SIG collectable, where the IPULSTYP is transitioning from MF to DTMF tone collection on the agent
- after completion of the FlexDial interaction, prior to terminating the call to a set treatment
- after completion of the FlexDial interaction, following allocation and seizure of the terminating trunk agent member

The ACKWINK option must be used for AXXESS agents mimicking a FGD protocol.

Fields

The ACKWINK option does not contain any field refinements. The presence of the option indicates that an acknowledgment wink is transmitted. The absence of the option indicates that an acknowledgment wink is not transmitted.

Note: Transmitting a wink without using the ACKWINK option is possible using the SNDSIGWNK collectable instruction.

ACKWINK option limitations and restrictions

The ACKWINK option is only applicable for the DS1 signaling type. An attempt to datafill the ACKWINK option against other signaling types results in the following error message:

```
ACKWINK is only applicable for DS1 SIGTYPE.
```

PTS signaling type ALTSEIZ option

The Alternate Seizure (ALTSEIZ) option indicates that a different AB bit seizure configuration is to be used.

Fields

The ALTSEIZ option does not contain any field refinements. The presence of the option indicates that the alternate AB bit seizure configuration is to be used. The absence of the option indicates that the alternate AB bit seizure configuration is not to be used.

ALTSEIZ option restrictions and limitations

The ALTSEIZ option is only applicable to FXS and FXO signaling types. An attempt to datafill the ALTSEIZ option against other signaling types results in the following error message:

```
ALTSEIZ is only applicable for FXS and FXO SIGTYPEs.
```

PTS signaling type ANSWFLTR option

The Answer Filter (ANSWFLTR) option identifies the non-standard filter time used to screen the off-hook answer signal on the terminating agent. The standard time is the default if this option is not present and is defined as 80 ms.

Fields

Table 10-4 contains the ANSWFLTR option field refinement.

Table 10-4
PTS signaling type ANSWFLTR option field

Field	Description	Values	Default
ANSWFLTR	This field identifies the answer filter time for the terminator. This is the amount of time that the terminator must remain off-hook before the answer signal is processed.	5 to 255 ten-millisecond intervals, excluding the value of 8	8 (80 ms)

ANSWFLTR option restrictions and limitations

A provisioned default value is not stored in the table entry. An attempt to provision the default value results in the following warning message:

```
Warning—The value selected for ANSWFLTR is the default value
of 8 (80 ms). Therefore, the ANSWFLTR option is not stored
with and subsequently not displayed for the table entry.
```

PTS signaling type ATDANS option

The Audio Tone Detector (ATD) Answer Detect (ATDANS) option specifies an ATD receiver to detect software answer for calls terminating to members of the trunk group.

Fields

Table 10-5 contains the ATDANS option field refinement.

Table 10-5
PTS signaling type ATDANS option field

Field	Description	Values	Default
DELAYTMR	This field identifies the amount of time after outpulsing that the ATD delays before being activated to detect answer.	1 to 100 in 160-millisecond increments.	N/A

ATDANS option restrictions and limitations

The ATDANS option does not have any identified restrictions.

PTS signaling type BCCOMPAT option

The TRKSIG table BCCOMPAT option as applicable for trunk groups is identical to the definition of the option for table FLEXFEAT. See “BCCOMPAT option” on page 7-3 for more information on BCCOMPAT option syntax.

PTS signaling type CICBLK option

The CICBLK option indicates that the DS1 multistage terminating trunk does not deliver CIC information.

Fields

Table 10-6 contains the CICBLK option field refinement.

Table 10-6
PTS signaling type CICBLK option field

Field	Description	Values	Default
CICBLK	This field identifies whether to block CIC delivery for national calls, international calls, or both types of calls.	BLK_CIP, BLK_TNS, BLK_BOTH BLK_CIP blocks the delivery of the CIC for national calls. BLK_TNS blocks the delivery of the CIC for international calls. BLK_BOTH blocks the delivery of the CIC for both national and international calls.	N/A

CICBLK option restrictions and limitations

- The PTS signaling type CICBLK option only applies to DS1 Multistage signaling agents. If you attempt to enter the CICBLK option on another PTS type signaling agent, the following error message is displayed:

  ```
  CICBLK only allowed on CCS7 and DS1 MLTSTAGE agents.
  ```

- Only one CICBLK option can be datafilled per TRKSIG tuple. If an attempt to datafill more than one CICBLK option per TRKSIG tuple is made, then the following error message is displayed:

  ```
  Cannot duplicate option CICBLK.
  ```

PTS signaling type CICSIZE option

The CICSIZE option specifies the number of digits to outpulse to the terminating trunk agency.

Table 10-7 contains the CICSIZE option field refinement.

Table 10-7
PTS signaling type CICSIZE option field

Field	Description	Values	Default
CICSIZE	This field identifies the size of the CIC.	3DIGS, 4DIGS	If the CIC4_TRANS_COMP office parameter is set to N, the default is 3DIGS. If the CIC4_TRANS_COMP office parameter is set to Y, the default is 4DIGS.

CICSIZE option restrictions and limitations

- A size of 3DIGS is not allowed when the CIC4_TRANS_COMP office parameter is set to Y. If you attempt to enter a size of 3DIGS when this parameter is set to Y, the following error message is displayed:

  ```
  ERROR: CANNOT DATAFILL CICSIZE=3DIGS WHEN OFFICE PARM
  CIC4_TRANS_COMP=Y
  ```

- The CIC4_TRANS_COMP office parameter cannot be set to Y if there are any tuples in table TRKSIG with a CICSIZE of 3DIGS.

  Note: For more information, see the *UCS DMS-250 Office Parameters Reference Manual*.

PTS signaling type CPIALLOW option

The Calling Party Identification (CPIALLOW) option indicates that the calling party identification is outpulsed on DAL outgoing trunk group type terminations.

Fields

The CPIALLOW option does not contain any field refinements. The presence of the option indicates that the calling party information is outpulsed according to the feature description. The absence of the option indicates that the CPI information is not outpulsed.

CPIALLOW option restrictions and limitations

The following restrictions and limitations apply for the CPIALLOW option:

- The CPIALLOW option cannot be provisioned for the TRKSIG table entry unless the NSER0001 software optionality control (SOC) is in an ON state. When transitioning this SOC from ON to IDLE, the transition will fail if there are any TRKSIG entries provisioned with the CPIALLOW option.
- The CPIALLOW option is only applicable for the DS1 signaling type and the DAL outgoing group type as identified by the TRKGRP OGRPTYP field. An attempt to datafill the CPIALLOW option against other signaling types or outgoing group types results in the following error message:

  ```
  Option CPIALLOW is only applicable to DS1 signaling type
  and OGRPTYP DAL.
  ```

PTS signaling type DELIVER option

The TRKSIG table DELIVER option as applicable for trunk groups is identical to the definition of the option for table FLEXFEAT. See “Deliver option” on page 7-14 for more information on DELIVER option syntax.

DELIVER option restrictions and limitations

The DELIVER option can only be provisioned for DS1 and CCS7 signaling types. An attempt to provision the DELIVER option for non-DS1 or SS7 signaling results in the following error message:

```
DELIVER is only applicable for DS1 and CCS7 sigtypes.
```

PTS signaling type DETDIAL option

The Detect Dialtone (DETDIAL) option indicates that dialtone has to be detected as the proceed to send signal for outpulsing situations.

Fields

The DETDIAL option does not contain any field refinements. The presence of the option indicates that the switch must detect dialtone as the Proceed To Send signal. The absence of the option indicates that the switch does not need to detect as the Proceed To Send signal for outpulsing.

DETDIAL option restrictions and limitations

The DETDIAL option is only applicable to DAL and ONAL outgoing trunk group types identified by the TRKGRP OGRPTYP field. An attempt to datafill DETDIAL against other OGRPTYP values results in the following error message:

```
DETDIAL is only applicable for DAL and ONAL OGRPTYPs.
```

PTS signaling type DIGSOUTP option

The Digits To Outpulse (DIGSOUTP) option identifies the number of digits to outpulse for DAL outgoing group type trunk groups.

Fields

Table 10-8 contains the DIGSOUTP option refinement field.

Table 10-8
PTS signaling type DIGSOUTP option field

Field	Description	Values	Default
DIGSOUTP	This field identifies the number of digits to outpulse.	0 to 10	N/A

DIGSOUTP option restrictions and limitations

The DIGSOUTP option is applicable only for the DAL outgoing group type as identified by the TRKGRP OGRPTYP field. An attempt to datafill the DIGSOUTP option against other outgoing group types results in the following error message:

```
DIGSOUTP is only applicable for DAL OGRPTYP.
```

PTS signaling type ECSTAT option

The Echo Canceller Status (ECSTAT) option identifies the status of the echo canceller for members of the trunk group.

Note: See the *UCS DMS-250 NT6X50EC Integrated Echo Canceller Application Guide*.

Fields

Table 10-9 contains the ECSTAT option field refinements.

Table 10-9
PTS signaling type ECSTAT option fields

Field	Description	Values	Default
STATUS	This field identifies the status of the echo canceller capability for the trunk group members.	INTERNAL, EXTERNAL, INNOTONE, UNEQ INTERNAL indicates the echo canceller on this trunk group is equipped with the NT6X50EC card in the DTC frame and is enabled by call processing if the call is not a data call. This value is not allowed when the echo suppressor is enabled. EXTERNAL indicates the echo canceller is not equipped on the DTC. No call processing control is involved. INNOTONE indicates that internal echo cancellation is active without 2100 Hz tone control. This value is not allowed when the echo suppressor is enabled.	UNEQ Identifies that echo cancellers are not equipped for members of the trunk group.
NSMATCH	If the status field is set to INTERNAL or INNOTONE, then this field is present identifies if noise matching is activated.	N or Y	N/A
AUTOON	If the status field is set to INTERNAL, then this field is present and indicates that auto re-enable control is activated.	N or Y	N/A

ECSTAT option restrictions and limitations

The STATUS value of INTERNAL is not allowed to be provisioned when the Echo Suppressor option is also provisioned. An attempt to do so results in the following error message:

```
INTERNAL ECSTAT value may not be used with the ESUPR option.
```

PTS signaling type ESUPR option

The Echo Suppressor (ESUPR) option identifies that the trunk group has echo suppressors activated.

Fields

Table 10-10 contains the ESUPR option field refinement.

Table 10-10
PTS signaling type ESUPR option field

Field	Description	Values	Default
ESUPR	This field identifies the activation of the echo suppressor.	H, F, N H—Half F—Full N—None	N Identifies that echo suppressors are not activated for members of the trunk group.

ESUPR option restrictions and limitations

The ECSTAT STATUS value of INTERNAL is not allowed to be provisioned when the ESUPR option is provisioned. An attempt to do so results in the following error message:

```
INTERNAL ECSTAT value may not be used with the ESUPR option.
```

PTS signaling type GLAREYD option

The Glare Yield (GLAREYD) option identifies that the members of the trunk group yield in glare situations.

Fields

The GLAREYD option does not contain any field refinements. The presence of the option indicates the members of the trunk group yield in glare situations. The absence of the option indicates that the members of the trunk group do not yield to glare.

GLAREYD option restrictions and limitations

The GLAREYD option does not have any identified restrictions.

PTS signaling type IRINGCHK option

The incoming ring check (IRINGCHK) option indicates that ringing is required for seizure.

Fields

The IRINGCHK option does not contain any field refinements. The presence of the option indicates that ringing is required for seizure. The absence of the option indicates that ringing is not required to accompany the incoming seizure signal.

IRINGCHK option restrictions and limitations

The IRINGCHK option is applicable only to FXO signaling types. An attempt to datafill the IRINGCHK option against other signaling types results in the following error message:

```
IRINGCHK is only applicable for FXO SIGTYPE.
```

PTS signaling type MLTSTAGE option

The Multistage (MLTSTAGE) option indicates that multistage outpulsing is to be used for FGD outgoing group type trunk groups.

Fields

The MLTSTAGE option does not contain any field refinements. The presence of the option indicates that multistage outpulsing is used according to the feature description. (For more information on multistage outpulsing, see the *UCS DMS-250 Feature Group D (FGD) Application Guide*.) The absence of the option indicates that multistage outpulsing is not used.

MLTSTAGE option restrictions and limitations

The MLTSTAGE option is applicable only for DS1 signaling type and the EANT outgoing group types as identified by the TRKGRP OGRPTYP field. An attempt to datafill the MLTSTAGE option against other signaling types or outgoing group types results in the following error message:

```
MLTSTAGE is only applicable for DS1 SIGTYPE and EANT OGRPTYP.
```

PTS signaling type ODSCFLTR option

The On-hook Disconnect Filter (ODSCFLTR) option identifies the non-standard filter time for screening the on-hook disconnect signal on the originating agent. The standard time is the default if this option is not present and is defined as 160 ms.

Fields

Table 10-11 contains the ODSCFLTR option field refinement.

Table 10-11
PTS signaling type ODSCFLTR option field

Field	Description	Values	Default
ODSCFLTR	This field identifies the disconnect filter time for the originator. This is the amount of time that the originator must remain on-hook before the disconnect signal is processed.	5 to 255 ten millisecond intervals, excluding the value of 16.	16 (160 ms)

ODSCFLTR option restrictions and limitations

A provisioned default value is not stored in results in the following warning message:

```
Warning—The value selected for ODSCFLTR is the default value
of 16 (160 ms). Therefore, the ODSCFLTR option is not stored
and subsequently not displayed for the table entry.
```

PTS signaling type ORIGFLTR option

The Originating Filter (ORIGFLTR) option identifies the non-standard filter time for screening the off-hook seizure signal on the originating agent. The standard time is the default if this option is not present and is defined as 160ms.

Fields

Table 10-12 contains the ORIGFLTR option field refinement.

Table 10-12
PTS signaling type ORIGFLTR option field

Field	Description	Values	Default
ORIGFLTR	This field identifies the originating signal filter time for the originating agent. This is the amount of time the seizure signal must be remain off-hook before the origination is processed.	5 to 255 ten-millisecond intervals, excluding the value of 16.	16 (160 ms)

ORIGFLTR option restrictions and limitations

A provisioned default value is not stored in the table entry. An attempt to provision the default value results in the following warning message:

```
Warning—The value selected for ORIGFLTR is the default value
of 16 (160 ms). Therefore, the ORIGFLTR option is not stored
and subsequently not displayed for the table entry.
```

PTS signaling type OUTCIC option

The OUTCIC option determines the CIC digits to outpulse over the terminating trunk.

Fields

Table 10-13 contains the OUTCIC option field refinement.

Table 10-13
PTS signaling type OUTCIC option field

Field	Description	Values	Default
CICDIGS	This field identifies the digits to outpulse.	Vector of 3 or 4 of {0 to 9}	N/A
OVERRIDE	This field specifies whether to override the received or default CIC.	Y, N	N

OUTCIC option restrictions and limitations

The following restrictions and limitations apply to the PTS signaling type OUTCIC option:

- The PTS signaling type OUTCIC option is only applicable for DS1 multistage signaling agents. If you attempt to enter the OUTCIC option on another PTS signaling agent, the following error message is displayed:

```
OUTCIC only allowed on CCS7 and DS1 MLTSTAGE agents.
```

- The CICDIGS field must be three or four digits. If you attempt to enter less than three digits, the following error message is displayed:

```
OUTCIC must be 3 or 4 digits in length.
```

- CICDIGS 000 and 0000 are invalid values. If you attempt to enter either of these values, the following error message is displayed:

```
OUTCIC digits <digits> are not valid.
```

- If the TRKSIG tuple does not have a CICSIZE option, the CICDIGS field must correspond with the default size of the CIC4_TRANS_COMP office parameter or the following error message is displayed:

```
OUTCIC digits must correspond with CIC4_TRANS_COMP.
```

- If the TRKSIG tuple has a CICSIZE option, the CICDIGS of the OUTCIC option must correspond with the size indicated by the CICSIZE option, otherwise the following error message is displayed:

```
OUTCIC digits must correspond with the CICSIZE option.
```

PTS signaling type REMBSY option

The Remote Make Busy (REMBSY) option identifies if the trunk group members provide RMB (Remote Make Busy) display status. A trunk group member enters the RMB state if the other side of the facility bearer channel enters a man busy (MB) state.

Fields

The REMBSY option does not contain any field refinements. The presence of the option indicates the Remote Make Busy feature is active for the trunk group. The absence of the option indicates that the Remote Make Busy feature is not active for the trunk group.

REMBSY option restrictions and limitations

The REMBSY option does not have any identified restrictions.

PTS signaling type RETOFFHK option

The Return Off-hook (RETOFFHK) option identifies when the answer signal is sent to the originating agency. Before the outgoing seizure point in the call, the answer signal may be sent through use of the SNDSIG protocol collectable.

Fields

Table 10-14 contains the RETOFFHK option field refinement.

Table 10-14
PTS signaling type RETOFFHK option field

Field	Description	Values	Default
RETOFFHK	This field identifies the return off-hook value for trunk group members. This identifies when the answer signal is be sent to the originating agency.	NIL, AOS, AOP, AAD NIL indicates that answer signal is never sent to the originating agent. AOS indicates that the answer signal is sent After Outgoing Seizure occurs. AOP indicates that the answer signal is sent After Outpulsing occurs.	AAD (After Answer Detected)

RETOFFHK option restrictions and limitations

The RETOFFHK option does not have any identified restrictions.

PTS signaling type TDSCFLTR option

The Terminating Screening Filter (TDSCFLTR) option identifies the non-standard filter time for screening the on-hook disconnect signal on the terminating agent. The standard time is the default if this option is not present and is defined as 160 ms.

Fields

Table 10-15 contains the TDSCFLTR option field refinement.

Table 10-15
PTS signaling type TDSCFLTR option field

Field	Description	Values	Default
TDSCFLTR	This field identifies the disconnect filter time for the terminator. This is the amount of time that the terminator must remain on-hook before the disconnect signal is processed.	5 to 255 ten-millisecond intervals, excluding the value of 16.	16 (160 ms)

TDSCFLTR option restrictions and limitations

A provisioned default value is not stored in the table entry. An attempt to provision the default value results in the following warning message:

```
Warning — The value selected for TDSCFLTR is the default
value of 16 (160 ms). Therefore, the TDSCFLTR option is not
stored and subsequently not displayed for the table entry.
```

PTS signaling type SPMECIDX option

The Spectrum echo canceller (ECAN) Index (SPMECIDX) option provisions a Spectrum trunk with a Spectrum ECAN. The SPMECIDX option is associated with an integer value that indexes table Spectrum ECAN (SPMECAN) for the selection of control parameters.

Fields

Table 10-16 contains the SPMECIDX option field refinement.

Table 10-16
PTS signaling type SPMECIDX option field

Field	Description	Values	Default
SPMECIDX	This field indexes table SPMECAN for the selection of control parameters for provisioning Spectrum ECANs.	0 to 255	N/A

SPMECIDX option restrictions and limitations

The following restrictions apply for the SPMECIDX option:

- The SPMECIDX option cannot be added to a tuple if the ESUPR option is present in the tuple with a value other than “N.” An attempt to do so results in the following error message:

  ```
  SPMECIDX option is NOT allowed along with the ESUPR option
  with value other than N.
  ```

- The index supplied to the SPMECIDX option must be present in table SPMECAN.

For more information on how Spectrum supports AXXESS trunks, see the *UCS DMS-250 Spectrum Reference Manual.*

CCS7 signaling type TRKSIG data refinements

The Common Channel Signaling #7 (CCS7) signaling type identifies its own data refinement for provisioning SS7 type trunks in the TRKSIG table.

Table 10-17 contains the CCS7 SIGTYPE refinement fields.

Table 10-17
CCS7 SIGTYPE refinement fields

Field	Description	Values	Default
PROTOCOL	This field identifies the Q.764 protocol variant for this ISUP agency.	UCP	N/A
ADJNODE	This field identifies the adjacent node data for the trunk group. This field is an index into the ADJNODE table.	An existing entry in table ADJNODE	N/A
ISUPIDX	This field identifies the ISUP index for ISUP message formatting and control.	UCS2EAEO	N/A
OPTIONS	This field consists of an options vector which contains all the options applicable to this trunk group.	NIL, ABCNTRL, BCCOMPAT, CICBLK, CICSIZE, COT, DELIVER, ECSTAT, ESUPR, GLARE, NO_HOP, OUTCIC, SPMECIDX Up to 23 options may be provisioned within the OPTIONS vector.	N/A

CCS7 signaling type NIL option

The TRKSIG table NIL option as applicable for trunk groups is identical to the definition of the option for table FLEXTYPE. See “NIL option” on page 5-2 for more information.

CCS7 signaling type ABCNTRL option

The A-bit Control (ABCNTRL) option identifies if A-bit bearer channel signaling is active for the trunk group members.

Fields

The ABCNTRL option does not contain any field refinements. The presence of the option indicates that bearer channel A-bit signaling is active for the trunk group. The absence of the option indicates that bearer channel A-bit signaling is not active.

ABCNTRL option restrictions and limitations

The ABCNTRL option does not have any identified restrictions.

CCS7 signaling type BCCOMPAT option

This option is identical to the BCCOMPAT option for PTS signaling types.

CCS7 signaling type CICBLK option

This option is identical to the CICBLK option for PTS signaling types.

CCS7 signaling type CICSIZE option

This option is identical to the CICSIZE option for PTS signaling types.

CCS7 signaling type COT option

The Continuity Test (COT) option identifies the percentage of calls on the trunk group that request a continuity test to be performed during call setup.

Fields

Table 10-18 contains the COT option fields.

Table 10-18
CCS7 signaling type COT option fields

Fields	Description	Values	Default
PERCENT	This field identifies the percentage of calls on this trunk group that request a continuity test to be performed.	1 to 100	0 Indicates that COT testing is not performed for members of the trunk group.
CHKTYPE	This field identifies the type of continuity test that is performed when requested.	THRL, TLRH, THRH, LOOPAROUND, 2W2W	N/A

COT option restrictions and limitations

The COT option does not have any identified table control restrictions.

CCS7 signaling type DELIVER option

This option is identical to the DELIVER option for PTS signaling types.

CCS7 signaling type ECSTAT option

This option is identical to the ECSTAT option for PTS signaling types.

CCS7 signaling type ESUPR option

This option is identical to the ESUPR option for PTS signaling types.

CCS7 signaling type GLARE option

The Glare Action (GLARE) option identifies how glare situations are processed by members of the trunk group. The glare action can be set to either carrier identification code (CIC) or STAND. By default if the option is not present for the TRKSIG entry, trunk group members YIELD in glare situations.

The CIC action identifies that the following algorithm is used for determining which side stands or yields to glare:

- Even numbered trunk group members stand to glare for the switch with the higher point code. Even numbered circuits on the switch with the lower point code yield to glare.
- Odd numbered trunk group members stand to glare for the switch with the lower point code. Odd numbered circuits on the switch with the higher point code yield to glare.

The trunk group member circuit numbers (circuit identification codes) are identified by the C7TRKMEM table.

The STAND action identifies that the trunk group members are to stand in glare situations.

Fields

Table 10-19 contains the GLARE option field refinement.

Table 10-19
CCS7 signaling type GLARE option field

Field	Description	Values	Default
ACTION	This field identifies the action for trunk group members to take in glare situations.	CIC, STAND, YIELD	YIELD Indicates that all members yield in glare situations.

GLARE option restrictions and limitations

A provisioned default value is not stored in the table entry. An attempt to provision the default value results in the following warning message:

```
Warning – The value selected for GLARE is the default value
of YIELD. Therefore, the GLARE option is not stored and
subsequently not displayed for the table entry.
```

CCS7 signaling type NO_HOP option

The NO_HOP option is applicable to originating trunk agents and disallows the UCS DMS-250 switch from modifying a received hop counter (HC) parameter value.

Note: Without this option, a received HC parameter value is decremented before the call is presented to the terminating switch.

Fields

The NO_HOP option does not contain any field refinements. The presence of the option indicates that the NO_HOP option is active for the trunk agent and the switch is unable to modify a received HC parameter value. The absence of the option indicates that the NO_HOP option is not active and the switch is able to modify a received HC parameter value.

CCS7 signaling type OUTCIC option

This option is identical to the OUTCIC option for PTS signaling types.

CCS7 signaling type SPMECIDX option

This option is identical to the SPMECIDX option for PTS signaling types.

System requirements

The TRKSIG table uses the same physical store as the TRKSGRP table.

Datafill order

The TRKSIG table requires that the ADJNODE table be previously provisioned when provisioning CCS7 signaling types.

The TRKSIG table must be datafilled with the necessary indexes before the TRKGRP table tuple can be properly provisioned.

When provisioning tables in which a subgroup value is entered (such as tables TRKMEM and ISUPDEST), the subgroup value of zero is used for AXXESS agents.

Dump and restore

A reformat procedure must be written for older tuples to assign “P” to the RESETDIGIT field in table TRKSIG.

The ability to perform a physical data move of the TRKSIG table is not required for one night process (ONP) requirements.

Restrictions and limitations

The following restrictions and limitations apply for table TRKSIG:

- The value of zero must be used when provisioning AXXESS agents in tables requiring a subgroup value to be entered, such as tables TRKMEM and ISUPDEST.
- The 16-character string key for the TRKSIG table is limited to 8192 unique values. An attempt to provision more than 8192 entries results in the following error message:

  ```
  Table TRKSIG is full. No more entries are possible.
  ```

- Individual signaling options cannot be duplicated for a TRKSIG table index. An attempt to duplicate options results in the following error message:

  ```
  Cannot duplicate option <option>.
  ```

  <Option> identifies the option that is duplicated.

- Indexes in the TRKSIG table may not be deleted if they are referenced by TRKGRP entries. An attempt to remove a referenced index results in the following error message:

```
TUPLE REFERENCED BY ANOTHER TABLE—USE TABREF TO GET
POTENTIAL TABLE LIST.
```

- The combination of a SIGTYPE value of FXO and an ISTARTSIG value of LS is not allowed. An attempt to provision these values results in the following error message:

  ```
  Only GS ISTARTSG values are applicable for FXO sigtype.
  ```

- The FXS SIGTYPE only supports GS and LS start signal types for incoming and outgoing start signals. An attempt to provision a value other than GS or LS results in the following error message:

  ```
  Only GS, LS ISTARTSG values are applicable for FXS
  sigtype.
  ```

  ```
  Only GS, LS OSTARTSG values are applicable for FXS
  sigtype.
  ```

- The DS1 sigtype does not support GS and LS start signal values for either incoming or outgoing start signals. An attempt to provision these values results in one of the following error messages:

  ```
  Only IM, DD, WK, SZ ISTARTSG values are applicable for DS1
  sigtype.
  ```

  ```
  Only IM, DD, WK, SZ OSTARTSG values are applicable for DS1
  sigtype.
  ```

- The IM start signal is only supported for the DS1 signaling type in combination with the DTMF pulse type. An attempt to provision the IM start signal with other signaling and pulse type combinations results in the following error message:

  ```
  IM start signal is only applicable for DS1 SIGTYPE and
  DTMF pulse type.
  ```

- Error messages related to individual TRKSIG options are displayed as necessary during table TRKGRP verification of the TRKFEAT and TRKSIG indexes specified.

All errors result in the failure of requested table control changes.

Table TRKFEAT

This chapter describes the FlexDial Trunk Group Features (TRKFEAT) table.

Purpose

The TRKFEAT table identifies the feature characteristics of the AXXESS trunk group. This table replaces the use of tables TRKGRP and TRKGRP1 in identifying features and characteristics for AXXESS agents. The TRKFEAT table provides the ability to group existing trunk group features under one table. This ability coupled with the use of table FLEXDIAL allows the AXXESS trunk group to duplicate the capabilities of many existing trunk groups.

The relationship of the TRKFEAT table to the TRKGRP table creates a new sequence to trunk group feature and characteristic provisioning. This enables many trunk groups with identical feature characteristics to share a single TRKFEAT table index.

General layout

The TRKFEAT table is indexed by a 16-character string (1 to 16 characters) as the key that identifies a tuple consisting of a list of feature and characteristic trunk group options. All trunk group features and characteristics are available for provisioning.

Key

The key for indexing the TRKFEAT table is a 16-character string. This index provides for 8192 table entries.

Table 11-1 contains the TRKFEAT table key.

Table 11-1
TRKFEAT table key

Key Field	Description	Values
KEY	The table key consists of a string of up to 16 characters. 8192 possible unique entries are provided.	Vector of up to 16 characters

Fields

The TRKFEAT table does not contain a fixed set of fields, but rather consists of two options vectors that can be datafilled with defined features and characteristics. Each option defines its own data refinement as needed for datafill purposes. Therefore each option is listed independently.

The options are divided into the originating and terminating options vectors. The originating options apply when the trunk group is an originating agent in the call. The terminating options apply when the trunk group is a terminating agent in the call. Some options can apply on both originating and terminating agents, and are therefore listed independently within each vector definition.

The default field for the options identifies the value assumed for feature options if the option is not present for the feature index. Normally when a tuple consists of an options vector, the default is to not have any options datafilled. In other words, normally an option not present has no default value. However, in table TRKFEAT, most options have a default value used by call processing even if the option is not present. The default value is used by the feature in place of the provisioned option. In other cases, the absence of the option indicates that the feature does not actively perform any function for the call.

When datafilling the field of an option, a default value for the field type can be specified by the DEFDATA table.

Table 11-2 provides table TRKFEAT fields.

Table 11-2
Table TRKFEAT fields

Field	Description	Values	Default
ORIGOPTS	This field consists of an options vector that contains a list of the features and characteristics applicable to an originating trunk group.	NIL, ALTTRTMT, ANSCDR, CAIN, CAINGRP, CDRTMPLT, CICRTE, CICSIZE, CITYCODE, COSOVE, DEFCIC, IEXCLINX, LATA, MSGCTR, NETSEC, OHQ, REORIGAL, SNPA, SNXX, SUSTMR, TIMEBIAS, TRANSTS, TRKCOS, ZONE, JIP Up to 26 options may be provisioned within the options vector. ***Note:*** The listed options can only be provisioned once within the vector, with the exception of MSGCTR, which can be provisioned up to 8 times.	N/A
TERMOPTS	This field consists of an options vector that contains a list of the features and characteristics applicable to an terminating trunk group.	NIL, CAIN, CAINGRP, ENFNPAOP, NETSEC, NOANSDUR, OHQTERM, ONNETTRK, SNPA, SNXX, TRKCOS, ZONE Up to 9 options may be provisioned within the options vector.	N/A

NIL option

The TRKFEAT table NIL option as applicable for trunk groups is identical to the definition of the option for table FLEXTYPE. See “NIL option” on page 5-2 for more information.

ALTTRTMT option

The Alternate Treatment (ALTTRTMT) option applies only to originating agents and indicates that applied call treatments are to be decoded using the ALTERNAT subtable within the TMTCNTL table instead of using the AXXESS subtable.

Fields

The ALTTRTMT option does not contain any field refinements. The presence of the option indicates that the ALTERNAT subtable is used. The absence of the option indicates that the AXXESS subtable is used.

Example

An example of the ALTTRTMT option is:

```
ALTTRTMT
```

In this example, the ALTERNAT subtable of the TMTCNTL table is used for treatment decoding for the trunk group.

ALTERNAT option restrictions and limitations

The ALTERNAT option does not have any identified table control restrictions.

ANSCDR option

The TRKFEAT table ANSCDR option as applicable for originating trunk groups is identical to the definition of the option for table FLEXTYPE. See “ANSCDR option” on page 5-3 for more information on ANSCDR option syntax.

Carrier AIN (CAIN) option

The Carrier AIN (CAIN) option indicates that NetworkBuilder services are enabled for the call based on the originating trunk group (option ORIGOPTS) or the terminating trunk group (option TERMOPTS).

Note: See the *UCS DMS-250 NetworkBuilder Application Guide* for more information on the NetworkBuilder feature. See the *UCS DMS-250 CAIN/FlexDial Interactions* or “FlexDial Interactions” appendix for information on FlexDial interactions with CAIN.

Fields

The CAIN option does not contain any field refinements. The presence of the option indicates that NetworkBuilder services are enabled for the trunk group. The absence of the option indicates that NetworkBuilder services are not enabled for the trunk group.

Example

An example of the CAIN option is:

```
CAIN
```

In this example, NetworkBuilder services are enabled for the trunk group. Activation of the AIN services depends on the provisioning of the CAINGRP TRKFEAT or FLEXFEAT table option.

CAIN option restrictions and limitations

The CAIN option does not have any identified table control restrictions.

CAINGRP option

The TRKFEAT table CAINGRP option as applicable for both originating and terminating trunk groups is identical to the definition of the option for table FLEXFEAT. See section “CAINGRP option” on page 7-4 for more information on CAINGRP option syntax. See the *UCS DMS-250 CAIN/FlexDial Interactions* or ”FlexDial Interactions” appendix for information on FlexDial interactions with CAIN.

For TRKFEAT call processing access, the default value of “NIL_GROUP” is used for CAINGRP when the option is not provisioned.

CAINGRP option restriction and limitation

Before datafilling the CAINGRP option, datafill the identical CAIN group in table CAINGRP.

CDRTMPLT option

The TRKFEAT table CDRTMPLT option as applicable for originating trunk groups is identical to the definition of the option for table FLEXTYPE. See “CDRTMPLT option” on page 5-9 for more information.

CICRTE option

The CIC Route (CICRTE) option applies only to originating agents and specifies that translations and routing use the CIC value received on the originating trunk.

Fields

The CICRTE option does not contain any field refinements. The presence of the option indicates that translations and routing use the CIC value received for the trunk group. The absence of the option indicates that the CIC is ignored for the trunk group.

Example

An example of the CICRTE option is:

```
CICRTE
```

In this example, translations and routing use the information in table CICROUTE to route the call if there is no ANI or authcode to determine routing.

CICRTE option restrictions and limitations

The CICRTE option does not have any identified table control restrictions.

CICSIZE option

The CICSIZE option applies only to originating agents and verifies whether the incoming CIC is a three-digit or four-digit CIC.

Fields

Table 11-3 contains the CICSIZE refinement.

Table 11-3
Table TRKFEAT CICSIZE option field

Field	Description	Values	Default
SIZE	This field indicates the size of the CIC.	3DIGS, 4DIGS	If the CIC4_TRANS_COMP office parameter is set to N, the default is 3DIGS. If the CIC4_TRANS_COMP office parameter is set to Y, the default is 4DIGS.

Example

An example of the CICSIZE option is:

```
CICSIZE 4DIGS
```

CICSIZE option restrictions and limitations

- A size of 3DIGS is not allowed when the CIC4_TRANS_COMP office parameter is set to Y. If you attempt to enter a size of 3DIGS when this parameter is set to Y, the following error message is displayed:

  ```
  ERROR: CANNOT DATAFILL CICSIZE=3DIGS WHEN OFFICE PARM
  CIC4_TRANS_COMP=Y
  ```

- The CIC4_TRANS_COMP office parameter cannot be set to Y if there are any tuples in table TRKFEAT with a CICSIZE of 3DIGS.
- The CIC, SUBR or CICPARM, SUBRPARM collectables in table FlexDial process the incoming CIC.

CITYCODE option

The TRKFEAT table CITYCODE option as applicable for originating trunk groups is identical to the definition of the option for table FLEXFEAT. See “CITYCODE option” on page 7-7 for more information on CITYCODE option syntax.

Note: A CITYVAL validation attempt requires that the CITYCODE option be provisioned against both the trunk group and subscriber number. See “SUBR CITYVAL option” on page 2-85.

For TRKFEAT call processing access, the default value of 000 is used for CITYCODE when the option is not provisioned. When a value of 000 is provisioned with the option, the following warning message is displayed:

```
Warning – The value selected for CITYCODE is 000. Therefore,
the CITYCODE option is not stored with and subsequently not
displayed for the table entry.
```

COSOVE option

The Class Of Service (COS) Override (COSOVE) option applies only to originating agents and indicates that COS screening failure result is overridden for this trunk group (that is, the call does not terminate to treatment). The COSOVE option specifies the FLEXDIAL index that is processed due to class of service overriding.

Fields

Table 11-4 contains the COSOVE field refinement.

Table 11-4
Table TRKFEAT COSOVE option field

Field	Description	Values	Default
INDEX	This field identifies the index into the FLEXDIAL table.	An existing entry in table FLEXDIAL	N/A

The REPLACE action is used for handling the FLEXDIAL index. See “REPLACE list modification action” on page 15-13 for a description of the REPLACE action.

Example

An example of the COSOVE option is:

```
COSOVE SUBR_IDX
```

In this example, COS overriding is in effect for the trunk group. If COS screening fails, then the SUBR_IDX FLEXDIAL index identifies collectables that are processed for the call.

COSOVE option restrictions and limitations

The COSOVE option does not have any identified table control restrictions.

DEFCIC option

The DEFCIC option applies only to originating agents and defines a default CIC for the call when no CIC was received.

Fields

Table 11-5 contains the DEFCIC option field refinements.

Table 11-5
Table TRKFEAT DEFCIC option field

Field	Description	Values	Default
CIC	This field contains the default CIC	Vector of 3 or 4 digits of {0 to 9}	N/A
SUBOPTION	This field consists of an options vector that contains a list of characteristics applicable to the DEFCIC option.	NIL, OUPULSE, OPERRTE, CAINCIC, AGNTDATA	N/A

For the NIL suboption, see “NIL option” on page 5-2 for more information.

The OUTPULSE suboption determines whether the UCS DMS-250 switch is capable of outpulsing the default CIC to the terminating agency.

The OPERRTE suboption determines whether the default CIC can be used to index into table OPERRTE.

The CAINCIC suboption determines whether the default CIC can be used in the outgoing TCAP query of a CAIN message.

The AGNTDATA option determines whether the default CIC can be placed in the digit buffer by an AGNTDATA collectable.

Example

An example of the DEFCIC option is:

```
DEFCIC 333 (OUTPULSE)(CAINCIC)(AGNTDATA)$
```

In this example, a default CIC of 333 is provisioned for calls that do not receive a CIC. The default CIC is outpulsed to the terminating agency, can be used in a CAIN TCAP message, and can be placed in the digit buffer by an AGNTDATA collectable. It is not allowed to be used to index table OPERRTE.

DEFCIC option restrictions and limitations

The CIC field must be either three or four digits. If you attempt to enter fewer than three digits, the switch displays the following error message:

```
DEFCIC must be 3 or 4 digits in length
```

The CIC digits 000 and 0000 are invalid. If you attempt to enter them, the switch displays the following error message:

```
DEFCIC digits <digits> are not valid.
```

You must enter at least one DEFCIC suboption. If you attempt to enter a DEFCIC option with no suboptions, the switch displays the following error message:

```
At least one DEFCIC suboption must be present.
```

You are unable to duplicate DEFCIC suboptions. If you attempt to do so, the switch displays the following error message:

```
Cannot duplicate suboption <suboption>.
```

If a tuple with the DEFCIC option has a CICSIZE option, the CIC field must correspond with the CICSIZE option or the switch displays the following error message:

```
DEFCIC digits must correspond with the CICSIZE option.
```

If a tuple with the DEFCIC option has no CICSIZE option, the CIC field must correspond with the default size indicated by the CIC4_TRANS_COMP office parameter or the switch displays the following error message:

```
DEFCIC digits must correspond with CIC4_TRANS_COMP.
```

ENFNPAOP option

The Enforce NPA Outpulsing (ENFNPAOP) option applies only to terminating agents and forces the UCS DMS-250 switch to outpulse a ten-digit called party address as a ten-digit called party address. In other words, it disallows the UCS DMS-250 switch from stripping the NPA from the outpulsed number.

Fields

The ENFNPAOP option does not contain any field refinements. The presence of the option indicates that the UCS DMS-250 switch is unable to strip the NPA from a ten–digit called party address. The absence of the option indicates that the UCS DMS-250 switch is able to strip the NPA from a ten–digit called party address.

IEXCLINX option

The Incoming Exclusion Index (IEXCLINX) option applies only to originating agents and identifies the index into table IEXCLUDE that provides for NPA-NXX blocking.

Fields

Table 11-6 contains the IEXCLINX option field refinement.

Table 11-6
Table TRKFEAT IEXCLINX option field

Field	Description	Values	Default
INDEX	This field contains the index value for table IEXCLUDE.	0 to 255	0

Example

An example of the IEXCLINX option is:

```
IEXCLINX 15
```

In this example, incoming exclusion screening is performed for the trunk group using index 15 of the IEXCLUDE table.

IEXCLINX option restrictions and limitations

The IEXCLINX option may not be provisioned with the default value of 0. An attempt to provision the option with a value of 0 results in the following warning message:

```
Warning – The value selected for IEXCLINX is 0. Therefore,
the IEXCLINX option is not stored with and subsequently not
displayed for the table entry.
```

JIP option

A Jurisdiction Information Parameter (JIP) is an SS7 IAM parameter that carries geographic and service provider information that is associated with the calling party. The parameter contains the originating switch’s Location Routing Number (LRN) and is used to enhance Local Number Portability (LNP) capabilities. JIP applies to originating FGD and SS7 IMT trunks (excluding global).

The TRKFEAT JIP option field allows customers to designate a JIP on a per-trunk group basis when one is not present on the incoming call. It is sent to the Switching Control Point (SCP) in a query message (for NetworkBuilder) as well as built and sent out on the outgoing IAM if the terminator is SS7. When it is sent, the ORIGLRN CDR is populated with the JIP.

Note 1: If a default JIP is provisioned on the originating trunk, then the building and sending of the JIP takes place for all calls, not just for NetworkBuilder calls.

Note 2: For more information on NetworkBuilder and LNP, see the *UCS DMS-250 NetworkBuilder Application Guide* and the *UCS DMS-250 Local Number Portability (LNP) Feature Application Guide*.

Fields

Table 11-7 contains the JIP option field refinement.

Table 11-7
Table TRKFEAT JIP option field

Field	Description	Values	Default
JIP	This field contains the JIP parameter for the trunk group.	up to 10 digits	N/A

Example

An example of the JIP option is:

```
JIP 2145550000
```

JIP option restrictions and limitations

This option does not have any identified table control restrictions.

LATA option

The LATA option is only applicable for originating trunk groups. It contains an index into table LATAID.

Fields

Table 11-8 contains the LATA option field refinement.

Table 11-8
Table TRKFEAT LATA option field

Field	Description	Values	Default
LATAID	This field contains the index into table LATAID.	valid index into table LATAID	N/A

Example

An example of the LATA option is:

```
LATA LATA1
```

In this example, the TRKFEAT tuple is indexing into the LATA1 tuple in table LATAID.

LATA option restrictions and limitations

Provision table LATAID before the LATA option in table TRKFEAT.

MSGCTR option

The TRKFEAT Table Message Center (MSGCTR) option as applicable for originating trunk groups is identical to the definition of the option for the FLEXFEAT table. Refer to “MSGCTR option” on page 7-24 for more information on MSGCTR syntax.

NETSEC option

The NETSEC option applies to both originating and terminating agents and specifies the UCS DMS-250 switch must generate a network security log or CDR when the call is answered.

Fields

Table 11-9 contains the NETSEC option field refinements.

Table 11-9
Table TRKFEAT NETSEC option field

Field	Description	Values	Default
TMPLTIDX	This field specifies the current CDR templates defined for the UCS DMS-250 switch.	current NUM_CDR_ TEMPLATES	N/A
USEEDIT	This field is a boolean field.	Y, N	N/A
PROFIDX	This field is an index into table NETSPROF, which provides profile screening. ***Note:*** To activate profile screening, SOC UBFR0003 must be in the ON state.	0 to 255 ***Note:*** The value 0 indicates that profile screening is not enabled.	0

Example

An example of the NETSEC option is:

```
NETSEC UCS12 Y 255
```

In this example, the NETSEC option is active, uses the UCS12 CDR template, and indexes into table NETSPROF where the UCS DMS-250 switch performs profile screening on the call.

NETSEC option restrictions and limitations

This option does not have any identified table control restrictions.

Note: For more information on the NETSEC option, see the *UCS DMS-250 Feature Group D Application Guide*.

NOANSDUR option

The TRKFEAT Table No Answer Duration (NOANSDUR) option as applicable for terminating trunk groups is identical to the definition of the option for the FLEXFEAT table. Refer to section “NOANSDUR Option” on page 7-25 for more information on NOANSDUR syntax.

OHQ option

The Off-hook Queuing (OHQ) option indicates that the originating trunk group can use the off-hook queuing feature when selecting a route choice.

Fields

The OHQ option does not contain any field refinements. The presence of the option indicates that off-hook queuing is enabled for the trunk group. The absence of the option indicates that off-hook queuing is not enabled for the trunk group.

Example

An example of the OHQ option is:

```
OHQ
```

In this example, off-hook queuing can occur for trunk group member selection for originations on this trunk group.

OHQ option restrictions and limitations

The OHQ option does not have any identified table control restrictions.

OHQTERM option

The Off-hook Queuing Terminator (OHQTERM) option applies only to terminating agents and indicates that the trunk group can be used for off-hook queuing terminations.

Fields

The OHQTERM option does not contain any field refinements. The presence of the option indicates that the trunk group can be used for off-hook queuing terminations. The absence of the option indicates that the trunk group may not be used for off-hook queuing terminations.

Example

An example of the OHQTERM option is:

```
OHQTERM
```

In this example, off-hook queuing can occur for selection of terminating trunk group members of this trunk group.

OHQTERM option restrictions and limitations

This option does not have any identified table control restrictions.

ONNETTRK option

The On-net Trunk (ONNETTRK) option only applies to terminating agents and indicates that the trunk group is an on-network trunk group.

Fields

The ONNETTRK option does not contain any field refinements. The presence of the option indicates that the trunk group is an on-network trunk group. The absence of the option indicates that the trunk group is an off-network trunk group.

Example

An example of ONNETTRK provisioning is:

```
ONNETTRK
```

In this example, the on-net characteristic applies to the trunk group.

ONNETTRK option restrictions and limitations

The ONNETTRK option does not have any identified table control restrictions.

REORIGAL option

The Reorigination Allowed (REORIGAL) option applies only to originating agents and indicates that reorigination is allowed on this trunk group. If the REORIGAL option is not provisioned for a trunk group, then reorigination capability for all subscribers originating calls on the trunk group is disabled.

To activate reorigination, the REORGACT and REORGTYP FLEXFEAT table options must be used. See “REORGACT option” on page 7-28 and “REORGTYP option” on page 7-30.

Fields

Table 11-10 contains the REORIGAL field refinement.

Table 11-10
Table TRKFEAT REORIGAL option field

Field	Description	Values	Default
REVERIFY	This field contains a vector of FLEXTYPE names that identify types of subscriber numbers that must be re-validated on reoriginated calls. If the subscriber number type was not processed for the original call, then the entry is skipped during call processing.	Vector of up to 4 (existing entry in table FLEXTYPE)	N/A

Example

An example of the REORIGAL option is:

```
REORIGAL (AUTH)$
```

In this example, reorigination is allowed for subscribers originating on the trunk group. If an AUTH subscriber number type is processed on the original call, then the number is revalidated for the reoriginated call.

REORIGAL option restrictions and limitations

The following restrictions and limitations apply:

- FLEXTYPE names may not be duplicated in the REVERIFY vector. An attempt to duplicate FLEXTYPE names in the REVERIFY vector results in the following error message:

  ```
  Cannot duplicate FLEXTYPE name <FLEXTYPE>.
  ```

- For CAIN/FlexDial interactions, if any subscriber numbers (ANIs, authcodes, etc.) fail validation on a given execution of FLEXDIAL, all re-validation of subscriber numbers is turned off on all re-originations from that point forward for that AXXESS origination. See *UCS DMS-250 CAIN/FlexDial Interactions* or Appendix D, “FlexDial Interactions.”

All errors result in failure of requested table control changes.

SNPA option

The Serving Numbering Plan Area (SNPA) option identifies the numbering plan area of the originating or terminating AXXESS trunk group.

As an option for the originating trunk group, the NPA field represents the NPA of the originator or subscriber.

As an option for the terminating trunk group, the NPA field represents the connecting NPA of the destination switch or the called party.

Fields

Table 11-11 contains the SNPA option field refinement.

Table 11-11
Table TRKFEAT SNPA option field

Field	Description	Values	Default
NPA	This field identifies the serving NPA value for the trunk group.	Vector of 3 of {0 to 9}	000

Examples

Examples of the SNPA option provisioning are:

```
SNPA 214
```

In this example, the SNPA characteristic of the trunk group is set to 214.

SNPA option restrictions and limitations

A provisioned default value is not stored in the table entry. An attempt to provision the default value results in the following warning message:

```
Warning – The value selected for SNPA is the default value of
000. Therefore, the SNPA option is not stored with and
subsequently not displayed for the table entry.
```

SNXX option

The Serving NXX (SNXX) option applies to both originating and terminating agents and identifies the NXX of the trunk group. Typically, SNXX is used in conjunction with the SNPA option.

Fields

Table 11-12 contains the SNXX field refinement.

Table 11-12
Table TRKFEAT SNXX option field

Field	Description	Values	Default
NXX	This field identifies the value for the serving NXX trunk group characteristic.	Vector of 3 of {0 to 9}	000

Example

An example of the SNXX option is:

```
SNXX 684
```

In this example, the SNXX characteristic of the trunk group is set to 684.

SNXX option restrictions and limitations

A provisioned default value is not stored in the table entry. An attempt to provision the default value results in the following warning message:

```
Warning – The value selected for SNXX is the default value of
000. Therefore, the SNXX option is not stored with and
subsequently not displayed for the table entry.
```

SUSTMR option

The Suspend/Resume Timer (SUSTMR) option applies only to originating agents and identifies the suspend/resume timer value used for the call.

Fields

Table 11-13 contains the SUSTMR field refinement.

Table 11-13
Table SUSTMR option field

Field	Description	Values	Default
TIMER	This field contains the timer value for suspend/resume timing for the call.	0 to 254 in 100ms increments	160

Example

An example of the SUSTMR option is:

```
SUSTMR 100
```

In this example, the suspend/resume timer value used for the call is 10 seconds.

SUSTMR option restrictions and limitations

A provisioned default value is not stored in the table entry. An attempt to provision the default value results in the following warning message:

```
Warning – The value selected for SUSTMR is the default value
of 160. Therefore, the SUSTMR option is not stored with and
subsequently not displayed for the table entry.
```

TIMEBIAS option

The Time Bias (TIMEBIAS) option only applies to originating agents and identifies that the trunk group destination and the switch are in different time zones. This information is used for time of day restriction COS screening.

Fields

Table 11-14 contains the TIMEBIAS field refinement.

Table 11-14
Table TRKFEAT TIMEBIAS option field

Field	Description	Values	Default
OFFSET	This is the amount of time that the trunk destination differs from the switch.	–12 to 12	0

Examples

Examples of the option provisioning are:

```
TIMEBIAS –2
```

This TIMEBIAS option identifies that the time zone of the trunk group destination is lagging the current time zone by two hours.

TIMEBIAS option restrictions and limitations

A provisioned default value is not stored in the table entry. An attempt to provision the default value results in the following warning message:

```
Warning – The value selected for TIMEBIAS is the default
value of 0. Therefore, the TIMEBIAS option is not stored with
and subsequently not displayed for the table entry.
```

TRANSTS option

The TRKFEAT Table Translation Serving Translation Scheme (TRANSTS) option as applicable for originating trunk groups is identical to the definition of the option for the FLEXFEAT table.

TRKCOS option

The Trunk To Trunk COS Screening (TRKCOS) option applies to both originating and terminating agents and identifies an index for screening the originating and terminating agencies of the call in table TRKCOS.

Fields

Table 11-15 contains the TRKCOS field refinement.

Table 11-15
Table TRKFEAT TRKCOS option field

Field	Description	Values	Default
INDEX	This field contains the index into the TRKCOS table for the trunk group.	0 to 119	0

Examples

An example of the option provisioning is:

```
TRKCOS 0
```

In this example, a TRKCOS index of zero is used for TRKCOS screening.

TRKCOS option restrictions and limitations

This option does not have any identified table control restrictions.

ZONE option

The Zone (ZONE) option applies to both originating and terminating agents and identifies when to use loop-around IMTs with external echo cancellers between the originating and terminating agents of the call (see the *UCS DMS-250 NT6X50 EC Integrated Echo Canceller Application Guide*).

Fields

Table 11-16 contains the ZONE field refinement.

Table 11-16
Table TRKFEAT ZONE option field

Field	Description	Values	Default
ZONE	This field contains the zone value.	1 to 15	0

Examples

Examples of the ZONE option provisioning are:

```
ZONE 5
```

In this example, a zone value of five is assigned to the AXXESS trunk group. If the sum of the originating and terminating trunk group zone values exceeds the limit specified by the ECHO_DELAY_THRESHOLD office parameter, then an IMT loop-around route as identified by the ECHOCAN_IMT_OFRT_INDEX office parameter is used to identify the destination for the call.

Zone option restrictions and limitations

This option does not have any table control restrictions.

System requirements

The TRKFEAT table allocates store on an as needed basis for a maximum of 8192 table entries. Store is allocated dynamically per entry, based on store requirements of each option provisioned. A minimum of 23 bytes and maximum of 442 bytes can be allocated per table entry. A maximum of 3.5 Mbytes of store can be allocated for the maximum number of table entries.

Datafill order

The TRKFEAT table requires the provisioning of the FLEXDIAL table to provision certain options.

The TRKFEAT table must be datafilled with the necessary indexes before the TRKGRP table tuple can be properly provisioned.

Dump and restore

This table does not require a dump and restore from any other tables.

Restrictions and limitations

The following restrictions and limitations apply to the TRKFEAT table:

- The 16-character string key for the TRKFEAT table is limited to 8192 unique values. An attempt to provision more than 8192 entries results in the following error message:

  ```
  Table TRKFEAT is full. No more entries are possible.
  ```

- Individual feature and characteristic options cannot be duplicated for a TRKFEAT table index, with the exception of the MSGCTR option. An attempt to duplicate options results in the following error message:

  ```
  Cannot duplicate option <option>.
  ```

<Option> identifies the option that is duplicated.

- A maximum of eight MSGCTR options may be provisioned within a TRKFEAT table entry. An attempt to provision more than eight MSGCTR options within a single table entry results in the following error message:

  ```
  Maximum MSGCTR Exceeded, Only 8 allowed.
  ```

- Indexes in the TRKFEAT table may not be deleted if they are referenced by TRKGRP entries. An attempt to remove a referenced index results in the following error message:

  ```
  TUPLE REFERENCED BY ANOTHER TABLE – USE TABREF TO GET
  POTENTIAL TABLE LIST.
  ```

- Error messages related to individual TRKSIG options are displayed as necessary during table TRKGRP verification of the TRKFEAT and TRKSIG indexes specified.

All errors result in the failure of requested table control changes.

Comparison to existing trunk group provisioning

The non-FlexDial framework scheme for provisioning trunk group features and signaling information is handled by the following tables for the DAL, FGA, FGB, FGC, and FGD trunk group types:

- TRKGRP
- TRKGRP1
- TRKSGRP

Tables 12-1, 12-2, and 12-3 identify how non-FlexDial trunk group feature and signaling information provisioning maps into the FlexDial framework provisioning scheme for AXXESS agents.

TRKGRP table mapping

Table 12-1 contains the TRKGRP table to FlexDial framework mapping.

Table 12-1
TRKGRP table to FlexDial framework mapping

Old field	New table	New field	Notes
COS	TRKFEAT	TRKCOS	
DIR			This field is eliminated. All AXXESS agents are 2-way agents.
PRTNM	FLEXDIAL	ADDR digit collectable	In VALIDATE option
	TRKFEAT	MSGCTR option	ADDR message, PRTNM message type
ODSCFLTR	TRKSIG	ODSCFLTR option	
ORIGFLTR	TRKSIG	ORIGFLTR option	

—continued—

Table 12-1
TRKGRP table to FlexDial framework mapping (continued)

Old field	New table	New field	Notes
ANSWFLTR	TRKSIG	ANSWFLTR option	
TDSCFLTR	TRKSIG	TDSCFLTR option	
CONNGNPA	TRKFEAT	SNPA option	In terminating options vector
DELAYATD	TRKSIG	ATDANS option	
IRINGCHK	TRKSIG	IRINGCHK option	
TRAFCLS	TRKFEAT		This field is eliminated.
AUTHDIAL	FLEXDIAL	SUBR digit collectable	Collectable identifies MIN and MAX values
	TRKFEAT	MSGCTR option	SUBR MINMAX message
DIALTONE	FLEXDIAL	Digit collectable PROMPT option	
DIGSOUTP	TRKSIG	DIGSOUTP option	Only for the DAL outgoing group type
RETOFFHK	FLEXDIAL	SNDSIG collectable	Use SNDSIG to send an off-hook signal after origination or after digits have been collected.
	TRKSIG	RETOFFHK option	
FASTIDGT	TRKSIG	MINRTMR	
OPART	TRKFEAT	CITYCODE option	The CITYCODE option is not used to derive an STS value. Other functions of the OPART field are not included in the FlexDial framework. The CITYCODE option is used for CITYVAL subscriber number validation.
	TRKFEAT	TRANSTS option	Used to identify the STS for the call. Overwritten through use of the FLEXFEAT TRANSTS option

—continued—

Table 12-1
TRKGRP table to FlexDial framework mapping (continued)

Old field	New table	New field	Notes
RECALLDT	TRKFEAT	REORIGAL option	Provides reorigination control along with the subscriber number REORGACT and REORGTYP options
	FLEXFEAT	REORGACT and REORGTYP options	Provisioned against the subscriber number or call type, and used in conjunction with the REORIGAL TRKFEAT option.
SNPA	TRKFEAT	SNPA option	In originating options vector
TIMEBIAS	TRKFEAT	TIMEBIAS option	
VAUTHFLD	FLEXDIAL	SUBR digit collectable	Use SUBR collectable FILED option
	TRKFEAT	MSGCTR option	SUBR message, FILED message type. Also use MINMAX message type to identify total number of digits needed.
ZEROMPOS	FLEXFEAT	MSGCTR option	ADDR message, OPER message type
	TRKFEAT	MSGCTR option	ADDR message, OPER message type
ZONE	TRKFEAT	ZONE option	
BCNAME	TRKSIG	BCCOMPAT option	
AUTHFRST	FLEXDIAL		The provisioned ordering of the collectables identifies the sequence that they are processed.
ADIN			This field is eliminated due to the new FLEXVAL provisioning scheme.
AIOD AIODDN AIODII AIODGRP			The AIOD fields refer to outpulsing features, and are not supported in the provisioning scheme for the AXXESS agent.
ONNETTRK	TRKFEAT FLEXFEAT	ONNETTRK option ONNET option	

—continued—

Table 12-1
TRKGRP table to FlexDial framework mapping (continued)

Old field	New table	New field	Notes
PANIVAL			Handled by the SUBR digit collectable along with FLEXFEAT and other FlexDial provisioning.
PANIINFO			The provisioned ordering of the collectables including the INFO digit collectable with FILED digits replaces this functionality.
CPIALLOW	TRKSIG	CPIALLOW option	
OPNOAUTH			The functionality provided by this field can be duplicated through FLEXDIAL table provisioning.
VANIDB		SUBR digit collectable	
ENHANSUP			This field is not used.
TRIGINDX			This field is not used.
VARDIAL			This field is not used in the FlexDial framework.
REORGVAL	TRKFEAT	REORIGAL option	The REVERIFY field identifies a list of subscriber number types that require revalidation on reoriginated calls.
	FLEXFEAT FLEXTYPE	REVALIDATE option REVALIDATE option	
INTERVPN			This field is not supported.
UATONE	FLEXDIAL	Digit collectable prompt option—PREDISTN field.	
DETDIAL	TRKSIG	DETDIAL option	
CONDHOTL			This field is not supported in the FlexDial framework.

—continued—

Table 12-1
TRKGRP table to FlexDial framework mapping (continued)

Old field	New table	New field	Notes
ALTSEIZ	TRKSIG	ALTSEIZ option	
SCRNCL			This field is not used.
ALTTRTMT	TRKFEAT	ALTTRTMT option	
ATDANS	TRKSIG	ATDANS option	See also DELAYATD
VPNSIX			This field is not supported.
ACPROMPT	FLEXDIAL	SUBR digit collectable	PROMPT option
IEXCLINX	TRKFEAT	IEXCLINX option	
OHQTERM	TRKFEAT	OHQTERM option	
OHQ	TRKFEAT	OHQ option	
MCCS			A similar service (without TCAP validation) can be provided through FLEXTYPE provisioning and the SUBR collectable.
SNDRPSIG			This field is eliminated due to the capability of the SIG protocol collectable.
SNDRPDIL			This field is eliminated due to the capability of the SIG protocol collectable.
ACSCRNIDX	TRKFEAT	MSGCTR option	SUBR message, INDEXES message type
FEATGRP			This field is not required in the FlexDial framework.
AUTHPIN			This field is not supported.
UACPRTNM			This field is not used.
ANIDIGS			The ANIDIGS option is not required in the FlexDial framework.

—continued—

Table 12-1
TRKGRP table to FlexDial framework mapping (continued)

Old field	New table	New field	Notes
IDPRTRAN	FLEXDIAL	OLI digit collectable	Use OLI collectable VALIDATE option.
	TRKFEAT	MSGCTR option	OLI message, PRTNM message type
ISUPIDX	TRKSIG	ISUPIDX	
TMANIDLV	TRKSIG	DELIVER	
CTRUAUTH			This field is not required in the FlexDial framework.
CTRUPANI			This field not required in the FlexDial framework.
BLOCKNB			This field is not supported.
ID24_ON			This field not required in the FlexDial framework.
DIRECTEO			This field is not used.
SNCDED			This field is not supported.
CASUALU	FLEXFEAT	CASUAL option	
	FLEXDIAL	SUBR collectable	CASUBLK validate option
COSOVE	TRKFEAT	COSOVE option	
ANSWSUPR			This field is not supported for AXXESS agents. The TRKSIG RETOFFHK option provides almost identical functionality.
MEMATTR			This field is not supported.
CAINGRP	TRKFEAT	CAINGRP option	Both originating and terminating versions
CAIN	TRKFEAT	CAIN option	
CICRTE	TRKFEAT	CICRTE option	
TMCICDLV	TRKFEAT	CICDELV option	

—continued—

Table 12-1
TRKGRP table to FlexDial framework mapping

Old field	New table	New field	Notes
FWDXMT			This field is not supported.
TMTID			This field is not supported.
TSUSR	TRKFEAT	SUSTMR option	
ANIBYP	FLEXDIAL	SUBR/SUBRPARM VALIDATE option	Do not use the VALIDATE option if validation is not desired.
NETSEC	TRKFEAT FLEXTYPE FLEXFEAT	ANSCDR option ANSCDR option ANSCDR option	

—end—

TRKGRP1 table mapping

Table 12-2 contains the TRKGRP1 table to FlexDial framework mapping.

Table 12-2
TRKGRP1 table to FlexDial framework mapping

Old field	New table	New field or option	Notes
DIALTONE	FLEXDIAL	Digit collectable PROMPT option	
AUTHFRST			This field is not required in the FlexDial framework.
ACPROMPT	FLEXDIAL	Digit collectable PROMPT option	
AUTHDIAL	FLEXDIAL	SUBR digit collectable	Collectable identifies MIN and MAX values.
	TRKFEAT	MSGCTR option	SUBR MINMAX message
MLTSTAGE	TRKSIG	MLTSTAGE option	Only for the EANT outgoing group type

—continued—

Table 12-2
TRKGRP1 table to FlexDial framework mapping (continued)

Old field	New table	New field or option	Notes
IDPRTRAN	FLEXDIAL	OLI digit collectable	Use OLI collectable VALIDATE option.
	TRKFEAT	MSGCTR option	OLI message, PRTNM message type
PANICLAS			This field is not supported.
PINADDRP	FLEXDIAL	Digit collectable PROMPT option	
CN	FLEXFEAT TRKFEAT	MSGCTR option MSGCTR option	ADDR message, OPER message type ADDR message, OPER message type
OSSSIG			This field is not supported.
ANICLAS			This field is not supported.
SNXX	TRKFEAT	SNXX option	In originating options vector
SPARE1			This field is not used.
SPARE2			This field is not used.
SPARE3			This field is not used.
SPARE4			This field is not used.
VAUTHFLD	FLEXDIAL	SUBR digit collectable	Use SUBR collectable FILED option
	TRKFEAT	MSGCTR option	SUBR message, FILED message type. Also use MINMAX message type to identify total number of digits needed.

—end—

TRKSGRP table mapping

Table 12-3 contains the TRKSGRP table to FlexDial framework mapping.

Table 12-3
TRKSGRP table to FlexDial framework mapping

Old field	New table	New field or option	Notes
CARDCODE	TRKSIG	SIGTYPE	The CARDCODE and SIGDATA values are mapped to a SIGTYPE value for the TRKSIG table.
SIGDATA	TRKSIG	SIGTYPE	The CARDCODE and SIGDATA values are mapped to a SIGTYPE value for the TRKSIG table.
DIR			This field is eliminated. All AXXESS trunk groups are 2-way.
IPULSTYP	TRKSIG	IPULSTYP	DP is not supported.
ISTARTSG	TRKSIG	ISTARTSIG	Values supported are based on the SIGTYPE and IPULSTYP values.
OVLP			This field is not used.
PSPDSEIZ	TRKSIG	PSEIZTMR	
PARTDIAL	TRKSIG	PDILTMR	
OPULSTYP	TRKSIG	OPULSTYP	DP not supported.
OSTARTSG	TRKSIG	OSTARTSIG	Values supported are based on the SIGTYPE and OPULSTYP values.
IDGTIME	TRKSIG	OIDGTMR	
NUMSTOPS			This field is not used.
GLAREYD	TRKSIG	GLAREYD option	For PTS (DS1, FXS, FXO)
CCONT			This field is not supported.
RNGBCK			This field is not required.
ESUPR	TRKSIG	ESUPR option	
SAT			This field is not supported for AXXESS trunk groups.

—continued—

Table 12-3
TRKSGRP table to FlexDial framework mapping (continued)

Old field	New table	New field or option	Notes
REMBSY	TRKSIG	REMBSY option	
DIALMODE	TRKSIG	DIALMODE	
ECSTAT	TRKSIG	ECSTAT option	
TRKGRDTM	TRKSIG	TRKGRDTM	
ABCNTL	TRKSIG	ABCNTRL	
PROTOCOL	TRKSIG	PROTOCOL	
CONTCHK	TRKSIG	COT option	
COTREQ	TRKSIG	COT option	
TMRNAME	TRKSIG	TMRNAME option	
GLARETYP	TRKSIG	GLARE option	For CCS7

—end—

Office parameter provisioning

This chapter describes the office parameters defined for interaction with the FlexDial framework. For more information, see the *UCS DMS-250 Office Parameters Reference Manual.*

FlexDial office parameters

FlexDial framework includes eight office parameters (OPs):

- FLEXDIAL_ACCESS_INDEX
- FLEXDIAL_ACCESS_CONTROL
- FLEXDIAL_MIN_CALLP_ALLOC
- FLEXDIAL_AUDIT_INTERVAL
- FLEXDIAL_N00_FLEXTYPE
- FLEXDIAL_MAX_LIST_BUILT
- FLEXDIAL_MAX_LIST_EXEC
- NUM_CPRC_EXT_BLKS

FLEXDIAL_ACCESS_INDEX

Range: Valid index to FLEXDIAL table.

This office parameter in table OFCENG identifies the FLEXDIAL table index that delimits tuples that can be changed by normal table control procedures, and tuples that may not be changed unless the FLEXDIAL_ACCESS_CONTROL write protected office parameter is set. Any tuples in the table prior to this index and including this index cannot be changed unless the FLEXDIAL_ACCESS_CONTROL office parameter is set to N.

This office parameter is defaulted to the upper bound of predefined FLEXDIAL table entries.

FLEXDIAL_ACCESS_CONTROL

Range: {N, Y}

This office parameter in table OFCENG is used in conjunction with the FLEXDIAL_ACCESS_INDEX office parameter. If it is set to Y, then the table control routines for the FLEXDIAL table are not allowed to modify tuples listed before and including the index identified by the FLEXDIAL_ACCESS_INDEX office parameter. If this office parameter is set to N, then all table tuples can be edited with the exception of the NIL tuple.

The default value is N.

FLEXDIAL_MIN_CALLP_ALLOC

Range: {0 to 100}

This office parameter resides in table OFCENG and identifies the minimum amount of FlexDial call processing system resources as a percentage of the defined maximum value. These resources are allocated for FlexDial call processing use. This office parameter must be set to a value greater than zero in order to perform FlexDial framework calls.

FlexDial framework call processing shared pool resources are allocated in 64K byte blocks, and the number of pooled elements is based upon the following equation:

Elements = (FLEXDIAL_MIN_CALLP_ALLOC/100) * MaxRatioValue * NCCBS

NCCBS represents the NCCBS office parameter.

Additionally, the number of pooled elements is rounded up to completely fill all allocated 64K byte blocks.

Increasing the value of the FLEXDIAL_MIN_CALLP_ALLOC office parameter may immediately increase shared pool resource sizes and additional memory resources are immediately allocated. Decreasing the value of the office parameter decreases pool resource sizes, however a cold restart is required in order to release memory resources.

Note that increasing the office parameter to a value greater than zero allocates a minimum of one 64K byte block of store for each shared pool resource defined.

See Chapter 23, “System Requirements,” for more information on the maximum ratio values defined for FlexDial framework call processing system shared pool resources.

This office parameter is defaulted to a value of zero.

FLEXDIAL_MSGCTR_MULT

This office parameter is no longer needed, since MSGCTR-related allocated elements now utilize the shared pool resource mechanisms.

FLEXDIAL_AUDIT_INTERVAL

Range: {0 to 960} in 15 minute increments

This office parameter resides in table OFCENG and identifies the time interval for executing the the FLEXAUDP call processing resource audit. The purpose of the audit is to verify the integrity of FlexDial call processing resource pools, and expand pool sizes as needed. The value of the office parameter identifies the number of 15-minute intervals between execution of each audit (a value of four indicates a one-hour audit interval).

The audit process is performed by routine FLEXAUDP, and may be queried using the QUERY command (for example, > query process flexaudp) on the switch.

Modifying the value of the office parameter causes immediate execution of the audit process, cancelling the current timer interval and enacting the new timer value upon completion of the audit.

This office parameter is defaulted to a value of 4.

FLEXDIAL_N00_FLEXTYPE

Range: Valid index to table FLEXTYPE.

This office parameter in table OFCVAR identifies the FLEXTYPE table index used in a SUBR MSGCTR message that is formatted containing an N00 or access number (typically 800, 888, or 950 numbers). This office parameter is used during ADDR or ADDRPARM collectable processing when processing a UAX STDPRTCT table selector or when processing a TCAP response message for the N00 application.

The default value is the first entry in table FLEXTYPE.

FLEXDIAL_MAX_LIST_BUILT

Range: {0 to 32,767}

This office parameter in table OFCENG limits the number of lists that can be built during processing of a single Flexdial action. This office parameter is designed to protect against erroneous recursive datafill. A Flexdial action may consist of either of the following:

- the action taken during an origination (the initial list is built from provisioning in table TRKGRM)
- the action taken when a Flexdial action is applied within collectable processing. (for example, an INSERT, APPEND, REPLACE, or EXEC Flexdial action that can be datafilled to occur as a response to a validation failure or an ”IF” collectable)

The FNAL treatment is set and the FLEX 306 log is generated whenever this limit is exceeded.

The default value is 100.

FLEXDIAL_MAX_LIST_EXEC

Range: {0 to 32,767}

This office parameter in table OFCENG limits the number of collectables executed on a list during the dialplan execution. This prevents infinite lists during execution.

The FNAL treatment is set and the FLEX 306 log is generated whenever this limit is exceeded.

The default value is 2000.

NUM_CPRC_EXT_BLKS

Range: {0 to 65535}

This office parameter resides in table OFCENG and identifies the number of FlexDial Call Processing Resource Class (CPRC) extension blocks that are available for Axxess agent calls. The CPRC is like an Overflow Call Condense Block (OCCB) extension block used for all UCS DMS-250 switch call processing but exists and is allocated solely for the support of Axxess agent calls. The number of CPRCs allocated must be equal to or greater than the maximum number of call processes that may be active involving an Axxess agent as either the originator or terminator of the call. This number must be greater than zero in order to perform Axxess agent call processing.

The CPRC follows general UCS DMS-250 switch rules regarding the provisioning, allocation, deallocation and auditing of extension block structures. Increasing the size of this office parameter immediately allocates more memory for the additionally allocated extension blocks. Decreasing the size of this office parameter reduces the number of CPRC extension blocks available, however a restart cold or reload is required in order to release memory resources.

The number of CPRC extension blocks is also used to control allocation of FlexDial call processing shared pool resources in conjunction with the FLEXDIAL_MIN_CALLP_ALLOC office parameter.

FlexDial framework call processing shared pool resources are allocated in 64K byte blocks, and the number of pooled elements is based upon the following equation:

Elements = (FLEXDIAL_MIN_CALLP_ALLOC/100) ∗ *MaxRatioValue* * NUM_CPRC_EXT_BLKS

Additionally, the number of pooled elements is rounded up to completely fill all allocated 64K byte blocks.

Increasing the value of the office parameter may immediately increase shared pool resource sizes and additional memory resources are immediately allocated. Decreasing the value of the office parameter decreases pool resource sizes, however a restart cold or reload is required in order to release memory resources.

Note that increasing the office parameter to a value greater than zero allocates a minimum of one 64K byte block of store for each shared pool resource defined.

See chapter “System Requirements” for more information on the maximum ratio values defined for FlexDial framework call processing system shared pool resources.

This office parameter is defaulted to a value of zero (0).

Note: The office parameter NUMPERMEXT represents available portperm extension blocks for call processing use, and is not specifically modified for FlexDial use. However, for FlexDial application of announcement prompts or announcement notifications, a portperm extension block is allocated to store announcement member information while the announcement member is connected to the call. The portperm extension block is deallocated when the announcement member is detached from the call.

FEATBYTE office parameters

This section describes the office parameters defined for interactions between the FlexDial framework and TCAP. For more information, see Appendix D, ”FlexDial interactions.”

FEATBYTE includes the following six office parameters:

- FEATBYTE_FIRST_INDEX

- FEATBYTE_FIRST_BYTE
- FEATBYTE_FIRST_MASK
- FEATBYTE_NOANSDUR_MULT
- FEATBYTE_NOANSDUR_OFFSET
- VER_2_BILL_FLEXTYPE

FEATBYTE_FIRST_INDEX

Range: {0 to 8191}

This office parameter resides in table OFCVAR and identifies the unique numeric index value used for the initial lookup of the first feature byte value in the FEATBYTE table.

The default value is zero (0).

FEATBYTE_FIRST_BYTE

Range: {is 1 to 8}

This office parameter resides in table OFCVAR and identifies the first feature byte (of the eight available) that is used for the initial lookup in the FEATBYTE table.

The default value is one (1).

FEATBYTE_FIRST_MASK

Range: {0 to 255}

This office parameter resides in table OFCVAR and identifies the mask applied to the first feature byte value. The resulting value is used in the initial lookup in the FEATBYTE table.

The default value is 255.

FEATBYTE_NOANSDUR_MULT

Range: {0 to 20}

This office parameter resides in table OFCVAR and identifies the multiplier to use in the equation ”offset + (featbyte_value * multiplier).” The value derived from the equation is used as the No Answer Duration timer.

The default value is one (1).

FEATBYTE_NOANSDUR_OFFSET

Range: {0 to 20}

This office parameter resides in table OFCVAR and identifies the offset to use in the equation ”offset + featbyte_value * multiplier).” The value derived from the equation is used as the No Answer Duration timer.

The default value is zero (0).

VER_2_BILL_FLEXTYPE

Range is 16 character string (any valid FLEXTYPE index).

This office parameter resides in table OFCVAR and identifies an index into the table FLEXTYPE. The BILLFLD option, within the FLEXTYPE tuple, identifies the field in the CDR that stores the billing number.

The billing number is not utilized from the TCAP response message when the tuple identified by this office parameter does not contain the BILLFLD option.

The default value is ANI.

QFLEXVAL, QTRK, and QSUBR CI commands

This chapter includes information on the QFLEXVAL command interpreter (CI) increment that provides an additional level of manipulation capability for FLEXVAL table. It also includes information on the QTRK CI command (to display provisioning information for an AXXESS trunk group), and the QSUBR CI command (to display provisioning information for a particular subscriber).

QFLEXVAL CI increment

The QFLEXVAL CI increment is used in conjunction with table FLEXVAL and provides an additional level of manipulation capability for the table. The command provides the following functionality:

- the ability to list FLEXVAL table entries according to a range of one or more of the three index values
- the ability to copy digit entries from one index and subscriber type to another
- the ability to delete all entries against a particular numeric index or combination of numeric index and subscriber number type
- the ability to change a set of FLEXVAL entries to contain the same FEATIDX field value. The set is a range of one or more of the three index values
- the ability to search entries for a particular FLEXFEAT index value

QFLEXVAL consists of a CI increment, which provides a subdirectory of commands. For QFLEXVAL, these commands are:

- DUMP (or DP)
- COPY (or CP)
- CLEAR (or CLR)
- SEARCH (or SCH)
- FEATCHA (or FCHA)
- QHELP (or QHLP)

- QUIT (or QT)

DUMP command

The DUMP (or DP) command displays a number of FLEXVAL table entries matching certain criteria.

```
DUMP
   [ FLEXTYPE <from> {string} [ <to> {string} ] ]
   [ IDX <from> { 0 to 1047999 } [ <to> { 0 to 1047999 } ] ]
   [ SNUM <from> { digit string } [ <to> { digit string } ] ]
```

For a successful DUMP operation, the following condition must be met:

- The FLEXTYPE names specified must exist in the FLEXTYPE table. If an invalid FLEXTYPE name is input, the operation fails and the following error message is displayed:

  ```
  Error – Invalid FLEXTYPE entry : <string>
  ```

  <string> identifies the invalid FLEXTYPE name input.

The DUMP command displays all matching FLEXVAL table entries. If no entries match the DUMP criteria, no entries are displayed. The DUMP operation displays the following information:

```
QFLEXVAL:
> dump flextype ani idx 1 snum 214 215

ANI 1   2145555453          10
ANI 1   2145555511          11
ANI 1   2145557467          12

QFLEXVAL:
>
```

COPY command

The COPY (or CP) command copies FLEXVAL table entries from one subscriber number type or numeric index or both to another.

```
COPY
   <from flextype> {string} [ <index> { 0 to 1047999 } ]
   <to flextype> {string} [ <index> { 0 to 1047999 } ]
```

For a successful copy operation, the following conditions must be met:

- The FLEXTYPE names must exist in the FLEXTYPE table. If an invalid FLEXTYPE name is input, the operation fails and the following error message is displayed:

  ```
  Error – Invalid FLEXTYPE entry : <string>
  ```

<string> identifies the invalid FLEXTYPE name input.

- The “from” index must identify at least one valid entry in the FLEXVAL table. If an entry does not exist, the operation fails and the following error message is displayed:

  ```
  Error – Invalid From FLEXVAL table index.
  ```

- The “to” index must not identify any valid entries in the FLEXVAL table. If the index is not empty, the operation fails and the following error message is displayed:

  ```
  Error – Invalid To FLEXVAL table index. Table index is not
  empty.
  ```

If the “from” numeric index value is specified, but the “to” numeric index is not specified, then the “to” numeric index is assumed equal to the “from” numeric index.

If the “from” numeric index value is not specified but the “to” numeric index is specified, then all entries against the “from” FLEXTYPE are copied to the “to” FLEXTYPE index for the specific numeric index. Duplicate digit entries are not copied.

If all input conditions are met successfully, then the copy operation provides the following output to the user:

```
QFLEXVAL:
> copy ani 1 ani 2
Copying FLEXVAL table tuples with keys matching:
   <Flextype Index> :    ANI 1
   To <Flextype Index> :    ANI 2
5000 entries copied. COPY operation complete.
```

CLEAR command

The CLEAR (or CLR) command removes all FLEXVAL table entries for a subscriber number type and/or numeric index.

CLEAR
[<flextype> {string}]
<index> { 0 to 1047999 }
[<prompt> { N, Y }]

For a successful clear operation, the following conditions must be met:

- The FLEXTYPE name must exist in the FLEXTYPE table. If an invalid FLEXTYPE name is input, the operation fails and the following error message is displayed:

  ```
  Error – Invalid FLEXTYPE entry : <string>
  ```

  <string> identifies the invalid FLEXTYPE name input.

- The overall index (FLEXTYPE and numeric index or just numeric index) must identify at least one valid entry in the FLEXVAL table. If an entry does not exist, the operation fails and the following error message is displayed:

  ```
  Error – Invalid FLEXVAL table index.
  ```

If the input conditions are met, then the clear operation continues.

If the PROMPT option is set to Y or not present, then the user is prompted for confirmation of the clear operation.

```
QFLEXVAL:
> clear ani 1
Removing all entries with key matching:
   The FLEXTYPE: ANI
   The Index : 1
Do you want to proceed ?
Please confirm : (“YES”, “Y”, “NO”, or “N”):
> Y

25 Entries removed. CLEAR operation complete.

QFLEXVAL:
>
```

Alternatively, if the user enters N to the prompt:

```
QFLEXVAL:
> clear 1
Removing all entries with key matching:
   The Index : 1
Do you want to proceed ?
Please confirm : (“YES”, “Y”, “NO”, or “N”):
> Y

CLEAR operation aborted.

QFLEXVAL:
>
```

Or if the PROMPT option is set to N, then the output becomes:

```
QFLEXVAL:
> clear ani 1 n
Removing all entries with key matching:
   The FLEXTYPE: ANI
   The Index : 1
55 Entries removed. CLEAR operation complete.

QFLEXVAL:
```

SEARCH command

The SEARCH (or SCH) command finds FLEXVAL table entries with the provisioned FLEXFEAT table index value.

QFLEXVAL SEARCH
<featidx> { 0 to 16777216}

In order for the search operation to be performed successfully, the following condition must be met:

- The featidx value specified must exist in the FLEXFEAT table. If the index is not provisioned in the table, the operation fails and the following error message is displayed:

  ```
  Error – Invalid FLEXFEAT entry : <index>
  ```

  <index> identifies the invalid FLEXFEAT index entered.

If the input conditions are met, then the search operation continues.

```
QFLEXVAL:
> search 11
Searching for entries matching FLEXFEAT index : 11

ANI     1   2145555511          11
ANI     2   2145555511          11
AUTH    1   5555511             11

SEARCH operation complete.

QFLEXVAL:
>
```

FEATCHA command

The FEATCHA (or FCHA) command modifies the FLEXFEAT index value for all the identified FLEXVAL table entries.

```
QFLEXVAL FEATCHA
   [ FLEXTYPE <from> {string} [ <to> {string} ] ]
   [ IDX <from> { 0 to 1047999 } [ <to> { 0 to 1047999 } ] ]
   [ SNUM <from> { digit string } [ <to> { digit string } ] ]
   FIDX <featidx> { 0 to 16777216}
```

For a successful change FLEXFEAT index operation, the following conditions must be met:

- The FLEXTYPE names specified must exist in the FLEXTYPE table. If an invalid FLEXTYPE name is input, the operation fails and the following error message is displayed:

  ```
  Error – Invalid FLEXTYPE entry : <string>
  ```

  <string> identifies the invalid FLEXTYPE name input.

- The index criteria must identify at least one valid entry in the FLEXVAL table. If an entry does not exist, the operation fails, and the following error message is displayed:

  ```
  Error – Invalid FLEXVAL table index.
  ```

- The featidx value specified must exist in the FLEXFEAT table. If the index is not provisioned in the table, the operation fails and the following error message is displayed:

  ```
  Error – Invalid FLEXFEAT index : <index>
  ```

  <index> identifies the invalid FLEXFEAT index entered.

If the input conditions are successfully met, then the change feature index operation provides the following output to the user:

```
QFLEXVAL:
> featcha flextype ani idx 1 2 snum 2145550001 ani 2
   2145559999 fidx 25
WARNING You are going to change the FEATIDX of a set of
    tuples to 25 with key matching:

    The Flextype: from ANI to ANI.
    The Index: from 1 to 2
    The Subscriber Number: from 2145550001 to 2145559999

Do you want to proceed ?
Please confirm (“YES”, “Y”, “NO”, or “N”):
```

```
> y

5000 entries modified. FEATCHA operation complete.

QFLEXVAL:
>
```

Alternatively, if the user enters N to the prompt:

```
QFLEXVAL:
> featcha flextype ani idx 1 2 snum 2145550001 ani 2
   2145559999 fidx 25
WARNING: You are going to change the FEATIDX of a set of
    tuples to 25 with key matching:

    The Flextype: from ANI to ANI.
    The Index: from 1 to 2
    The Subscriber Number: from 2145550001 to 2145559999

Do you want to proceed ?
Please confirm (“YES”, “Y”, “NO”, or “N”):

> n

FEATCHA operation aborted.

QFLEXVAL:
>
```

QHELP command

The QHELP (or QHELP) command provides a functional description of each QFLEXVAL command.

QHELP
[<topic> { dump, copy, clear, search, featcha }]

Without specifying a topic, the QHELP command displays the following information:

```
QFLEXVAL:
> qhelp
   QFLEXVAL TOOL is the table edit utility that is only
applicable
for table FLEXVAL.

The available subcommands include:

DUMP (or DP) –– The DUMP command provides the ability to
```

```
    list a range of FLEXVAL table entries based on the
    identified parameter criteria. This criteria
    can be by either FLEXTYPE index, numeric index,
    digit entries, or combinations of the three
    indexes.

COPY (or CP) –– The COPY command provides the ability to
   copy FLEXVAL table entries from one identified index
   to another. The “from” index must have valid table
   entries while the “to” index must not identify any
   FLEXVAL table entries.

CLEAR (or CLR) –– The CLEAR command provides the ability
   to delete a range of FLEXVAL table entries. By default
   the user is prompted to confirm the request to remove
   the table entries.

SEARCH (or SCH) –– The SEARCH command provides the ability
    to list all FLEXVAL table entries which provision the
    specified FLEXFEAT table index.

FEATCHA (or FCHA) –– The FEATCHA (FLEXFEAT index change)
        command provides the ability to modify an entire
        range of FLEXVAL table indexes to use the
        identified FLEXFEAT index.

QHELP (or QHLP) –– The QHELP command provides a brief
        synopsis of the command.

QUIT (or QT) –– quit QFLEXVAL.
```

Each QHELP subcommand briefly describes the sub-command.

```
QFLEXVAL:
> qhelp dump

DUMP (or DP)    –– The DUMP command provides the ability
    to list a range of FLEXVAL table entries based on the
    identified parameter criteria. This criteria
    can be by either FLEXTYPE index, numeric index,
    digit entries, or combinations of the three
    indexes.

Parms: [<Key_Part>... {FLEXTYPE <from> STRING
                [<to> STRING],
            IDX <from> {0 TO 1047999}
                [<to> {0 TO 1047999}],
            SNUM <from> STRING
                [<to> STRING]}]
```

```
QFLEXVAL:
> qhelp copy

COPY (or CP) –– The COPY command provides the ability to
    copy FLEXVAL table entries from one identified index
    to another. The “from” index must have valid table
    entries while the “to” index must not identify any
    FLEXVAL table entries.

Example:

The contents of table FLEXVAL:
----------------------------------------------------------
   Flextype   Numeric_Idx Subr_Num     FEATIDX
   XXX                  20         123                40

   XXX                  30         456         50
----------------------------------------------------------
1). If the user enters:

   COPY XXX 20 YYY 40

The contents of table FLEXVAL will be:
----------------------------------------------------------
   Flextype   Numeric_Idx Subr_Num     FEATIDX
   XXX       20         123             40
   XXX       30         456             50
   YYY       40         123             40
----------------------------------------------------------
2). Now, if the user enters:

   COPY XXX ZZZ

The contents of table FLEXVAL will be:
----------------------------------------------------------
   Flextype   Numeric_Idx Subr_Num     FEATIDX
   XXX       20         123             40
   XXX       30         456             50
   YYY       40         123             40
   ZZZ       20         123             40
   ZZZ       30         456             50
----------------------------------------------------------

Parms:  <from_flextype> STRING
   [<from_index { 0 to 1047999 }]
   <to_flextype> STRING
   [<to_index> { 0 to 1047999 }]
```

```
QFLEXVAL:
> qhelp clear

CLEAR (or CLR) -- The CLEAR command provides the ability
    to delete a range of FLEXVAL table entries. By default
    the user is prompted to confirm the request to remove
    the table entries.

Examples:

    1).   CLEAR XXX 30

    2).   CLEAR 20

Parms:  [<flextype> STRING]
        <index> {0 TO 1047999}
        [<prompt> {Y,
                    N}]

QFLEXVAL:
> qhelp search

SEARCH (or SCH) -- The SEARCH command provides the ability
        to list all FLEXVAL table entries which provision
        the specified FLEXFEAT table index.

Parms: <featidx> {1 TO 16777216}

QFLEXVAL:
> qhelp featcha

FEATCHA (or FCHA) -- The FEATCHA (FLEXFEAT index change)
        command provides the ability to modify an entire
        range of FLEXVAL table indexes to use the
        identified FLEXFEAT index.

Parms: [<Key_Part>... {FLEXTYPE <from> STRING
                [<to> STRING],
            IDX <from> {0 TO 1047999}
                [<to> {0 TO 1047999}],
            SNUM <from> STRING
                [<to> STRING]}]
   <Featidx> {FIDX <featidx> {1 TO 16777216}}
```

QUIT command

The QUIT (or QT) command exits the QFLEXVAL CI increment.

QUIT

QTRK CI command

The QTRK CI command is used to quickly display all provisioning information for an AXXESS trunk group. In a simple display format, the associated datafill from the following tables is displayed for the AXXESS trunk group CLLI specified.

- TRKGRP
- TRKSIG
- ISUPDEST (only displayed for CCS7 signaling agents)
- TRKFEAT
- MSGCTR (all indexes referenced in the TRKFEAT tuple are displayed)
- FLEXDIAL (all indexes used in the TRKGRP DPIDX field are displayed)
- TRKMEM (optionally all members are displayed).
- C7TRKMEM (only displayed for CCS7 signaling agents in combination with TRKMEM display).

The syntax of the QTRK command involves receipt of the AXXESS trunk group CLLI and QTRK, as follows:

```
qtrk
         <CLLI> {string}
         [<Display TRKMEM Data> { True, False }]
```

The “Display TRKMEM Data” parameter is optional, and defaults to True.

If an undefined CLLI is entered, the following error message is displayed:

```
CI:
> qtrk notdefinedclli
CLLI: NOTDEFINEDCLLI is not valid.
```

If the CLLI entered is not a trunk group CLLI, then no output is provided.

```
CI:
> qtrk nottrunkclli
>
```

If the CLLI entered is not an AXXESS trunk group, then the following error message is displayed:

```
CI:
> qtrk notAXXESSclli
CLLI: NOTAXXESSCLLI is not an AXXESS Trunk.

 Only AXXESS Trunk information is displayed.
>
```

For a valid PTS AXXESS agent, the following is an example output of the QTRK command.

```
   CI:
   > qtrk ptsAXXESSclli
   AXXESS Trunk Group Display for CLLI: PTSAXXESSCLLI

   TRKGRP:
    PTSAXXESSCLLI AXXESS 127 NPDGP NCON MIDL
    TRKSIG_0001 TRKFEAT_0001
    (I_OLI_ANI_V) (ITC_AU_CD_AD) $ EANT

   TRKSIG Index: TRKSIG_0001
    TRKSIG_0001 DS1 WK MF 5 5 5 KP $ ALL M WK MF 6 70
    (ORIGFLTR 7) (ANSWFLTR 16) $

   TRKFEAT Index: TRKFEAT_0001
    TRKFEAT_0001 (CITYCODE 111) (OHQ ) (SNPA 214)
    (TRANSTS PARTITION 111 0) (MSGCTR 2016) $ (OHQTERM )
    (SNPA 214) $

   MSGCTR Indexes: 2001
            2001 (ADDR N Y PRTNM AXX) (ADDR N Y OPER
NORMAL OFRT
    1 AXX OFRT 2) (SUBR UAX N N FILED DIGS 05 PREFIX)
    (OLI N Y PRTNM EAPT) $

   FLEXDIAL Indexes:    I_OLI_ANI_V     ITC_AU_CD_AD
    I_OLI_ANI_V (COLDIG 0 15 CALLING $ $) (DO 1NX_0ZZ_CHK)
        (DO OA_OCHK_V) $ N
    ITC_AU_CD_AD (IFNOA CALLED IS ( 950_CT) (NO_NUM_CT) $
        TCT_AU_CD_AD)$ Y I_MF_ADDR

   TRKMEM Member List: 1 trunk member
    PTSAXXESSCLLI 1 0 DTC 2 7 14
>
```

For a CCS7 AXXESS agent, the following is an example output of the QTRK command.

```
CI:
> qtrk ss7AXXESSclli
AXXESS Trunk Group Display for CLLI: SS7AXXESSCLLI
```

```
TRKGRP:
 SS7AXXESSCLLI AXXESS 50 NPDGP NCOF MIDL
 SS7UCP_0001 TRKFEAT_1010
 (S7_OLI_ANI_VCB) (S7_AU_SD_AD) $ EANT

TRKSIG Index: SS7UCP_0001
 SS7UCP_0001 CCS7 UCP DMSNODE UCS2EAEO (ABCNTRL )
 (GLARE CIC) (ESUPR N) (COT 100 THRH) $

ISUPDEST Index: SS7AXXESSCLLI 0
 SS7AXXESSCLLI 0 C7RTESET1

TRKFEAT Index: TRKFEAT_1010
 TRKFEAT_1010 (CITYCODE 111) (OHQ ) (REORIGAL ( ANI) $)
 (SNPA 214) (TRANSTS PARTITION 111 0) (MSGCTR 2015) $
 (SNPA 214) $

MSGCTR Indexes: 2015
 2015 (ADDR N Y PRTNM AXX) (ADDR N Y OPER NORMAL OFRT
 50 AXX OFRT 50) (SUBR UAX N N FILED DIGS 05 PREFIX)
 (OLI N Y PRTNM EAPT) $

FLEXDIAL Indexes:    S7_OLI_ANI_VCB      S7_AU_SD_AD
 S7_OLI_ANI_VCB (DO S7_TNS_CHK) (DO S7_OA_OCHK_VCB)
     $ N
 S7_AU_SD_AD (IFNOA CALLED IS ( 950_CT) (NO_NUM_CT) $
     TCT_AU_SD_AD)$ Y I_S7_ADDR

TRKMEM Member List: 1 trunk member
 SS7AXXESSCLLI 0 0 DTC 3 17 7

C7TRKMEM Member List: 1 trunk member
 SS7AXXESSCLLI 0 322
```

QSUBR CI command

The QSUBR CI command is used to display all provisioning information for a specific subscriber number. In a simple display format, the associated datafill from the following tables is displayed for the subscriber number specified:

- FLEXTYPE
- FLEXVAL
- FLEXFEAT
- MSGCTR
- FLEXDIAL

The syntax of the QSUBR command involves receipt of the key to table FLEXVAL to display the desired provisioned information. This key consists of three parts:

1 the subscriber number type (the FLEXTYPE)

2 the numeric index

3 the subscriber number digits

The following is an example of using the QSUBR CI command:

```
> q qsubr
   Parms: <FLEXTYPE IDX> STRING
          <UNIQUE ID> {0 TO 1047999}
          <SUBSCRIBER NUM> STRING
```

If the identified FLEXTYPE is invalid, the following message is output:

```
> qsubr bob 1 1234
   Error – FLEXTYPE BOB is invalid.
```

IF the numeric index is out of range, you are reprompted to enter the correct value:

```
> qsubr ani 2334984 1234
   Out of range: <UNIQUE ID> {0 TO 1047999}
   Enter: <UNIQUE ID> <SUBSCRIBER NUM>

> abort
>
```

If the entry does not exist, nothing is displayed:

```
> qsubr ani 1 2145551212
   Wrong type: <UNIQUE ID> {0 TO 1047999}
   Enter: <UNIQUE ID> <SUBSCRIBER NUM>
```

An example of using QSUBR for one field at a time:

```
> qsubr
   Next par is: <FLEXTYPE IDX> STRING
   Enter: <FLEXTYPE IDX> <UNIQUE ID> <SUBSCRIBER NUM>

> ani
   Next par is: <UNIQUE ID> {0 TO 1047999}
   Enter: <UNIQUE ID> <SUBSCRIBER NUM>
> 1
   Next par is: <SUBSCRIBER NUM> STRING
   Enter: <SUBSCRIBER NUM>
> 214
```

```
 FLEXTYPE:
    ANI (BILLFLD ANISP REPLACE) (CALLING ) (OPERDISP CLG
REPLACE) $
Wrong type: <UNIQUE ID> {0 TO 1047999}

 FLEXVAL:
    ANI 1 214 200090

 FLEXFEAT:
    200090 (CITYCODE 111) (ONNET ) (CITYVAL TRMT Y Y RODR)
$
```

An example, showing all possible datafill displayed:

> qsubr auth 1 6112214

```
 FLEXTYPE:
    AUTH (BILLFLD BILLNUM REPLACE) (FLEXLOG ) (OPERDISP
SPL REPLACE)
         (CAINFLG AUTH) $
 FLEXVAL:
    AUTH 1 6112214 110114
 FLEXFEAT:
    110114 (DPIDX AC_NOP APPEND) (REORGACT
SIG_AD_CD_AC_H50 10014 15)
         (TRANSTS PARTITION 111 0) (PVSPDIDX 52) (CITYCODE
111)
         (REORGTYP ONKEY STR P 15 5) (MSGCTR 10014) $
 MSGCTR Indices: 10014
    10014 (SUBR ACCT N N MINMAX 3 3) $
 FLEXDIAL Index: AC_NOP
    AC_NOP (IFNOA IS CALLED ( NO_NUM_OP) (SUBR_OP)
(NATL_OP) (INTL_OP) $ REPLACE
         NOOP NIL)$ Y AC
```

Another example:

> qsubr ani 1 2146112215

```
 FLEXTYPE:
    ANI (BILLFLD ANISP REPLACE) (CALLING ) (OPERDISP CLG
REPLACE) $
 FLEXVAL:
    ANI 1 2146112215 200415
 FLEXFEAT:
    200415 (DPIDX ACV_SD_NOP APPEND) (TRANSTS PARTITION
111 0) (DELIVER ALWAYS)
         (MSGCTR 20015) $
 MSGCTR Indices: 20015
    20015 (SUBR ACCT N N MINMAX 3 3)
         (SUBR ACCT N N INDEXES ( 1) $)$
 FLEXDIAL Index: ACV_SD_NOP
```

```
       ACV_SD_NOP (IFNOA IS CALLED ( NO_NUM_OP) (SUBR_OP)
   (NATL_OP) (INTL_OP) $
            REPLACE NOOP NIL)$ Y ACV_SD
```

If the FLEXTYPE, UNIQUE_ID, and subscriber number digits do not index table FLEXVAL, only table FLEXTYPE provisioning is output from the command.

```
>  qsubr ani 1 2146848801
      FLEXTYPE:
               ANI (BILLFLD ANISP REPLACE) (CALLING )
               (OPERDISP CLG
           REPLACE) (CAINFLG CLGPTYADD) $
```

For a valid FLEXVAL table index, all the identified provisioning is displayed. Provisioning for tables MSGCTR and FLEXDIAL is optionally displayed and is dependent upon the use of the MSGCTR and DPIDX options in the FLEXFEAT tuple respectively.

```
>  qsubr ani 1 2146112216
      FLEXTYPE:
               ANI (BILLFLD ANISP REPLACE) (CALLING )
                (OPERDISP CLG REPLACE) (CAINFLG CLGPTYADD) $
      FLEXVAL:
               ANI 1 2146112216 201616

      FLEXFEAT:
               201616 (DPIDX ACV_SD_NOP APPEND)
               (REORGACT SIG_AD_CD_ACV 10016 15)
                (TRANSTS PARTITION 111 0) (DELIVER ALWAYS)
                (REORGTYP ONKEY STR P 15 5) (MSGCTR 10016) $

      MSGCTR Indices: 10016
               10016 (SUBR ACCT N N MINMAX 5 5)
               (SUBR ACCT N N INDEXES ( 2) $)$

       FLEXDIAL Index: ACV_SD_NOP
           ACV_SD_NOP (IFNOA IS CALLED ( NO_NUM_OP) (SUBR_OP)
           (NATL_OP) (INTL_OP) $ REPLACE NOOP NIL)$ Y ACV_SD
```

If the entered FLEXTYPE index is not valid, then the following message is displayed:

```
Error – FLEXTYPE <string>is invalid.
```

Where <string> represents the string of characters entered for the FLEXTYPE index.

If the FLEXTYPE, UNIQUE_ID, and subscriber number digits do not index table FLEXVAL, then only table FLEXTYPE provisioning is output from the command.

FlexDial call processing applications overview

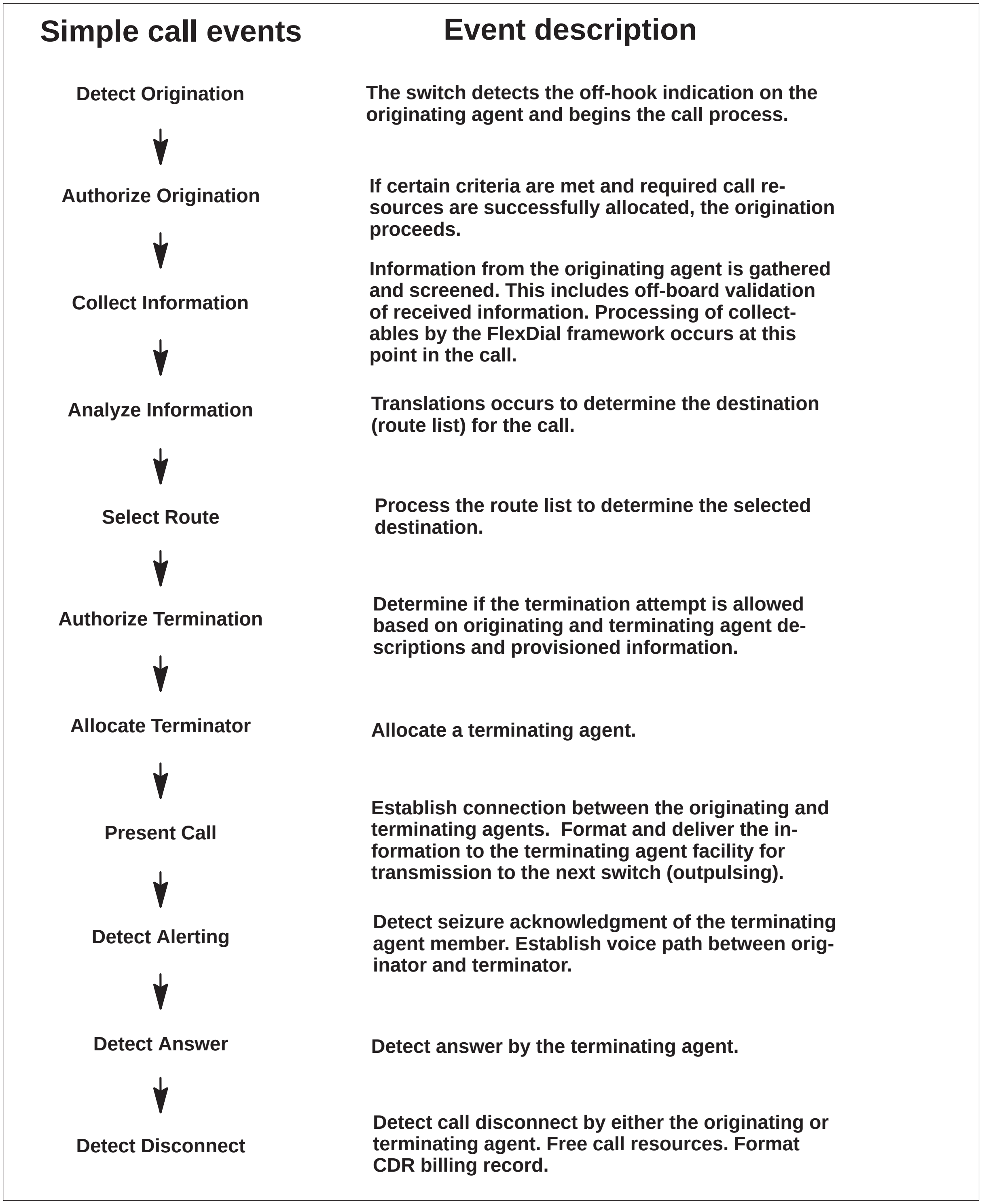
This chapter describes an overview of the FlexDial call processing applications.

The call processing application of the FlexDial framework begins with the provisioning data models outlined by tables FLEXDIAL, FLEXTYPE, FLEXFEAT, and table TRKFEAT. For the FlexDial framework, what is provisioned in table FLEXDIAL is executed by call processing.

The execution characteristics of the applications within the framework depend on the application itself (the instruction or collectable), and the environment it is executing within. This environment identifies characteristics such as the type of originating agent that is being used, and the activation of other interworking features of the system.

The application of features within the switch revolve around the execution of the basic call model in Figure 15-1.

Figure 15-1
Basic call model



Simple call events	Event description
Detect Origination	The switch detects the off-hook indication on the originating agent and begins the call process.
Authorize Origination	If certain criteria are met and required call resources are successfully allocated, the origination proceeds.
Collect Information	Information from the originating agent is gathered and screened. This includes off-board validation of received information. Processing of collectables by the FlexDial framework occurs at this point in the call.
Analyze Information	Translations occurs to determine the destination (route list) for the call.
Select Route	Process the route list to determine the selected destination.
Authorize Termination	Determine if the termination attempt is allowed based on originating and terminating agent descriptions and provisioned information.
Allocate Terminator	Allocate a terminating agent.
Present Call	Establish connection between the originating and terminating agents. Format and deliver the information to the terminating agent facility for transmission to the next switch (outpulsing).
Detect Alerting	Detect seizure acknowledgment of the terminating agent member. Establish voice path between originator and terminator.
Detect Answer	Detect answer by the terminating agent.
Detect Disconnect	Detect call disconnect by either the originating or terminating agent. Free call resources. Format CDR billing record.

Under more complicated call processing scenarios, call events may be entered again to properly handle the situation (such as returning to “Select Route” in order to route advance).

The FLEXDIAL table provides the mechanism for defining all of the interactions with the originating agent required to set up the call. This table defines instructions, termed collectables, which are executed in order to complete this interaction.

Activation

The instructions (collectables) provisioned through the FLEXDIAL table are only applicable when using the AXXESS trunk group type. The trunk group table (TRKGRP) definition for the AXXESS agent identifies the initial interaction with the originating agent through the DPIDX field. This field identifies up to four FLEXDIAL table indexes containing a list of provisioned collectables to execute. This list of collectables identifies the required interaction for members of the AXXESS trunk group.

Note: The trunk group defines the initial interaction. The interaction can be altered during processing.

The execution of collectables only occurs at the collect information point in the call model. Normally this occurs right after origination has been detected and authorized to continue, but the collect information point in the call may be entered again under other circumstances.

When the collect information point in the call is entered again, collectable processing begins with the next unprocessed collectable. If there are no more collectables to process for the call, the collect information point in the call model is exited and a FLEX 301 trouble log is generated with a trouble code value of:

```
No additional collectables to process
```

The FLEX 301 log does not include a FLEXTYPE index, a FLEXDIAL index, or the received digits.

AXXESS trunk terminal specifications

To minimize the impact to non-FlexDial agents, the AXXESS trunk uses a terminal type defined as ABAXX. Non-FlexDial agents use the AB250 terminal type.

In table LTCINV, the ABAXX terminal type must be provisioned using the AXX250 exec lineup, as this is the only exec lineup supported for PTS AXXESS trunk group members. Non-FlexDial agents continue using the DTC250 or UTR250 exec lineups.

The amount of store in the XPM for exec lineups is limited to 12151 bytes. The sizes of the exec lineups used for the UCS DMS-250 switch are:

- UTR250—5225 bytes (used with AB250 terminal type)
- DTC250—4754 bytes (used with AB250 terminal type)
- AXX250—4366 bytes (used with ABAXX terminal type)
- DTCEX—2941 bytes (used with ABTRK terminal type)

With the current limitations on exec lineup store, the UTR250, AXX250 and DTCEX exec lineups cannot be included on a single DTC. Combinations of exec lineups with associated terminal types may be provisioned providing the store limitation is not exceeded.

Additionally, the XPM on which members of the AXXESS trunk group are provisioned must contain UTR hardware. If the XPM does not contain UTR hardware, then members of the AXXESS agent provisioned will fail to return to service (RTS).

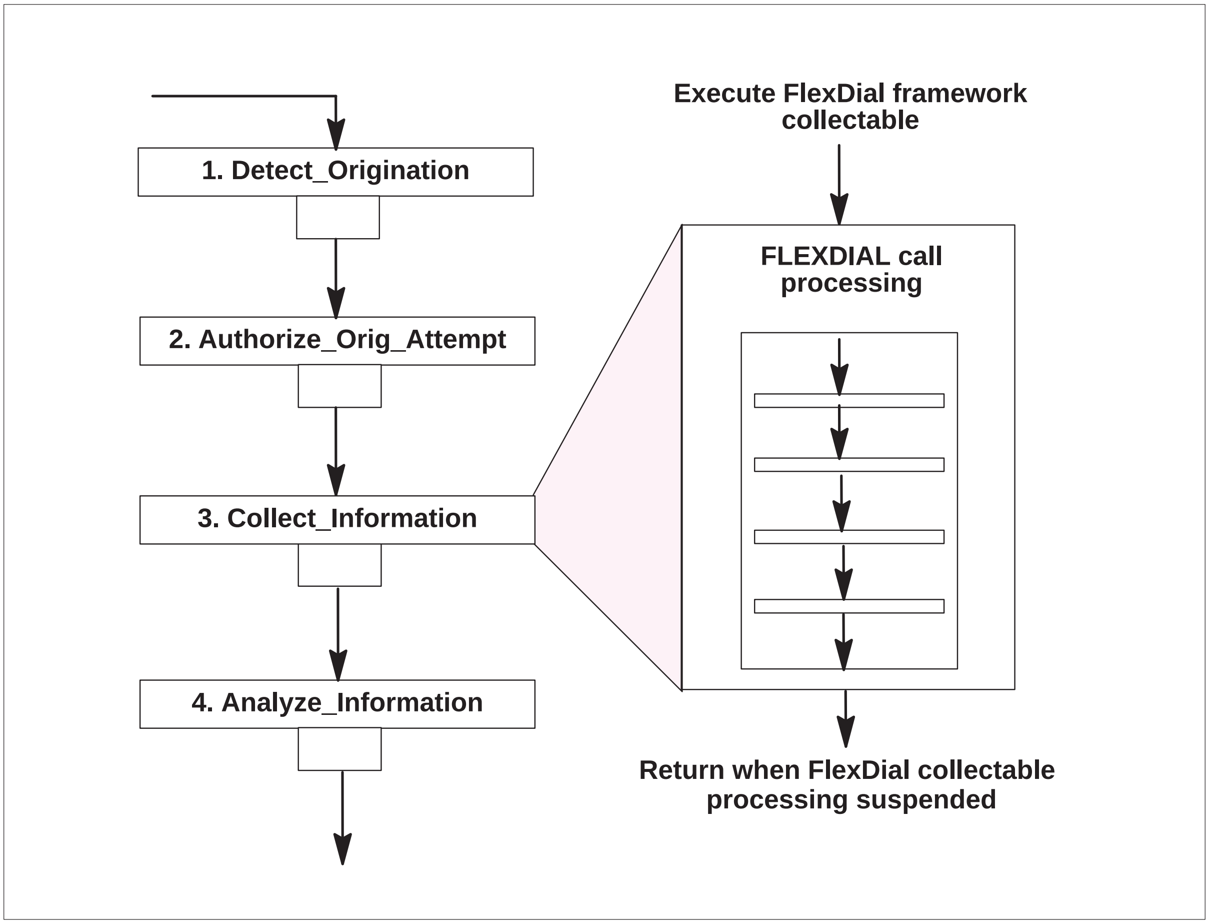
AXXESS agents requiring UTR ONKEY reorigination capability require UTR hardware in the XPM maintaining the agent terminal. The limited availability of UTR resources in the XPM require use of the UTR for reorigination purposes to be limited. This is to avoid potential UTR resource exhaustion for call processing.

Call processing activation

The processing of FLEXDIAL table collectables occurs during the “collect information” point in the basic call processing model, after origination has been detected and authorized by the UCS DMS-250 switch.

Figure 15-2 provides the basic call processing model.

Figure 15-2
Agent interface and call processing


1. Detect_Origination
2. Authorize_Orig_Attempt
3. Collect_Information
4. Analyze_Information
Execute FlexDial framework collectable
FLEXDIAL call processing
Return when FlexDial collectable processing suspended

Origination occurs when a seizure signal is received by the agent terminal controller (XPM). The handling of the seizure signal differs based on the type of AXXESS agent involved:

- For two- and four-wire AB-bit signaling DTMF and MF PTS agents, the terminal controller responds to the seizure signal according to the start signal provisioned for the agent in the TRKSIG table. The seizure signal is forwarded to the Central Module (CM), and after initialization of the call process, the first FlexDial collectable is processed. For DTMF type signaling, the initial instruction typically provides an audible prompt signal as a stimulus for the subscriber to enter digits.

- For Common Channel Signaling AXXESS Agents (CCS7), the seizure signal consists of an Initial Address Message (IAM). In addition to a seizure indication, this message typically contains all the information required to set up the call. The received IAM is forwarded to the CM, and after initialization of the call process, the first FlexDial collectable is executed.

  The collectables processed for Common Channel Signaling agents center on processing parameters in the IAM, unless inband digit collection is to be performed on the originating agent.

The activation of the FlexDial call processing framework results in the execution of the collectables provisioned in table FLEXDIAL.

Call processing model

The call processing entity that executes the provisioned FLEXDIAL table collectables and constitutes the call processing interface to the FlexDial framework is the Setup Collector.

Interacting with the agent and existing call processing software, the FlexDial call processing framework gives life to the collectables provisioned in the FLEXDIAL table. These collectables define the interaction with the originating agent and allow all necessary information to be collected and processed.

Figure 15-3 provides the FlexDial framework subsystem interactions.

Figure 15-3
FlexDial framework subsystem interactions

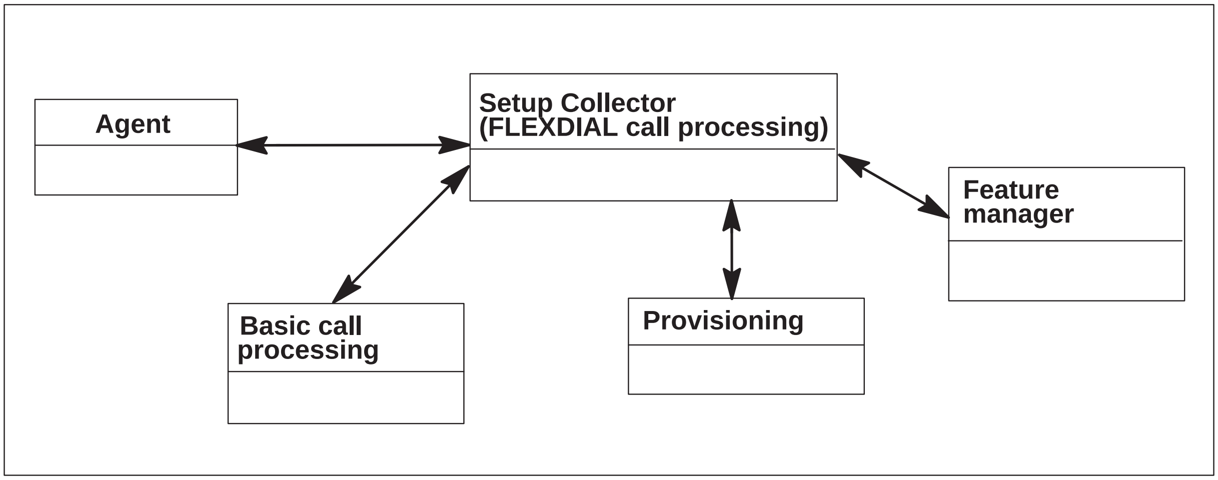


Table 15-1 provides Setup Collector interface associations.

Table 15-1
Setup Collector interface associations

Association	Rationale
Setup Collector and basic call processing	The Setup Collector encapsulates the functionality necessary to service the defined interactions with the originating agent. This functionality is defined in the role of the originating call model, and basic call processing uses the Setup Collector to achieve these goals.
Setup Collector and agent	The agent class provides the representation and interface for the AXXESS agent for the Setup Collector. When call processing needs to perform an agent specific task or reference agent related information, it uses the agent class.
Setup Collector and provisioning	The Setup Collector subsystem interworks with the provisioning system which identifies interaction with the originating agent and provides databases for screening digits received.
Setup Collector and Feature Manager	The Feature Manager maintains the set of applicable feature data for the call, which may be set and processed at any point in the life of the call. Many features are identified through processing of subscriber numbers by the Setup Collector.

Setup Collector subsystem

The Setup Collector consists of the following elements to process provisioned FLEXDIAL table collectables:

- Collectable Manager

  The Collectable Manager manages the list of collectables being processed.

- Call Processing collectable

  Call Processing collectables process specific instructions related to the definition of the interaction with the originating agent. The call processing collectable is considered a “twin sibling” to the provisioned collectable.

- Digit Buffer

  The Digit Buffer manages the incoming digits received from the originating agent or manipulated by sequence collectables, until they are taken and processed by digit collectables. The digit buffer can hold up to 64 digits.

- Message Center

  The Message Center identifies dynamic per-call information (in the form of messages) for collectables being processed. The message center provides a method of indirect communication among the collectables.

- Setup Collector events

  The defined set of Setup Collector events defines the event model for the Setup Collector and provide the state machine processing necessary to execute the collectables identified.

Figure 15-4 provides the Setup Collector subsystem model.

Figure 15-4
Setup Collector subsystem model

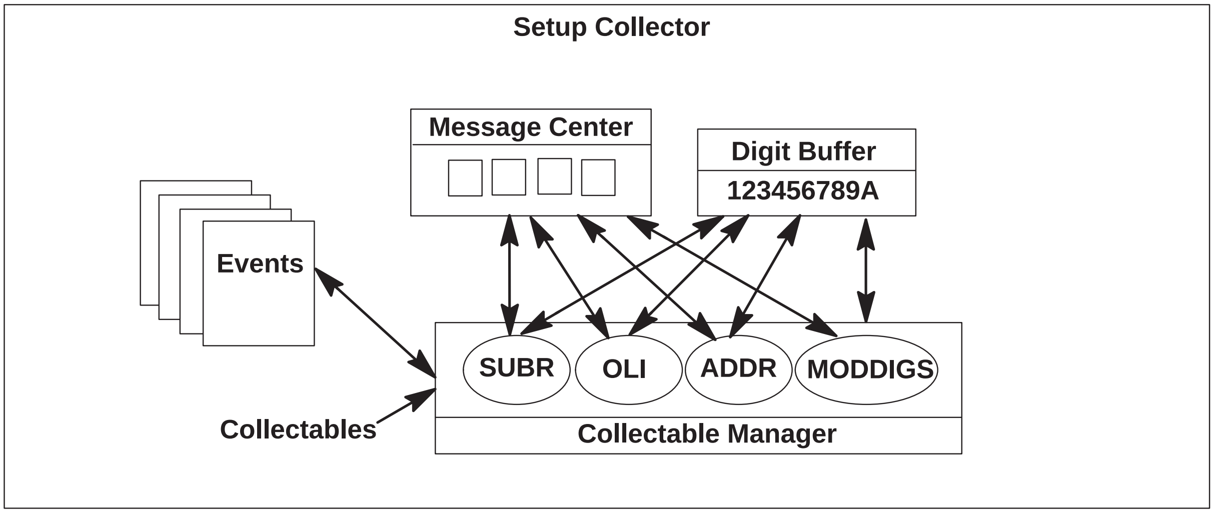


Table 15-2 provides Setup Collector interface associations.

Table 15-2
Setup Collector interface associations

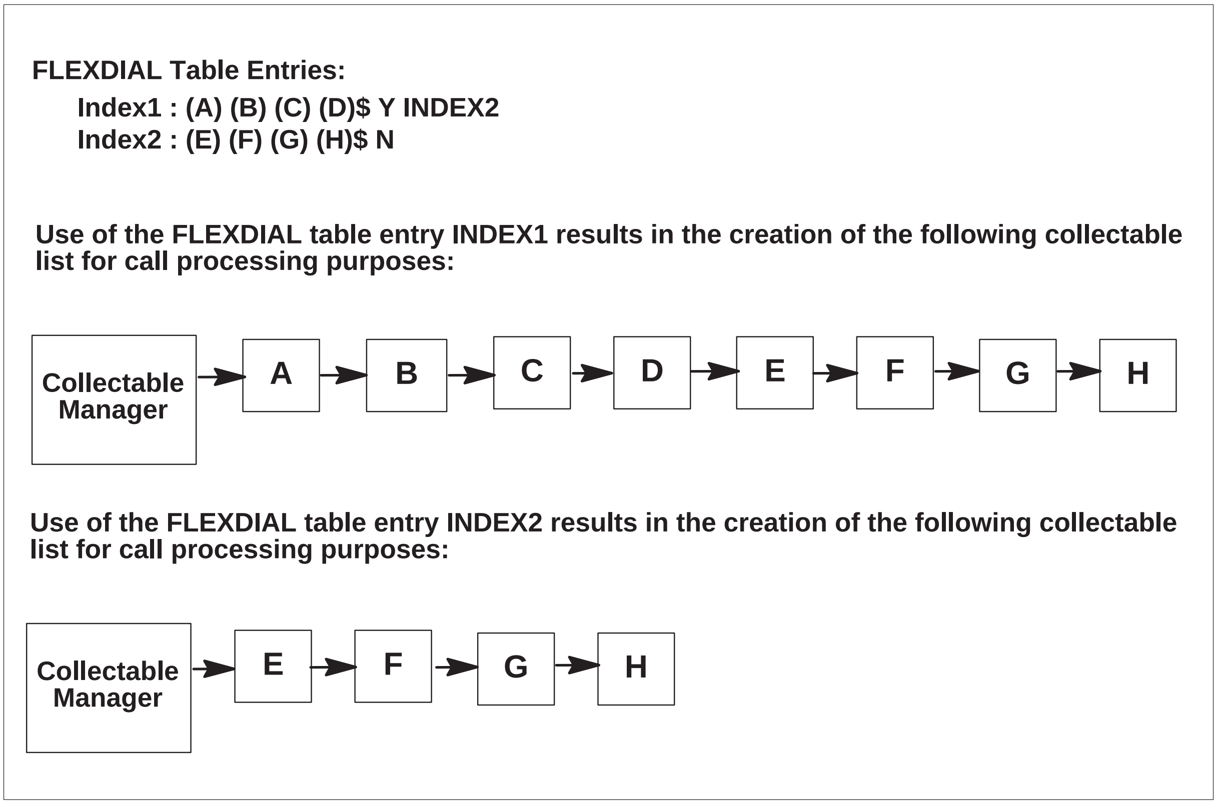
Association	Rationale
Setup Collector events and collectable manager	The Setup Collector events interact with the collectable manager to process the list of provisioned collectables that identify the complete interaction with the originating agent.
Collectable Manager and call processing collectables	The Collectable Manager contains the call processing collectables. Each collectable is a part of the overall picture identifying the interaction with the originating agent. The collectables can modify the collectable list represented by the Collectable Manager.
Call Processing collectables and Message Center	The Call Processing collectables retrieve pertinent dynamic call data in the form of messages from the Message Center. The data overrides the provisioned data associated with the collectable. The collectables can also leave messages at the Message Center for other collectables that have yet to be processed.
Call Processing collectables and digit buffer	The Call Processing collectables that require digits retrieve them from the digit buffer. Some collectables have the capability of not only taking digits from the buffer, but also adding and modifying digits in the buffer.

Collectable list management

A collectable list is identified as the vector of collectables provisioned for a particular FLEXDIAL table index. While a particular FLEXDIAL table entry is limited to provisioning up to four collectables in the list, use of the CONTINUE and INDEX fields of the tuple can increase the number of possible collectables in a particular collectable list.

Figure 15-5 provides a collectable list example.

Figure 15-5
Collectable list example



FLEXDIAL Table Entries:
Index1 : (A) (B) (C) (D)$ Y INDEX2
Index2 : (E) (F) (G) (H)$ N

Use of the FLEXDIAL table entry INDEX1 results in the creation of the following collectable list for call processing purposes:

Use of the FLEXDIAL table entry INDEX2 results in the creation of the following collectable list for call processing purposes:

Initial collectable list for call process

The initial collectable list to process is identified by the trunk group DPIDX field that identifies a vector of FLEXDIAL indexes to process for the call.

The FLEXDIAL table indexes identified by the TRKGRP table DPIDX vector each contain a list of collectables. These collectables initially identify the interaction with the originating agent, and each index is executed as though contained within an INCLUDE framework collectable. The result of this method of handling the FLEXDIAL indexes is that all the collectables from the FLEXDIAL table indexes are sequentially grouped to form the initial collectable list to be processed for the call.

Figure 15-6 provides an initial collectable list example.

Figure 15-6
Initial collectable list example

Table TRKGRP:

Axxess1 AXXESS ... (INDEX1)(INDEX2)(INDEX3)$ ...

Table FLEXDIAL:

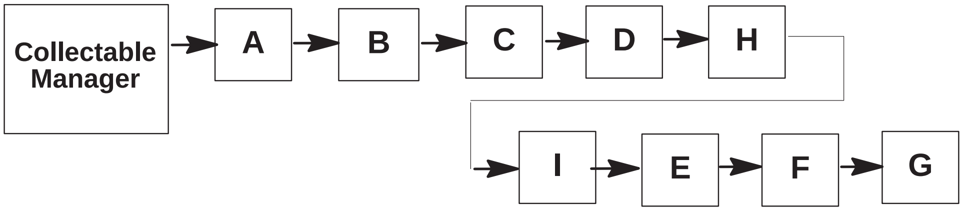
Index1 : (A) (B) (C) (D)$ Y INDEX10
Index2 : (E)$ N
Index3 : (F) (G)$ N

Index10 : (H) (I)$ N

The DPIDX values identified are treated as if contained within INCLUDE collectables:

(INCLUDE index1) (INCLUDE index2) (INCLUDE index3)

This results in the following initial collectable list to be created for the call:



This initial list of collectables is treated as a single list, and not as separate lists identified by each index. This affects the FlexDial list modification actions APPEND, INSERT, REPLACE, and EXEC.

The APPEND, INSERT, REPLACE, and EXEC modification actions only occur during the execution of a collectable. Modification actions are always identified through provisioning, and only apply to the "current collectable list" being processed. For example, the DPIDX FLEXFEAT table option provides this ability when validating subscriber numbers.

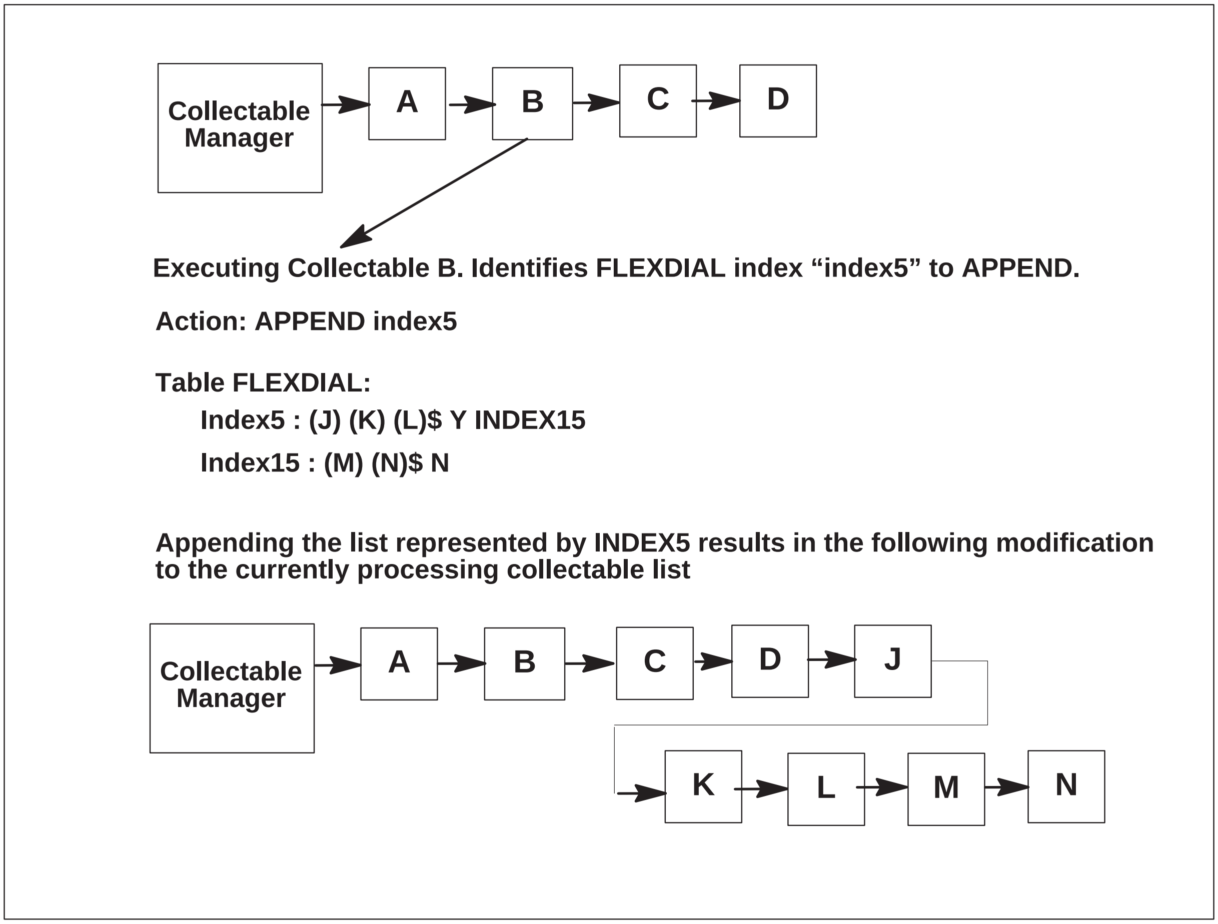
Note: The current collectable list is identified as the list of collectables being executed at a moment in time. The currently processing collectable list changes when branching collectables are executed (such as IFDIGS, IFNOA, GOTO, or DO collectables).

APPEND list modification action

The APPEND modification action appends the list of collectables identified by the associated FLEXDIAL table index to the current collectable list. The new collectable list is merged into the identity of the currently processing collectable list.

Figure 15-7 shows an APPEND list example.

Figure 15-7
APPEND list modification example



Executing Collectable B. Identifies FLEXDIAL index “index5” to APPEND.

Action: APPEND index5

Table FLEXDIAL:

Index5 : (J) (K) (L)$ Y INDEX15

Index15 : (M) (N)$ N

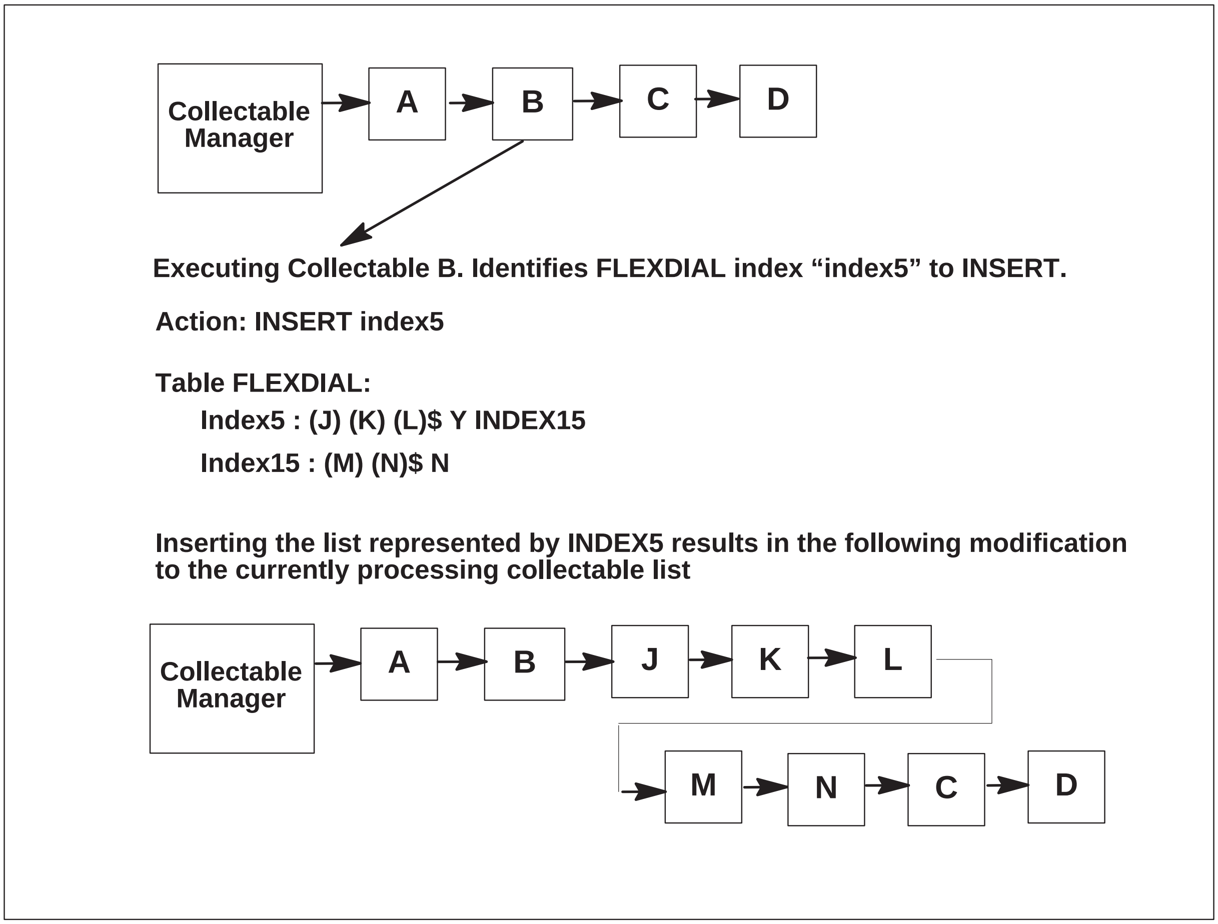
Appending the list represented by INDEX5 results in the following modification to the currently processing collectable list

INSERT list modification action

INSERT inserts the list of collectables identified by the associated FLEXDIAL table index into the current collectable list before the next collectable executed. The new collectable list is merged into the currently processing collectable list. The INSERT action performs the same action as that of the INCLUDE collectable.

Figure 15-8 shows an INSERT list example.

Figure 15-8
INSERT list modification example



Executing Collectable B. Identifies FLEXDIAL index “index5” to INSERT.

Action: INSERT index5

Table FLEXDIAL:

Index5 : (J) (K) (L)$ Y INDEX15

Index15 : (M) (N)$ N

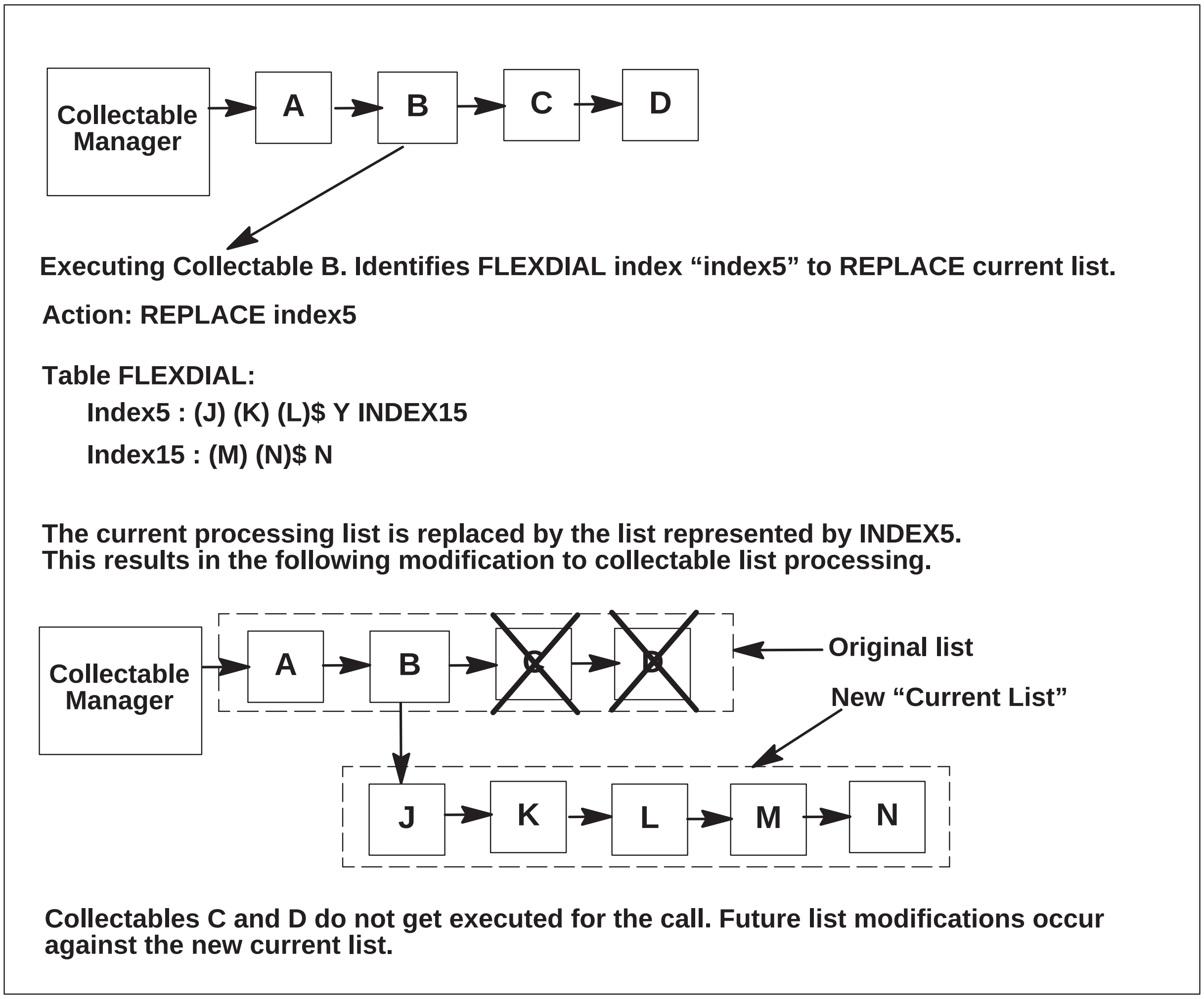
Inserting the list represented by INDEX5 results in the following modification to the currently processing collectable list

REPLACE list modification action

The REPLACE modification action replaces the remainder of the current list of collectables executed with a new collectable list. This new collectable list remains separate, identified by the associated FLEXDIAL table index into the current collectable list before the next collectable to be executed. The new collectable list replaces the currently processing collectable list. The REPLACE action is the same as the action performed by the GOTO collectable.

Figure 15-9 shows a REPLACE list example.

Figure 15-9
REPLACE list modification example



Executing Collectable B. Identifies FLEXDIAL index “index5” to REPLACE current list.

Action: REPLACE index5

Table FLEXDIAL:
Index5 : (J) (K) (L)$ Y INDEX15
Index15 : (M) (N)$ N

The current processing list is replaced by the list represented by INDEX5. This results in the following modification to collectable list processing.

Collectables C and D do not get executed for the call. Future list modifications occur against the new current list.

All collectables identified for processing are executed sequentially. Some collectables such as IFDIGS and GOTO cause branching to occur in FlexDial call processing.

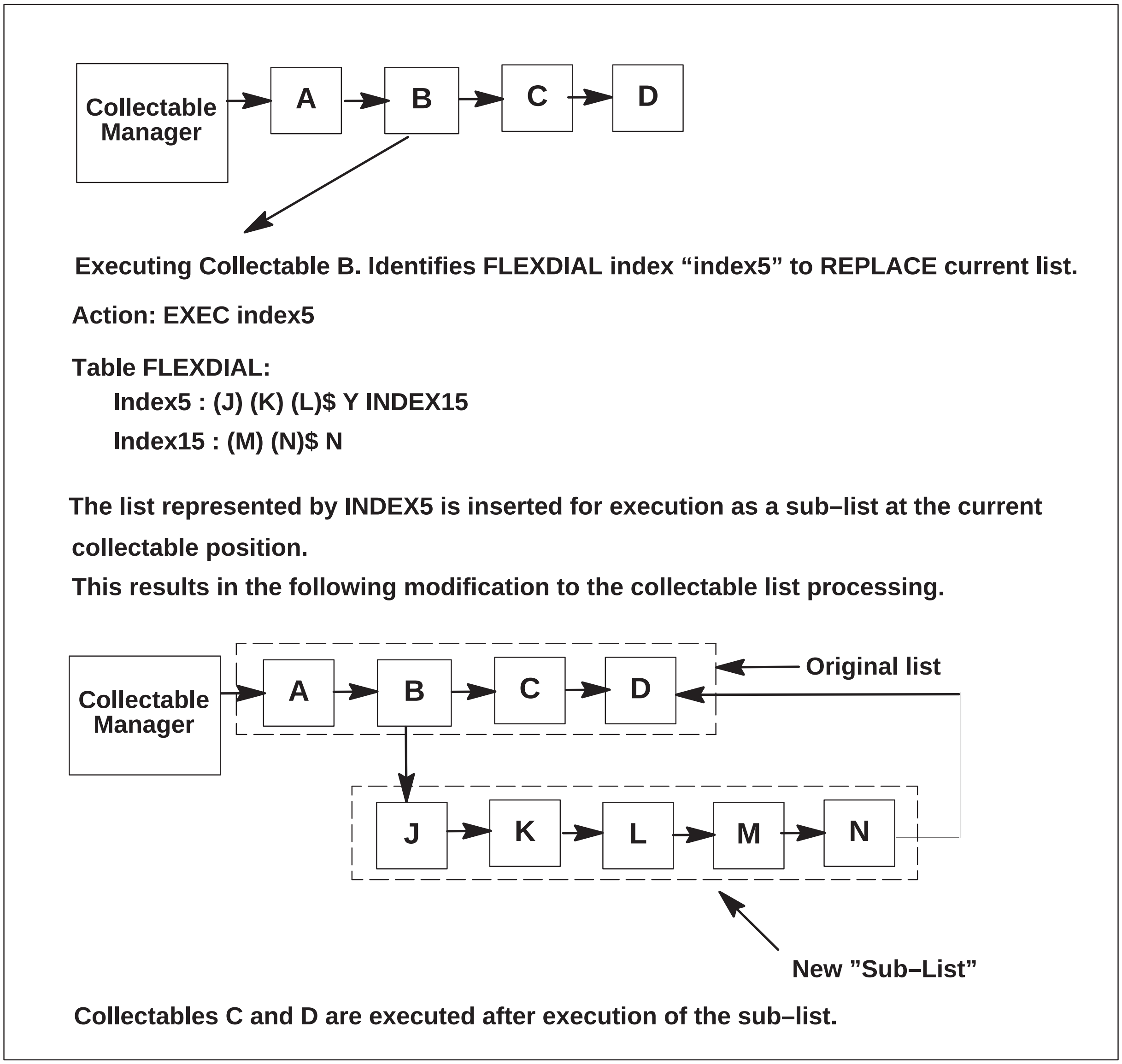
EXEC list modification action

The EXEC modification action inserts for execution a sub–list into the current list at the current collectable. Execution continues into the sub–list and returns to the original list when execution of the sub–list is complete. Any list actions performed by collectables in the sub–list only affect the sub–list. Execution will always return to the original list. The EXEC action performs the same action as that of the DO collectable.

Figure 15-10 shows an EXEC list example.

Figure 15-10
EXEC list modification example

FW-xxxxx



Executing Collectable B. Identifies FLEXDIAL index “index5” to REPLACE current list.

Action: EXEC index5

Table FLEXDIAL:
Index5 : (J) (K) (L)$ Y INDEX15
Index15 : (M) (N)$ N

The list represented by INDEX5 is inserted for execution as a sub–list at the current collectable position.

This results in the following modification to the collectable list processing.

Collectables C and D are executed after execution of the sub–list.

Execution of collectables

All collectables are executed within the event model of the Setup Collector. Each collectable is processed in three general steps:

- initialization
- service execution

- completion

Collectable initialization

In the initialization of the call processing data for the collectable, the variables are initialized with the collectable’s provisioned information as well as information in the form of messages from the message center.

Message center message processing

The message center holds messages for specific collectables. Messages may be left at the message center during different points in the call process:

- At initialization of the Setup Collector messages are left by the agent (the messages are initiated through provisioning of the TRKFEAT MSGCTR option).
- At validation processing of SUBR collectables post messages (the messages are initiated through provisioning of the FLEXFEAT MSGCTR option retrieved from validation).
- When processing collectables messages may be posted (the message is initiated internally by call processing).

Messages are stored in a FIFO (first in–first out) order at the message center, with new messages appended to the end of the message list for a collectable. The information contained within the message is typically provisioned with the message in table MSGCTR, with the exception of the FILED message.

Instead of provisioning the digits within the FILED message, the digits may be determined at run-time from two sources:

- VALDIGS filed digits

  Through the VALDIGS filed digits type, the subscriber number digits validated are used in the FILED digits message posted.

  This option is only applicable when the MSGCTR message is being posted through execution of a SUBR or SUBRPARM collectable whose digits have been validated in table FLEXVAL.

  It is not possible to identify the “validated subscriber number digits” when this option is used through table TRKFEAT or through CALLTYPE collectable processing. Therefore the use of this option in these scenarios results in no FILED message being posted at the message center.

- CALLING filed digits

  Through the CALLING filed digits type, the calling party address digits processed for the call are included in the FILED message. The calling party address digits do not have to be identified at the time the message is posted, but must be identified when the message is retrieved for processing. If the calling party address digits are not identified by the time the message is retrieved, then call processing effectively processes a FILED message with zero digits, and the message is of no consequence.

Calling party address digits are received through SUBR or SUBRPARM collectable processing, where the subscriber number type (FLEXTYPE field) used contains the provisioned CALLING option in table FLEXTYPE.

Typically a collectable instance consumes all messages addressed to that collectable type. Multiple copies of a consumed message overwrite information from a previous message, and only the information in the last message processed of a particular message type is used by the collectable.

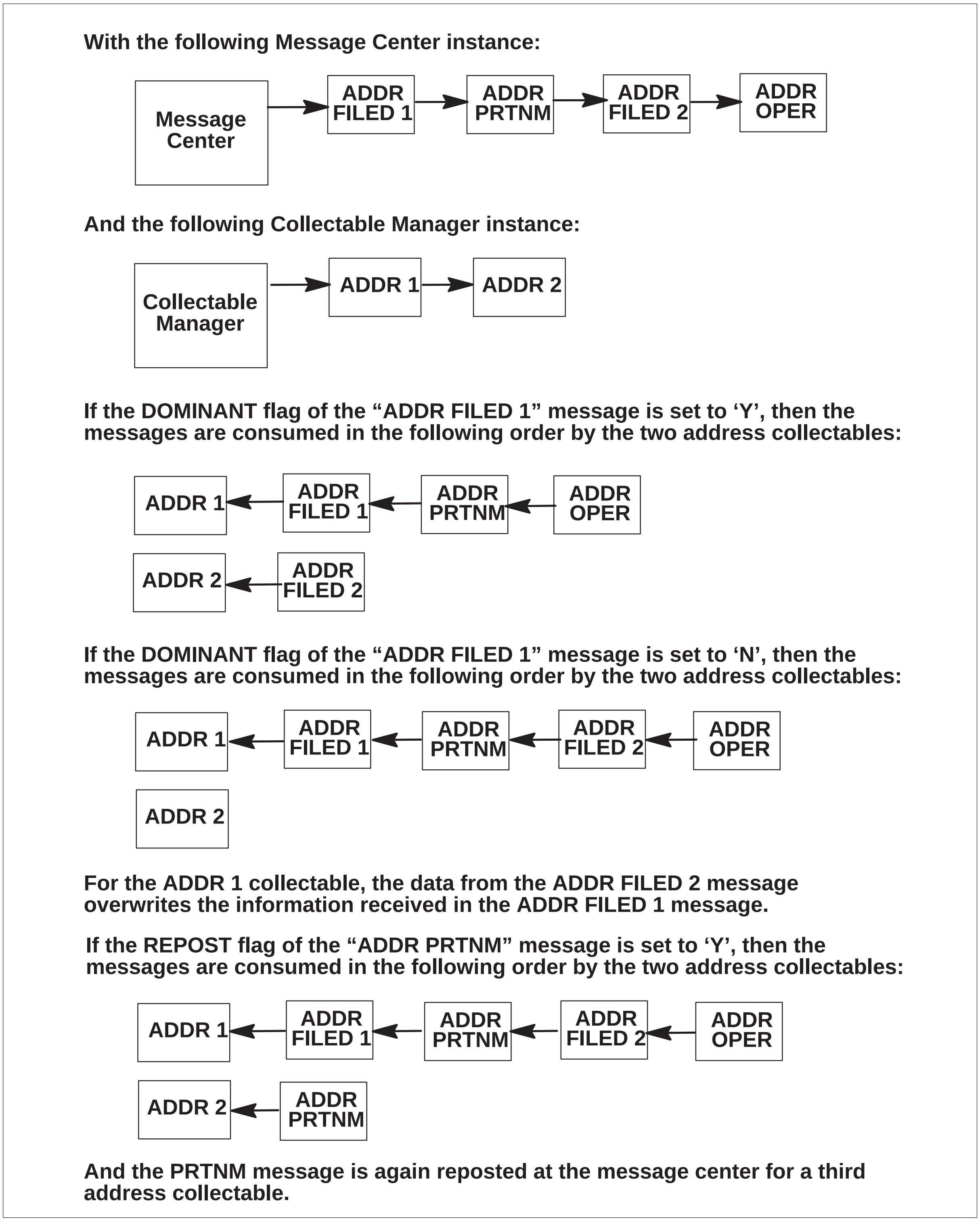
However, when a collectable consumes a message with the DOMINANT flag set (equal to Y), that collectable may not consume another message of that message type. When another message of the message type is reached, then consumption of MSGCTR messages is halted and unprocessed messages (including the message with the identical message type) remain for another collectable instance.

Note: This rule includes the attributes of the message as well (such as the REPOST indicator). A message is only reposted if it is the last message type instance that is processed by the collectable.

If a REPOST indicated message is processed by the collectable (that is, it is a DOMINANT message or the last message of the particular message type processed), then after processing the information in the message, the collectable reposts the message at the message center at the front of the remaining MSGCTR list for that collectable type. The message is then set to be processed by the next collectable instance of that collectable type. When a DOMINANT message is also identified as a REPOST message, then the message loses its DOMINANT characteristic when it is reposted at the message center.

Figure 15-11 shows a message center message retrieval example.

Figure 15-11
Message center message retrieval example



With the following Message Center instance:

And the following Collectable Manager instance:

If the DOMINANT flag of the “ADDR FILED 1” message is set to ‘Y’, then the messages are consumed in the following order by the two address collectables:

If the DOMINANT flag of the “ADDR FILED 1” message is set to ‘N’, then the messages are consumed in the following order by the two address collectables:

For the ADDR 1 collectable, the data from the ADDR FILED 2 message overwrites the information received in the ADDR FILED 1 message.

If the REPOST flag of the “ADDR PRTNM” message is set to ‘Y’, then the messages are consumed in the following order by the two address collectables:

And the PRTNM message is again reposted at the message center for a third address collectable.

In the example above, during initialization of the first ADDR collectable (ADDR 1), processing of message center messages ends when the second FILED message is scanned if the DOMINANT field of the first FILED message is set to Y. The second FILED and remaining OPER messages do not get processed until a second ADDR collectable (ADDR 2) is executed.

If the DOMINANT field of the first FILED message is set to N, then all four messages are consumed by the first ADDR collectable, and the data from the first FILED message is overwritten by data from the second FILED message.

If the REPOST indicator is set on the PRTNM message, then after being processed by the first ADDR collectable, the PRTNM message is reposted at the message center and is processed and subsequently reposted by the second ADDR collectable.

Service execution

The service execution of a collectable is where the collectable provides its function. The applications of the collectables are individually discussed beginning with “Signaling collectable (SIG)” on page 16-1.

Completion

At the completion of the collectable, the collectable manager marks the collectable as processed, and selects the next available collectable in the list for processing.

If no more collectables are left to be processed, then the setup collector terminates normally, and the “collect information” point in the call process is complete.

Collectable completion

Collectable processing is complete or suspended when one of the following events occurs:

- The originating agent abandons the call by disconnecting.
- All required collectables in the specified collectable lists have been processed.
- An identified signal is not received by the RCVSIG collectable and treatment is identified as the failure action. In this case, collectable processing is suspended and the call is immediately terminated to the treatment specified.
- The TERMINATE collectable with the STOP process direction has been executed.

- The APTRMT collectable has been executed where a treatment is specified for the call.
- A Feature Not Allowed (FNAL) or Resource Unavailable (RESU) treatment is set for the call.
- A permanent signal (PSIG) or partial dial (PDIL) exception is generated by a digit collectable and the treatment is set for the call.
- A reset limit for a digit collectable is exceeded and a treatment is set for the call.

When FlexDial collectable processing is complete, the call is routed to the destination identified by the ROUTE collectable, validation screening, translations, or according to a specified treatment.

Note: If the call has already been connected to a terminating agent, processing is immediately suspended and awaits further call progress indication (such as alerting, answer, or disconnect).

FlexDial collectables call processing applications

This chapter describes how FlexDial collectables are used in call processing.

Signaling collectable (SIG)

The Signaling (SIG) collectable alters digit collection signaling characteristics during the call setup interaction with the originating agent. An inherent function of the SIG collectable is to identify that from this point in the interaction with the originating agent, digit collection is being performed using inband signaling techniques.

Note: Digits are collected by listening for tone patterns in the bearer channel of the agent.

Application

The SIG collectable modifies the digit collection methodology of the current call and instructs the agent terminal controller (XPM) to collect digits according to the revised collection signaling parameters.

The SIG collectable can be used to perform a simple operation such as altering the partial dial timer, to a more visible operation such as modifying the pulse type for receiving digits. The SIG collectable can be used multiple times during the interaction with the originating agent to alter digit collection signaling characteristics as required.

Unique agent types interpret the action of the SIG collectable differently. For PTS agent types (DS1, FXS, FXO), the initial digit collection signaling parameters are defined by the TRKSIG table for the trunk group. When a SIG collectable is executed for these agents, the agent terminal controller is instructed to collect digits according to the revised collection signaling parameters. PTS agents always collect digits inband the bearer channel.

When the ACKWINK option is used for a DS1 PTS agent, and the SIG collectable executed is transitioning the incoming pulse type (IPULSTYP) from MF to DTMF tone collection, then execution of the SIG collectable also triggers transmission of the acknowledgment wink. The pre-wink duration timer value used is taken from the table OFCSTD PRE_SND_WK_DD_TIME office parameter, and the wink duration timer value used is taken from one of two office parameters:

- SND_MF_WK_TIME (table OFCSTD) for national calls
- EA_INT_WINK_DUR (table OFCVAR) for international calls

For Common Channel Signaling agents (CCS7), the SIG collectable marks the transition from processing digits received through out-of-band signaling (the Initial Address Message [IAM]) to collecting digits inband from the subscriber. Before a SIG collectable is executed, all digits processed in the interaction with the originating agent come from the Received Seizure (IAM) message using out-of-band digit collectables. After the SIG collectable is executed, digits processed may be collected using inband collection techniques according to the collection signaling parameters specified by the SIG collectable. (This applies to inband digit collectables. Even after execution of the SIG collectable, it is still possible to execute out-of-band collectables.).

To perform inband digit collection on common channel signaling agents, the SIG collectable must first be used to specify the collection signaling parameters for the process. Execution of the SIG collectable for common channel signaling agents causes an Address Complete message (ACM) to be sent on the signaling channel for the agent. The ACM is only delivered once in case the SIG collectable is executed multiple times for the interaction with the originating agent.

The execution of the SIG collectable does not require an acknowledgment for the agent terminal controller, and therefore processing continues immediately to the next collectable.

Use case scenarios

The SIG collectable is used in the following call scenarios:

- modifying partial dial timers between digit collectable processing
- MF calls requiring subscriber digit collection

  For AXXESS agents impersonating PTS FGD protocols, the SIG collectable is required on cut-through, transitional, and other call types where the switch from MF signaling to DTMF signaling is required.

- inband collection on CCS7 agents

The use of the SIG collectable is required to transition to inband digit collection on common channel signaling agents.

Restrictions and limitations

The following restrictions and limitations apply to the SIG collectable:

- The SIG collectable may not be used for Nx64 (Wideband) defined AXXESS agents. If a SIG collectable execution is attempted, all collectable processing is suspended, call processing continues with the analyze information point in call, and a FLEX 301 trouble log is output with a trouble code value of:

  ```
  Inband digit collection not supported for NX64
  ```

  The flextype field of the log is set to SIG and no digits are included in the report.

- If a request for inband digits is made on CCS7 agents (by using either the COLDIG, SUBR, ADDR, OLI, or CIC collectables) before a SIG collectable is executed, then all collectable processing is suspended and feature not allowed (FNAL) treatment is immediately applied to the call.
- Execution of the SIG collectable for CCS7 agents causes call progress indication in the form of an address message (ACM) message to be sent on the signaling channel for the agent.

Receive signal collectable (RCVSIG)

The Receive Signal (RCVSIG) collectable is used on PTS agent types to receive an inband wink, answer, or release signal in order to adhere to a certain protocol outlining the interaction with the originating agent.

Note: This collectable is currently not supported. Generally, interactions with the originating agent require signals to be sent rather than received. If an off-hook type of signal is received, call processing will be initiated. If an on-hook (release) type of signal is received, the call will be torn down. However, the capability to define Receive Signal interactions may eventually exist, so information on the collectable is included.

Application

Unique agent types interpret the action of the RCVSIG collectable differently. PTS agent types instruct the agent terminal controller to receive either an off-hook-on-hook-off-hook transition (wink), or a hook state transition according to the specified timer values. For wink signals, the duration of the on-hook cannot exceed the filter timer value specified by the TRKSIG OSDCFLTR option. If the duration of the on-hook exceeds the ODSCFLTR timer value, then the signaling transition is treated as a disconnect.

Note: The on-hook-off-hook signal is indicated by the A-bit in inband ABCD bit signaling.

Execution of the RCVSIG collectable for release signals supersedes normal hook state monitoring by call processing. For example, if an on-hook signal is received, then the collectable is successfully executed, and call processing continues executing the remaining collectables.

Note: Because the call is initially off-hook (a received off-hook signal initiates the call process), the RCVSIG ANSWER only provides value if following a RCVSIG RELEASE instruction.

Normally, a received on-hook signal is handled as originating agent disconnect and the call is taken down.

Once the signal is successfully received, the processing related to the RCVSIG collectable is complete.

If the signal is not received within the time specified, then collectable processing is suspended and the treatment identified is immediately applied to the call.

The RCVSIG collectable can be used multiple times on the PTS agent type to receive multiple signals.

For common channel signaling (CCS7) agent types, the RCVSIG collectable provides no call processing function and the next collectable is processed. When the RCVSIG collectable is executed for a CCS7 agent type, a FLEX 601 information log is generated with a report indicator of:

```
No operation performed for CCS agent.
```

The flextype field of the log is also set to RCVSIG, and no digits are included in the report.

Use case scenario

The RCVSIG collectable is used in the following call scenarios:

- matching protocol specifications

  In order to match defined protocol specifications of the interaction with the originating agent, the RCVSIG collectable receives a wink, answer, or release signal on the agent signaling channel. This type of signal typically separates stages of digits being received.

Restrictions and limitations

The RCVSIG collectable performs no call processing function for common channel signaling agent types, and if executed for a CCS7 type of agent, a FLEX 601 information log is generated with a report indicator of “No operation performed for CCS agent.” The Flextype field of the log is also set to RCVSIG, and no digits are included in the report.

Send signal collectable (SNDSIG)

The Send Signal (SNDSIG) collectable sends an inband wink, an out-of-band alerting, or an answer or release signal to the originating agent in order to adhere to a certain protocol outlining the interaction with the originating agent. The answer or release signal may be sent inband or out-of-band.

Inband signals may only be sent on inband signaling (PTS) agents. Out-of-band signals may only be sent on common channel signaling (CCS) agents.

Application

Unique agent types interpret the action of the SNDSIG collectable differently. For PTS agent types, the signaling types assume the following meanings:

- WINK

  Delivers a wink signal opposite to the current hook state. Once the wink is delivered, bearer channel PCM is connected and any tones applied are disconnected.

  For the WINK signal, the type of wink sent is opposite of the current hook state being transmitted to the originating agent. If the facility is on-hook, then an off-hook wink is sent. If the facility is off-hook, then an on-hook wink is sent.

- ALERTING

  For alerting, an audible ringback signal is applied on the bearer channel of the agent. The ringback tone may be disabled through use of any other of the SNDSIG signaling types.

  For ALERTING, standard North American ringback tone is applied.

- ANSWER

  Connects bearer channel PCM and disconnects any tone applied to the agent. Sends answer signal on the agent terminal facility.

The ANSWER signaling type causes an off-hook signal to be sent. The RELEASE signaling type causes an on-hook signal to be sent.

- RELEASE

  Disconnects bearer channel PCM and sends a release signal to the agent terminal facility.

The SNDSIG collectable can be used multiple times on PTS agent types to send multiple signals of different or the same type.

For common channel signaling (CCS7) agent types, execution of the SNDSIG collectable causes either an ACM, answer message (ANM), or release message (RLS) to be sent on the originating agent on the signaling channel. The messages map in the following fashion to the SNDSIG signal types:

- ALERTING signal—ACM message
- ANSWER signal—ANM message
- RELEASE signal—RLS message

Only one ACM is delivered per call before answer on the agent signaling channel. Therefore if the ACM or ANM has already been sent, execution of the SNDSIG collectable for CCS7 agents results in no call processing action and a FLEX 601 information log is generated with a report indicator of:

```
Alerting signal already delivered.
```

The Flextype field of the log is also set to SNDSIG, and no digits are included in the report.

Only one ANM or RLS is delivered per call on the agent signaling channel. Therefore if the ANM has already been sent, execution of the SNDSIG collectable for CCS7 agents results in no call processing action and a FLEX 601 information log is generated with a report indicator of:

```
Answer signal already delivered.
```

The Flextype field of the log is also set to SNDSIG, and no digits are included in the report.

If the RLS is being sent, the release cause value identifies normal disconnect. After the release is delivered on the signaling channel, the call is taken down.

When the signal is successfully delivered, processing related to the SNDSIG collectable is complete.

Use case scenario

The SNDSIG collectable is used in the following call scenarios:

- matching protocol specifications

  In order to match defined protocol specifications of the interaction with PTS originating agents, the SNDSIG collectable transmits acknowledgment wink signals on the agent signaling channel. This type of signal typically separates stages of digits being received.

Restrictions and limitations

The following restrictions and limitations apply to the SNDSIG collectable:

- The WINK signal type is not supported for CCS type agents. If executed for a CCS agent type, no call processing action is performed and a FLEX 301 trouble log is generated with a trouble code value of:

  ```
  Wink not supported for CCS agents
  ```

  The Flextype field of the log is set to SNDSIG and no digits are included in the report.

- For common channel signaling agents, only one ACM is delivered per call before answer. If the ACM or ANM has already been sent, execution of the SNDSIG collectable results in no call processing action and a FLEX 601 information log is generated with a report indicator of:

  ```
  Alerting signal already delivered.
  ```

  The Flextype field of the log is also set to SNDSIG, and no digits are included in the report.

- For common channel signaling agents, only one ANM is delivered per call. If the ANM has already been sent, execution of the SNDSIG collectable results in no call processing action and a FLEX 601 information log is generated with a report indicator of:

  ```
  Answer signal already delivered.
  ```

  The Flextype field of the log is also set to SNDSIG, and no digits are included in the report.

- For common channel signaling agents, only one RLS is delivered per call. If the RLS has already been sent, execution of the SNDSIG collectable results in no call processing action and a FLEX 601 information log is generated with a report indicator of:

  ```
  Release signal already delivered.
  ```

  The Flextype field of the log is also set to SNDSIG, and no digits are included in the report.

- When a release signal is sent on a CCS agent (RLS message), the cause indicator is set to normal clearing.

TERMINATE Call to Destination collectable (TERMINATE)

The TERMINATE Call To Destination (TERMINATE) collectable is used when the call must be completed to the terminating agent before remaining collectables can be processed.

Application

When the TERMINATE collectable is executed, collectable processing is suspended, and the call is connected to the terminating agent based on ROUTE collectable specifications, validation screening, or appropriate translation of the address digits. If a treatment has been set for the call, then the only call processing action performed by the TERMINATE collectable is to disallow inband digit collection requests from this point onward, and the next collectable in the list is executed.

If the route has not been identified and no address digits have been received at this point in the interaction with the originating agent, then the translation attempt results in the call being terminated to vacant code (VACT) treatment, as not enough information is available to determine the destination for the call.

If the process direction indicator is set to STOP, then processing of collectables is suspended. If the indicator is set to CONTINUE, then collectable processing resumes with the next collectable (following the TERMINATE collectable) after the call has been established to the terminating agent.

When the NOTIFY option is used, the tone or announcement is connected to the originating agent for the specified period or number of cycles, before the call is connected to the terminating agent.

If an announcement member cannot be allocated at the time of the request, then the NOTIFY announcement is skipped, and a FLEX 301 trouble log is generated with a trouble code value of:

```
Announcement not available: <announcement clli>
```

The Flextype field of the log is set to TERMINATE and no digits are included in the report.

After the log is generated the call is immediately connected to the terminating agent.

After the TERMINATE collectable is executed, remaining digit collectables requiring inband digit collection cannot be processed for the call. (When the call state gets beyond dialing and into setup, collection of inband digits from the agent terminal controller is not supported.) Digit collection attempts requesting action from the agent terminal controller (XPM) to collect digits are blocked by call processing. At this point, execution of the digit collectable is halted, the execution of further collectables is suspended, and a FLEX 303 failure to execute log is generated with a failure reason of:

```
Cannot perform inband collection after TERMINATE.
```

If a treatment is set for the call, then no action is performed by the TERMINATE collectable, including any NOTIFY action. The only effect of executing the TERMINATE collectable is that from this point onward, digit collection requests are not allowed. Treatment is applied to the call after remaining collectables have been processed.

If the originating agent is already connected to a terminating agent when a TERMINATE collectable is processed, then one of the following actions is performed based on the process direction indicator:

- For the STOP process direction, the execution of collectables is suspended and the call process is suspended until the next call event occurs.
- For the CONTINUE process direction, the next collectable in the list is processed.

Use case scenario

The TERMINATE collectable is used in the following call scenario:

- exit collectable processing

  The TERMINATE collectable provides a way to prematurely exit collectable processing.

Restrictions and limitations

The following restrictions and limitations apply to the TERMINATE collectable:

- Inband digit collection cannot be performed after the TERMINATE collectable has been executed and the call has been connected to the identified destination.
- When the call is already connected to the terminating agent, use of the TERMINATE collectable either simply suspends collectable processing or is essentially ignored based on the process direction indicator.

- When collectable processing resumes after a TERMINATE collectable is executed, the next collectable in the list (following the TERMINATE collectable) is executed.
- If a treatment is set for the call, then the TERMINATE collectable does not perform any action, including NOTIFY. The only effect of executing the TERMINATE collectable is that from this point onward, digit collection requests are not permitted. All digits processed must already be in the digit buffer or be FILED digits.

Notify collectable (NOTIFY)

The NOTIFY collectable provides the ability to play an uninterruptable tone or announcement during the interaction with the originating agent. Prior to UCS08, the only mechanism for playing an uninterruptable announcement within FlexDial call processing was through the TERMINATE collectable NOTIFY option.

Application

The NOTIFY collectable provides the ability to play an uninterruptable tone or announcement for the originating agent. This collectable is provisioned with a key element of information needed for processing:

- The TONE or ANNC CLLI

The TONE or ANNC CLLI is played for the complete duration and number of cycles defined, after which the next identified collectable is executed.

The tone or announcement is played as an uninterruptable notification, and may precede, come between, or follow digit collection requests. Digits entered during the notification do not stop it from playing to completion, though any digits entered are buffered and apply to the next collection request.

Usecase Scenarios

The NOTIFY collectable provides a generic mechanism for FlexDial call processing whereby an uninterruptable tone or announcement may be integrated into the desired interaction with the originating agent.

Interactions

The NOTIFY collectable requires the use of PORTPERM extension blocks in order to connect and process announcement notifications. The number of available PORTPERM extension blocks is identified in table OFCENG by the NUMPERMEXT office parameter.

Restrictions and limitations

The following call processing restrictions or limitations exist for the NOTIFY collectable:

- Bearer channel interactions, including tone and announcement notifications specified by the NOTIFY collectable, may not be performed from the following triggers:
  - — Following a TERMINATE collectable with CONTINUE process direction.
  - — Through the CALLCOND collectable execution trigger.

  If an attempt to execute the NOTIFY collectable under these conditions is performed, then a FLEX 303 failure to execute log is generated with a failure reason of “Cannot perform inband processing” and the execution of further collectables is suspended.

- If an announcement member cannot be allocated at the time of the request, then the NOTIFY announcement is skipped and a FLEX 301 trouble log is generated with a trouble reason of “Announcement not available: <announcement CLLI>.”

RETRIEVE Collectable

The RETRIEVE collectable provides the ability to retrieve the identified digits that have been already processed for the call and add them to the digit buffer at the desired location. Digit manipulation or alternative processing may then be performed on the buffered digits.

Application

The RETRIEVE collectable does not request any digits from the originating agent, but retrieves digit that have already been processed for the call. This collectable is provisioned with two key elements of information needed for processing:

- The specific digits to retrieve.
- The location in the digit buffer to place the digits.

The following types of digits may be retrieved for insertion into the digit buffer:

- Dialed number address digits (DIALED)
- Called party number address digits (ADDR)
- Originating line information digits (OLI)
- Carrier identification code digits (CIC)
- Subscriber number digits (SUBR)

The identification of subscriber number digits is further defined by specification of the FLEXTYPE, or subscriber number type, and either the first or last occurrence of the processing of these particular subscriber number type digits. If only a single subscriber number of the type identified has been processed by the call, then either value of the OCCURRENCE field retrieves the processed subscriber number digits for insertion into the digit buffer.

If the type of digits to be retrieved have not been processed for the call at the time of the execution of the RETRIEVE collectable, then no digits are inserted into the digit buffer, and the RETRIEVE collectable performs no operation for the call.

When the identified digits have already been processed for the call, then the RETRIEVE operation inserts the identified digits into the digit buffer. Once the digits are in the buffer, it is just as if they were received from the originating agent, and can therefore be processed accordingly by remaining digit or sequence collectables.

The number of digits to be inserted into the digit buffer may be limited by the DIGLEN field, which identifies the maximum number of digits from the digit source that are to be inserted into the digit buffer. If the value of DIGLEN is greater than or equal to the number of digits of the digit source, then all the digits retrieved are added to the digit buffer.

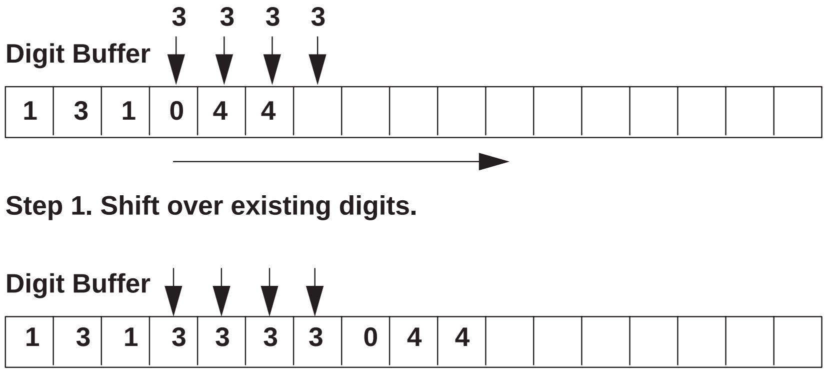
If the value of the DIGLEN field is less than the number of available digits of the digit source, then only the first DIGLEN number of digits are added to the digit buffer.

Digits identified by the RETRIEVE collectable are inserted into the digit buffer before digits at the identified location. The existing digits are shifted and not overwritten.

Figures 16-1 and 16-2 contain RETRIEVE applications.

Figure 16-1
RETRIEVE application

```
> RETRIEVE CIC 4 POS 4
```



Step 1. Shift over existing digits.

Step 2. Insert digits.

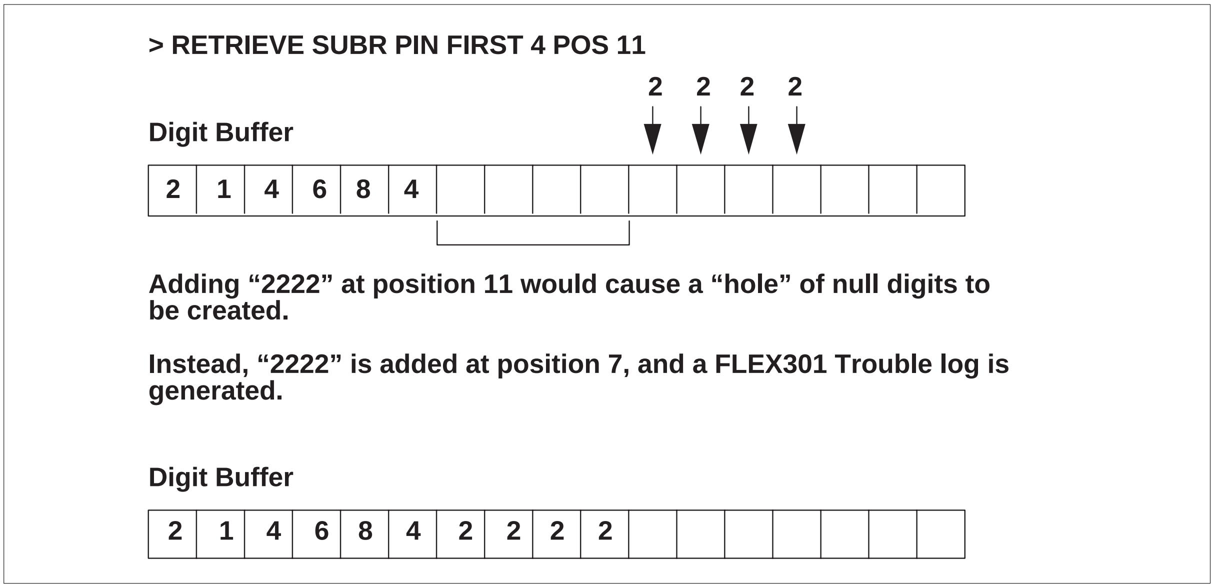
The RETRIEVE operation may not cause holes to occur within the digit buffer[1]. If such an attempt is made, then the digits are added at the closest possible location, and a FLEX 301 trouble log is generated with the following trouble code value:

```
Digits force appended to buffer
```

The flextype field in the log is set to “RETRIEVE”, and the digits added to the buffer by the RETRIEVE collectable are included in the report.

[1] This could only occur through use of the POS buffer location value.

Figure 16-2
RETRIEVE application



> RETRIEVE SUBR PIN FIRST 4 POS 11

Digit Buffer

2	1	4	6	8	4											

Adding “2222” at position 11 would cause a “hole” of null digits to be created.

Instead, “2222” is added at position 7, and a FLEX301 Trouble log is generated.

Digit Buffer

2	1	4	6	8	4	2	2	2	2							

Usecase Scenarios

The RETRIEVE collectable provides a generic mechanism for FlexDial call processing whereby digit manipulation may occur by retrieving digits previously processed for the call and inserting them into the digit buffer.

Interactions

The RETRIEVE collectable requires that the identified digits be previously processed by an ADDR, SUBR, CIC, or OLI digit collectable.

Restrictions and limitations

The following call processing restrictions or limitations exist for the RETRIEVE collectable:

- The RETRIEVE collectable does not request any digits from the originating agent, but works with what has already been processed for the call.
- If the requested digits to retrieve haven’t been processed for the call, then no digits are added to the digit buffer, and processing continues with the next collectable in the list.
- Correct use of the RETRIEVE collectable requires intimate knowledge of the defined interaction with the originating agent.

- If an attempt is made to add digits to the digit buffer which exceeds the 64 digit capacity of the buffer, then the excess digits either being appended to the buffer or pushed out of the buffer due to the insert action of the RETRIEVE collectable are discarded and not added to the buffer. In addition, a FLEX 301 trouble log is generated with the following trouble code value:

```
Digits truncated from buffer
```

The flextype field in the log is set to “RETRIEVE” and the digits truncated from the buffer are included in the digits field of the report.

TIMER Collectable

The TIMER collectable provides an additional triggering mechanism to initiate FlexDial collectable execution for the originating agent. The triggering mechanism is based upon the expiration of the identified FlexDial timer, initiating execution of the collectable list identified by the FLEXDIAL table index.

Call Processing Application

The TIMER collectable activates one of three available FlexDial timers for service processing use, where the expiration of the timer results in the execution of the collectable list for the identified FLEXDIAL table index. This collectable is provisioned with three key elements of information needed for processing:

- The Timer ID
- The duration of the timer
- The FLEXDIAL table index identifying the collectable list to execute.

Three independent timers are available for activation, generically named TIMER1, TIMER2, and TIMER3. If an attempt is made to activate a timer that is already currently active, then the previous activation is cancelled, and the timer is reactivated using the new parameters. All active timers are automatically cancelled when disconnect occurs and the call is completed.

The duration of the FlexDial timer may be identified through either a defined integer value or through a FlexDial variable. For FEATVAR FLEXFEAT variable use, the variable must be activated prior to its use by the TIMER collectable. If the desired FEATVAR variable is not activated prior to its use by the TIMER collectable, then a FLEX 307 log is generated with a trouble reason of “Uninitialized variable use”, and the timer is not activated.

If a value of zero (0) is specified for the timer duration, then the timer does not get activated, and the net result is to cancel a potentially active FlexDial timer identified by the timer ID.

The FLEXDIAL table index specified identifies the collectable list that is to be executed upon expiration of the FlexDial timer. If the FLEXDIAL table index used is the “NIL” index, then the FlexDial timer does not get activated, and the net result is to cancel a potentially active FlexDial timer identified by the timer ID.

While a FlexDial timer may only be activated through FlexDial processing, expiration of the timer may occur in one of a number of call states:

- During the interaction with the originating agent.
- After call setup, prior to answer, or when connected to a tone or announcement.
- During the talking state of the call.

When the timer expires during the interaction with the originating agent, the REPLACE action is used to replace the remainder of the current collectable list or sub-list with the list identified by the TIMER collectable FLEXDIAL table index.

When the timer expires after call setup prior to answer, then the connection to the terminating agent is broken and the terminating agent is released from the call and idled. After completion of FlexDial collectable processing, the connection to the terminating agent may be re-established through the normal channels of translations and routing.

When the timer expires during the talking state of the call, then the connection to the terminating agent is broken. However, the terminating agent is simply put on hold and remains allocated to the call. For the originating agent, STR reorigination capabilities are disabled during FlexDial collectable processing. After completion of collectable processing, the connection to the terminating agent is re-established, the bearer channel link between the two agents is enabled, and STR reorigination is re-enabled for the talking state of the call.

For each case of timer expiration collectable execution, full bearer channel capabilities including the abilities to play inband tones and announcements as well as request and collect inband digits are available.

FlexDial timers may be cancelled through use of the KILLTMR collectable.

For reset actions, the TIMER collectable cancels the timer activated. However, timers cancelled due to the execution of the TIMER collectable are not restored.

Usecase Scenarios

The TIMER collectable provides a new mechanism for triggering FlexDial call processing after call setup, before and after answer has occurred for the call. Collectable processing through timer expiry has the added benefit of full bearer channel capability.

Interactions

FlexDial timers enabled through the TIMER collectable may be disabled before timer expiration through use of the KILLTMR collectable.

A FlexDial timer enabled through the TIMER collectable may expire and trigger FlexDial collectable processing during the interaction with the originating agent, after call setup and before answer, or after answer during the talking state of the call.

The TIMER collectable remains active and utilizes real-time time measurements when used with the FLEXSIM FlexDial call simulator tool.

Restrictions and limitations

The following call processing restrictions or limitations exist for the TIMER collectable:

- Use of a timer ID for a timer already active disables the currently active timer before the new one is activated.
- If the identified duration for the timer is zero, then the FlexDial timer is not enabled.
- If the identified FLEXDIAL table index is “NIL”, then the FlexDial timer is not enabled.
- If the TIMER collectable specifies use of a FEATVAR variable which has not been activated for the call, then a FLEX 307 log is generated and the identified timer is not activated for the call.

KILLTMR Collectable

The KILLTMR (kill timer) collectable provides the ability to disable a FlexDial timer activated through execution of the TIMER collectable.

Call Processing Application.

The KILLTMR collectable deactivates one of three available FlexDial timers, identifying the specific timer which is disabled.

If the identified timer is not currently active, then no action is performed by the KILLTMR collectable.

The KILLTMR collectable does not perform any reset action, and does not restore the FlexDial timer cancelled.

Usecase Scenarios

The KILLTMR collectable provides an offsetting mechanism to the TIMER collectable, enabling the ability to cancel an active FlexDial timer.

Interactions

FlexDial timers must be enabled through use of the TIMER collectable.

Restrictions and limitations

The following call processing restrictions or limitations exist for the KILLTMR collectable:

- No reset action is performed for the KILLTMR collectable, in particular, the cancelled timer is not restored through KILLTMR reset processing.

CALLCOND Collectable

The CALLCOND (call condition) collectable provides an additional triggering mechanism to initiate FlexDial collectable execution for the originating agent. The triggering mechanism is based upon occurrence of the specified call condition for the call, initiating execution of the collectable list identified by the FLEXDIAL table index.

Currently, only two call conditions are supported by the CALLCOND collectable, answer and disconnect.

Call Processing Application

The CALLCOND collectable queues a trigger against the specified call condition for service processing use, where the execution of the call event results in the execution of the collectable list for the identified FLEXDIAL table index. This collectable is provisioned with two key elements of information needed for processing:

- The call condition to trigger upon.
- The FLEXDIAL table index identifying the collectable list to execute.

The supported call conditions are answer and disconnect.

For the answer call event, FlexDial processing occurs after all other answer processing has occurred, including CAIN O_ANSWER TDP processing. Once FlexDial collectable processing has completed, the call is returned to the talking state.

For the disconnect call event, FlexDial processing occurs prior to completing the disconnect actions. Once FlexDial collectable processing has completed, the disconnection actions are resumed and call processing resources are deallocated.

For all applications of call condition execution, bearer channel capabilities including the abilities to play inband tones and announcements as well as request and collect inband digits are not available. If an attempt is made to play a tone or announcement, modify digit collection request parameters, or request inband digits from the originating agent is made, then a FLEX 303 log is generated with a trouble reason of “Cannot perform INBAND prompt or collection activities” and FlexDial collectable processing is suspended.

The FLEXDIAL table index specified identifies the collectable list that is to be executed when the call condition event is reached for the call. If the FLEXDIAL table index used is the “NIL” index, then the call condition does not get activated, and the condition is disabled for the call.

Multiple call condition collectables may be executed throughout the call, with each execution overwriting prior values set. The call condition collectable must be executed prior to handling of the call event, or no application of FlexDial collectable processing can occur. For instance, the CALLCOND collectable specifying the ANSWER condition must be executed prior to receipt of answer for the call in order to trigger FlexDial collectable processing. If the FlexDial call condition is enabled after answer occurs, then the identified FlexDial collectable processing can never be triggered for execution.

Usecase Scenarios

The CALLCOND collectable provides a new mechanism for triggering FlexDial call processing for specific points in the call processing call model.

Interactions

FlexDial call conditions enabled through the CALLCOND collectable may be disabled through use of the “NIL” FLEXDIAL table index in the collectable provisioning.

Inband interaction with the originating agent is not permitted for CALLCOND FlexDial collectable processing, and if attempts, results in a FLEX 303 log and suspension of further collectable processing for that trigger.

Restrictions and limitations

The following call processing restrictions or limitations exist for the CALLCOND collectable:

- Use of a call condition that is already active overwrites the FLEXDIAL table index to use when the condition is triggered. Specifying a FLEXDIAL index of “NIL” clears the call condition trigger for the call.
- Inband interaction with the originating agent is not permitted for CALLCOND FlexDial collectable processing, and if attempts, results in a FLEX 303 log and suspension of further collectable processing for that trigger.

Apply Treatment collectable (APTRMT)

The Apply Treatment (APTRMT) collectable is a special case of the TERMINATE to destination capability where the call is terminated to treatment if one has been specified for the call. If a treatment has not been specified, then the APTRMT collectable performs no call processing function and the next collectable is executed.

Treatments set through failed validation attempts of digit collectable processing are not immediately processed so that the required interaction with the originating agent can be completed. APTRMT provides a decision point capability where collectable processing is halted and the call is terminated to the destination specified by treatment decoding.

Application

When the APTRMT collectable is executed and a treatment has been set for the call by digit collectable processing, remaining collectable processing is suspended, and the call is connected to the destination specified by treatment decoding. Collectable processing is not resumed.

When the APTRMT collectable is executed and a treatment has not been set for the call, then no call processing associated with APTRMT occurs, and the next collectable is processed.

Use case scenario

The APTRMT collectable is used in the following call scenario:

- apply call treatment before processing additional collectables.

  It may be desirable to apply an identified treatment for a call instead of continuing digit collectable processing, in particular if the additional digits collected cannot reverse the application of treatment for the call. This enables call resources to be released and made available for other calls.

Restrictions and limitations

If a treatment has not been identified for the call when the APTRMT Collectable is executed, then no call processing action is performed by APTRMT and the next collectable is immediately processed.

Branch Execute Without Return collectable (GOTO)

The Branch Execute Without Return (GOTO) collectable provides the ability to identify additional collectables for execution. The FLEXDIAL table index provisioned identifies a list of collectables for additional processing.

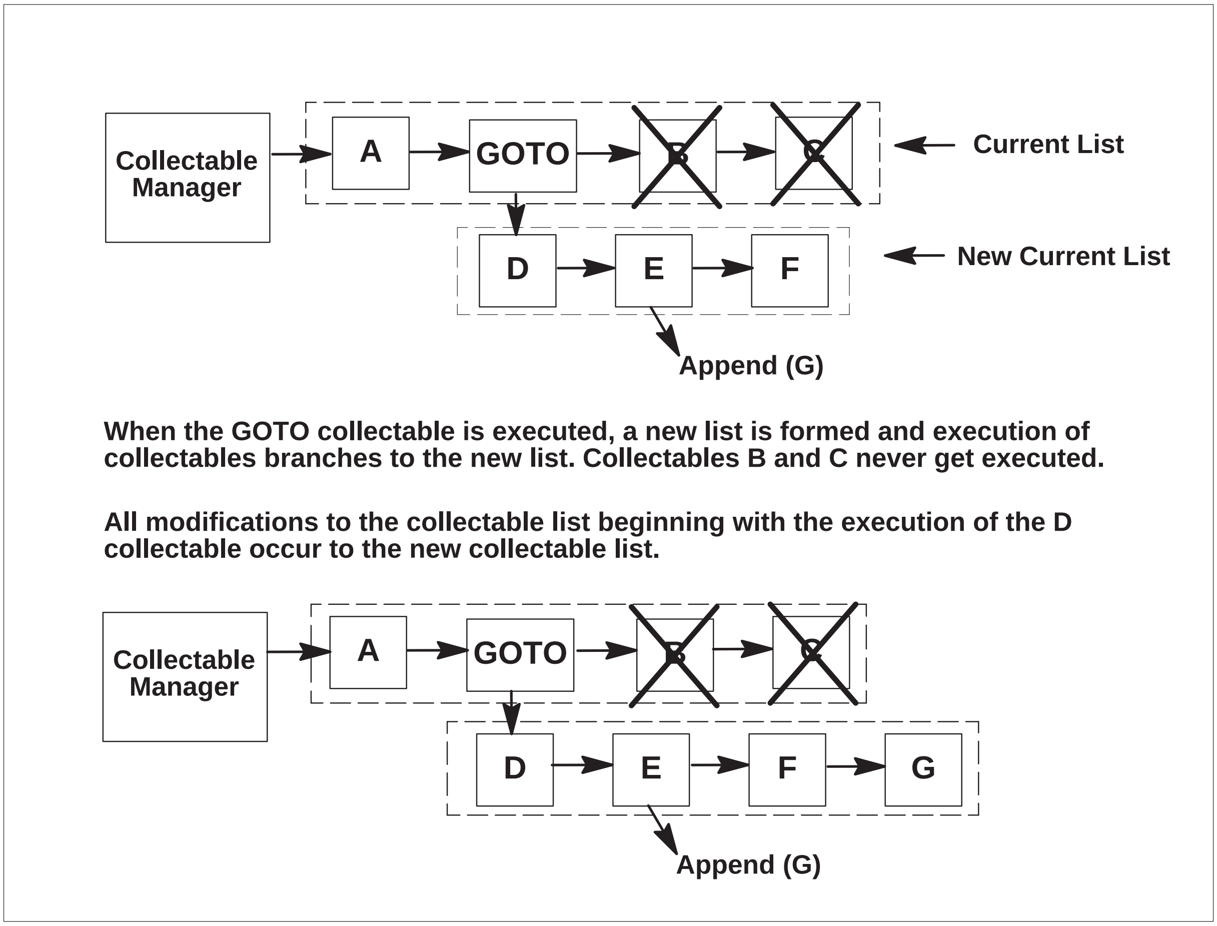
Application

The GOTO execution is similar to a JUMP instruction. Processing “jumps” to the list of collectables identified by the FLEXDIAL index.

Execution of the GOTO collectable results in modifications to the list of remaining unprocessed collectables. When GOTO is executed, remaining collectables in the current FlexDial list being processed are ignored, and the collectables identified by the FLEXDIAL table index are queued for processing.

Figure 16-3 contains a GOTO collectable example.

Figure 16-3
GOTO collectable example



When the GOTO collectable is executed, a new list is formed and execution of collectables branches to the new list. Collectables B and C never get executed.

All modifications to the collectable list beginning with the execution of the D collectable occur to the new collectable list.

Call processing can execute an unlimited number of GOTO collectables.

Use case scenario

The GOTO collectable extends processing of collectables to an additional FLEXDIAL index. The GOTO collectable functions in a similar manner to the REPLACE FLEXDIAL action.

Restrictions and limitations

Remaining collectables in the current list are never executed after a GOTO operation is performed.

Branch Execute With Return collectable (DO)

The Branch Execute With Return (DO) collectable provides the ability to identify additional collectables for execution. The FLEXDIAL table index provisioned identifies a list of collectables for additional processing.

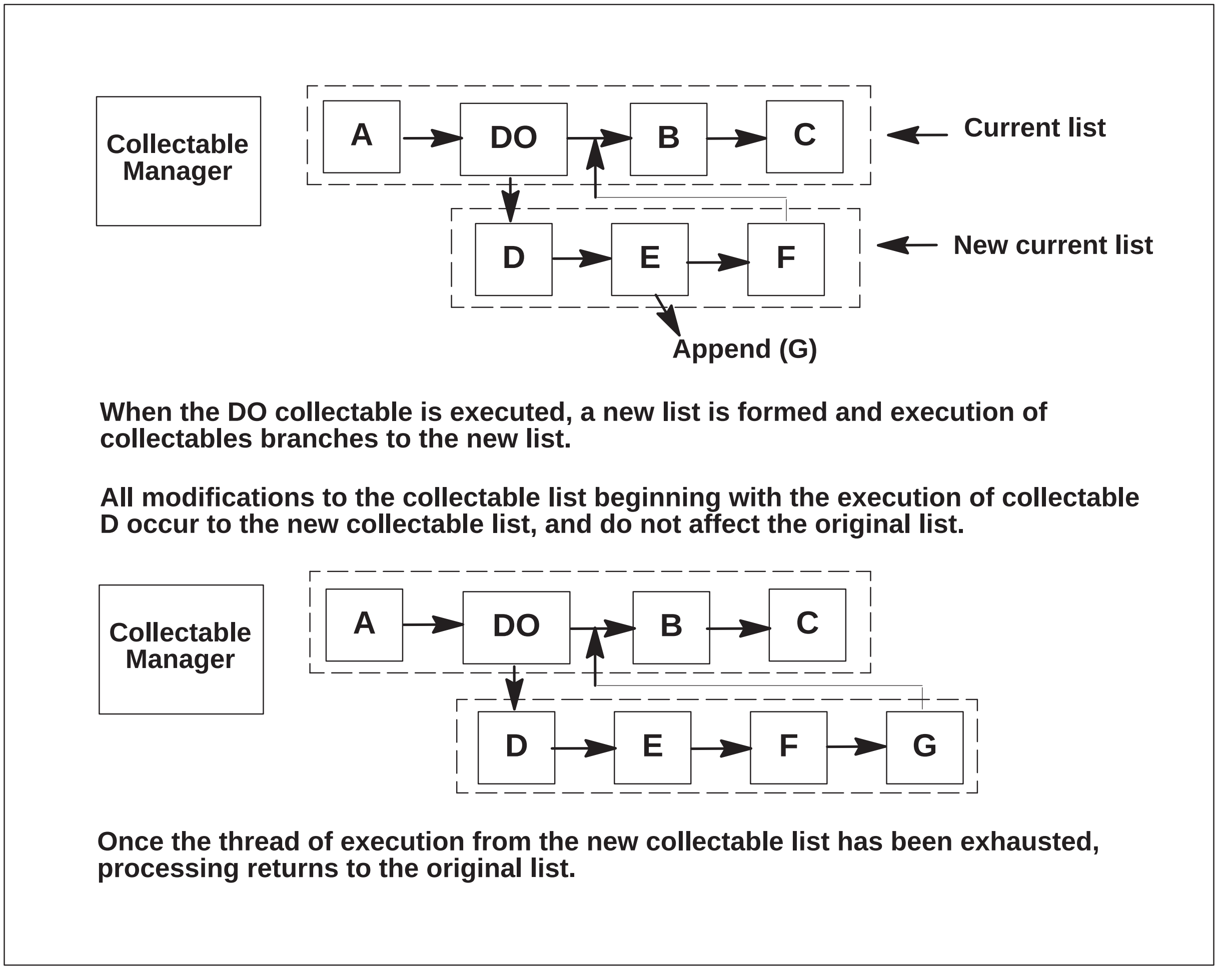
Application

The DO execution is similar to a sub-routine call. Processing “jumps” to the list of collectables identified by the FLEXDIAL index. Once the thread of execution identified by the list has been completed, processing “returns” to the collectable following the DO instruction in the original list.

Execution of the DO collectable results in modifications to the list of remaining unprocessed collectables. When DO is executed, the collectables identified by the FLEXDIAL table index are inserted into the list and queued for processing.

Figure 16-4 contains a DO collectable example.

Figure 16-4
DO collectable example



When the DO collectable is executed, a new list is formed and execution of collectables branches to the new list.

All modifications to the collectable list beginning with the execution of collectable D occur to the new collectable list, and do not affect the original list.

Once the thread of execution from the new collectable list has been exhausted, processing returns to the original list.

Call processing can execute an unlimited number of DO collectables.

Use case scenario

The DO collectable groups related collectable processing for provisioning purposes. The DO collectable functions in a similar manner to the EXEC FLEXDIAL action.

Restrictions and limitations

There are no restrictions or limitations associated with the DO collectable.

Include List collectable (INCLUDE)

The Include List (INCLUDE) collectable provides the ability to identify additional collectables for execution. The FLEXDIAL table index provisioned identifies a list of collectables for additional processing.

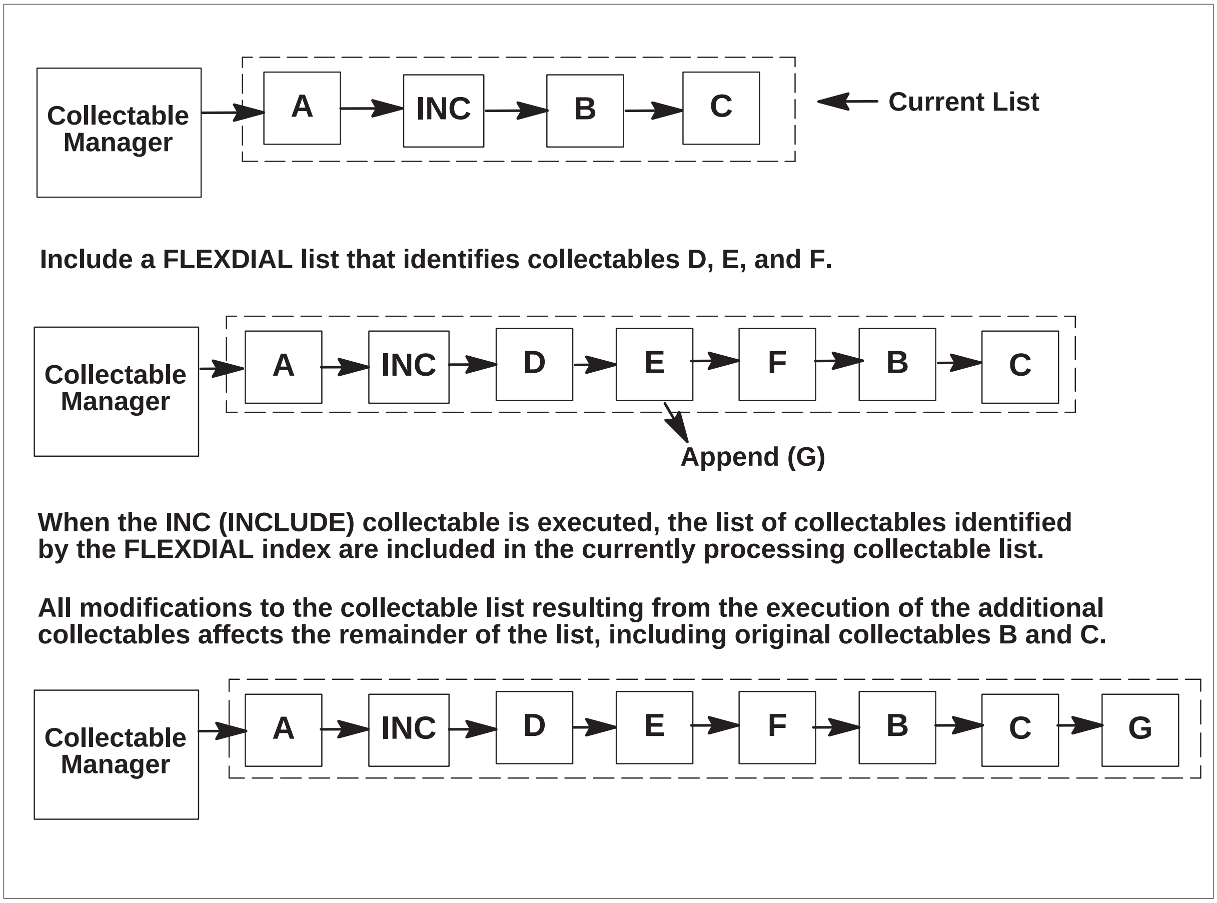
Application

The INCLUDE collectable execution inserts the identified list of collectables into the currently processing collectable list. The list of collectables identified by the FLEXDIAL table index is inserted before the next collectable to be processed.

This differs from the DO collectable in that modifications to the collectable list do not affect a separate collectable list, but affect the original list.

Figure 16-5 contains an INCLUDE collectable example.

Figure 16-5
INCLUDE collectable example



Include a FLEXDIAL list that identifies collectables D, E, and F.

When the INC (INCLUDE) collectable is executed, the list of collectables identified by the FLEXDIAL index are included in the currently processing collectable list.

All modifications to the collectable list resulting from the execution of the additional collectables affects the remainder of the list, including original collectables B and C.

Call processing can execute an unlimited number of INCLUDE collectables.

Use case scenario

The INCLUDE collectable groups related collectable processing for provisioning purposes. The INCLUDE collectable functions in a similar manner to the INSERT FLEXDIAL action.

Restrictions and limitations

There are no restrictions or limitations associated with the INCLUDE collectable.

Identify Destination collectable (ROUTE)

The Identify Destination (ROUTE) collectable specifies the destination for the call, superceding a route determined through pre-translation validation of digit collectables, normal translations of the received address digits, or other ROUTE collectables.

Application

Execution of the ROUTE collectable results in the route being assigned for the call. After the provisioned route is assigned, the remaining collectables are executed. Once collectable processing is complete, the call is connected to the identified terminating agent.

The ROUTE collectable can be executed multiple times in the call, where the provisioned destination overrides the destination specified by a previously executed ROUTE collectable.

Use case scenario

The ROUTE collectable is used in the following call scenario:

- associate route with subscriber number

  In unique situations where the destination of the call is specified by the calling party subscriber number received, the ROUTE collectable can be used to provide this capability. The FLEXFEAT table DPIDX option can identify a FLEXDIAL table index that contains the route for the call.

Restrictions and limitations

The following restrictions and limitations apply to the ROUTE collectable:

- The destination identified by the ROUTE collectable overrides a destination set by pre-translation validation of digit collectables, normal translations of the received address digits, or other ROUTE collectables.
- After the ROUTE collectable is executed, remaining collectables are executed.

- After collectable processing is complete, the call is connected to the terminating agent specified by the ROUTE collectable.
- The route set by the ROUTE collectable does not override the effect of a treatment set for the call. A set treatment always take precedence over a route set for a call.

No Operation collectable (NOOP)

The No Operation (NOOP) collectable is simply a provisioning placeholder for FLEXDIAL table provisioning.

Application

If executed, the NOOP collectable does not perform any call processing related function, and the next collectable to be processed is executed.

Use case scenario

This collectable is not used for any call processing related function.

Restrictions and limitations

The NOOP collectable does not perform any call processing related function.

Treatment Match Conditional Branch collectable (IFTRMT)

The Treatment Match Conditional Branch (IFTRMT) collectable allows dynamic branching of collectable processing, based on the current treatment setting for the call.

Application

The IFTRMT collectable compares the current treatment set for the call with a vector of up to four treatment definitions. This collectable is provisioned with the following information needed for processing:

- the method of comparison to use
- a vector of up to four treatments to search for
- the FLEXDIAL IFTRUE and IFFALSE indices to use
- the FLEXDIAL action to take

For the ”IS” comparison method, if the comparison is successful (just one of the identified treatments matches the current treatment set for the call), then the IFTRUE index is selected. Otherwise, the IFFALSE index is selected.

For the ”NOT” comparison method, if none of the datafilled treatments match the current treatment set for the call, then the IFTRUE index is selected. Otherwise, the IFFALSE index is selected.

The FLEXDIAL action taken by the collectable (INSERT, APPEND, REPLACE or EXEC) is applied using the FLEXDIAL index supplied by the indicated IFTRUE or IFFALSE index, according to the comparison results. If the NIL index is indicated, then the collectable takes no action.

The IFTRMT collectable may be executed multiple times during the processing of a call.

Use case scenario

This collectable is not currently used in any call scenario; however, an example scenario is provided:

- route a failed authcode entry to an operator after collecting address information from the subscriber on an FGD cut-thru call

  The IFTRMT collectable would be placed in the collectable execution chain after collection of the subscriber authcode and called address. The IFTRMT collectable would search for INAU treatment and, if matched, would replace the remaining collectables in the list with an index that routes to an operator position and ends with a TERMINATE STOP collectable. The operator could then interact with the subscriber to determine how to process the call.

Restrictions and limitations

There are no itemized restrictions specific to the execution of IFTRMT collectable in FLEXDIAL call processing.

Match Prompt Type Conditional Branch collectable (IFPRMT)

The Match Prompt Type Conditional Branch (IFPRMT) collectable allows dynamic branching of collectable processing, based on the most recently played subscriber prompt.

Application

The IFPRMT collectable compares the most recently played prompt with a vector of prompt types. This collectable is provisioned with the following information needed for processing:

- the method of comparison to use
- a vector of up to four prompts to search for
- the FLEXDIAL IFTRUE and IFFALSE indices to use
- the FLEXDIAL action to take

For the ”IS” comparison method, if the comparison is successful (just one of the identified prompts matches the last prompt played), then the IFTRUE index is selected. Otherwise, the IFFALSE index is selected.

For the ”NOT” comparison method, if none of the identified prompts match the most recent prompt, then the IFTRUE index is selected. Otherwise, the IFFALSE index is selected.

The FLEXDIAL action taken by the collectable (INSERT, APPEND, REPLACE or EXEC) is applied using the FLEXDIAL index supplied by the indicated IFTRUE or IFFALSE index, according to the comparison results. If the NIL index is indicated, then the collectable takes no action.

The IFPRMT collectable may be executed multiple times during the processing of a call.

Use case scenario

This collectable is not currently used in any call scenario; however, an example scenario is provided:

- play a prompt if the previous prompt was a voice announcement

  If you are collecting a PIN code and use a voice prompt to indicate to users to enter their PIN digits, you may select to apply, on validation failure, a tone prompt for all attempts following the first. You would then use the IFPRMT collectable to execute a FLEXDIAL index that sends a prompt message to a SUBR PIN collectable, datafilled with an announcement prompt. The message would indicate that a standard tone prompt be used instead of replaying the announcement.

Restrictions and limitations

There are no itemized restrictions specific to the execution of IFPRMT collectable in FLEXDIAL call processing.

Time-of-Day Comparison Conditional Branch collectable (IFTOD)

The Time-of-Day Comparison Conditional Branch (IFTOD) collectable allows dynamic branching of collectable processing, based on the current time of day.

Application

The IFTOD collectable compares the current time of day from the switch’s clock with datafill in the collectable, and performs a FLEXDIAL action accordingly. This collectable is provisioned with the following information needed for processing:

- the method of comparison to use
- a vector of up to four time intervals to search for
- the FLEXDIAL IFTRUE and IFFALSE indices to use
- the FLEXDIAL action to take

For the ”IS” comparison method, if the comparison is successful (the current time lies within one of the datafilled time intervals), then the IFTRUE index is selected. Otherwise, the IFFALSE index is selected.

For the ”NOT” comparison method, if the current time matches none of the datafilled time intervals, then the IFTRUE index is selected. Otherwise, the IFFALSE index is selected.

The FLEXDIAL action taken by the collectable (INSERT, APPEND, REPLACE or EXEC) is applied using the FLEXDIAL index supplied by the indicated IFTRUE or IFFALSE index, according to the comparison results. If the NIL index is indicated, then the collectable takes no action.

The IFTOD collectable may be executed multiple times during the processing of a call.

Use case scenario

This collectable is not currently used in any call scenario; however, some example scenarios are provided:

- translate calls to a different STS at different times of day

  After collection of subscriber information, an IFTOD collectable could be used to route calls to translations in one STS from 8 a.m. to 5 p.m., while translating these calls using a different STS at all other times. This would be accomplished by conditional execution of a CALLTYPE collectable used to apply a feature set that modifies the current STS for the call.

- route calls to a different CIC depending on the current time-of-day on SS7 FGD agents

  You could apply IFTOD to file CIC digits that override the CIC digit parameters in an IAM, and thus route subscribers to different carriers, depending on when the call originated.

Restrictions and limitations

There are no itemized restrictions specific to the execution of IFTOD collectable in FLEXDIAL call processing.

Match Digits Conditional Branch collectable (IFDIGS)

The Match Digits Conditional Branch (IFDIGS) collectables allows dynamic branching of collectable processing, based on a comparison of actual digits received (digits stored in the digit buffer) to digits identified.

Application

The IFDIGS collectable does not request digits from the originating agent, but bases comparisons against what is currently in the digit buffer.

Note: In order to collect the proper digits to perform the comparison, the COLDIG or COLPARM collectable must be used first.

The collectable is provisioned with the following information needed for processing:

- the method of comparison to use
- the digits or digit patterns to use for the comparison. Up to four patterns can be specified. The number of digits required for the comparison is derived from this field.
- the FLEXDIAL IFTRUE and IFFALSE indices
- the FLEXDIAL action to take

For the “ARE” comparison method, if the first group of digits in the digit buffer match any of the digit patterns identified by the DIGITS vector, then the datafilled IFTRUE index is selected by the collectable to modify the current collectable list. Otherwise, the IFFALSE index is selected.

For the “NOT” comparison method, if the first group of digits in the digit buffer do not match any of the digit patterns identified by the DIGITS vector, then the IFTRUE index is used by the collectable. Otherwise, the IFFALSE index is used.

The FLEXDIAL action taken by the collectable (INSERT, APPEND, REPLACE, or EXEC) is applied using the FLEXDIAL index supplied by the indicated IFTRUE or IFFALSE index according to the results of the comparison. If the NIL index is indicated, then the collectable takes no action.

If there are not enough digits in the digit buffer to successfully perform the comparison, then the comparison automatically fails and the IFFALSE index is selected.

The IFDIGS collectable does not remove digits from the digit buffer for the comparison. The received digits remain in the digit buffer for other comparisons or for digit collectable or digit manipulation processing.

The IFDIGS collectable can be executed multiple times during the processing of a call.

Use case scenario

The collectable is used in the following call scenario:

- international FGD 1NX versus information digit comparison

  The IFDIGS collectable distinguishes between international 1NX digits received and ANI information digits, and processing branches accordingly to handle the 1NX international stream.

- dynamic execution of collectables based on received digits

  With the IFDIGS collectable, unique digit streams received can be specifically handled in the FlexDial framework.

Restrictions and limitations

The IFDIGS collectable does not request digits from the originating agent, but bases the comparison against digits currently in the digit buffer. If there are not enough digits in the digit buffer for the comparison, then the comparison result is unsuccessful.

Match Digit Count Conditional Branch collectable (IFCNT)

The Match Digit Count Conditional Branch (IFCNT) collectable allows dynamic branching of collectable processing, based on receipt of a specified number of digits (the number of digits currently in the digit buffer).

Application

The IFCNT collectable does not request digits from the originating agent, but bases the comparison against what is currently in the digit buffer. This collectable is provisioned with the following information needed for processing:

- the method of comparison to use
- the minimum count value to use for the comparison
- the maximum count value to use for the comparison
- the FLEXDIAL IFTRUE and IFFALSE indices to use
- the FLEXDIAL action to take

For the ”IS” comparison method, if the comparison is successful (the number of digits within the incoming buffer is within the range specified by the MIN and MAX fields), then the IFTRUE index is selected by the collectable. Otherwise, the IFFALSE index is selected.

For the ”NOT” comparison method, if the digit count does not match the number of digits in the digit buffer, the IFTRUE index is selected. Otherwise, the IFFALSE index is selected.

The FLEXDIAL action taken by the collectable (INSERT, APPEND, REPLACE, or EXEC) is applied using the FLEXDIAL index supplied by the indicated IFTRUE or IFFALSE index, according to the comparison results. If the NIL index is indicated, then the collectable takes no action.

The IFCNT collectable does not remove digits from the digit buffer for the comparison. The received digits remain in the digit buffer for other comparisons or for digit collectable or digit manipulation processing.

The IFCNT collectable may be executed multiple times during the processing of a call.

Use case scenario

The collectable is used in the following call scenario:

- speed dial number versus normal address digit processing

  With the IFCNT collectable, the distinction of speed dial numbers versus normal address digits can be made. A speed dial number branching scenario would branch to identify a unique pretranslator name to use for speed dial number screening.

- dynamic execution of collectables based on digit count

  With the IFCNT collectable, unique digit counts received can be specifically handled in the FlexDial framework.

Restrictions and limitations

The IFCNT collectable does not request digits from the originating agent, but bases the comparison against digits currently in the digit buffer.

Match NOA Conditional Branch collectable (IFNOA)

The Match NOA (nature of address) Conditional Branch (IFNOA) collectable allows dynamic branching of collectable processing, based on a comparison of the NOA of the calling or called party numbers.

Application

The IFNOA collectable does not request digits from the originating agent, but bases the comparison against the NOA values currently recorded for the calling and called party numbers.

Note: See section ”Setting calling party NOA” on page 16-86 for a description of how call processing sets the nature of address values for the calling and called party numbers.

The collectable is provisioned with the following information needed for processing:

- an indicator identifying which NOA value to use for the comparison (either called, calling, or dialed)

  Note: The DIALED nature of address is defined as the first called party number NOA set for the call (similar to dialed number digits). Typically the dialed number NOA value is set simultaneously with the dialed number digits for the call, however situations exist (such as for the “no number cut-thru” value) where the dialed NOA may be specified without the presence of dialed number digits. The IFNOA collectable using the DIALED party compares the dialed number nature of address value set for the call with the identified values. If the dialed number nature of address value has not yet been identified for the call, then the value of “UNKNOWN” is used as the dialed number nature of address for the call.

- the method of comparison to use
- a vector of up to four called or calling number NOA values to use for the comparison
- the FLEXDIAL IFTRUE and IFFALSE indices to use
- the FLEXDIAL action to take

For the “IS” comparison method, if the NOA value received matches any of the NOA values specified by the NATOFADD vector, then the IFTRUE index is selected. Otherwise, the IFFALSE index is selected.

For the “NOT” comparison method, if the NOA value received does not match any of the NOA values specified by the NATOFADD vector, then the IFTRUE index is selected. Otherwise, the IFFALSE index is selected.

The FLEXDIAL action taken by the collectable (INSERT, APPEND, REPLACE, or EXEC) is applied using the FLEXDIAL index supplied by the indicated IFTRUE or IFFALSE index, according to the comparison results. If the NIL index is indicated, then the collectable takes no action.

The IFNOA collectable does not reset or modify the NOA values once the comparison is performed.

The IFNOA collectable can be executed multiple times during the processing of a call.

Use case scenario

The collectable is used in the following call scenario:

- detection of cut-through and transitional FGD type calls

  Using the IFNOA collectable, the detection of a cut-through or transitional call can be made based on the called party number NOA. The cut-through or transitional call can then be blocked or handled based on FlexDial provisioning.

- dynamic execution of collectables based on received NOA

  With the IFNOA collectable, unique NOA values received can be specifically handled in the FlexDial framework.

Dialed number NOA comparison ability to the IFNOA collectable provides the ability to specifically handle unique dialed number NOA scenarios in the case where the dialed number NOA value is not equivalent to the called party number NOA value.

Restrictions and limitations

The IFNOA collectable does not request digits from the originating agent, but bases the NOA comparison against the values currently recorded for the calling and called party numbers.

Match Parameter Name Conditional Branch collectable (IFPARM)

The Match Parameter Name Conditional Branch (IFPARM) collectable allows dynamic branching of collectable processing, based on receipt of specified parameters in the incoming out-of-band message (the IAM message for CCS7 agents).

Application

The IFPARM collectable does not request or modify an out-of-band message from the originating agent, but bases the comparison against the currently available out-of-band message. This collectable is provisioned with the following information needed for processing:

- the method of comparison to use
- a vector of up to four parameters to search for
- the FLEXDIAL IFTRUE and IFFALSE indices to use
- the FLEXDIAL action to take

For the ”IS” comparison method, if the comparison is successful (just one of the identified parameters is available in the incoming out-of-band message), then the IFTRUE index is selected. Otherwise, the IFFALSE index is selected.

For the ”NOT” comparison method, if none of the parameters match any of those identified in the incoming message, then the IFTRUE index is selected. Otherwise, the IFFALSE index is selected.

The FLEXDIAL action taken by the collectable (INSERT, APPEND, REPLACE, or EXEC) is applied using the FLEXDIAL index supplied by the indicated IFTRUE or IFFALSE index, according to the comparison results. If the NIL index is indicated, then the collectable takes no action.

The IFPARM collectable may be executed multiple times during the processing of a call.

Use case scenario

The collectable is used in the following call scenario:

- verify existence of optional parameters before processing the parameter

  For CCS7 processing, certain parameters are only processed if they are present. If the parameter is not present, processing simply continues down another path.

- dynamic execution of collectables based on parameter existence

  With the IFPARM collectable, unique parameters received can be specifically handled in the FlexDial framework.

Restrictions and limitations

The IFPARM collectable does not request or modify out-of-band messages from the originating agent, but bases the comparison against an available out-of-band message. If an out-of-band message is not available for the comparison, the IFPARM collectable performs no action, and processing continues with the next collectable.

Delete Digits collectable (DELDIGS)

The Delete Digits (DELDIGS) collectable provides the ability to manipulate the digits received by removing digits from the digit buffer before they can be processed by the remaining digit or sequence collectables.

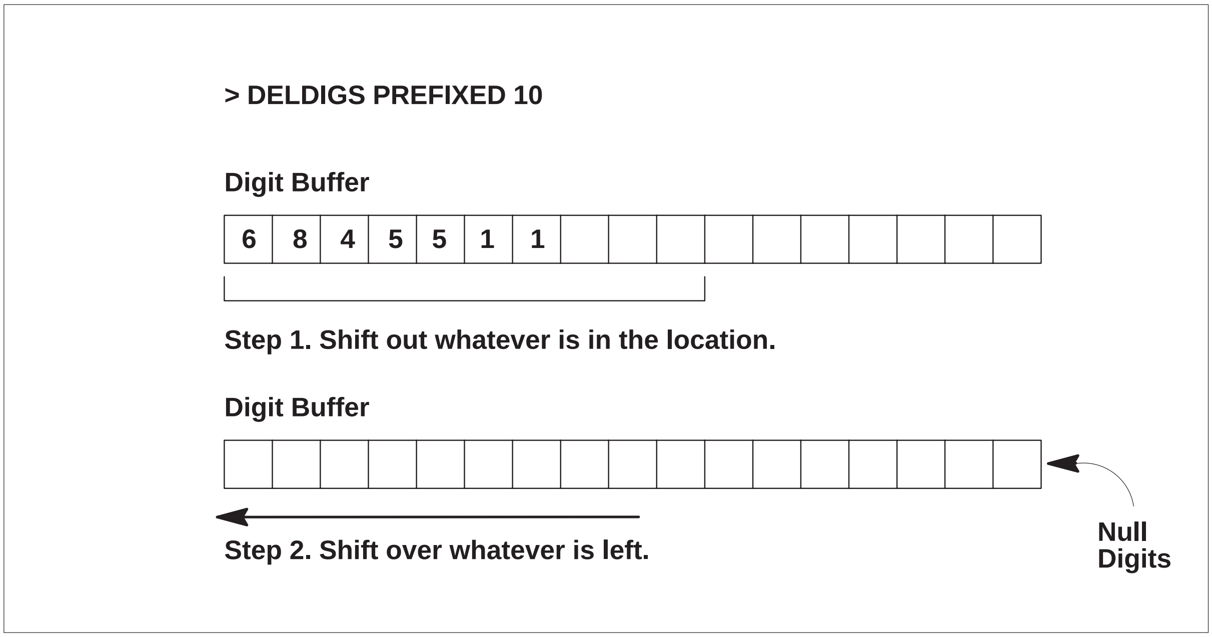
Application

The DELDIGS collectable does not request digits from the originating agent, but uses digits currently in the digit buffer.

The DELDIGS operation removes digits from the digit buffer so that they cannot be processed by the call. There are no error scenarios associated with the DELDIGS operation.

Figure 16-6 contains a DELDIGS application.

Figure 16-6
DELDIGS application


> DELDIGS PREFIXED 10
Digit Buffer
6
8
4
5
5
1
1
Step 1. Shift out whatever is in the location.
Digit Buffer
Step 2. Shift over whatever is left.
Null Digits

Use case scenario

The collectable is used in the following call scenario:

- digit manipulation on incoming digits

With the DELDIGS collectable, received digit information can be altered before being processed for the call.

Restrictions and limitations

The following restrictions and limitations apply to the DELDIGS collectable:

- The DELDIGS collectable does not request any digits from the originating agent, but uses digits currently in the digit buffer.
- Correct use of the DELDIGS collectable requires in depth knowledge of the defined interaction with the originating agent.
- If an attempt is performed to delete more digits than there are available in the digit buffer, then all available digits are removed from the buffer and the call continues, without the generation of any error or exception case.

Digit Manipulation collectable (ADDDIGS)

The Add Digits (ADDDIGS) collectable provides the ability to manipulate the digits received by adding digits to the digit buffer so they can be processed by digit or other sequence collectables.

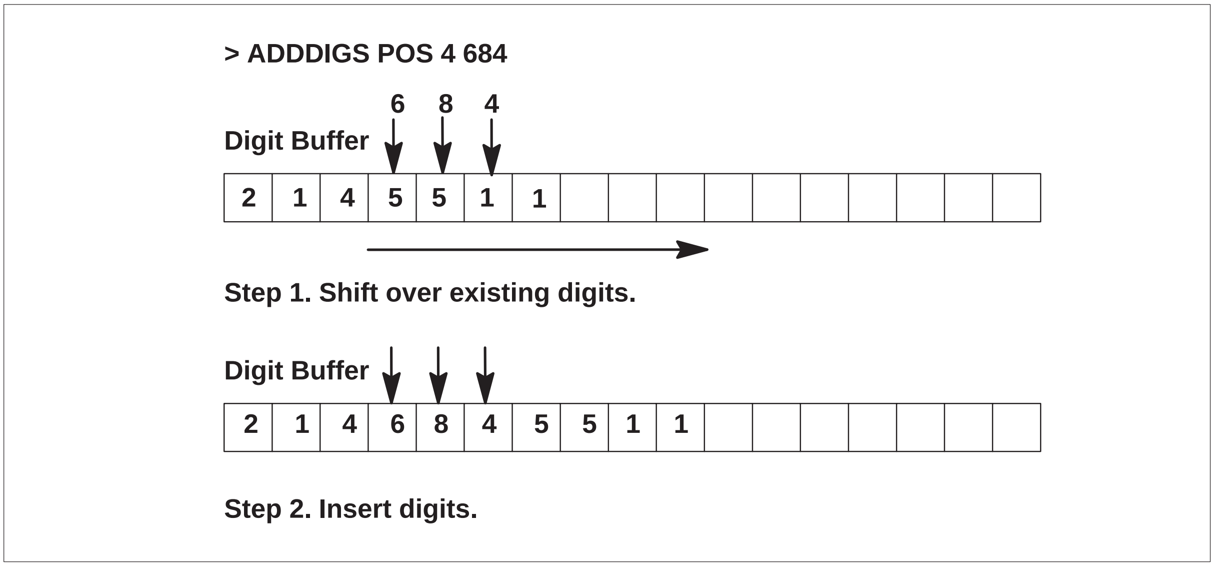
Application

The ADDDIGS operation inserts digits into the digit buffer. Once the digits are in the buffer, it is just as if they were received from the originating agent, and can therefore be processed accordingly by remaining digit or sequence collectables.

Digits identified by the ADDDIGS collectable are inserted into the digit buffer before digits at the identified location. The existing digits are shifted and not overwritten.

Figures 16-7 and 16-8 contain ADDDIGS applications.

Figure 16-7
ADDDIGS application



```
> ADDDIGS POS 4 684
```

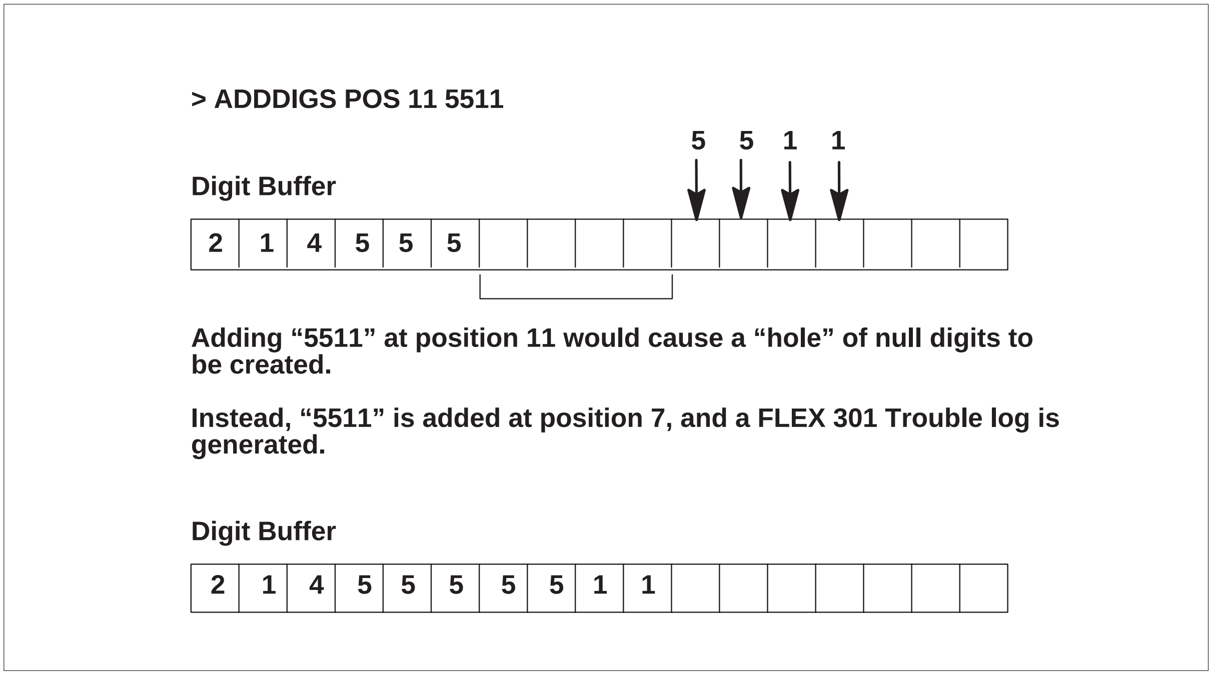
The ADDDIGS operation may not cause holes to occur within the digit buffer. (This could occur only through the POS buffer location value.) If such an attempt is made, then the digits are added at the closest possible location, and a FLEX 301 trouble log is generated with the following trouble code value:

```
Digits force appended to buffer
```

The Flextype field in the log is set to ADDDIGS, and the digits added to the buffer by the ADDDIGS collectable are included in the report.

Figure 16-8 contains an ADDDIGS application where the digits are force appended to the buffer.

Figure 16-8
ADDDIGS application with digits force appended



```
> ADDDIGS POS 11 5511
```

Adding “5511” at position 11 would cause a “hole” of null digits to be created.

Instead, “5511” is added at position 7, and a FLEX 301 Trouble log is generated.

Use case scenario

The collectable is used in the following call scenario:

- digit manipulation on incoming digits

With the ADDDIGS collectable, received digit information can be altered before being processed for the call.

Restrictions and limitations

The following restrictions and limitations apply to the ADDDIGS collectable:

- The ADDDIGS collectable does not request any digits from the originating agent, but uses digits currently in the digit buffer.
- Correct use of the ADDDIGS collectable requires intimate knowledge of the defined interaction with the originating agent.

- If an attempt is made to add digits to the digit buffer which exceeds the 64 digit capacity of the buffer, then the excess digits either being appended to the buffer or pushed out of the buffer due to the insert action of the ADDDIGS collectable are discarded and not added to the buffer. In addition, a FLEX 301 trouble log is generated with the following trouble code value:

  ```
  Digits truncated from buffer
  ```

  The Flextype field in the log is set to ADDDIGS and the digits truncated from the buffer are included in the digits field of the report.

Digit Manipulation collectable (MODDIGS)

The Modify Digits (MODDIGS) collectable provides the ability to manipulate the digits received by replacing an identified series of digits in the digit buffer for processing by digit or other sequence collectables.

Application

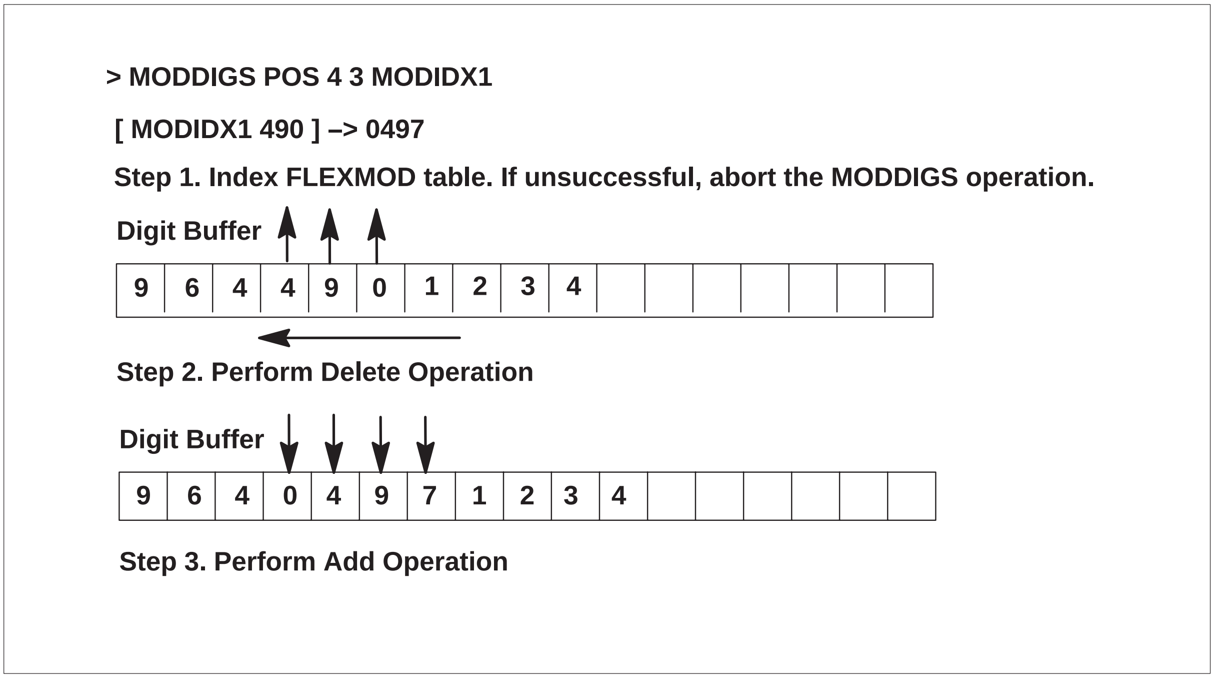
The MODDIGS operation is essentially a sequence of DELDIGS and ADDDIGS operations, contingent upon successfully indexing the FLEXMOD table to retrieve digits to add back to the buffer. The digits to remove are used to index table FLEXMOD. Once the digits are added to the buffer, it is as if they were received from the originating agent, and they are processed by remaining digit or sequence collectables.

If the digits to be removed from the buffer do not successfully index the FLEXMOD table, then the MODDIGS operation is aborted, and the buffer remains unchanged.

Digits identified by table FLEXMOD are inserted into the digit buffer before digits at the identified location. The existing digits are shifted and not overwritten.

Figure 16-9 contains a MODDIGS application.

Figure 16-9
MODDIGS application



```
> MODDIGS POS 4 3 MODIDX1
```

[MODIDX1 490] –> 0497

Step 1. Index FLEXMOD table. If unsuccessful, abort the MODDIGS operation.

Digit Buffer

9	6	4	4	9	0	1	2	3	4							

Step 2. Perform Delete Operation

Digit Buffer

9	6	4	0	4	9	7	1	2	3	4						

Step 3. Perform Add Operation

The INDEX message received from the message center can alter processing of the MODDIGS collectable. This message identifies a FLEXMOD table index for the collectable, and is used in place of the index provisioned with the MODDIGS collectable.

Use case scenario

The collectable is used in the following call scenario:

- digit manipulation on incoming digits

With the MODDIGS collectable, received digit information can be altered before being processed for the call.

Restrictions and limitations

The following restrictions and limitations apply to the MODDIGS collectable:

- The MODDIGS collectable does not request any digits from the originating agent, but uses digits currently in the digit buffer.
- Correct use of the MODDIGS collectable requires intimate knowledge of the defined interaction with the originating agent.

- If an attempt is made to add digits back to the digit buffer which exceeds the 64 digit capacity of the buffer, then the excess digits either being appended to the buffer or pushed out of the buffer due to the insert action of the MODDIGS collectable are discarded and not added to the buffer. In addition, a FLEX 301 trouble log is generated with the following trouble code value:

  ```
  Digits truncated from buffer
  ```

  The Flextype field in the log is set to MODDIGS and the digits truncated from the buffer are included in the digits field of the report.

- Null digits from the digit buffer do not successfully index table FLEXMOD. No trouble log is generated as this situation is not treated differently than when valid digits fail to index the FLEXMOD table.

Digit Manipulation collectable (COPYDIGS)

The Copy Digits (COPYDIGS) collectable provides the ability to manipulate the digits received by replacing an identified series of digits in the digit buffer for processing by digit or other sequence collectables.

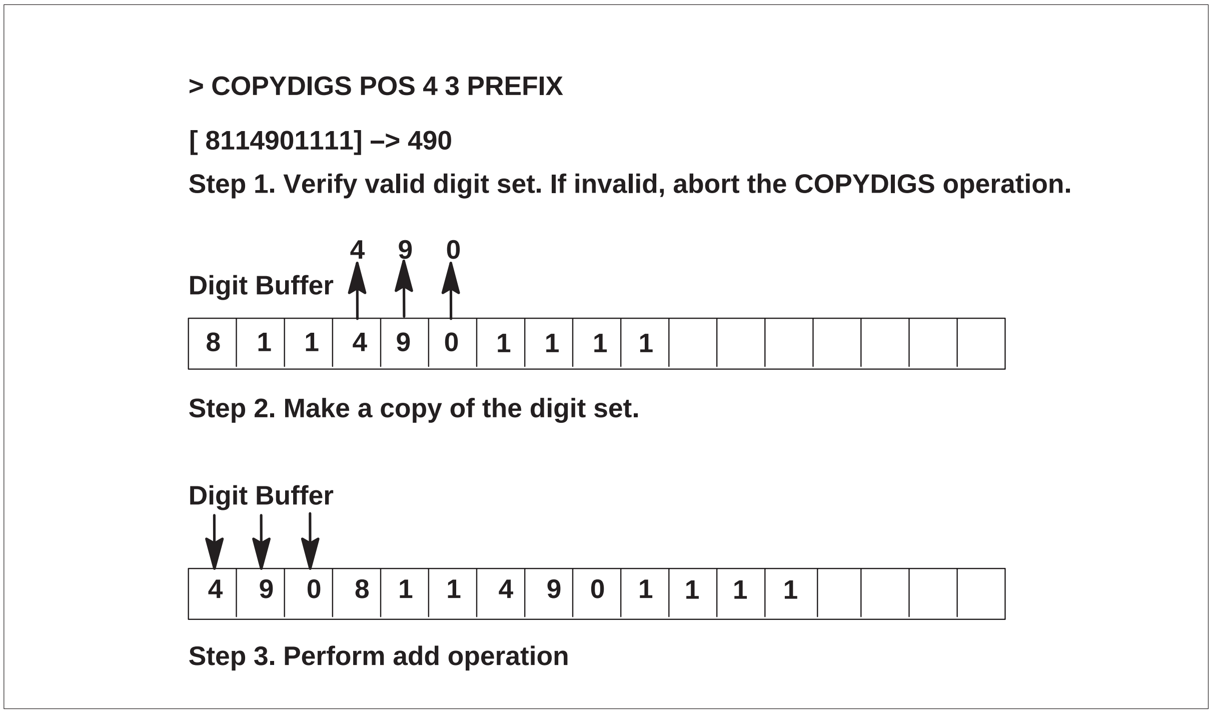
Application

The copy digits operation duplicates digits in the digit buffer at another location within the buffer. Once the digits have been copied in the buffer, it is just as if they were received from the originating agent, and can therefore be processed accordingly by remaining digit or sequence collectables.

The copy operation is performed only if the digits within the digit buffer represented by the FBUFLOC and DIGCNT fields represent actual digits and not empty spaces (null digits) within the buffer. All digits represented by the FBUFLOC and DIGCNT fields must be valid digits within the buffer. A valid digit is any digit collected or assigned by a digit or sequence collectables.

Figures 16-10, 16-11, and 16-12 contain COPYDIGS applications.

Figure 16-10
COPYDIGS application



> COPYDIGS POS 4 3 PREFIX

[8114901111] –> 490

Step 1. Verify valid digit set. If invalid, abort the COPYDIGS operation.

8	1	1	4	9	0	1	1	1	1							

Step 2. Make a copy of the digit set.

4	9	0	8	1	1	4	9	0	1	1	1	1				

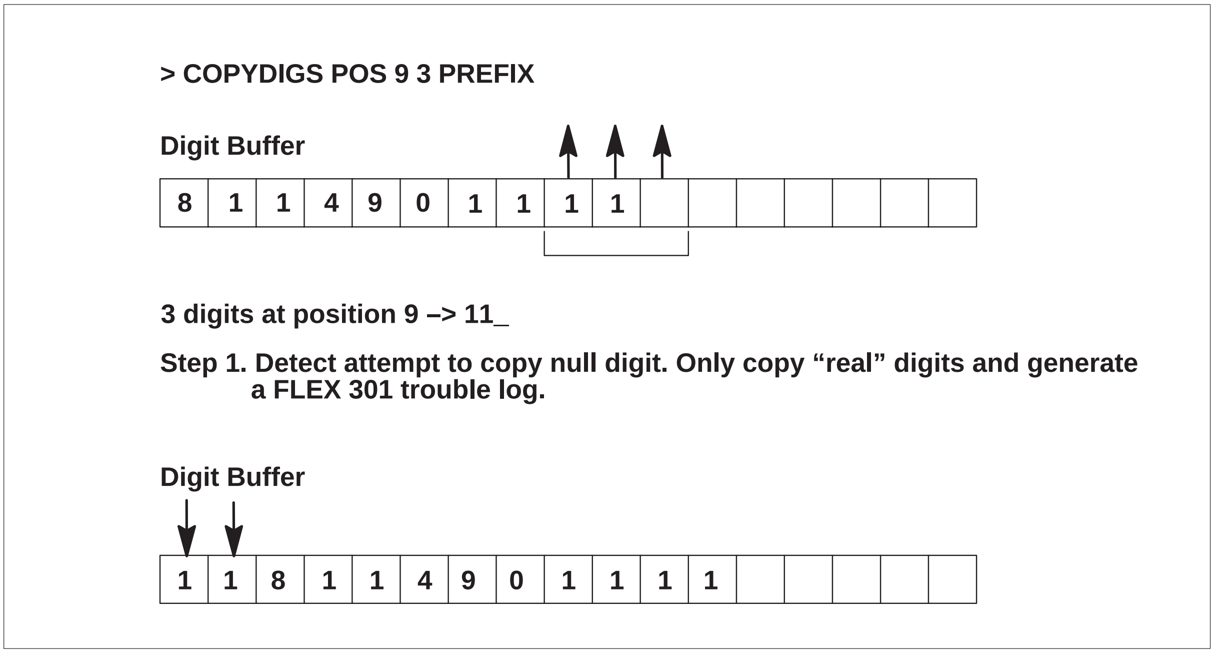
Step 3. Perform add operation

Null digits in the identified copy set are not copied by the COPYDIGS collectable. If a null digit is represented in the digit set to be copied, then only the non-null digits are copied, and a FLEX 301 trouble log is generated with the following trouble code value:

```
Only available digits copied
```

The Flextype field in the log is set to COPYDIGS and just the digits copied are included in the digits field of the report.

Figure 16-11
COPYDIGS application with null digit



> COPYDIGS POS 9 3 PREFIX

Digit Buffer

8	1	1	4	9	0	1	1	1	1							

3 digits at position 9 –> 11_

Step 1. Detect attempt to copy null digit. Only copy “real” digits and generate a FLEX 301 trouble log.

Digit Buffer

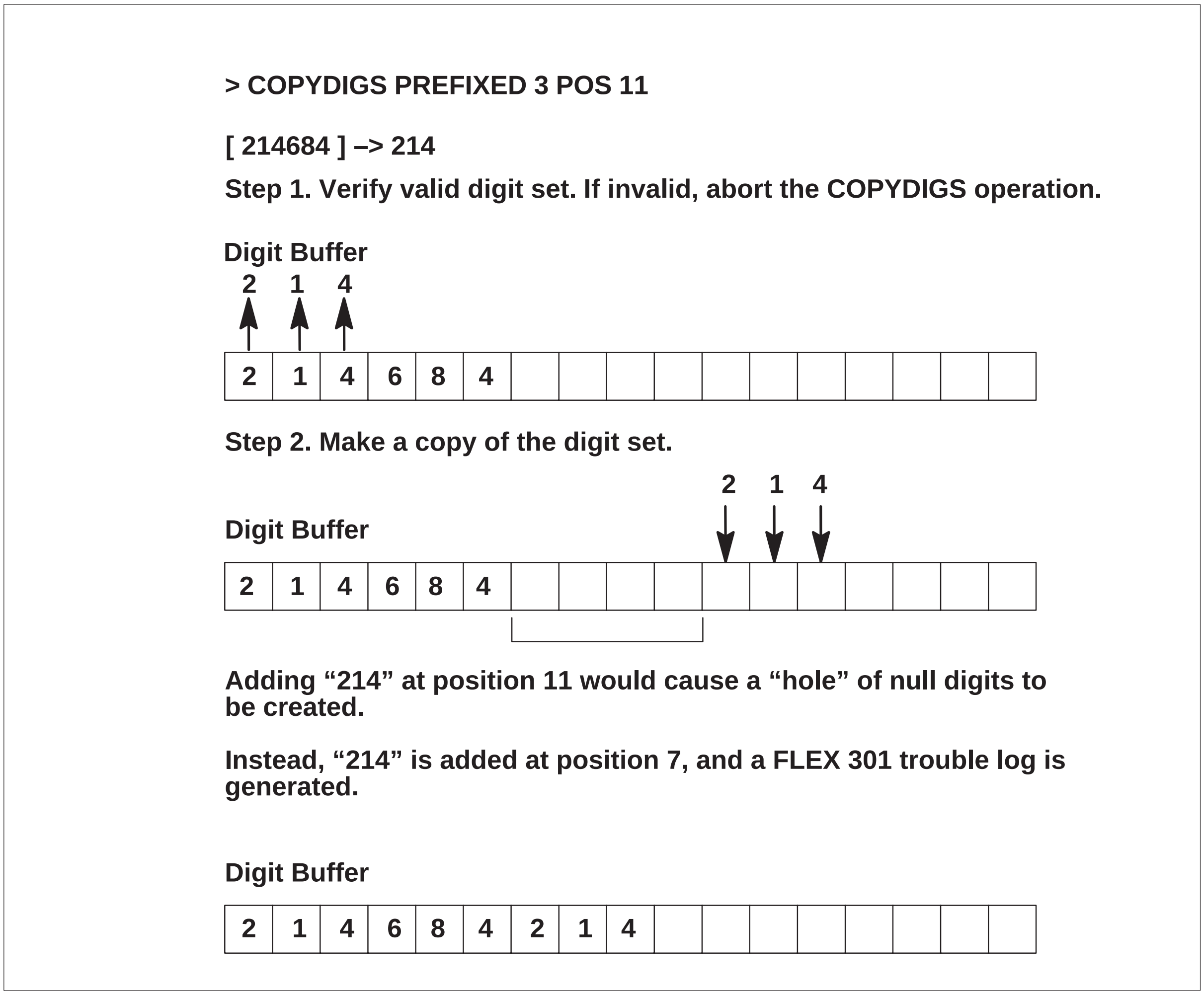
1	1	8	1	1	4	9	0	1	1	1	1					

The COPYDIGS operation may not cause holes to occur within the digit buffer. (This could occur only through the POS buffer location value.) If such an attempt is made, then the digits are added at the closest possible location, and a FLEX 301 trouble log is generated with the following trouble code value:

```
Digits force appended to buffer
```

The Flextype field in the log is set to COPYDIGS, and the digits added to the buffer by the COPYDIGS collectable are included in the report.

Figure 16-12
COPYDIGS application with digits force appended



> COPYDIGS PREFIXED 3 POS 11

[214684] –> 214

Step 1. Verify valid digit set. If invalid, abort the COPYDIGS operation.

Digit Buffer

2	1	4	6	8	4											

Step 2. Make a copy of the digit set.

Digit Buffer

2	1	4	6	8	4											

Adding “214” at position 11 would cause a “hole” of null digits to be created.

Instead, “214” is added at position 7, and a FLEX 301 trouble log is generated.

Digit Buffer

2	1	4	6	8	4	2	1	4								

Use case scenario

The collectable is used in the following call scenario:

- digit manipulation on incoming digits

  With the COPYDIGS collectable, received digit information can be altered before being processed for the call.

Restrictions and limitations

The following restrictions and limitations apply to the COPYDIGS collectable:

- The COPYDIGS collectable does not request any digits from the originating agent, but uses digits currently in the digit buffer.
- If an attempt is made to copy digits in the digit buffer that exceed the 64-digit capacity of the buffer, then the excess digits either being appended to the buffer or pushed out of the buffer due to the insert action of the COPYDIGS collectable are discarded and not added to the buffer. In addition, a FLEX 301 trouble log is generated with the following trouble code value:

  ```
  Digits truncated from buffer
  ```

  The Flextype field in the log is set to COPYDIGS and the digits truncated from the buffer are included in the digits field of the report.
- Correct use of the COPYDIGS collectable requires intimate knowledge of the defined interaction with the originating agent.

NOA Manipulation collectable (MODNOA)

The Modify NOA (MODNOA) collectable provides the ability to modify the received NOA value.

Application

The MODNOA collectable does not request digits from the originating agent, but directly overwrites the NOA value previously recorded in the OCCB. (See “Result of Digit Collection” on pages 16-86 and 16-97 for a description of how call processing sets the NOA values for the calling and called party numbers.) Once the new NOA value has been set, the MODNOA collectable is finished.

The MODNOA collectable can be executed multiple times during the processing of a call.

Use case scenario

The collectable is used in the following call scenario:

- manipulation of identified NOA

With the MODNOA collectable, the calling or called party NOA can be altered for the call.

Note: Dialed number NOA comparison ability to the MODNOA collectable provides the ability to set the dialed number NOA to a specific value for the call.

Restrictions and limitations

The following restrictions and limitations apply to the MODNOA collectable:

- The MODNOA collectable does not request any digits from the originating agent, but directly overwrites the NOA value previously recorded.
- Correct use of the MODNOA collectable requires intimate knowledge of the defined interaction with the originating agent.

Digit Manipulation collectable (AGNTDATA)

The Agent Data (AGNTDATA) collectable provides the ability to manipulate the digits received by adding digits to the digit buffer so that they can be processed by other digit or sequence collectables. The digits added to the buffer are retrieved from agent provisioned information.

Application

The AGNTDATA operation inserts digits provisioned against the agent into the digit buffer. Once the digits are in the buffer, it is just as if they were received from the originating agent, and can therefore be processed accordingly by remaining digit or sequence collectables.

The following options from the TRKFEAT table can be used by the AGNTDATA collectable:

- SNPA
- SNXX
- CITYCODE

Each of these options represent a vector of three digits. The SNPA and SNXX digits are retrieved from the ORIGOPTS vector field of the TRKFEAT table entry.

If the TRKFEAT option specified is not provisioned within the TRKFEAT table entry, then the following actions occur:

- SNPA

  For the SNPA option, the default digits of 000 are used and added to the digit buffer if the SNPA option is not provisioned in the ORIGOPTS field of the TRKFEAT table entry.

- SNXX

  For the SNXX option, the default digits of 000 are used and added to the digit buffer if the SNXX option is not provisioned in the ORIGOPTS field of the TRKFEAT table entry.

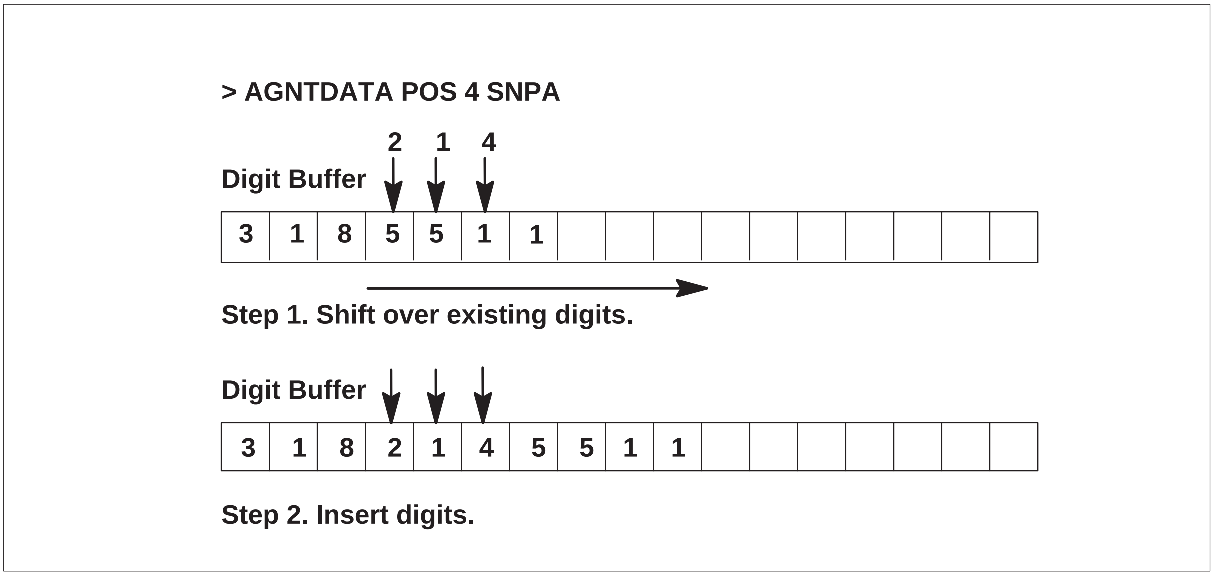
- CITYCODE

  For the CITYCODE option, no digits are added to the digit buffer if the CITYCODE option is not provisioned in the ORIGOPTS field of the TRKFEAT table entry.

The digits identified by the TRKFEAT option are inserted into the digit buffer before existing digits at the identified location. The existing digits are shifted and not overwritten.

Figures 16-13 and 16-14 contain AGNTDATA applications.

Figure 16-13
AGNTDATA application

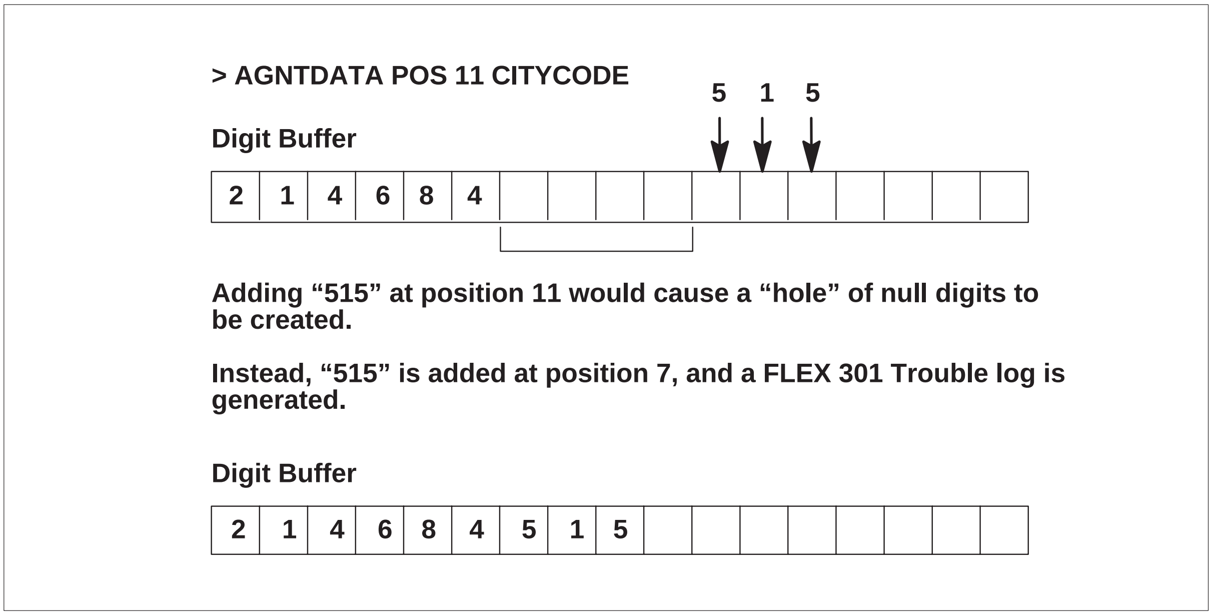


The AGNTDATA operation may not cause holes to occur within the digit buffer. (This could occur only through the POS buffer location value.) If such an attempt is made, then the digits are added at the closest possible location, and a FLEX 301 trouble log is generated with the following trouble code value:

```
Digits force appended to buffer
```

The Flextype field in the log is set to AGNTDATA, and the digits added to the buffer by the AGNTDATA collectable are included in the report.

Figure 16-14
AGNTDATA application with digits force appended



```
> AGNTDATA POS 11 CITYCODE
```

Digit Buffer

2	1	4	6	8	4											

Adding “515” at position 11 would cause a “hole” of null digits to be created.

Instead, “515” is added at position 7, and a FLEX 301 Trouble log is generated.

Digit Buffer

2	1	4	6	8	4	5	1	5								

Use case scenario

The collectable is used in the following call scenario:

- digit manipulation on incoming digits

With the AGNTDATA collectable, received digit information can be altered before being processed for the call.

Restrictions and limitations

The following restrictions and limitations apply to the AGNTDATA collectable:

- The AGNTDATA collectable does not request any digits from the originating agent, but uses digits currently provisioned against the agent in table TRKFEAT.
- Correct use of the AGNTDATA collectable requires intimate knowledge of the defined interaction with the originating agent.
- If an attempt is made to add digits to the digit buffer which exceeds the 64-digit capacity of the buffer, then the excess digits either being appended to the buffer or pushed out of the buffer due to the insert action of the AGNTDATA collectable are discarded and not added to the buffer. In addition, a FLEX 301 trouble log is generated with the following trouble code value:

  ```
  Digits truncated from buffer
  ```

  The Flextype field in the log is set to AGNTDATA and the digits truncated from the buffer are included in the digits field of the report.
- If the CITYCODE option identified by the AGNTDATA collectable is not provisioned in the TRKFEAT table entry, then no digits are added to the digit buffer, and no log is generated.

General application of inband digit collection

Inband digit collection collects digit information from the originating agent, where digits consist of pulsed tones delivered in the bearer channel of the connection to the originating agent. The collected digit information is eventually used to connect the call to the destination desired by the subscriber.

The FlexDial framework provides the ability to collect digits from the originating agent through MF or DTMF inband tone signals through the COLDIG, SUBR, ADDR, OLI, and CIC collectables. The validation of received digits occurs as directed by the specific collectable.

The inband collectables are identical in validation and service application as their out-of-band siblings.

Table 16-1 contains inband and out-of-band digit collectable applications.

Table 6-1
Inband and out-of-band digit collectable applications

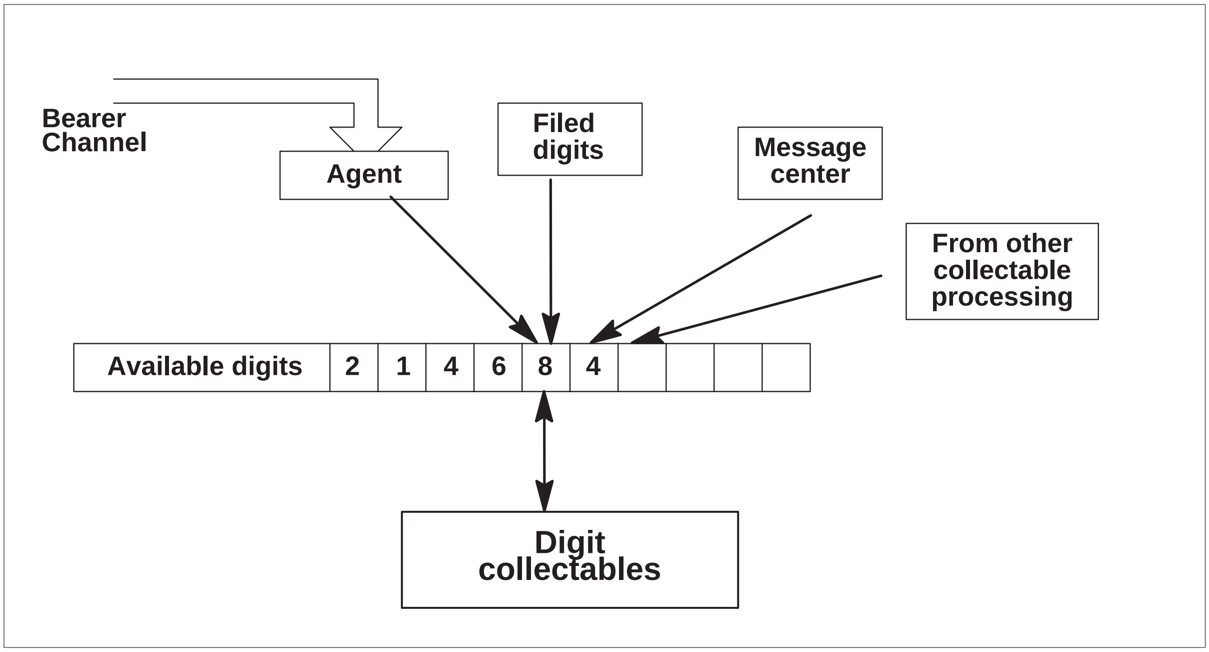
Inband	Out-of-band	Description
COLDIG	COLPARM	Collect raw digit and NOA information.
SUBR	SUBRPARM	Collect and process subscriber numbers
ADDR	ADDRPARM	Collect and process called party address digits
OLI	OLIPARM	Collect and process originating line information digits
CIC	CICPARM	Collect and process carrier identification code digits
REPLDIG		Collect inband digits to replace identified digits within the digit buffer

Application

All inband digit collectables (COLDIG, SUBR, ADDR, OLI, CIC, REPLDIG) can perform a digit collection request for digits from the originating agent. The request is completed by collecting inband MF and DTMF digits and storing them in the digit buffer. The collectable then consumes the required number of digits from the digit buffer for collectable processing.

Figure 16-15 contains sources of digits for digit collectables.

Figure 16-15
Sources of digits for digit collectables

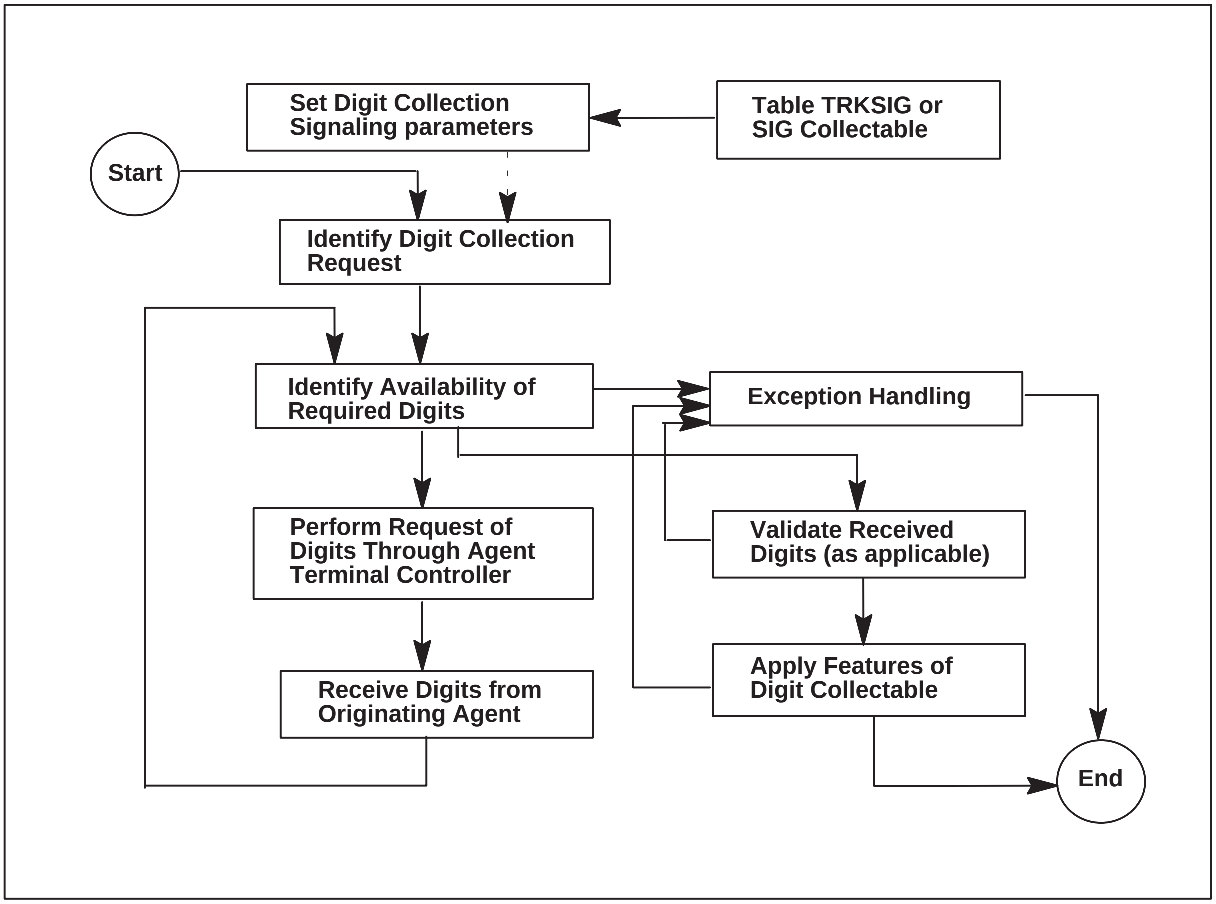


For DTMF inband digit collection, with the exception of the COLDIG digit collectable and the SUBR digit collectable where the REPBDIGS field is set to Y, digit collectables consume all inband digits collected from the agent terminal controller by the collectable, and also consume all filed digits identified for the collectable.

For MF inband digit collection, through use of the FDIGMASK and LDIGMASK fields (specified in a SIG collectable, for example), it is possible to receive more than the identified MAX number of digits. In this event, the extra digits collected remain in the digit buffer.

Figure 16-16 contains the flow for the inband digit collection process.

Figure 16-16
Inband digit collection process flow


Set Digit Collection Signaling parameters
Table TRKSIG or SIG Collectable
Start
Identify Digit Collection Request
Identify Availability of Required Digits
Exception Handling
Perform Request of Digits Through Agent Terminal Controller
Validate Received Digits (as applicable)
Apply Features of Digit Collectable
Receive Digits from Originating Agent
End

This flow identifies the following processing areas for digit collection:

- specify digit collection signaling parameters
- identify digit collection request
- identify availability of required digits
- perform digit collection request
- receive digits
- validate applicable digits
- apply digit collectable features
- exception handling

Specify digit collection signaling parameters

The collection request requires the following inband digit collection signaling parameters to be defined. This information is defined by either table TRKSIG for PTS agents, or by the SIG protocol collectable for both PTS and CCS type agents.

- pulse type (either MF or DTMF)
  - For MF signaling, the first and last digit masks (FDIGMASK, LDIGMASK).
  - For DTMF signaling, the digit mask (DIGMASK) and terminating digit (TRMDIGIT).
- permanent signal timer (PSEIZTMR)
- partial dial timer (PDILTMR)
- minimum digit received partial dial timer (MINRTMR)

For PTS type agents, the pulse type is initially defined by table TRKSIG. The SIG protocol collectable alters the digit collection signaling parameters during the processing of the call, and can be used to change the pulse type currently in use.

The digit mask fields identify which digits are acceptable to use to fulfill the digit collection request. In the case of MF digit collection, the first and last digit masks provide delimiters to receive a stream of digits from the originating agent.

The terminating digit identifies an end of dialing sequence by the subscriber. (See “Application of reset capability” on page 16-64 for terminating digit versus reset digit handling.) The receipt of a terminating digit identifies that the subscriber has completed entry of digits for the particular sequence (digit collectable). If enough digits have not been entered to successfully process the digit collectable, then either permanent signal (when no digits have been entered) or partial dial (when some but not enough digits have been entered) treatment is applied to the call.

The permanent signal timer identifies the amount of time allowed to receive the first digit before a permanent signal timeout exception occurs.

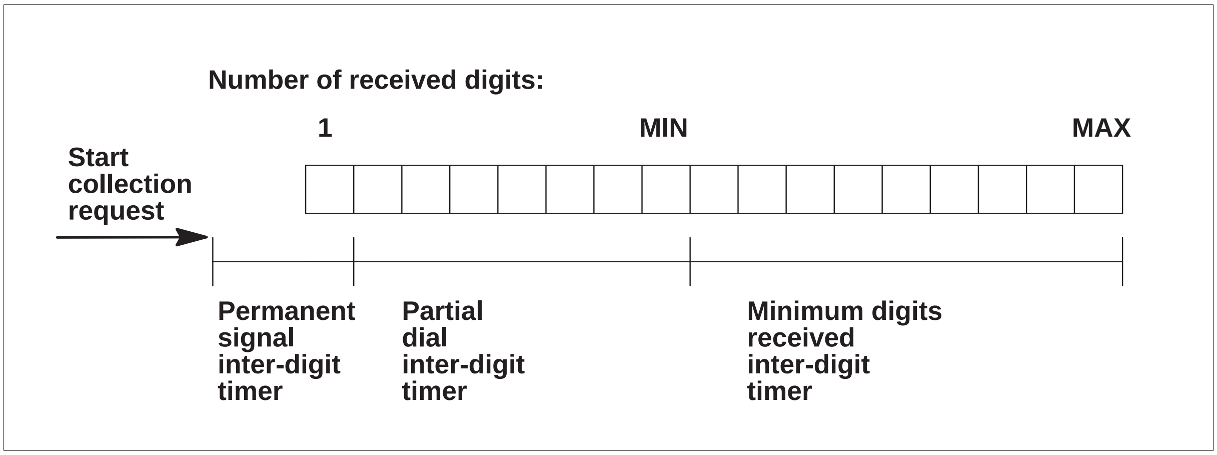
The partial dial timer identifies the amount of time to receive each digit, prior to collection of the minimum amount of required digits, before a partial dial timeout exception occurs.

The minimum digits received timer identifies the amount of time to receive each digit, after the minimum amount of required digits has been collected, before a minimum digits received timeout occurs.

The permanent signal, partial dial, and minimum digits received timers are categorized as digit reception interdigit timers.

Figure 16-17 contains the digit collection interdigit timer use.

Figure 16-17
Digit collection interdigit timer use


Number of received digits:
1
MIN
MAX
Start collection request
Permanent signal inter-digit timer
Partial dial inter-digit timer
Minimum digits received inter-digit timer

Common channel signaling agents do not contain inband digit collection signaling parameters within the TRKSIG table definition. Therefore, on CCS type originating agents, a SIG collectable must precede use of an inband digit collectable in order to specify this information.

If a SIG protocol has not been executed prior to executing an inband digit collectable on common channel signaling agents, then the inband digit collection application is aborted. A FLEX 303 failure to execute log is generated with a failure reason of:

```
Need SIG to perform inband collection,
```

Collectable processing is suspended, and “feature not allowed” (FNAL) treatment is immediately applied to the call.

Identify digit collection request

An inband collection request for digits from the originating agent occurs through execution of the COLDIG, SUBR, ADDR, OLI, CIC, and REPLDIG digit collectables. Each collectable provides specific information for the inband digit request. This information includes:

- applicable filed digits (FILED option)
- applicable prompt information (PROMPT option)
- the minimum number of digits to collect (MIN field)
- the maximum number of digits to collect (MAX field)
- applicable reset digit information (RESET option)
- applicable validation and other digit collectable service processing information.

This information controls the characteristics of the interaction with the originating agent for the specific digit collectable.

Identify availability of required digits

The actual number of digits requested from the originating agent is based upon the MIN and MAX values provisioned, less the number of available digits in the digit buffer and the number of available filed digits.

Note: Filed digits may be received either through the FILED digit collectable option, or through receipt of filed digits from a message center FILED message.

Figure 16-18 contains the formula for collection request.

Figure 16-18
Formula for collection request

Formula:

#Digits to Collect
– #Available Digits in Digit Buffer
– #Filed Digits

# of Digits to Request

Example:

7 (MIN number to collect ; MAX = 11)
– 3 Digits already in Digit Buffer
– 2 Filed Digits

---

2 Digits required for inband digit request.
An additional 4 optional digits are also requested from the agent.

Note: After the first two digits have been received, the request for 4 additional digits is executed. Only two digits are required for the collectable to successfully complete.

An initiated digit collection request from the originating agent is compete when one of the following events occur:

- All required digits are available from the combination of the digit buffer and filed digit specifications.
- The terminating digit is received.
- A reset digit is received.
- A timeout occurs during the collection process.

If all required digits are available (the minimum amount of digits) before an initial inband digit collection request is made of the originating agent, then the inband digit collection request is not performed. In this case, the digit collectable receives all required digits and available optional digits from the digit buffer and FILED digits options.

If on the initial analysis of available digits no digits are available through either the digit buffer or FILED digit sources, and the MIN count to collect is equal to zero, then an initial request is still performed to receive up to the MAX number of digits. (MIN can only be zero through use of the COLDIG collectable.) The request always completes successfully even if no digits are received. COLDIG uses a zero-based digit collection request. A request for digit is always performed unless the number of available digits is greater than the MIN count to collect.

For MF digit collection, if the required number of digits are received in the FDIGMASK/LDIGMASK delimited digit stream, then the digit collectable does not wait for the reception of additional inband digits, but completes processing with the available digits.

Once all required digits are available, and an attempt to collect optional digits is performed as allowed or needed, the following events occur:

- The required digits and available optional digits are consumed (removed from the digit buffer to be used and stored in the proper place for call processing) by the digit collectable.
- Validation (for those digit collectables provisioned with a VALIDATE option) of the digits occur as applicable to the digit collectable.
- Execution of digit collectable features occur as applicable to the digit collectable.

A terminating or reset digit can be received from the originating agent, or the digit may have been added to the digit buffer by a previously executed ADDDIGS, MODDIGS, or COPYDIGS collectable. In both cases, the handling of the reset or terminating digit is identical.

Note: If the terminating digit was in the digit buffer, then no more digits following the terminating digit are pulled from the digit buffer for use by the digit collectable.

Receipt of the terminating digit or timeout indicates that the last digit for this request has been received. If at least the minimum number of digits required is available for the collectable, then the required number of digits has been successfully received.

If the minimum number of digits required has not been received when a terminating digit or timeout is received, then treatment is set for the call. If no digits have been collected from the agent terminal controller or no digits are available in the digit buffer, then a permanent signal exception timeout is generated. If at least one digit has been collected, then a partial dial exception timeout is generated.

If more than the maximum number (identified by the MAX field) of digits are received, then only the MAX number of digits are consumed by the digit collectable, and the extra digits remain in the digit buffer. This is possible using MF pulsing, not possible using DTMF pulsing.

Application of filed digits

Filed digits are used when a particular originating agent or group of agents originating on a particular access terminal group (trunk group) use the same number (same digits) for a particular digit collection application.

For instance, all subscribers on a particular trunk group may use the same authorization (AUTH) code, or a particular subscriber number may always use the same called party address digits, such as a hotline call.

In these cases, the digits are provisioned on the UCS DMS-250 switch for use by call processing. Digits may be filed for a particular digit collectable through a handful of methods:

- use of the FILED option on the provisioned digit collectable
- use of the MSGCTR FILED message for applicable digit collectables (COLDIG, SUBR, ADDR, OLI, REPLDIG)

These methods for identifying filed digits may be used individually or in combination. For example, processing of a subscriber number through the SUBR collectable may use the ADDR FILED message center message to specify four filed digits for the called party address. The ADDR collectable may use the digit collectable FILED option to identify three more filed digits for a combined total of seven filed digits for the ADDR collectable.

Note: Multiple MSGCTR FILED messages cannot be used in combination by a digit collectable. Additional FILED messages processed overwrite previous filed information received.

Filed digits received via the message center are applied after digits provisioned in the digit collectable FILED option. If the message center filed digits are able to satisfy the minimum (MIN) number of digits required for the digit collectable, digits provisioned in the digit collectable FILED option will not be used. Similarly, if enough digits are available from both filed message center and digit collectable FILED option to satisfy the MIN, no digits will be removed from the digit buffer or requested from the agent. An exception is made for the COLDIG collectable. The COLDIG collectable will continue to pursue digits from other sources until its minimum (MIN) is exceeded or its maximum (MAX) is satisfied.

If the number of filed digits exceeds the maximum (MAX) number of digits required for the digit collectable, then the excess filed digits are unused. In this scenario, the available digits are truncated to the MAX value, and the excess digits are discarded.

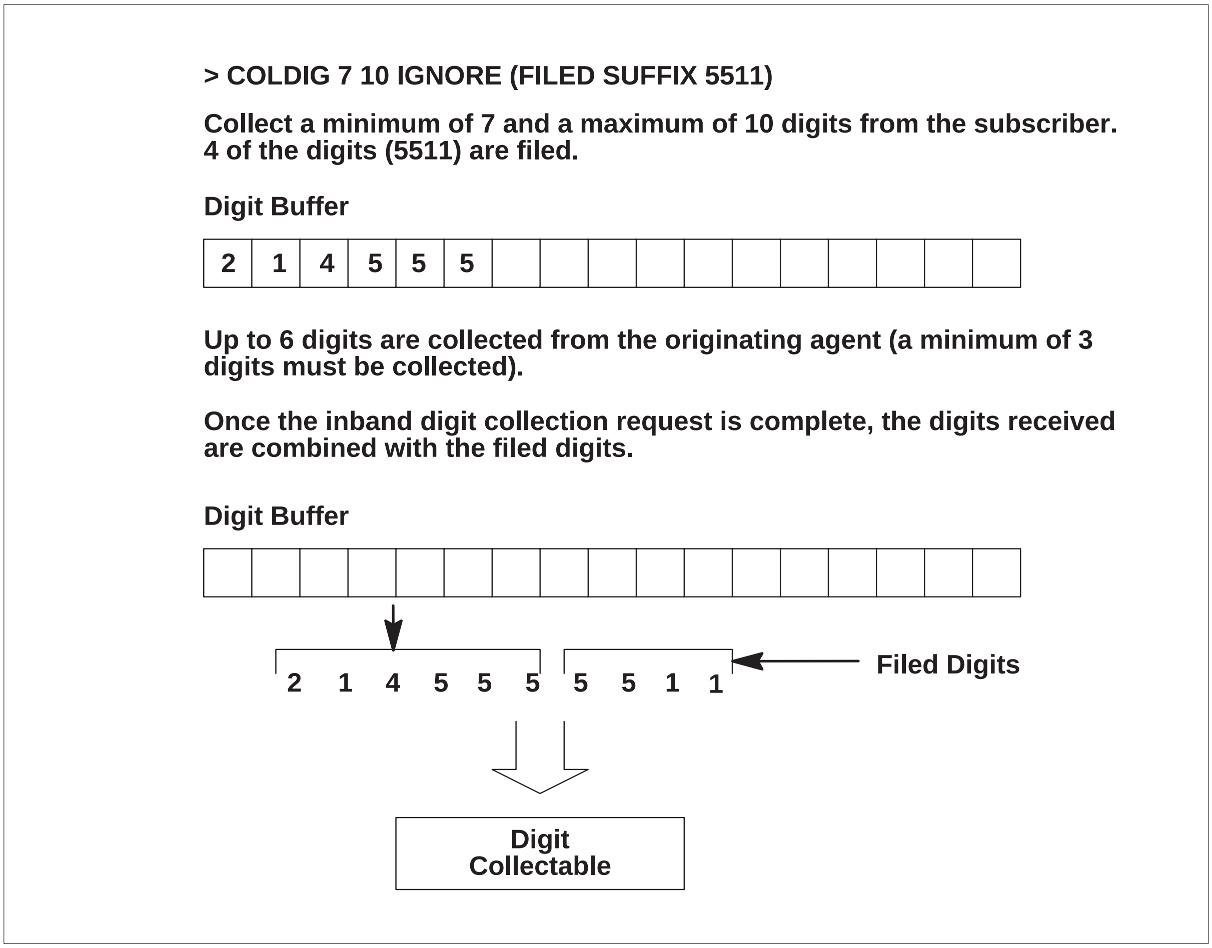
Filed digits may be used to fulfill the entire request (if the number of filed digits is greater than or equal to the minimum number of digits to collect), or fulfill a partial amount of the request (the number of filed digits is less than the minimum number of digits to collect). For a partially filed number scenario, the required number of digits are collected, and then the received digits are combined with the filed digits to provide up to the MAX amount of digits for the collectable. Typically, filed digits are not inserted into the digit buffer for processing, but are handled by the digit collectable independent of digit buffer use.

The number of filed digits available is subtracted from the number of digits to collect from the originating agent. Once all the requested digits have been collected, the filed digits are included for collectable processing.

For the SUBR or SUBRPARM collectables, when the replace buffer digits field (REPBDIGS) of the collectable is set to Y, all of the digits processed by the collectable are re-inserted into the digit buffer. This action includes any filed digits processed by the collectable. For example, a seven-digit subscriber number is being processed, and four of the seven digits are filed. Three digits are therefore collected from the originating agent, and then combined with the four filed digits to meet the requirements of the collectable. If the REPBDIGS field for the collectable is set to Y, then all seven digits processed are reinserted into the digit buffer upon completion of the collectable.

Figure 16-19 contains the FILES digit application.

Figure 16-19
FILED digits application



```
> COLDIG 7 10 IGNORE (FILED SUFFIX 5511)
```

Collect a minimum of 7 and a maximum of 10 digits from the subscriber. 4 of the digits (5511) are filed.

Digit Buffer

2	1	4	5	5	5											

Up to 6 digits are collected from the originating agent (a minimum of 3 digits must be collected).

Once the inband digit collection request is complete, the digits received are combined with the filed digits.

Digit Buffer

Filed digits may be prefixed or suffixed to the received digits.

It is possible (but not recommended) that filed digits in the FlexDial framework can include either a terminating or reset digit. Processing one of these digits as a filed digit is handled according to Table 16-2.

Table 6-2
Reset or terminating filed digit actions

Filed method	Digit filed	Action
PREFIX	Reset	The reset without digits action is performed for the digit collectable.
	Terminating	The terminating digit and subsequent filed digits are ignored. If the minimum number of digits are available, then the call continues with validation or apply features. If the minimum number of digits are not available, then a permanent signal exception (no digits are available) or a partial dial exception (if 1 or more digits are available) is generated. No digits from the digit buffer are consumed.
	Reset = Terminating	The identified digit and subsequent filed digit are ignored. If the minimum number of digits are available, then the digit is interpreted as a terminating digit and the call continues with validation or apply features. If less than the minimum number of digits are available, then the digit is interpreted as a reset digit and the reset without digits action is performed for the digit collectable.
SUFFIX	Reset	If digits are available in the digit buffer, then up to the required number of digits are consumed from the digit buffer and a reset with digits action is performed on the collectable. If no digits are available in the digit buffer, then an attempt to collect one digit from the originating agent is performed. If a digit is collected, then the reset with digits action is performed. If a digit is not collected, then the reset without digits action is performed.

—continued—

Table 6-2
Reset or terminating filed digit actions (continued)

Filed method	Digit filed	Action
SUFFIX	Terminating	The terminating digit and subsequent filed digits are ignored, and the MIN/MAX parameters for the required number of digits are calculated based on the revised available number of filed digits. Collectable processing then proceeds normally.
SUFFIX	Reset = Terminating	The identified digit and subsequent filed digits are ignored, and the MIN/MAX parameters for the required number of digits are calculated based on the revised available number of filed digits. If the minimum number of digits are not available, then the digit is treated as a reset digit and either the reset without digits action is performed (if no digits are available) or the reset with digits action is performed (if at least one digit is available). If at least the minimum number of digits are available, collectable processing proceeds to either validation or apply features as appropriate.
—end—		

Application of reset capability

The subscriber uses reset to restart digit collection if digits are entered incorrectly. Reset occurs when the subscriber enters the identified reset digit for the processing digit collectable, or the reset digit is added to the digit buffer through use of a sequence collectable (either ADDDIGS, MODDIGS, or COPYDIGS), and subsequently processed in the “Identify Availability of Required Digits” phase of digit collectable execution.

Note: Reset for telephones is like the backspace key on a computer. The basic telephone though lacks the powerful sensory feedback (visual feedback) a computer can provide. Therefore, the service provider must carefully program reset actions to deliver proper and discernible feedback to the subscriber.

A reset digit can be processed only if the RESET option is provisioned with the digit collectable being processed. The RESET option contains the following information for managing the reset event:

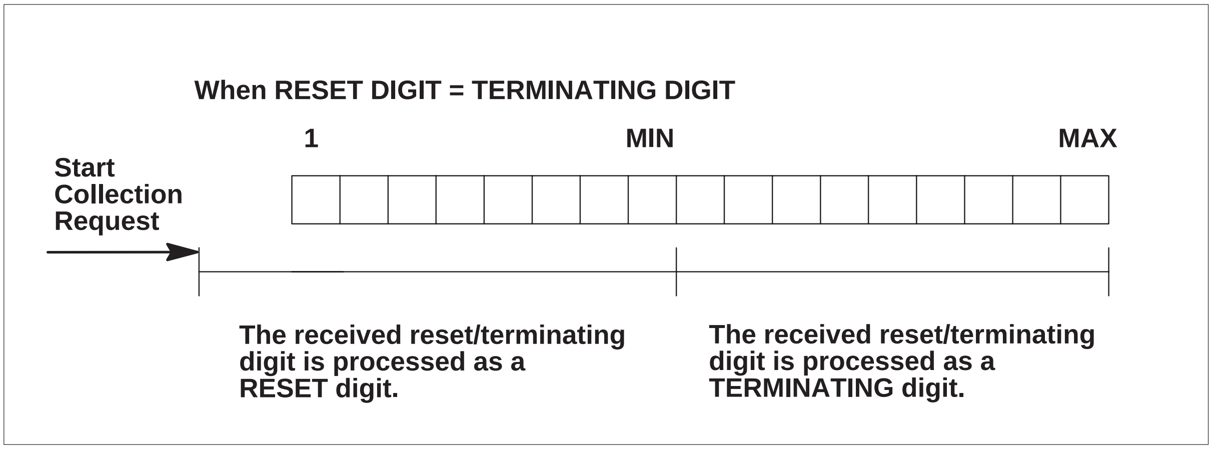
- reset digit (either * [asterisk] or # [octothorpe]).
- a limit to the number of times reset can occur and the treatment that is set if the limit is exceeded
- the action to take if reset occurs after some digits have been collected for the collectable
- the action to take if reset occurs before any digits have been collected for the collectable

The RESET digit can be changed for an entire trunk group (RESETDIGIT) field in table TRKSIG) or changed for each digit collectable processed. The reset digit should not be set to the same digit as the terminating digit (see “SIG protocol collectable” on page 2-7). If the identified reset digit is identical to the terminating digit, then the defined MIN value for the collectable identifies how the digit is processed:

- If less than the minimum number of digits have been collected, then the digit is interpreted as a reset digit.
- If equal to or greater than the minimum number of digits have been collected, then the digit is interpreted as a terminating digit.

Figure 16-20 contains the reset/terminating digit interaction.

Figure 16-20
Reset/terminating digit interaction

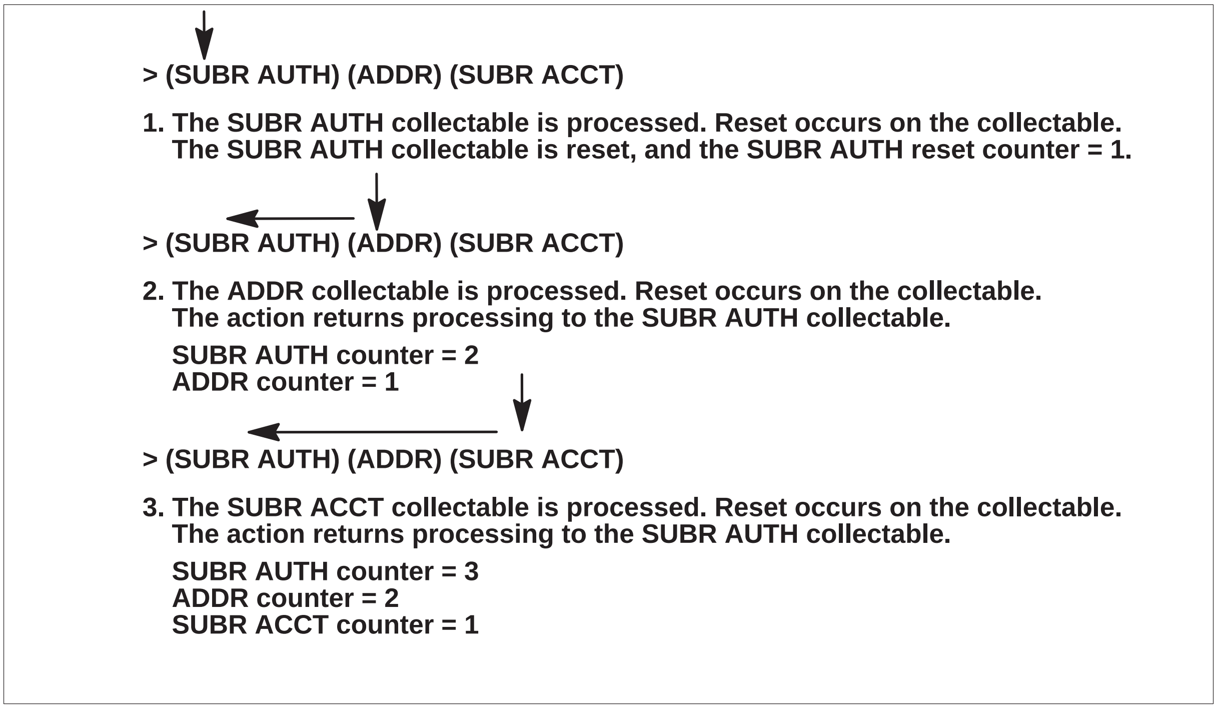

When RESET DIGIT = TERMINATING DIGIT
1
MIN
MAX
Start Collection Request
The received reset/terminating digit is processed as a RESET digit.
The received reset/terminating digit is processed as a TERMINATING digit.

Because the RESET option can be provisioned with each digit collectable, the reset limit is maintained on a per collectable basis. The counter for resets by a digit collectable can be incremented through two events:

- Reset occurs on the digit collectable being processed.
- Reset occurs on a digit collectable processed later in the call, and the reset action specified causes the previously executed collectable to be reset.

Figure 16-21 contains the reset/terminating digit interaction.

Figure 16-21
Reset/Terminating digit interaction



> (SUBR AUTH) (ADDR) (SUBR ACCT)

1. The SUBR AUTH collectable is processed. Reset occurs on the collectable. The SUBR AUTH collectable is reset, and the SUBR AUTH reset counter = 1.

> (SUBR AUTH) (ADDR) (SUBR ACCT)

2. The ADDR collectable is processed. Reset occurs on the collectable. The action returns processing to the SUBR AUTH collectable.

SUBR AUTH counter = 2
ADDR counter = 1

> (SUBR AUTH) (ADDR) (SUBR ACCT)

3. The SUBR ACCT collectable is processed. Reset occurs on the collectable. The action returns processing to the SUBR AUTH collectable.

SUBR AUTH counter = 3
ADDR counter = 2
SUBR ACCT counter = 1

If the reset counter for a digit collectable exceeds the reset limit specified, then an exception is generated and the treatment provisioned with the RESET option is set. All further collectable processing is suspended, and the treatment set is immediately applied to the call.

The reset event can occur in one of three states of digit collectable processing:

- after the digit collectable has completed all execution

  This state is handled by the Reset To Action.

- after some digits have been collected (one or more) by the digit collectable

  This state is handled by the Reset With Digits Action.

- before any digits have been collected by the digit collectable

  This state is handled by the Reset Without Digits Action.

Reset to action

The reset to action applies after a collectable has completed processing and is being reset due to the use of a PREVIOUS or PREPRMPT reset action by a digit collectable executed later in the call.

Because reset occurs after the collectable has completed, all processing that occurred by the collectable must be reversed. This involves a number of actions for the different collectables as shown in Table 16-3.

Table 16-3
Reverse action for FlexDial framework collectables

Collectable	Reverse Action
SIG	The signaling data (timer values, digit masks) is returned to its value prior to execution of the SIG collectable. Reset may only occur on a SIG collectable that has not changed the method or pulse type of receiving digits from the originating agent.
RCVSIG	No action (cannot undo)
SNDSIG	No action (cannot undo)
TERMINATE	Reset may not occur on a TERMINATE collectable
RETRIEVE	The digit buffer is returned to its state prior to execution of the RETRIEVE collectable.

—continued—

Table 16-3
Reverse action for FlexDial framework collectables (continued)

Collectable	Reverse Action
NOTIFY	No action
TIMER	The timer activated by the TIMER collectable is deactivated. A prior timer against the TIMER ID is not restored.
KILLTMR	No action (the timer killed is not restored)
CALLCOND	The call condition identified is deactivated.
APTRMT	No action
GOTO	The collectable list is restored to its pre-execution state.
DO	The collectable list is restored to its pre-execution state.
INCLUDE	The collectable list is restored to its pre-execution state.
ROUTE	The routing data for the call is returned to its value prior to execution of the ROUTE collectable.
NOOP	No action
IFTRMT	The collectable list is restored to its pre-execution state.
IFPRMT	The collectable list is restored to its pre-execution state.
IFTOD	The collectable list is restored to its pre-execution state.
IFDIGS	The collectable list is restored to its pre-execution state.
IFCNT	The collectable list is restored to its pre-execution state.
IFNOA	The collectable list is restored to its pre-execution state.
IFPARM	The collectable list is restored to its pre-execution state.
DELDIGS	The digit buffer is returned to its state prior to execution of the DELDIGS collectable. (The digits removed are added back.)

—continued—

Table 16-3
Reverse action for FlexDial framework collectables (continued)

Collectable	Reverse Action
ADDDIGS	The digit buffer is returned to its state prior to execution of the ADDDIGS collectable. (The digits added are removed.)
MODDIGS	The digit buffer is returned to its state prior to execution of the MODDIGS collectable. (Digits from the FLEXMOD table added to the buffer are removed, and digits which were removed are added back.)
COPYDIGS	The digit buffer is returned to its state prior to execution of the COPYDIGS collectable. (Digits copied are removed.)
MODNOA	The calling or called party NOA is reset to the value held prior to execution of the MODNOA collectable.
AGNTDATA	The digit buffer is returned to its state prior to execution of the AGNTDATA collectable. (Digits added are removed.)
COLDIG	A number of events occur when resetting a COLDIG collectable: • The digit buffer is returned to its state prior to execution of the COLDIG collectable. • Any messages retrieved from the message center by the COLDIG collectable are replaced.
SUBR	A number of events occur when resetting a SUBR collectable: • The digit buffer is returned to its state prior to execution of the SUBR collectable. • Any messages added to the message center by SUBR processing are removed. • Any messages retrieved from the message center by the SUBR collectable are replaced. • Any feature data set through SUBR processing (data from table FLEXFEAT/FLEXTYPE) is reset to values prior to execution of the SUBR collectable. Any feature data that is acted upon immediately cannot be undone (for example, a FLEX log that was generated). • The call treatment data is returned to its value prior to execution of the SUBR collectable. • Any data captured in the call detail record is removed.

—continued—

Table 16-3
Reverse action for FlexDial framework collectables (continued)

Collectable	Reverse Action
ADDR	A number of events occur when resetting an ADDR collectable: • The digit buffer is returned to its state prior to execution of the ADDR collectable. • Any messages added to the message center by ADDR processing are removed. • Any messages retrieved from the message center by the ADDR collectable are replaced. • Any feature data set through ADDR processing (data from table STDPRTCT) is reset to values prior to execution of the ADDR collectable. Any data that is acted upon immediately cannot be undone (for example, performing a TCAP query for N00 numbers and acting on the response). • The call treatment data is returned to its value prior to execution of the ADDR collectable. • Any data captured in the call detail record is removed.
OLI	A number of events occur when resetting an OLI collectable: • The digit buffer is returned to its state prior to execution of the OLI collectable. • Any messages added to the message center by OLI processing are removed. • Any messages retrieved from the message center by the OLI collectable are replaced. • Any feature data set through OLI processing (data from table STDPRTCT) is reset to values prior to execution of the OLI collectable. Any data that is acted upon immediately cannot be undone. • The call treatment data is returned to its value prior to execution of the OLI collectable. • Any data captured in the CDR is removed.
CIC	A number of events occur when resetting a CIC collectable: • The digit buffer is returned to its state prior to execution of the CIC collectable. • Any messages added to the message center by CIC processing are removed. • Any feature data set through CIC processing is reset to values prior to execution of the CIC collectable (this includes CIC routing). Any data that is acted upon immediately cannot be undone. • Any data captured in the CDR is removed.

—continued—

Table 16-3
Reverse action for FlexDial framework collectables (continued)

Collectable	Reverse Action
REPLDIG	A number of events occur when resetting a REPLDIG collectable: • The digit buffer is returned to its state prior to execution of the REPLDIG collectable. • Any messages retrieved from the message center by the REPLDIG collectable are replaced.
COLPARM	The reset action is identical as described for the COLDIG collectable.
SUBRPARM	The reset action is identical as described for the SUBR collectable.
ADDRPARM	The reset action is identical as described for the ADDR collectable.
OLIPARM	The reset action is identical as described for the OLI collectable.
CICPARM	The reset action is identical as described for the CIC collectable.
CALLTYPE	A number of events occur when resetting a CALLTYPE collectable: • Any messages added to the message center by CALLTYPE processing are removed. • Any messages retrieved from the message center by the CALLTYPE collectable are replaced. • Any feature data set through CALLTYPE collectable processing (data from table FLEXFEAT/FLEXTYPE) is reset to values prior to execution of the CALLTYPE collectable. Any feature data that is acted upon immediately cannot be undone. • The call treatment data is returned to its value prior to execution of the CALLTYPE collectable.
CLRFTRS	A number of events occur when resetting a CLRFTRS collectable: • Any messages removed from the message center by the CLRFTRS collectable are restored. • The history of executed collectables is restored. • The list of collectables to revalidate on a reoriginated call is restored. • Any feature data reset through CLRFTRS collectable processing is restored to values held prior to execution of the CLRFTRS collectable.
SETTRANS	The translation system is restored to the value set prior to execution of the SETTRANS collectable.

—continued—

Table 16-3
Reverse action for FlexDial framework collectables (continued)

Collectable	Reverse Action
SETTRMT	The treatment setting for the call is restored to its value prior to execution of the SETTRMT collectable.
APRESET	No action.
OM	The OM pegged by the collectable is decremented by one.
VAROP	Variables are restored to their pre-execution values and MSGCTR messages are reposted.
IFVAR	The collectable list is restored to its pre-execution state.

—end—

After the results of the previous execution of the collectable have been reversed, the collectable is then re-executed. All parameters and options applied during the previous execution of the collectable also apply in an identical fashion to the re-execution of the collectable. For example, if a PROMPT option is used on a digit collectable, then the PROMPT is re-applied for the digit collectable collection process.

Reset with digits action

When digits have been received by the collectable before the reset digit is detected, then the reset action is identified by the WDIGACT field is applied. This action may be one of four values:

- NOPROMPT
- PROMPT
- PREVIOUS
- PREPRMPT

The NOPROMPT or PROMPT actions identify that the digit collectable is re-executed either with or without an identified prompt. The PREVIOUS action identifies that the previously executed digit collectable that collected at least one digit from the originating agent is reset and re-executed.

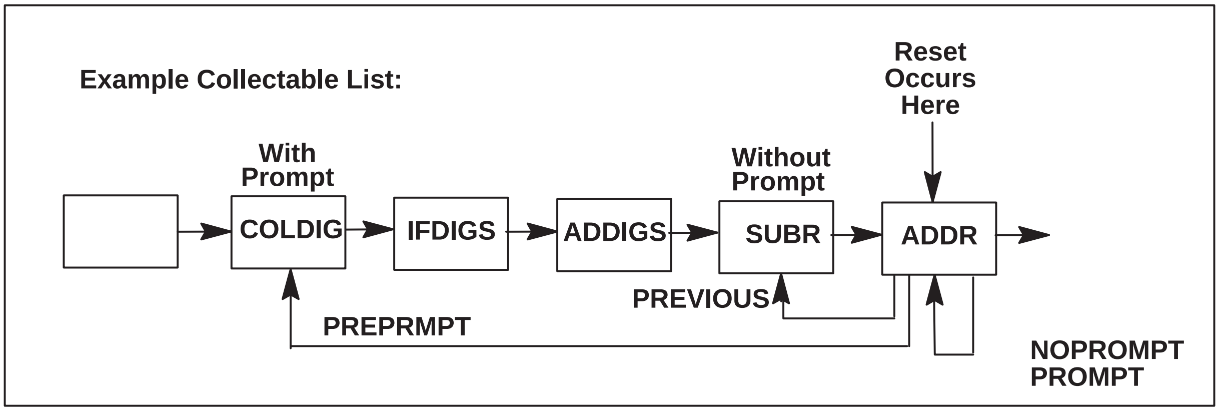
Note: If at least one digit is not collected from the originating agent, then there is no purpose to simply re-executing a digit collectable that processes the same digits. An inband digit collectable that retrieved all required digits from the digit buffer or from FILED digit sources does not collect any digits from the originating agent. Whether or not any digits are collected from the originating agent by the collectable is based on the previous execution of the collectable.

The PREPRMPT action identifies that the previously executed digit collectable containing a PROMPT option or RESET PROMPT option that collected at least one digit from the originating agent is reset and re-executed.

Note: A PROMPT or RESET PROMPT option is not needed if the collectable is not required to collect any digits from the originating agent. Whether or not any digits are collected from the originating agent by the collectable is based on the previous execution of the collectable.

Figure 16-22 contains an example collectable list for the reset with digit actions.

Figure 16-22
Reset with digit actions



NOPROMPT/PROMPT with digits action

The NOPROMPT and PROMPT actions identify that only the currently processing collectable is being reset. Before the collectable is re-executed, the following actions are performed:

- All digits placed in the digit buffer through execution of the collectable are erased.
- Messages consumed from the message center are reposted. (The messages are re-consumed when the collectable is re-executed.)

If the WDIGACT field is set to NOPROMPT, then a prompt is not applied on the reset occurrence. If the WDIGACT field is set to PROMPT, then a prompt is re-applied for the subscriber. The prompt applied may either be the original prompt (from the collectable's PROMPT option), or a reset prompt (from the RESET option's PROMPT option). The prompt used is determined by the logic in Table 16-4.

Table 16-4
Logic for determining with digits reset action prompt

Collectable PROMPT option provisioned	Reset option PROMPT option provisioned	Action
No	No	Cannot occur. At least one PROMPT option must be provisioned in order for the WDIGACT field to be set to PROMPT.
Yes	No	The original prompt is applied for the subscriber.
No	Yes	The RESET option prompt is applied for the subscriber.
Yes	Yes	The RESET option prompt is applied for the subscriber.

PREVIOUS/ PREPRMPT with digits action

The PREVIOUS and PREPRMPT reset actions each identify that the history of previously executed collectables is searched to find the most recently executed digit collectable matching the following criteria:

- For PREVIOUS, the most recently executed inband digit collectable that collected at least one digit from the originating agent is required.
- For PREPRMPT, the most recently executed inband digit collectable that collected at least one digit from the originating agent and applied a prompt for subscriber digit collection or contains a RESET option with a provisioned PROMPT option is required.

The search can be terminated in any of the following ways:

- When the beginning of the list of processed collectables is reached (through use of CLRFTRS (DPIDX)$, the CLRFTRS collectable becomes the first collectable in the list of processed collectables.) In this case, the first collectable is re-initialized and re-executed (no exceptions).
- When a signaling (SIG) collectable is reached where the method of receiving digits has changed (out-of-band to inband, or a change in pulse type). In this case, the search is discontinued and the collectable following the SIG collectable is re-initialized and re-executed.

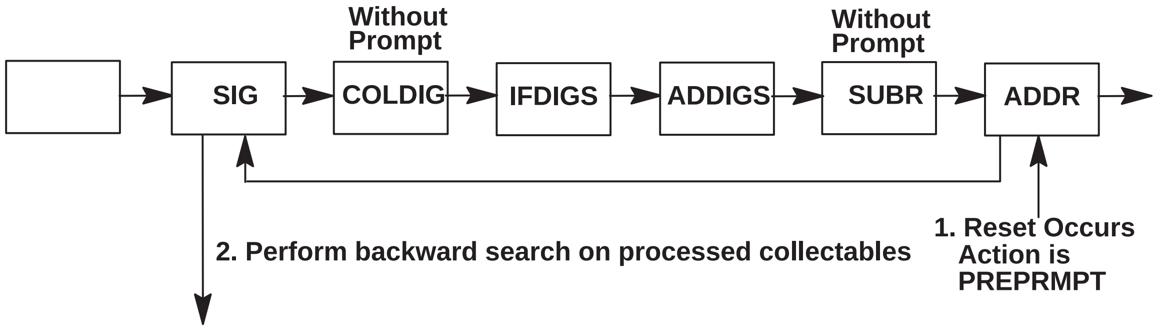
  If the SIG collectable had not changed the pulse type or signaling method, then the search continues.
- When a TERMINATE collectable is reached. In this case, the search is discontinued and the collectable following the TERMINATE collectable is re-initialized and re-executed.
- For an action of PREVIOUS, the search is terminated if an inband digit collectable is reached where the collectable requested at least one inband digit from the originating agent.
- For an action of PREPRMPT, the search is terminated if a digit collectable with a PROMPT option or RESET PROMPT option is reached where the collectable requested at least one inband digit from the originating agent.

Once the search has terminated, all processing performed by the collectables between the currently processing digit collectable and the “reset to” collectable is reversed (see Table 16-3 for a description of collectable reversal actions). Once the reversal actions are complete for all collectables up to the “reset to” collectable, then the “reset to action” is applied to the collectable.

Figure 16-23 contains an example collectable list for the reversal of collectable processing on a PREPRMPT reset option.

Figure 16-23
Reversal of collectable processing on a PREPRMPT reset example

Example collectable list:


Without Prompt
Without Prompt
SIG
COLDIG
IFDIGS
ADDIGS
SUBR
ADDR
2. Perform backward search on processed collectables
1. Reset Occurs Action is PREPRMPT

3. Search terminated when SIG collectable reached (assume change in pulse type). COLDIG collectable marked.

4. All processing between ADDR and COLDIG is reversed:

a. SUBR collectable:

- **– Digit Buffer returned to state prior to SUBR processing.**
- **– Message Center returned to state prior to SUBR processing.**
- **– Non-immediate features or treatments set by SUBR are reset.**

b. ADDDIGS collectable:

- **– Digit Buffer returned to state prior to ADDDIGS processing.**

5. Processing switches to the COLDIG collectable which is reset (reset to action) and re-executed.

After reversing the processing results of the executed collectables, the Reset To collectable is reversed and re-executed through the “reset to action”.

Reset without digits action

When reset occurs before any digits have been received from the subscriber, then the reset action to take is based upon the WODIGACT field of the RESET option. This field is provisioned with one of the four possible values:

- SAME
- PREVIOUS
- PREPRMPT
- DPIDX

The SAME action identifies that the processing digit collectable is re-executed, in a similar fashion as the “reset to action” method. The PREVIOUS action identifies that the previously executed digit collectable which collected at least one digit from the originating agent is reset and re-executed.

Note: If at least one digit is not collected from the originating agent, then there is no purpose to simply re-executing a digit collectable that processes the same digits. An inband digit collectable that retrieved all required digits from the digit buffer or from FILED digit sources does not collect any digits from the originating agent. Whether or not any digits are collected from the originating agent by the collectable is based on the previous execution of the collectable.

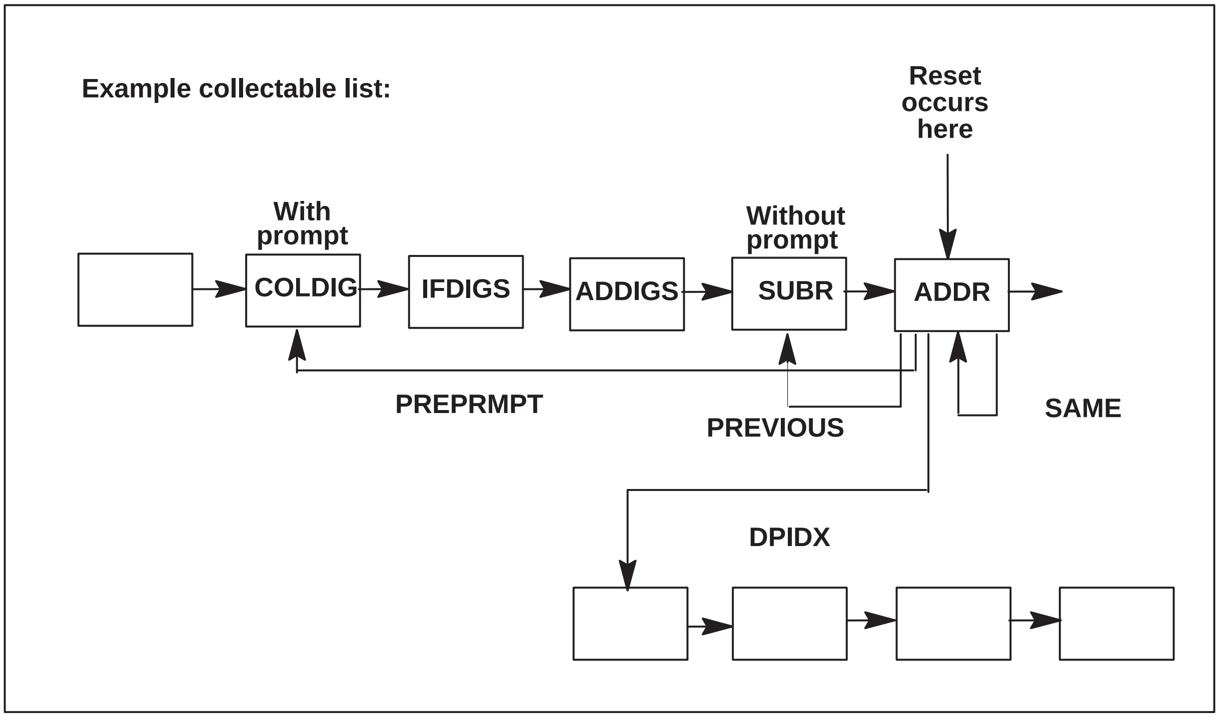
The PREPRMPT action identifies that the previously executed digit collectable which requested at least one inband digit from the originating agent and contains a PROMPT option or RESET PROMPT option is reset and re-executed.

Note: A PROMPT or RESET PROMPT option is not needed if the collectable is not required to collect any digits from the originating agent. Whether or not any digits are collected from the originating agent by the collectable is based on the previous execution of the collectable.

The DPIDX action identifies that collectable processing branches and continues processing on a new collectable list.

Figure 16-24 contains an example collectable list for reset without digit actions.

Figure 16-24
Reset without digit actions



SAME without digits action

The SAME action identifies that the currently processing collectable is re-initialized and the execution of the collectable starts over. The processing is identical to the reset to action, although no digits have to be removed from the digit buffer. See “Reset to action” on page 16-67 for a description.

PREVIOUS without digits action

The PREVIOUS without digits action is identical to the PREVIOUS with digits action, except no digits have to be removed from the digit buffer. See section ”PREVIOUS/ PREPRMPT With Digits Action” on page 16-75 for a description.

PREPRMPT without digits action

The previous prompt (PREPRMPT) without digits action is identical to the PREPRMPT with digits action, except no digits have to be removed from the digit buffer. See “PREVIOUS/ PREPRMPT with digits action” on page 16-75 for a description.

DPIDX without digits action

The FLEXDIAL index (DPIDX) value identifies a FLEXDIAL table index that is to apply to the call. This FLEXDIAL table index contains a list of collectables that are executed for the call as the reset action. An implied action of REPLACE applies to the handling of the collectable list identified by the FLEXDIAL index, where the remaining collectables to be processed are replaced by the collectable list identified.

Once the collectable list for the call is updated, handling of the currently processing collectable is aborted, and the next collectable in the list (the first collectable of the list identified by the FLEXDIAL index) is executed.

Unlike the PREVIOUS or PREPRMPT reset actions, the DPIDX action does not reset any feature or call data, message center messages, or digit buffer data for previously executed collectables. (Therefore, it is recommended that the CLRFTRS collectable be the first collectable to be executed in the DPIDX FlexDial collectable list.)

Deferred reset

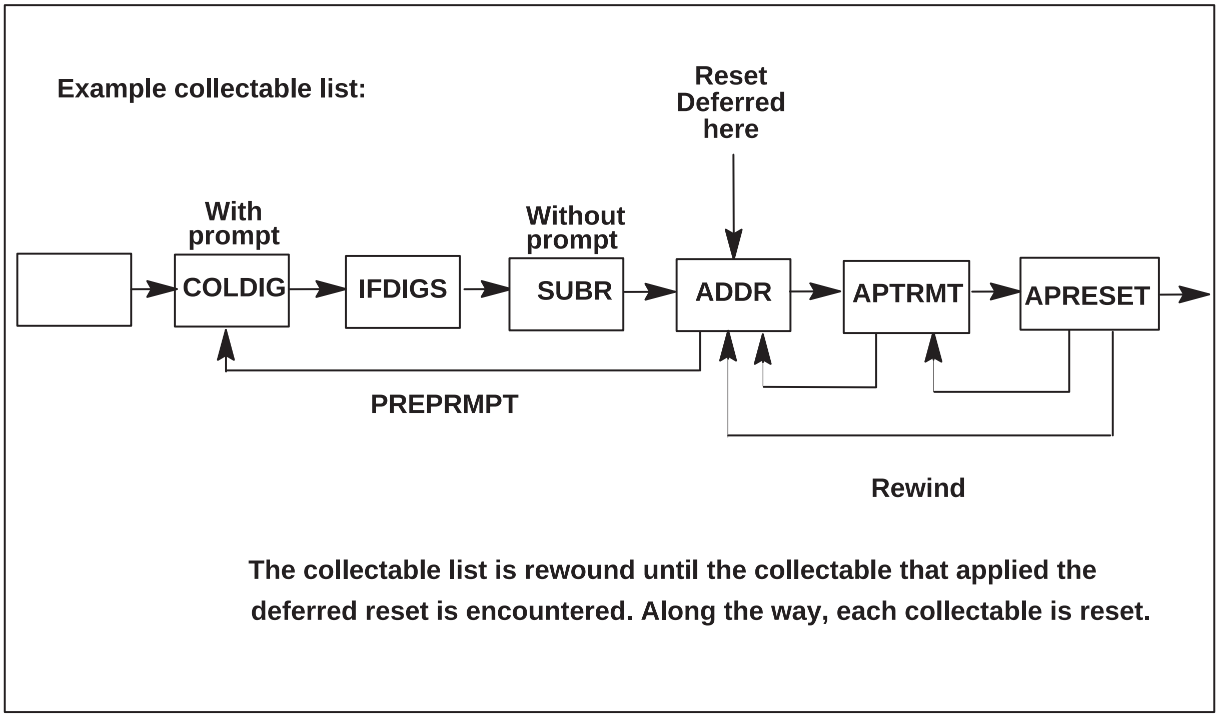
By setting the DEFERED field to Y in the SUBR, ADDR, or OLI digit collectable’s VALIDATE option, the desired reset action can be delayed until further collectable processing has occurred. When DEFERED reset is used, the currently processing digit collectable indicates reset but does not apply it until later in FLEXDIAL collectable processing. The reset indication is processed when one of the following events occur in FLEXDIAL processing:

- execution of an APRESET collectable
- execution of the TERMINATE or ROUTE collectable
- before triggering a CAIN event
- termination of the collectable list

When one of the above events occur and a reset was deferred, then the reset action indicated by the deferred reset is taken from the point in FLEXDIAL processing that the deferred reset was indicated. In other words, the collectable processing is rewound to the collectable that applied the deferred reset and, at that point, the desired reset action is taken.

Figure 16-25 contains an example collectable list for deferred reset actions.

Figure 16-25
Reset without digit actions


Example collectable list:
Reset Deferred here
With prompt
Without prompt
COLDIG
IFDIGS
SUBR
ADDR
APTRMT
APRESET
PREPRMPT
Rewind
The collectable list is rewound until the collectable that applied the deferred reset is encountered. Along the way, each collectable is reset.

Perform digit collection request

The digit collection request deals with the proper prompt handling and digit collection timers used for the process, as well as handling and reporting of the digits received. How the request is performed is based upon the digit collection signaling parameters and specific request information.

Prompts as proceed to send

The PROMPT option identifies an audible “proceed to send” signal that is provided to instruct the originating agent to begin transmitting digit information. If an audible proceed to send signal is desired or required for the digit collection request, then a PROMPT option must be provisioned for the inband digit collectable. Without a PROMPT option, no audible proceed to send signal is provided. (No dial tone is automatically provided. The PROMPT option must be used to identify all audible prompts applied for digit collection requests.)

Three variations of prompting capability is provided by the FlexDial framework for inband digit collection:

- announcement prompts

  Announcement prompts are specified by setting the PRMTTYPE field in the PROMPT option to ANNC and identify announcements provisioned in tables ANNS, ANNMEMS, and DRAMTRK.

- tone prompts—table TONES

  Table TONES tone prompts are initiated by setting the PRMTTYPE field in the PROMPT option to TONE and the TONETYPE field to TONES. Defined through table TONES, the prompt is typically specified as a “periodic” type tone (a tone which is played in a defined on-off pattern for a certain length of time).

- tone prompts—simple tones

  Simple tone prompts are initiated by setting the PRMTTYPE field in the PROMPT option to TONE and the TONETYPE field to STD. Through the PROMPT option STDTONE and TONEDUR fields, a standard tone type (tone frequency) and a tone duration are specified, providing a simple tone as the prompt signal.

  Note: A slight variation of a simple tone is the stutter dial tone, which is also supported by these fields.

Setting up an announcement prompt involves the allocation of two switch resources:

- an announcement member
- a network connection

If neither of these resources is available when the announcement prompt is required, then a No Software Resource (NOSR) Treatment is set and immediately applied to the call.

Tone prompts are always available as all necessary hardware to provide a tone prompt exists in the XPM for the agent.

A prompt signal applied for a digit collection request is disabled with the reception of the first digit from the originating agent. For announcement prompts, the announcement member and network connection are released from the call after reception of the first digit for the request.

Handling of collection timers

The agent terminal controller (XPM) is instructed to wait for a time value equal to the PSEIZTMR value before receipt of the first digit. If the first digit is not received within the value specified, then a timeout occurs and a permanent signal exception is generated in the central module (CM).

For MF digit collection, digits are collected by the agent terminal controller from the receipt of the first digit (identified by the FDIGMASK) through receipt of the last digit (identified by the LDIGMASK). When the final digit is received, then all the digits are reported to the CM. Upon the receipt of each digit, a timer is started with either the PDILTMR or MINRTMR timer value. If the timer expires before the next digit is received, then a partial dial timeout occurs and an exception is generated in the CM.

From receipt of the first digit through receipt of the last digit (with other digits in between) is termed a “stream” of MF digits.

Multiple streams of MF digits may be collected in order to fulfill the requirements of the digit collectable. Although if the minimum number of digits required has been received within an MF digit stream, then the requirements for the collectable are considered complete and no additional streams of MF digits are processed for the collectable.

For DTMF digit collection, the CM identifies the minimum number of digits to be collected by the agent terminal controller. Upon receipt of each digit, a timer is started with the PDILTMR timer value. If the timer expires before the next digit is received (before the minimum number of digits have been received), then a timeout occurs, a report is sent to the CM containing the digits collected before the timeout event occurred, and a partial dial timeout exception is generated in the CM.

Note: The partial dial timeout exception is generated because a digit matching the LDIGMASK value was not received, even though the actual number of received digits may be greater than the identified minimum value.

Once the minimum number of digits has been received, the agent terminal controller attempts to collect up to the MAX number of digits specified. Upon receipt of each digit, a timer is started with either the MINRTMR timer value. If the timer expires before the next digit is received, then CM is notified of the timeout, and digit collectable processing completes.

If the terminating digit, a reset digit, or a special first digit is received, then the CM is immediately notified of the collection event, and appropriate action is taken.

Receive digits

Digits received from the originating agent terminal controller are loaded into the digit buffer for processing, with the exception of the following digits:

- the first digit matching the FDIGMASK set for MF pulse type collection
- the last digit matching the LDIGMASK set for MF pulse type collection
- the terminating digit
- the reset digit

The MF FDIGMASK and LDIGMASK digits are used to determine the NOA of the called or calling party number.

While not being included in the digit buffer when received from the originating agent, the terminating and reset digits may be included in the digit buffer through use of the ADDDIGS, MODDIGS, or COPYDIGS collectables. The “Identify Availability of Required Digits” phase properly processes the terminating or reset digit whether or not it was received from the agent or through ADDDIGS, MODDIGS, or COPYDIGS collectable processing.

Digits required for the collectable are then consumed from the digit buffer.

Validate applicable digits

Before validation occurs, the digit collectable consumes the required and available optional digits from the digit buffer. It is on these digits that the validation occurs.

Validation of applicable digits for the collectable only occurs for digit collectables which contain a VALIDATE option. If validation is not performed, or validation is successful, then the features applicable to the digit collectable are executed.

Note: A VALIDATE option only exists for the SUBR, ADDR, and OLI digit collectables.

If validation is not successful, then the action specified by the FAILACT field is applied to the call. This action may be one of four values:

- TRMT
- RESET
- IGNORE
- DPIDX

TRMT action for validation failure

The treatment (TRMT) action identifies the treatment that is set for the call. All treatments set by digit collectables are delayed treatments with the exception of PSIG, PDIL, FNAL, and RESU treatments, meaning that the treatment is not immediately applied to the call.

If the OVERRIDE field is set to N and a treatment has already been identified for the call, then the treatment identified by the TRMT field is not set for the call (the treatment does not override a previous treatment set). If the OVERRIDE is set to Y, then the treatment identified by the TRMT field is set for the call irregardless of a previous treatment being set for the call.

Note 1: PSIG, PDIL, FNAL, and RESU treatments if set are immediately applied to the call.

Note 2: Use of the APTRMT collectable causes delayed treatments to be immediately applied to the call. To apply the treatment to the call requires the processing of collectables to be suspended and the call to continue on to analyze information and select route.

RESET action for validation failure

The reset validation failure action identifies that upon failing the validation attempt or occurrence of a timeout exception, the reset action as provisioned by the RESET option is applied to the collectable. Any treatment set by the validation failure or timeout exception is cleared, and the reset with or without digits action is applied. See section ”Application of reset capability” on page 16-64.

IGNORE action for validation failure

The ignore action identifies that validation failure is ignored, and any treatment set for the call by the collectable is cleared. The occurrence of a timeout exception is not ignored.

DPIDX action for validation failure

The FLEXDIAL index (DPIDX) action identifies that upon failing the validation attempt or occurrence of a timeout exception, the list of collectables provisioned against the FLEXDIAL index is applied to the call. How to apply the collectable list is identified by the ACTION field (INSERT, APPEND, REPLACE, or EXEC).

Once the collectable list for the call is updated, handling of the currently processing collectable is aborted, and any treatment set by the validation failure or timeout exception is cleared. The next collectable in the list (the first collectable of the list identified by the FLEXDIAL index) is then executed.

Deferred reset capability

All digit collectable validations indicating reset also have the option of delaying application of reset until later in FLEXDIAL processing. This is useful if one wants to go on to collect address digits after a subscriber has entered an invalid authcode, for example. After address collection, the APRESET collectable would be executed to cause processing of reset so that the subscriber could have the opportunity to re-enter their authcode instead of receiving immediate application of treatment to the call.

Apply digit collectable features

Each digit collectable identifies and processes its own set of applicable features. Typically applicable features are identified through validation.

Features are identified as those which are immediately executed, and those which are delayed. Information for delayed features is stored with the Feature Manager until such a time when the feature is to be setup or triggered.

Exception processing

Exceptions occur in digit collectable processing for incorrect use of the inband digit collectable, timeouts, unavailable resources, and validation failures. When an attempt to execute an inband digit collectable on a common channel agent is performed before use of the SIG collectable, then feature not allowed (FNAL) treatment is set for the call.

When a permanent signal timeout exception occurs, then permanent signal (PSIG) treatment is set for the call. When a partial dial timeout exception occurs and the minimum number of digits required has not been collected from the originating agent, then partial dial (PDIL) treatment is set for the call. When a resource unavailable exception occurs, then resource unavailable (RESU) treatment is set for the call.

When a partial dial timeout exception occurs for MF pulse type signaling before the last digit matching the LDIGMASK set is received, then PDIL treatment is set for the call.

When permanent signal, partial dial, or resource unavailable exceptions occur, all collectable processing is suspended and the treatment set is immediately applied to the call.

When a validation failure occurs for the call, then the action specified by the VALIDATE option FAILACT field is applied to the collectable/call.

Execution of remaining FlexDial collectables is not suspended when treatment is set for the call by a digit collectable validation attempts, providing the treatment is not set to PSIG, PDIL, FNAL, or RESU. Therefore a treatment set is not immediately applied and is termed a “delayed” treatment.

Note: For PSIG, PDIL, FNAL, and RESU, all collectable processing is suspended and the treatment set is immediately applied to the call.

Result of digit collection

The results of successful digit collection are on a per digit collectable type basis. Generally, the required digits are gathered and consumed by the executed digit collectable (although the purpose of the COLDIG digit collectable is to add digits to the digit buffer). These digits and other related information is typically captured in the call detail record (CDR) for the call.

For the COLDIG, SUBR, and ADDR collectables, the NOA value for the calling or called party may also be determined.

Setting calling party NOA

The calling party NOA may be set through processing of the following collectables:

- COLDIG

  Processing the COLDIG collectable sets the calling party NOA value when the PROCNOA field is set to CALLING.

- SUBR

  Processing the SUBR collectable sets the calling party NOA when the subscriber number FLEXTYPE used contains the FLEXTYPE table CALLING option provisioned (see “CALLING option” on page 5-8).

The calling party NOA value is determined by call processing as shown in Table 16-5.

Table 16-5
Logic for determining with digits reset action prompt

Signaling Method	Factor	Result
PTS	Number > 10 digits	NOA is set to INTL (international number)
	Number = 10 digits	NOA is set to NATL (national number)
	Number = 7 digits	NOA is set to SUBR (subscriber number)
	Otherwise	NOA is set to unknown

Processing the calling party NOA may also set the called party NOA value for MF type signaling. The value set may be overwritten during the collection and processing of address digits:

- MF STP digit is received in calling number stream.

  If an STP digit is received in the calling number digit stream, then the no number present, cut-through (NO_NUM_CT) NOA value is set for the called party number in addition to the value set for the calling party number.

- MF ST2P or ST3P digit is received in calling number stream.

  If an ST2P or ST3P digit is received in the calling number digit stream, then the “no number present cut-through” (950_CT) NOA value is set for the called party number in addition to the value set for the calling party number.

Setting called party NOA

The called party NOA may be set through processing of the following collectables:

- COLDIG

  Processing the COLDIG collectable sets the called party NOA value when the PROCNOA field is set to CALLED.

- ADDR

  Processing the ADDR collectable sets the called party NOA.

The calling party NOA value is determined by call processing as shown in Table 16-6.

Table 16-6
Calling party NOA assignment

Signaling Method	Factor	Result
PTS	Single zero digit received	NOA is set to NO_NUM_OP (no number present, operator requested)
	First three digits are 011	NOA is set to INTL (international number)
PTS	First three digits are 01N	NOA is set to INTL_OP (international number, operator requested)
	First two digits are 0N and the number of digits is equal to 8.	NOA is set to SUBR_OP (subscriber number, operator requested)
	First two digits are 0N and the number of digits is equal to 11.	NOA is set to NATL_OP (national number, operator requested)
	First digit is not 0, and the number of digits is greater than 11.	NOA is set to INTL (international number)
PTS	First digit is not 0 or 1, and the number of digits is greater than 10.	NOA is set to INTL (international number)
	First digit is not 0, number of digits is 10.	NOA is set to NATL (national number)
	First digit is a 1 and the number of digits is 11.	NOA is set to NATL (national number)
PTS	First digit is not 0, and the number of digits is 7.	NOA is set to SUBR (subscriber number)
	Otherwise	NOA is set to SUBR

The called party NOA can also be set through special circumstances when processing a calling party number digits. See “Setting calling party NOA” on page 16-86. The NOA value set through calling party number processing may be overwritten by the value determined through processing of called party address digits later in the call.

Processing the originating line information digits may also set the called party NOA if the information digits are screened using the INTOA or INTDD call condition values. The NOA value set through originating line information digit processing cannot be overwritten by the value determined through processing of called party address digits later in the call (specifically, through COLDIG [where PROCNOA = CALLED] or ADDR/ADDRPARM collectable processing.)

Note 1: The NOA used for the call can always be modified by the MODNOA collectable.

Note 2: Only the call conditions INTOA and INTDD set the called party NOA. Other call conditions or pretranslator selectors used do not modify the nature of address of either the called or calling party.

Restrictions and limitations for inband digit collection

The following restrictions and limitations apply to inband digit collection:

- To perform inband digit collection on common channel agents, the signaling (SIG) collectable must be executed prior to execution of the inband digit collectable. If the SIG collectable is not executed, then execution of the digit collectable results in the immediate application of FNAL Treatment for the call.
- A dial tone prompt is not automatically provided for DTMF digit collection. The PROMPT option must be used to provide any desired or required audible proceed to send signal.

General application of out-of-band digit collection

Out-of-band digit collection collects digit information from the originating agent, where digits have been included in a message delivered in the signaling channel of the connection to the originating agent. The received digit information is eventually used to connect the call to the destination desired by the subscriber.

The FlexDial framework provides the ability to process digits from an originating agent using a message based protocol through the COLPARM, SUBRPARM, ADDRPARM, OLIPARM, and CICPARM digit collectables. The FGDPARM collectable combines the functions of the above collectables to process the SS7 FGD signaling protocol in one collectable. The validation of received digits occurs as directed by the specific collectable.

The out-of-band collectables are identical in validation and service application as their inband siblings.

Table 16-7 contains in band and out-of-band digit collectable applications.

Table 6-7
Inband and out-of-band digit collectable applications

Inband version	Out-of-band version	Description
COLDIG	COLPARM	collect raw digit and NOA information
SUBR	SUBRPARM	collect and process subscriber numbers
ADDR	ADDRPARM	collect and process called party address digits
OLI	OLIPARM	collect and process originating line information digits
CIC	CICPARM	collect and process carrier identification code digits
N/A	FGDPARM	perform within one collectable the same functions as COLPARM, SUBRPARM, ADDRPARM, OLIPARM, and CICPARM for the SS7 FGD protocol
REPLDIG		collect inband digits to replace identified digits within the digit buffer

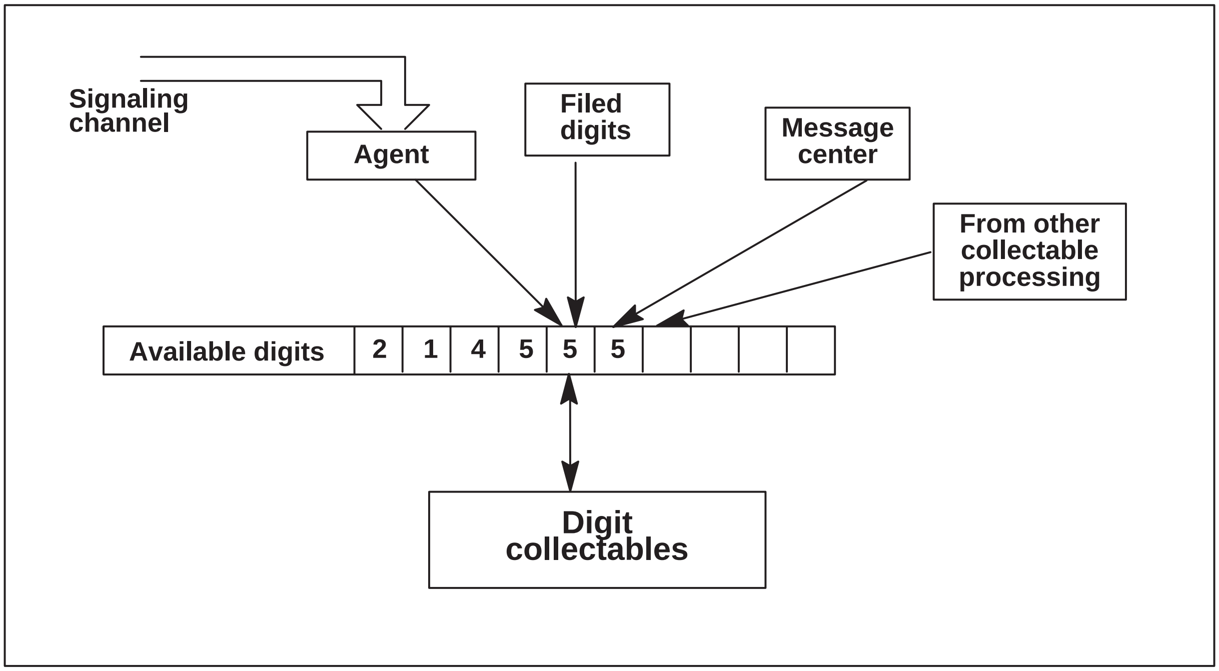
Out-of-band digit collection can occur only on common channel signaling (CCS) type agents. If the out-of-band collectables are executed on per-trunk signaling (PTS) agents, then FNAL Treatment is immediately applied to the call.

Application

All out-of-band digit collectables (COLPARM, SUBRPARM, ADDRPARM, OLIPARM, CICPARM, FGDPARM) retrieve digits from the initial address message (IAM) received on originating agent’s signaling channel. The retrieved digits are stored in the digit buffer. The collectable then consumes the required number of digits from the digit buffer for collectable processing.

Figure 16-26 contains sources of digits for digit collectables.

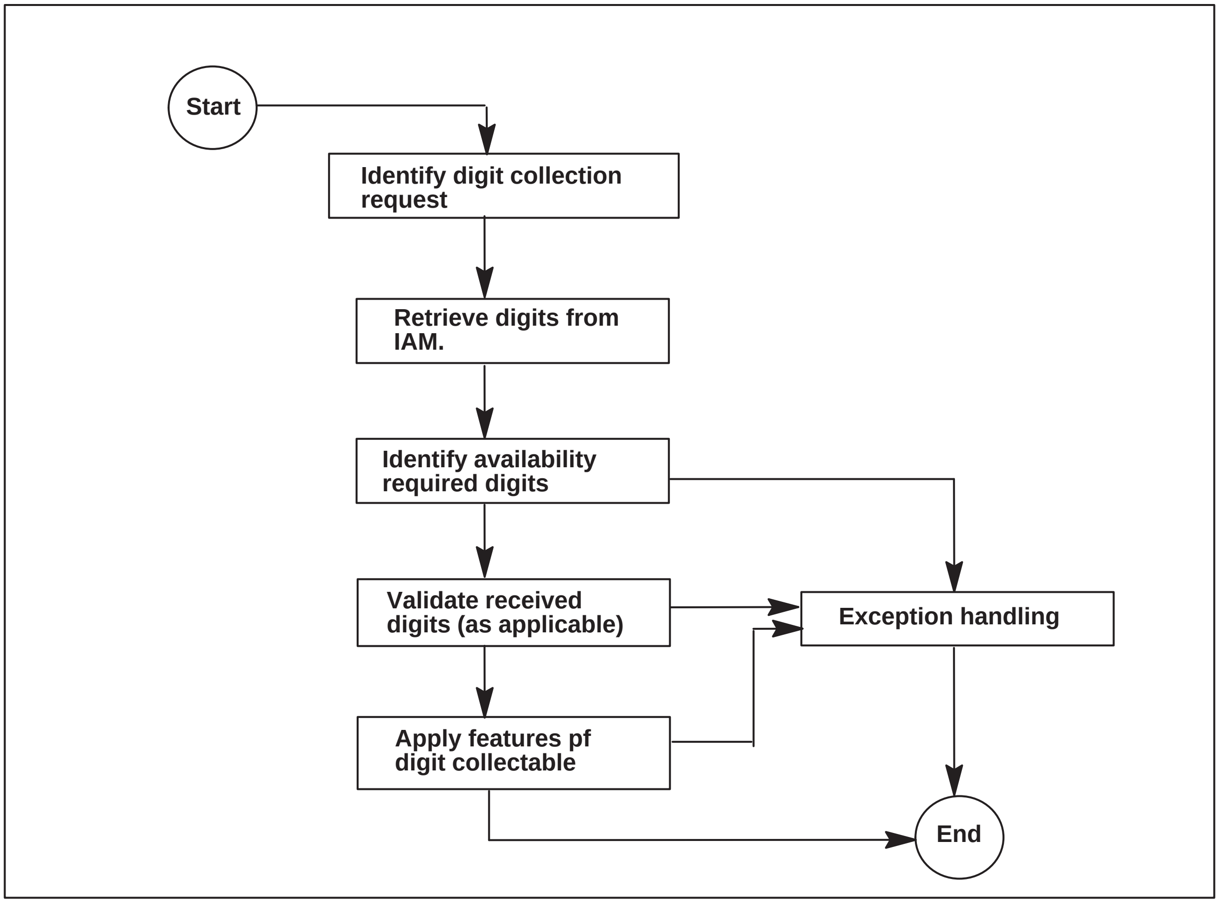
Figure 16-26
Sources of digits for digit collectables



For out-of-band digit collection, with the exception of the COLPARM digit collectable and the SUBRPARM digit collectable where the REPBDIGS field is set to Y, digit collectables consume all out-of-band digits retrieved from the IAM, and also consume all filed digits identified for the collectable.

Figure 16-27 shows the out-of-band digit collectable process flow..

Figure 16-27
Out-of-band digit collection process flow


Start
Identify digit collection request
Retrieve digits from IAM.
Identify availability required digits
Validate received digits (as applicable)
Exception handling
Apply features pf digit collectable
End

This flow identifies the following processing areas for digit collection:

- identify digit collection request
- retrieve digits from IAM
- identify availability of required digits
- validate applicable digits
- apply digit collectable features
- exception handling

Identify digit collection request

An out-of-band collection request for digits from the originating agent occurs through the execution of the COLPARM, SUBRPARM, ADDRPARM, OLIPARM, and CICPARM digit collectables or through the execution of FGDPARM. Each collectable provides specific information for the out-of-band digit request. This information includes:

- applicable filed digits (FILED option)
- up to two identified message parameters that contain the requested digits
- applicable validation and other digit collectable service processing information

The execution of out-of-band digit collectables is not allowed for per-trunk signaling (PTS) agents. An attempt to execute an out-of-band collectable on PTS agents causes the out-of-band digit collection application to be aborted and suspends execution of collectables. Additionally FNAL Treatment is immediately applied to the call.

The use of the SIG collectable for CCS agents enables either inband or out-of-band digit collection.

Note: Out-of-band digit collection is not disabled through execution of the SIG collectable.

Retrieve digits from IAM

Out-of-band collectables differ in requirements from inband collectables in that the switch has already received the information required for out-of-band collectables, whereas inband collectables must typically gather more information from the originating agent.

The out-of-band collectable may identify up to two parameters that are used to retrieve digits from the IAM for the collectable, but both parameters cannot be used in combination. If the first identified parameter is present in the message, then it provides digits and NOA information for the collectable. The second identified parameter, regardless of its availability in the message, is ignored.

If the first parameter is not present in the message, and the second parameter is present in the message, then the second parameter provides the digits and NOA information for the collectable.

If neither parameter is available in the message, then the out-of-band collectable relies on other sources for digit information.

Table 16-8 contains out-of-bound parameter identification.

Table 16-8
Out-of-band parameter identification.

Parameter name	CCS7 parameter	Description of parameter use in IAM
CLD	Called Party Address	mandatory
CLG	Calling Party Address	optional ID = #0A
CHG	Charge Number	optional ID = #EB
OLI	Originating Line Information	optional ID = #EA
GADDR	Generic Address	optional ID = #C0. One byte subtype.
GDIGS	Generic Digits	optional ID = #C1. One byte subtype.
TNS	Transit Network Selector	optional ID = #23. Used for carrier identification code digits.
CIP	Carrier Identification Parameter	optional ID = #C5. Used for carrier identification code digits.
TNSCKT	Transit network selector	optional ID = #23. The parameter circuit code value and circuit code digits are used in combination with tables OCCNAME, OCCINFO, and CKTDIGIT to identify the digits placed in the digit buffer. If the proper provisioning is not performed for these tables, no digits are added to the digit buffer.

Identify availability of required digits

Unlike inband digit collectables, out-of-band digit collectables do not contain MIN and MAX values. The out-of-band digit collectable relies on the availability of all the required digits in the identified message parameter or in available digits in the digit buffer, in combination with identified FILED digits.

Note: FILED digits can be received either through the FILED digit collectable option, or through receipt of a FILED message with digits from the message center.

Table 16-9 contains identification of available digits for out-of-bound digit collectables.

Table 16-9
Identification of available digits for out-of-band digit collectables

IAM parameter present	Filed digits available	Available digits in digit buffer	Digits used by out-of-band collectable
No	No	No	A permanent signal exception is generated for the call, with the exception of the COLPARM collectable. ***Note:*** The COLDIG collectable supports a MIN value of zero. Similarly, the COLPARM can handle the appropriate parameter not available in the message and simply allow the call to proceed.
No	No	Yes	All available digits in the digit buffer are consumed by the collectable.
No	Yes	No	All filed digits are consumed by the collectable.
No	Yes	Yes	Only filed digits are consumed by the collectable. Available digit in the digit buffer remain unaffected.
Yes	No	No	All digits contained in the parameter are consumed by the collectable.
Yes	No	Yes	All digits contained in the parameter are consumed by the collectable. Available digits in the digit buffer remain unaffected.

Note: For the IAM parameter present column, “No” indicates that neither of the identified parameter is present in the message. “Yes” indicates that one of the parameters was available.

—continued—

Table 16-9
Identification of available digits for out-of-band digit collectables (continued)

IAM parameter present	Filed digits available	Available digits in digit buffer	Digits used by out-of-band collectable
Yes	Yes	No	Only filed digits are consumed by the collectable. Available digit in the digit buffer remain unaffected.
Yes	Yes	Yes	All digits contained in the parameter and all filed digits are consumed by the collectable. Available digits in the digit buffer remain unaffected.

Note: For the IAM parameter present column, “No” indicates that neither of the identified parameter is present in the message. “Yes” indicates that one of the parameters was available.

—end—

Once available digits have been identified, one of the following actions occur:

- Available digits cannot be identified for the collectable, and a permanent signal exception is generated for the call, with the exception of the COLPARM collectable. For the COLPARM collectable, if the specified parameter is not contained in the message, then the COLPARM collectable is marked as completed and call processing continues with the next collectable in the list.
- Available digits are identified. In this case, a number of additional actions occur:
  - — The identified digits are consumed by the digit collectable.
  - — Validation of the digits occur as applicable to the digit collectable.

  Note: Validation occurs for those digit collectables provisioned with a VALIDATE option.

  - — Execution of digit collectable features occur as applicable to the digit collectable.

Because out-of-band collectables do not contain MIN/MAX values, all digits from the identified location are consumed by the collectable. There is not a possibility for excess digits to remain in the digit buffer.

Note: The exception to this rule is for ADDRPARM processing. Because table STDPRTCT contains MIN/MAX values, the MAX value can be used to determine if excess digits are being consumed by the collectable. In this case, the excess digits are not consumed and remain in the Digit Buffer.

Because out-of-band digit collectables do not collect digits directly from the subscriber, these collectables do not process terminating or reset digits.

Application of filed digits

The application of filed digits for out-of-band digit collectables is similar to the application of filed digits for inband collectables.

All filed digits are consumed by the out-of-band digit collectable.

Validate applicable digits

Validation of applicable digits for out-of-band digit collectables is identical to validation for inband digit collectables.

Apply digit collection features

The application of out-of-band digit collectable features is identical to the application of features for inband digit collectables.

Exception processing

Exception handling for out-of-band digit collectables is identical to exception handling for inband digit collectables.

Result of digit collection

The results of digit collection are on a per-digit collectable type basis. Generally, the required digits are gathered and consumed by the executed digit collectable (although the purpose of the COLPARM digit collectable is to add digits to the digit buffer). These digits are typically captured in the call detail record (CDR) for the call.

For the COLPARM, SUBRPARM, and ADDRPARM collectables, the NOA value for the calling or called party may also be set.

Setting calling party NOA

The calling party NOA can be set through processing of the following collectables:

- COLPARM

  Processing the COLPARM collectable sets the calling party NOA value when the PROCNOA field is set to CALLING.

- SUBRPARM

  Processing the SUBRPARM collectable sets the calling party NOA when the subscriber number FLEXTYPE used contains the FLEXTYPE table CALLING option provisioned (see “CALLING call processing application” on page 17-7).

The calling party NOA value is determined by call processing as shown in Table 16-10.

Table 16-10
Calling party NOA assignment

Signaling method	Factor	Result
CCS7	Message Parameter	Specific NOA value is contained within the address parameter in the signaling message. For parameters that do not contain a NOA value, the value of UNKNOWN is set.

Setting called party NOA

The called party NOA can be set through processing of the following collectables:

- COLPARM

  Processing the COLPARM collectable sets the called party NOA value when the PROCNOA field is set to CALLED.

- ADDRPARM

  Processing the ADDRPARM collectable sets the called party NOA.

The calling party NOA value is determined by call processing the same way as shown in Table 16-10.

Processing the originating line information digits can also set the called party NOA if the information digits are screened using the INTOA or INTDD call condition values. The NOA value set through originating line information digit processing may not be overwritten by the value determined through processing of called party address digits later in the call (specifically, through COLDIG [where PROCNOA = CALLED] or ADDR/ADDRPARM collectable processing).

Note: Only the call conditions INTOA and INTDD set the called party NOA. Other call conditions or pretranslator selectors used do not modify the nature of address of either the called or calling party.

Restrictions and limitations for out-of-band digit collection

Out-of-band digit collection is not permitted on inband or PTS type agents. If an out-of-band digit collectable is executed on a PTS agent, then execution of the collectable results in the immediate application of FNAL treatment for the call.

Raw data digit collectable (COLDIG/COLPARM)

The Collect Digits (COLDIG/COLPARM) collectables provide the ability to collect raw information in the form of digits and a NOA value from the originating agent. The collect digits collectables do not process or validate the digits received. Other digit or sequence collectables process the information.

Application

The COLDIG collectable follows the general application of inband digit collection as described in “Application of inband digit collection” on page 16-51.

The COLPARM collectable follows the general application of out-of-band digit collection as described in “Application of out-of-band digit collection” on page 16-57.

Identify digit collection request

The messages received from the message center may alter the request parameters for the collect digits collectable.

The FILED message identifies filed digits for the collectable. See “Application of filed digits” on page 16-60 for more information on the handling of filed digits.

The MINMAX message identifies the minimum and maximum values for the digit collection request, overriding the values provisioned in the MIN and MAX fields of the collectable (see “Identify availability of required digits” on page 16-57). The MINMAX message is only applicable for the COLDIG collectable. The message is consumed, but not used by the COLPARM collectable.

Note: If the REPOST indicator is set for the MINMAX message, the message is still reposted by a COLPARM collectable.

Perform digit collection request

The COLDIG collectable follows the general application of requesting digits from the agent terminal controller as described in “Perform digit collection request” on page 16-80.

The COLPARM collectable does not perform a request of inband digits from the agent terminal controller.

SIGFEAT capability

The SIGFEAT option identifies two unique signalling features for the DTMF inband digit collection request, PDIL234 and SPLFDIG/SPLFTMR.

The PDIL234 timer identifies that a special partial dial timer is in effect after collection of the second, third, and fourth digits. The PDILTMR or MINRTMR timer value as appropriate is used after collection of the other digits. A value of zero (0) for the PDIL234 field indicates that this digit collection signalling feature is not in effect. Additionally, the MAX value for the digit collection request must be greater than or equal to five (5) for activation of the PDIL234 timer.

The SPLFDIG and SPLFTMR fields identify that special first digit timing is in effect for the received special first digit, and that special first digit handling is in effect for the asterisk and octothorpe digits, separate from their possible use as reset and/or terminating digits. If the first digit received matches the SPLFDIG identified, then the SPLFTMR timer value is used until the receipt of the second digit instead of using the PDILTMR or MINRTMR timer value. Currently due to implementation restrictions, only a single zero digit may be used for the SPLFDIG field when the SPLFTMR field is provisioned with a value greater than zero.

When the SPLFTMR field is provisioned with a value greater than zero, then special digit handling is also enabled for receipt of the asterisk or octothorpe digit as the first digit, and specifies that:

- The asterisk or octothorpe is treated as a normal digit and not as a reset or terminating digit. Reset or terminating digit handling is reasserted for the collection process after receipt of the first digit.
- The MIN/MAX values for the collection process are set to two (2) and five (5) respectively, regardless of the MIN/MAX values provisioned or used for the digit collection request. For Smart ADDR MIN/MAX processing, the revised collection request must fulfil all digit requirements for the ADDR collectable. Note that with an asterisk or octothorpe as the first called party address digit, the special pretranslator name provisioned in table PRETNAME is used for STDPRTCT table screening of the received address digits.

Typically special first digit handling of the asterisk or octothorpe as the first digit is used for special hotline or speed dial number calling, and a short SPLFTMR timer value is used to provide a quicker time-out for zero-minus operator handled calls.

Interactions

The SIGFEAT option is designed and typically used for called party address digit processing. When interworking with Smart ADDR MIN/MAX Collection, the SIGFEAT timer specifications provide additional timer requirements for the first phase of the digit collection process. Additionally, receipt of an asterisk or octothorpe as the first digit when the SPLFTMR timer is enabled guarantees that a maximum of five digits are collected for the ADDR digit collectable.

Currently the special AST / OCT first digit handling supported by the SIGFEAT option is used for speed dialing services.

Restrictions and limitations

The following three call processing restrictions or limitations are defined for the SIGFEAT option:

- The PDIL234 timer must be set to a value greater than zero and the MAX value for the collection request must be greater than or equal to five in order to enable the PDIL234 timer for the digit collection request.
- Asterisk and octothorpe special first digit handling is enabled when the SPLFTMR timer field is set to a value greater than zero. Asterisk and octothorpe special first digit handling involves treating the asterisk or octothorpe as a normal digit and resetting the MIN/MAX for the request to two and five digits respectively.

- The SIGFEAT option is not applicable for and subsequently ignored for MF inband digit collection. SIGFEAT only applies to DTMF inband collection requests.

Validate applicable digits

The COLDIG/COLPARM collectables do not perform any validation on the digits received. Digits collected by the collectable are not stored in any CDR field.

Apply digit collectable features

The COLDIG digit collectable feature collects digits that are to be processed by other sequence and digit collectables. Therefore, the required and optional digits collected by the COLDIG collectable remain in the digit buffer and are not consumed by the COLDIG collectable.

The COLDIG digit collectable can be used multiple times during collectable processing to gather raw digit information from the originating agent. However, COLDIG collectables cannot be used consecutively as each subsequent collectable, finding the required number of digits already in the digit buffer, will not perform any action for the call.

Results of digit collection

The collect digits collectables gather and append digits received to those in the digit buffer for use by other digit and sequence collectables.

The PROCNOA field identifies how the COLDIG/COLPARM collectable determines/sets the NOA value. The collect digits collectable either ignores the NOA value, or it sets the calling or called NOA value for the call.

If the PROCNOA value identifies that the calling party NOA value is set, then the following field in the CDR is also captured:

- CLGNOA

  This field identifies the NOA value set during call processing for the calling party subscriber number.

If processing the calling party NOA value also sets the called party NOA value (see “Setting Calling Party NOA” on page 16-98), then the following field in the call detail record is also captured:

- CLDNOA

  This field identifies the NOA value set during call processing of the called party address.

If the PROCNOA value identifies that the called party NOA value is set, then the following fields are captured in the CDR:

- CLDNOA

  This field identifies the NOA value set during call processing of the called party address.

- DIALEDNO

  If the dialed number has not been captured for the call, then this field is captured and identifies the called party address digits received before they are processed by the call.

- DIALNOA

  If the dialed number has not been captured for the call, then this field is also captured and holds the NOA of the dialed number.

Use case scenario

The collectable is used in the following call scenario:

- Collect digits and NOA values before using conditional branching and digit manipulation collectables.

  Using the COLDIG collectable, raw information can be gathered separately from the processing of that information, which may take several steps. More importantly, the information can be gathered with the proper prompts and reset capabilities.

Restrictions and limitations

COLDIG collectables cannot be “chained” together to collect raw digit data due to how the availability of required digits is identified for inband digit collection (see “Identify digit collection request” on page 16-57). However, COLPARM collectables can be chained together to insert digits from multiple parameters into the digit buffer.

Note: The net result of the subsequent COLDIG collectable would be to shift digits by taking the MAX number of digits from the start of the buffer and appending the digits to the end of the buffer.

Subscriber Number collectable (SUBR/SUBRPARM)

The Subscriber Number (SUBR, SUBRPARM) collectables collect subscriber numbers from the originating agent. The type of subscriber number digits is identified by the FLEXTYPE field entry. The SUBR collectable collects only digits that are processed as subscriber number digits.

Application

Subscriber number collectables are executed when the call is identified as a “calling party billed” call versus a “called party billed” call. For “called party billed calls,” SUBR or SUBRPARM collectables may or may not be executed based upon the following criteria:

- If the subscriber number collectable is included in the collectable list to be processed before the call is identified as a called party billed call, then the collectable is not executed.
- If the subscriber number collectable is included in the collectable list to be processed after the call is identified as a called party billed call, then the collectable is executed.

Note: Called party billed calls can be identified through N00 address application processing or through the FLEXFEAT CLDPBILL option. The call can be reset to calling party billed through use of the CLRFTRS collectable with the CLDPBILL option.

Subscriber number collectables are always executed when the call is marked as “calling party billed.”

Processing the CLDPBILL FLEXFEAT option by a subscriber number collectable identifies the call as called party billed, but does not affect the remaining processing of the subscriber number collectable.

The SUBR digit collectable follows the general application of inband digit collection; the SUBRPARM digit collectable follows the general application of out-of-band digit collection

Identify digit collection request

The messages received from the message center may alter the request parameters for the subscriber number collectable:

- The MINMAX message identifies the minimum and maximum values for the digit collection request, overriding the values provisioned in the MIN and MAX fields of the collectable (see “Identify availability of required digits” on page 16-57). The MINMAX message is only applicable for the SUBR collectable. The message is consumed, but not used by the SUBRPARM collectable.

> ***Note:*** If the REPOST indicator is set for the MINMAX message, the message is still reposted by a SUBRPARM collectable.

- The FILED message allows for the use of filed digits, instead of collecting raw information from the originating agent.
- The PROMPT message is used to modify the prompt played by a SUBR digit collectable. The MSGCTR PROMPT option fields have a similar definition to the PROMPT option for the digit collectables. This message is ignored by the SUBRPARM digit collectables.
- The VALIDATE message, in conjunction with the FAILACT message and either the MATCH or INDEXES message, allows call processing events to determine how subscriber digits are validated. The VALIDATE message overrides the VALIDATE option provisioned for the SUBR/SUBRPARM collectables in table FLEXDIAL.
- The MATCH message identifies the subscriber number digits to be used for validation.
- The INDEXES message identifies the table FLEXVAL indexes to be used for validation of the identified subscriber number type.
- The FAILACT message identifies a fail action to replace the FAILACT provisioned for the SUBR/SUBRPARM collectable in table FLEXDIAL.
- The FEATIGN message identifies a list of table FLEXFEAT features that the SUBR or SUBRPARM collectable processing must ignore.

Note: For more information on these messages, see “SUBR addressee messages” on page 4-9.

Validate applicable digits

The subscriber number collectable supports multiple types of validation for subscriber number digits:

- MATCH validation
- FLEXVAL validation
- CITYCODE validation
- CASUAL number blocking

For all types of subscriber number validation, the validation result is overruled and considered successful by default if the call is identified as called party billed through previous use of the CLDPBILL FLEXFEAT table option, or through use of the CLDPBILL option for the currently processing collectable. (The called party billed option can only be identified for the currently processing collectable through FLEXVAL table validation.)

If no validation attempt occurs, then the FLEXTYPE specific OM NOVAL counter is incremented.

MATCH validation

Match type validation occurs when the received digits must be the same as the digits identified by the collectable; that is, there is only one sequence of digits that validate successfully.

Identification of the use of the match validation scheme can occur through one of the following:

- use of the SUBR/SUBRPARM VALIDATE option with the in the switch validation type (IVALTYPE) set to MATCH
- use of the SUBR/SUBRPARM VALIDATE option in addition to receiving a MSGCTR MATCH message type for the SUBR digit collectable

Use of the MSGCTR MATCH message type overrides any type of validation that is set in the VALIDATE option, and MATCH validation is performed for the collectable.

Validation of the subscriber number for MATCH validation is successful if the following condition is met:

- The received digits are identical to the digits provisioned with the MATCH option, or contained in a received MSGCTR MATCH message.

If the received digits do not match, then the validation is unsuccessful. For unsuccessful validation, an exception is generated and the action identified by the FAILACT field is executed.

If the FLEXLOG option is provisioned for the subscriber number type in table FLEXTYPE, then on match validation failure, a FLEX 302 validation failure log is generated with the following validation failure reason:

```
MATCH Screening Failure
```

When a match validation attempt occurs, either one of two operational measurement counters for the specific subscriber number type are incremented:

- VALSUCC—match validation successful counter
- VALFAIL—match validation failure counter

For successful validation, the features identified by the FLEXTYPE table entry are applied to the call.

FLEXVAL validation

FLEXVAL table validation occurs when the received subscriber number digits are screened in table FLEXVAL.

Identification of the use of the FLEXVAL table validation scheme occurs through:

- use of the SUBR/SUBRPARM VALIDATE option with the in the switch validation type (IVALTYPE) set to FLEXVAL
- use of the SUBR/SUBRPARM VALIDATE option in addition to receiving a MSGCTR INDEXES message type for the SUBR digit collectable

FLEXVAL table validation requires three pieces of information to screen the number in table FLEXVAL:

- subscriber number type

  The subscriber number type is the FLEXTYPE table index provisioned in the SUBR/SUBRPARM collectable FLEXTYPE field.

- numeric index

  The numeric index to table FLEXVAL is retrieved from VALIDATE option FLEXIDXS field or the MSGCTR INDEXES message type.

- subscriber number digits

  The digits received for the collectable provide the third part of the index into the FLEXVAL table.

The number of digits validated is the lesser count value of either:

- the number of received digits for the subscriber number collectable
- the FLEXIDXS DIGCNT field provisioned in the VALIDATE option

In addition to the identified FLEXVAL table index, the FLEXIDXS vector field identifies the number of received digits to screen using the index. If the value set is less than the actual number of identified digits for the collectable, then only the number of digits identified by the FLEXIDXS DIGCNT field are validated by the collectable.

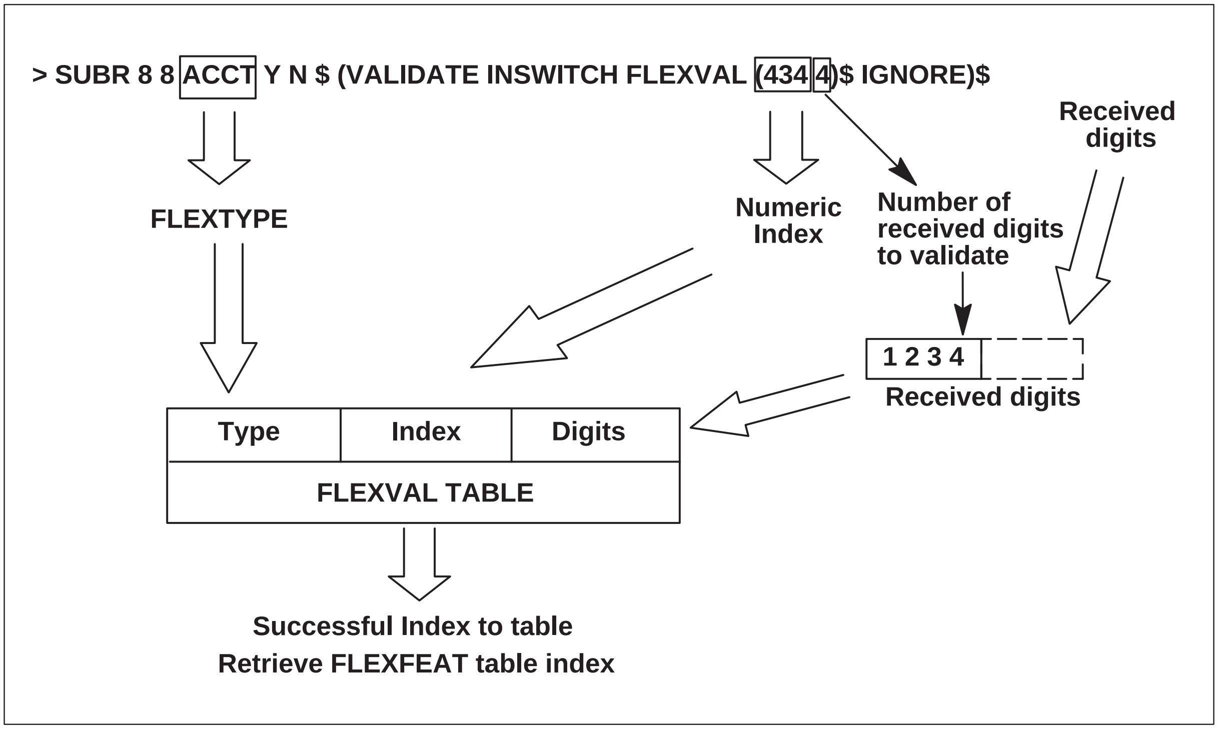
If the value in the FLEXIDXS DIGCNT field is greater then the actual number of digits identified for the collectable, then only the actual number of digits identified are validated, and the FLEXIDXS DIGCNT field is ignored.

Note: When the MIN value is less than the MAX value for the collectable, this scenario is likely to frequently occur.

Partial validation of the number can occur if the FLEXIDXS DIGCNT field is set to a value less than the number of received digits.

Figure 16-28 illustrates FLEXVAL table validation.

Figure 16-28
FLEXVAL table validation


> SUBR 8 8 ACCT Y N $ (VALIDATE INSWITCH FLEXVAL (434 4)$ IGNORE)$
FLEXTYPE
Numeric Index
Number of received digits to validate
Received digits
1 2 3 4
Received digits
Type
Index
Digits
FLEXVAL TABLE
Successful Index to table
Retrieve FLEXFEAT table index

The FLEXIDXS field may also identify up to two FLEXVAL table indexes, with the corresponding number of digits to validate at each index. In this double index validation scheme, the first index screens the first *x* number of digits received, and the second index screens the next *y* number of digits received. Both screenings must pass to successfully validate the digits.

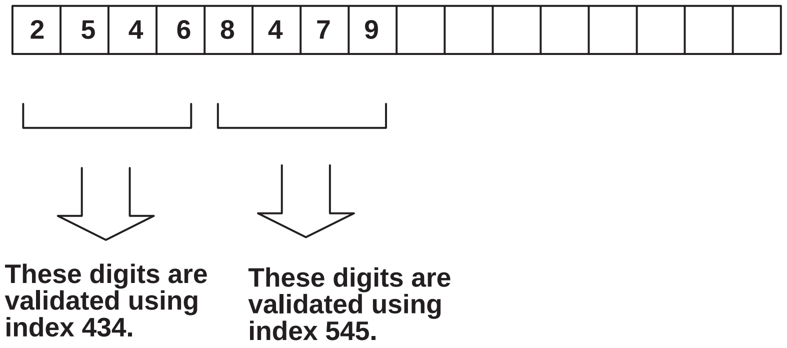
Figure 16-29 illustrates the double index validation scheme.

Figure 16-29
Double index validation scheme

```
> SUBR 8 8 ACCT Y N $ (VALIDATE INSWITCH FLEXVAL (434 4)(545 4)$ IGNORE)$
```

Eight digits are received by the collectable. Four digits are to be validated using index 434, and then four more digits are to be validated using index 545.

SUBR Collectable Digits



2	5	4	6	8	4	7	9								

These digits are validated using index 434.
These digits are validated using index 545.

If the number of digits identified for the subscriber number collectable is less than the FLEXIDXS DIGCNT value, then only the number of digits identified are validated. If there are no available digits to validate at the second FLEXVAL index specified, then the second validation attempt is not performed, and the subscriber number is processed as a single validation attempt.

Use of the MSGCTR INDEXES message type overrides any type of validation set by the VALIDATE option, and FLEXVAL table validation is performed for the collectable. If the subscriber number collectable VALIDATE option does not identify INSWITCH FLEXVAL table validation for the collectable, then the index specified in the MSGCTR INDEXES option screens all of the received subscriber number digits. If more than one index is identified in the MSGCTR INDEXES option, then only the first one is used.

Note: If the VALIDATE option is not provisioned for the collectable, FLEXVAL table validation is not performed, and the MSGCTR INDEXES message is ignored.

If the subscriber number collectable VALIDATE option is present and identifies that validation should only occur at one FLEXVAL table index, then the index specified in the MSGCTR INDEXES option screens the number of digits identified by the VALIDATE option FLEXIDXS field (or the actual number of digits received, if less than the FLEXIDXS DIGCNT value). If more than one index is identified in the MSGCTR INDEXES option, then only the first one is used.

If the subscriber number collectable VALIDATE option is present and identifies that validation should only occur at two FLEXVAL table indexes, then the index specified in the MSGCTR INDEXES option screens the number of digits identified for both indexes (identified by the FLEXIDXS DIGCNT value or the actual number of digits identified, whichever is less). If two indexes are identified in the INDEXES option, then each index screens the amount of digits identified.

Validation of the subscriber number in table FLEXVAL is successful if the following conditions are met:

- The identified FLEXTYPE, numeric index, and identified number of received digits successfully index table FLEXVAL.

  Alternatively, if the numeric index used to index table FLEXVAL does not contain any entries against it for the identified subscriber number type (FLEXTYPE), and the FLEXTYPE table entry for the subscriber number type contains the EMPTYIDX option, then although the received digits cannot successfully index table FLEXVAL, the validation attempt is considered successful (see “EMPTYIDX call processing application” on page 17-10). In this case, a FLEXFEAT index is not retrieved and therefore not processed.

- The FLEXFEAT table entry identified by the index retrieved from the FLEXVAL table lookup does not contain the FAILVAL option.
- All identified digits for the subscriber number were FILED digits if the FLEXFEAT table entry contains the FLDONLY option. If just one digit was consumed from the digit buffer, validation is unsuccessful.

The validation attempt is unsuccessful if any single condition in the above criteria is not met.

For double index validation, both sets of received digits must successfully index table FLEXVAL and both FLEXFEAT table entries retrieved must not contain a FAILVAL option. If one set of digits does not successfully pass the screening test, then a validation failure occurs for the collectable.

For unsuccessful validation, an exception is generated. If the FLEXFEAT FAILVAL option is the cause of the unsuccessful validation attempt, then the action identified by the FAILVAL option is applied to the call. Otherwise, the action identified by the FAILACT field is executed.

The FAILVAL option supports two actions:

- Set treatment for the call.
- Apply the action identified by the FAILACT field of the subscriber collectable VALIDATE option.

See “Validate applicable digits” on page 16-105 for a description of the FAILACT field and setting treatments for unsuccessful validation attempts.

If the FLEXLOG option is provisioned for the subscriber number type in table FLEXTYPE, then on validation failures a FLEX 302 validation failure log is generated with one of the following validation failure reasons:

```
FLEXVAL Numeric Index not Provisioned
```

This reason is used when the FLEXVAL table is not successfully indexed because no entries are provisioned against the numeric index.

```
FLEXVAL Screening Failure
```

This reason is used when the FLEXAL table is not successfully indexed.

```
FAILVAL option enforced for Subscriber Number
```

This failure reason is used when the FAILVAL option is contained in the FLEXFEAT table entry retrieved from table FLEXVAL.

```
Subscriber digits must be FILED
```

This failure reason is used when the FLDONLY option is contained in the FLEXFEAT table entry retrieved from table FLEXVAL, and at least one digit for the collectable was retrieved from the digit buffer.

The digits field of the log contains the subscriber number digits that were used in the validation attempt. (This may be a subset of the total number of digits processed for the collectable.)

When a FLEXVAL validation attempt occurs, one of the following operational measurement (OM) counters for the specific subscriber number type is incremented:

```
VALSUCC—Validation successful counter.
```

This counter is incremented on successful validation attempts.

`VALFAIL—Validation failure counter.`

This counter is incremented on unsuccessful validation attempts.

`VALEMPTY—Validation successful using the EMPTYIDX FLEXTYPE table option.`

This counter is incremented on successful validation attempts when the validation is successful due to use of the EMPTYIDX FLEXTYPE table option.

After the validation attempt (either success or failure), the FLEXTYPE options are processed.

For successful validation, the features identified by the FLEXFEAT table entry are processed for the call.

For successful double index validation, both of the FLEXFEAT table entries retrieved are processed for the call, in the order they were retrieved from the validation attempts.

CITYCODE validation

Citycode validation compares the CITYCODE of the subscriber number being processed to the CITYCODE value set for the call. If the CITYCODE values are identical, then validation is successful. Otherwise the validation attempt fails.

Citycode validation is triggered through one of two mechanisms:

- through use of the CITYVAL option in table FLEXFEAT for the validated subscriber number. Citycode validation is provided on a per-subscriber number basis.
- through use of the CITYVAL option for the subscriber number collectable in table FLEXDIAL. Citycode validation is provided on a per-trunk group basis.

When both triggering mechanisms are activated for the call, then the specifications for the FLEXFEAT table CITYVAL option are used for the validation attempt. Citycode validation can only occur after successful FLEXVAL table validation has occurred.

Citycode validation requires that the following information is provided for the validation attempt:

- FLEXFEAT table CITYCODE option

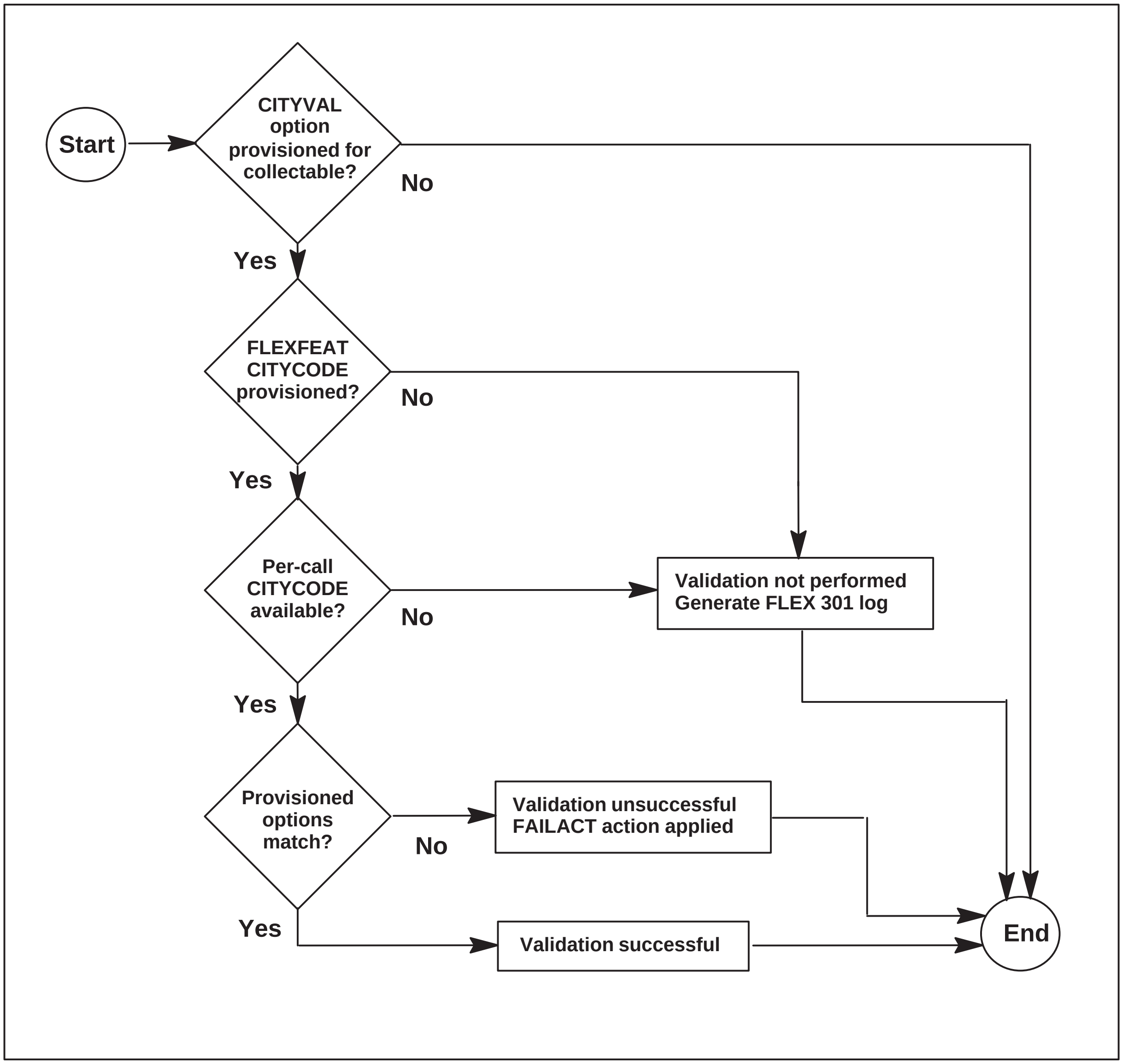
  The CITYCODE option must be provisioned for the subscriber number in the FLEXFEAT table index retrieved during FLEXVAL table validation. A CITYCODE option processed by a previous subscriber number or calltype collectable cannot be used.

- Per-call CITYCODE value

  The per-call CITYCODE value is the information contained within a CITYCODE option previously processed in the call. The initial CITYCODE value for the call may be provided by the originating agent through use of the TRKFEAT table CITYCODE option. Alternatively, a previously processed subscriber number or calltype collectable may have specified the CITYCODE value for the call.

Figure 16-30 illustrates the CITYCODE validation logic.

Figure 16-30
CITYCODE validation



If either CITYCODE value is not available, then completion of the city code validation attempt does not occur, and a FLEX 301 trouble log is generated with one the following trouble code values:

```
FLEXFEAT CITYCODE not available for CITYVAL
```

This error is used when the CITYCODE option is not provisioned for the subscriber number in table FLEXFEAT.

```
TRKFEAT CITYCODE not available for CITYVAL
```

This error is used when the per-call city code value is not identified.

The result of the validation attempt is then successful by default.

If both CITYCODE values are available, then the city code digits identified in both values must match for city code validation to be successful. If the digits in the two options do not match, then city code validation is unsuccessful.

If validation is successful, then the features identified by the FLEXTYPE and FLEXFEAT tables entry are applied to the call.

If validation is unsuccessful, then the action identified by the CITYVAL FAILACT field is applied to the call. The FAILACT field supports three actions:

- identify a FLEXDIAL table index (FAILACT = DPIDX)

  The FLEXDIAL table index identifies a list of collectables that must be processed due to the validation failure.

- apply the action identified by the FAILACT field of the subscriber collectable VALIDATE option (FAILACT=VALIDATE)
- apply the specific treatment identified (FAILACT = TRMT)

For the DPIDX FAILACT value, see “Validate applicable digits” on page 16-83 for a description of the FAILACT field and applying a FLEXDIAL table index for unsuccessful validation attempts.

When the validation is unsuccessful, the remaining features and characteristics within the FLEXFEAT table entry are not applied to the call, with one exception. The FAILACT value of TRMT contains a boolean field (APPLYFTR) that identifies whether or not the remaining FLEXFEAT options are to be processed on city code validation failures. When set to N, remaining options are not processed. When set to Y, remaining options are processed.

If the FLEXLOG option is provisioned for the subscriber number type in table FLEXTYPE, then on city code validation failure, a FLEX 302 validation failure log is generated with the following failure reason:

```
CITYVAL Screening Failure
```

When a CITYCODE validation attempt occurs, one of the following OM counters for the specific subscriber number type is incremented:

- TVALSUCC—validation successful counter

  This counter is incremented on successful CITYCODE validation attempts.

- TVALFAIL—Validation failure counter

  This counter is incremented on unsuccessful CITYCODE validation attempts.

For double-index validation, city code validation occurs independently with each FLEXVAL validation attempt.

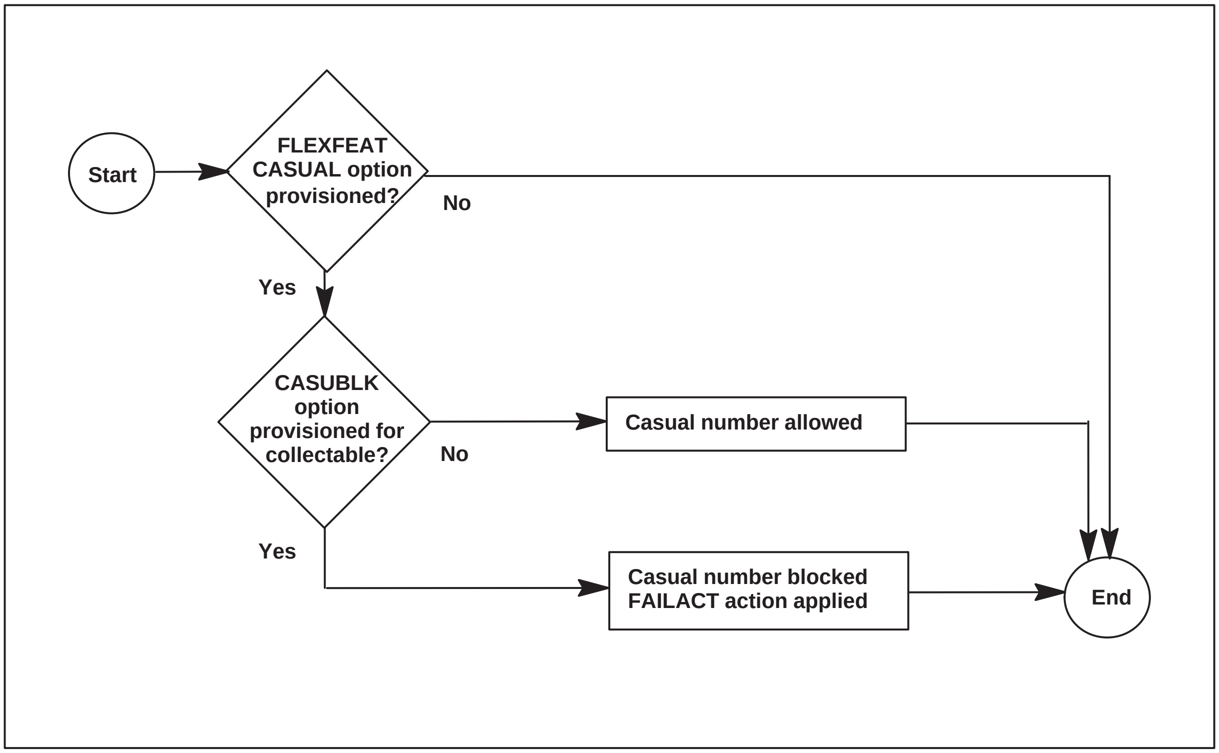
CASUAL number blocking

Casual subscriber number blocking occurs when the CASUBLK option is provisioned for the subscriber number collectable. Casual subscriber number blocking can occur only after successful FLEXVAL table validation has occurred.

A casual subscriber number is identified by the FLEXFEAT table CASUAL option. If this option is provisioned for the subscriber number in the FLEXFEAT table index retrieved during FLEXVAL table validation, then the subscriber number is identified as a casual subscriber number.

Figure 16-31 shows the CASUAL number blocking validation logic.

Figure 16-31
CASUAL number blocking validation


Start
FLEXFEAT CASUAL option provisioned?
No
Yes
CASUBLK option provisioned for collectable?
No
Casual number allowed
Yes
Casual number blocked FAILACT action applied
End

If the subscriber collectable CASUBLK option is provisioned for the collectable, then the collectable cannot successfully process casual subscriber numbers as identified by the FLEXFEAT CASUAL option.

If the processed subscriber number is not blocked by the CASUBLK option (that is, the subscriber number is not a casual number), the features identified by the FLEXTYPE and FLEXFEAT tables entry are applied to the call.

If the subscriber number is blocked by the CASUBLK option, then one of two possible actions is applied to the call as identified by the CASUBLK option FAILACT field:

- Set treatment for the call.
- Apply the action identified by the FAILACT field of the subscriber collectable VALIDATE option.

See “Validate applicable digits” on page 16-105 for a description of the FAILACT field and setting treatments for unsuccessful validation attempts.

If the FLEXLOG option is provisioned for the subscriber number type in table FLEXTYPE, then on casual number blocking, a FLEX 302 validation failure log is generated with the following failure reason:

```
Casual Subscriber Number Blocked
```

When a CASUAL number validation attempt occurs, one of the following OM counters for the specific subscriber number type is incremented:

- CVALSUCC—validation successful counter

  This counter is incremented on successful CASUAL number validation attempts.

- CVALFAIL—validation failure counter

  This counter is incremented on unsuccessful CASUAL number validation attempts.

Apply digit collectable features

Tables FLEXTYPE and FLEXFEAT identify features and characteristics that apply to the call due to the processing of the subscriber number collectable. In general, the options provisioned in the FLEXTYPE table are applied whether or not a validation attempt is performed or is successful, while options in table FLEXFEAT can only be retrieved through successful screening of the subscriber number in table FLEXVAL.

Because many different subscriber numbers or call types can be processed for the call, the setting of feature data for the call is according to the following rule:

A processed option overrides data set by a previously processed option.

For example, a processed ANI number may identify a multiple COS screening index (MLTCOSID) of 1. Further along in the call, a processed authcode may identify another MLTCOSID option with a value of 2, overriding the data set by the processed ANI.

FLEXTYPE table options

The FLEXTYPE table entry is retrieved from the FLEXTYPE field of the SUBR or SUBRPARM collectable, and is always processed by the subscriber number collectable. Processing the FLEXTYPE table entry for the subscriber number type is not dependent upon performing a validation attempt or a successful result of a validation attempt.

The FLEXTYPE table options identify certain areas of processing that immediately apply to the subscriber number collectable:

- BILLFLD

  Identifies the CDR field where the subscriber number digits are stored for CDR formatting.

- BILLFLGS

  Changes to the access, service, and call type as identified by the BILLFLGS option are captured in the CALLTYPE field of the CDR.

Other options in the FLEXTYPE table entry identify feature data for areas of processing that occur at a later time in the call process. This data is subject to being overwritten by the processing of other subscriber collectables, a CALLTYPE collectable, or cleared by the CLRFTRS collectable:

- ANSCDR

  The Answer CDR option identifies that a CDR is generated when the terminating agent answers the call.

- CDRTMPLT

  The CDR Template option contains a CDRTMPLT table index that identifies the template used to format the CDR generated.

- CALLING

  The Calling option identifies that the subscriber number is identified as the calling party address for the call (also known as the ANI digits).

- CIC

  The CIC option identifies that the subscriber number type represents a carrier identification code (CIC).

FLEXFEAT table options

The FLEXFEAT table entry is retrieved from table FLEXVAL when FLEXVAL validation occurs for the identified subscriber number digits. When double index validation occurs, each FLEXFEAT table index retrieved is processed for the call.

The FLEXFEAT table options identify certain areas of processing that immediately apply to the subscriber number collectable:

- CASUAL

  The Casual option immediately identifies the subscriber number as a casual number.

- CITYCODE

  The Citycode option identifies the city code of the subscriber number and is used for CITYVAL type validation.

- CLDPBILL

  The Called Party Billed option identifies that the call is being charged to the called party number. Therefore any treatment set by the processing of a subscriber number is cleared.

- DPIDX

  The FLEXDIAL table index identifies interactions with the originating agent identified for the particular subscriber.

- FAILVAL

  The Fail Validation option identifies that FLEXVAL validation is unsuccessful. Normally, since table FLEXVAL was successfully indexed, the validation attempt would be successful.

- GENLOG

  The Generate Log option generates a FLEX 601 information log due to the use of the subscriber number.

- MSGCTR

  The Message Center option identifies messages that are posted at the message center for other collectables due to the use of the subscriber number.

Other options in the FLEXFEAT table entry identify feature data for areas of processing that occur at a later time in the call process. This data is subject to being overwritten by the processing of other subscriber collectables, a CALLTYPE collectable, or cleared by the CLRFTRS collectable:

- ANSCDR

  The Answer CDR option identifies that a call detail record is generated when the terminating agent answers the call.

- BCCOMPAT

  The Bearer Channel Compatibility option identifies the bearer capability for the subscriber number used in bearer capability screening.

- CAINGRP

  The CAIN Group Option identifies the index into the CAINGRP table for AIN identified services.

- CDRTMPLT

  The CDR Template option contains a CDRTMPLT table index that identifies the template used to format the CDR generated.

- CITYCODE

  The Citycode option identifies a city code value that indexes table CITYCODE for public speed dial number processing.

- DELIVER

  The Calling Party Number Deliver option identifies when the calling party number is to be delivered to the next network switch as the call is connected and the terminating agent is seized and outpulsed upon.

- MLTCOSID

  The Multiple COS Index option identifies that COS screening of the address digits is to be performed using the identified MULTICOS table index.

- NOANSDUR

  The No Answer Duration option identifies that if answer is not received within a certain time period after seizure of the terminating agent occurs, then the connection is taken down and the identified action is applied to the call.

- ONNET

  The Onnet option identifies that the call is identified as an on network call rather than an off network call.

- PVSPDIDX

  The Private Speed Dial Index option identifies the private speed dial index for speed dial number conversion.

- REORGACT; REORGTYP

  The Reorigination options identify how reorigination is to be applied for the call for the particular subscriber.

- SPLASHBK

  The Splash Back option identifies an index into the SPLASHID table to provide a special splashback tone which must be applied before local treatment is applied to the call.

- TCAPANNC

  The TCAP announcement option identifies a subscriber specific index for customized announcements in interworking with Transactions Capabilities Application Part (TCAP) validation processing.

  Note: For more information on the interaction of FlexDial AXXESS agents and the TCAP feature, see Appendix D, ”FlexDial interactions” and the *UCS DMS-250 Transaction Capabilities Application Part (TCAP) Application Guide*. For information on table FEATBYTE, which identifies call types and interaction requirements related to information contained in a TCAP response message, see the *UCS DMS-250 Data Schema Reference Guide.*

- TRANSTS

  The Translation STS option identifies the STS used for translations.

Results of digit collection

If the subscriber collectable REPBDIGS (replace buffer digits) field is set to Y, then the digits consumed by the subscriber collectable from the digit buffer are returned to the digit buffer so that they may be processed by other sequence or digit collectables. If the REPBDIGS field is set to N, then the digits remain consumed by the subscriber number collectable.

If the CALLING option is present for the subscriber number type in the associated FLEXTYPE table entry, then the calling party NOA is set by the SUBR collectable. If the CALLING option is not present for the subscriber number type, then the calling party NOA value is not modified.

Additionally for CALLING party identified numbers, the following field is captured as processed by the ANI delivery feature:

- PRESIND

  Contains the presentation indicator set for the call.

If the subscriber collectable BILLCAPT field is set to Y, and the BILLFLD option is present for the subscriber number type in the FLEXTYPE table entry, then the subscriber number digits received are captured in the CDR field identified by the FLEXTYPE table BILLFLD value.

The BILLFLD option identifies one of the following fields in which to capture the subscriber number digits:

- ANISP

  capacity of 10 digits

- BILLNUM

  capacity of 24 digits

- PINDIGS

  capacity of 4 digits

- ACCTCD

  capacity of 12 digits

- CLGPTYNO

  capacity of 15 digits

- SUBRNUM1—SUBRNUM2

  capacity of 16 digits

- UNIVACC

  capacity of 10 digits

If the CLGPTYNO field is used, then the following field in the CDR is also captured:

- INC_INTL

  This field identifies whether the CLGPTYNO number is international.

Note: The CLGPTYNO field is used to identify that the subscriber number being processed is identified as the calling party number for the call.

Based upon the ACTION field value of the BILLFLD option, the processed digits can either be appended to or replace digits already captured in the CDR billing record field.

If the subscriber is identified as the calling party number through the FLEXTYPE table CALLING option, then the following CDR field is also captured:

- CLGNOA

  This field identifies the NOA value set during call processing for the calling party subscriber number.

If a result of the execution of the subscriber number collectable is to set the called party NOA value (see “Setting calling party NOA” on page 16-86), then the following CDR field is also captured:

- CLDNOA

  This field identifies the NOA value set during call processing for the called party address.

When the FLEXFEAT TRANSTS option is used, then the following fields in the call detail record are also captured.

- STS

  The STS field contains the serving translation scheme used for call translations. This field is overwritten each time an STS is identified through use of the TRANSTS option.

- ORIGSTS

  The ORIGSTS field contains the first serving translation scheme set for the call. If the field has not been captured yet when the TRANSTS option is processed, then the STS value set is captured. The ORIGSTS CDR field is not overwritten on processing of multiple TRANSTS options.

With the exception of ANISP and UNIVACC, the received subscriber number digits are appended to digits already captured in the identified recording unit field. This occurs if two or more subscriber number types processed for a call identify the same field for capturing the received subscriber number digits.

For ANISP and UNIVACC use, the captured digits replace digits previously captured.

If there is not enough room in the identified recording unit field to store all of the received digits, then the excess digits are truncated and cannot be captured. A FLEX 304 CDR capture failure log is generated.

If the BILLFLD is set to either SUBRNUM1 or SUBRNUM2 (see the *UCS DMS-250 Billing Records Application Guide*), then information for the following associated CDR fields is also captured:

- SUBRTYPx

  This field identifies the FLEXTYPE table index for the subscriber number processed.

- SUBRIDXx

  Captured if FLEXVAL table validation occurs, this field identifies the numeric index used to validate the subscriber number digits received.

  Note: Only one numeric index is stored. If validation occurs at multiple indexes, then the last index used is captured.

- SUBRVALx

  This field identifies if validation occurred for the subscriber number received.

- SUBRLOGx

  This field identifies if a FLEX 601 information log was generated through use of the GENLOG option (see “GENLOG call processing application” on page 18-21) for the subscriber number processed.

  Note: x represents 1 or 2.

Use case scenario

The collectable is used in the following call scenario:

- collect and process subscriber number digits

  All types of subscriber numbers (including authcodes, PIN digits, ANI numbers, account code digits, and TCN numbers) are collected and potentially validated using the subscriber number collectables.

Restrictions and limitations

The following restrictions and limitations apply to the SUBR/SUBRPARM collectable:

- If both INDEXES and MATCH message center message types are received by a SUBR/SUBRPARM collectable containing a VALIDATE option, the message processed last identifies the type of validation that occurs for the collectable.
- Empty index validation is only successful if the numeric index used for FLEXVAL table validation is not present in any table entries against the subscriber number type (FLEXTYPE).
- CITYVAL type validation and CASUAL number blocking can only be processed if successful FLEXVAL table validation of the subscriber has occurred.
- FLEXTYPE table options are processed whether or not validation was performed for the subscriber number, or whether it was successful.
- FLEXTYPE and FLEXFEAT options overwrite values set for the call by previously processed subscriber number collectables and CALLTYPE collectables.
- The received subscriber number is only captured in the CDR if the BILLCAPT field of the subscriber collectable is set to Y and the billing record field is identified for the subscriber number type by the BILLFLD option of the FLEXTYPE table entry.
- The calling party NOA value is only set by the subscriber number collectable if the subscriber number type is identified as a CALLING subscriber number.
- For multiple index validation, only the last index used in the validation attempt is stored in the SUBRIDXx billing record field.

Address digit collectable (ADDR/ADDRPARM)

The address digit (ADDR/ADDRPARM) collectables collect called party address digits from the originating agent. The ADDR collectable collects digits that are processed as called party address digits.

Application

The ADDR digit collectable follows the general application of inband digit collection.

The ADDRPARM digit collectable follows the general application of out-of-band digit collection.

Identify digit collection request

The messages received from the message center may alter the request parameters for the address collectable.

The FILED message identifies filed digits for the collectable.

The MINMAX message identifies the minimum and maximum values for the digit collection request, overriding the values provisioned in the MIN and MAX fields of the collectable. The MINMAX message is only applicable for the ADDR collectable. The message is consumed, but not used by the ADDRPARM collectable.

The OPER message identifies information used for screening and translation of operator call types (0-, 0+, 01+). Unlike other message center messages, the OPER message contains information not provisioned with the collectable. The table MSGCTR ADDR addressee OPER message contains three routing methods for an operator call:

- NORMAL
- OPERRTE
- OPCHOICE

If the routing method is set to NORMAL, the OPER message identifies:

- The route for zero-minus (0-) calls.
- The pretranslator name for zero-plus (0+) or zero-one-plus (01+) calls.
- The route for international operator assisted (01+) calls. This route is only used for 01+ calls when the OLI collectable identifies the call as an international operator-assisted (INTOA) call (see “Results of digit collection” on page 16-144).

For calls where the OLI collectable does not identify the call as a 01+ call, then the pretranslation name specified for zero-plus (0+) calls is used to screen the received digits and identify the call as an international operator-assisted call.

If the routing method is OPERRTE, the OPER message contains:

- The CIC that identifies the OPERRTE table index for all operator call types. A CIC can be used when it is desirable to uniquely route the operator call on subsequent network switches.

If the routing method is OPCHOICE, the OPER message identifies:

- The index into table OPCHOICE. Table OPCHOICE provides
  - — for zero-minus (0-) and international operator-assisted (INTOA) calls, both a position and a table office route (OFRT) index. The position is used unless the value in the position field is NONE, in which case the table OFRT index is used. If both the position and the table OFRT index are set to NONE, the ADDR or ADDRPARM collectable applies the VACT treatment.

    Note: If the table OPCHOICE index is empty, the ADDR or ADDRPARM collectable applies the VACT treatment to 0- and INTOA calls.

  - — An alternate pretranslator name, for zero-plus (0+) calls, to validate the address digits. If the pretranslator name is NPRT, the ADDR or ADDRPARM collectable does not use table OPCHOICE to route the call, but uses the last pretranslator name encountered through call processing.

    If the table OPCHOICE index is empty, the ADDR or ADDRPARM collectable does not use table OPCHOICE to route the call, but uses the last pretranslator name encountered through call processing.

  All calls that are routed using table OPCHOICE will populate the OPCHOICE field of the CDR with the table OPCHOICE index.

If an OPER message is not received for an address collectable, then the following actions occur:

- For zero-minus (0-) calls, partial dial treatment is set for the call.
- For zero-plus (0+) and zero-one-plus (01+) calls, the same pretranslator name identified for non-operator calls is used for pretranslations.
- For 01+ calls identified as INTOA by the OLI collectable, the call receives vacant code treatment (VACT).

Use of the carrier identification number (CIC) is mutually exclusive from the use of the other information. When an operator NOA or operator call type is identified in address digit processing, this information is applied to determine the route for the operator call type.

Normally, address digits identifying a request for operator services are screened with the same information used to screen non-operator calls.

The PRTNM message identifies the pretranslator name used for address digit validation in table STDPRTCT, overriding the pretranslator name provisioned in the VALIDATE option. If the VALIDATE option is not provisioned, then the message is consumed but not used by the collectable.

There are no conflicting interactions between the ADDR MSGCTR messages.

Perform digit collection request

The ADDR collectable follows the general application of requesting digits from the agent terminal controller as described in “Perform digit collection request” on page 16-80.

The ADDRPARM collectable does not perform a request of digits from the agent terminal controller.

SIGFEAT capability

The handling of the SIGFEAT option provisioned with the ADDR collectable is identical to SIGFEAT processing for the COLDIG collectable. See “SIGFEAT capability” on page 16-100.

Validate applicable digits

The address collectable supports validation of the received address digits through table STDPRTCT. The identification of table STDPRTCT screening occurs through use of the VALIDATE option for the address collectable. STDPRTCT table screening requires the following information to screen the address digits:

- pretranslator name

  The pretranslator name is identified by the PRTNM field of the VALIDATE option or by the MSGCTR PRTNM message type.

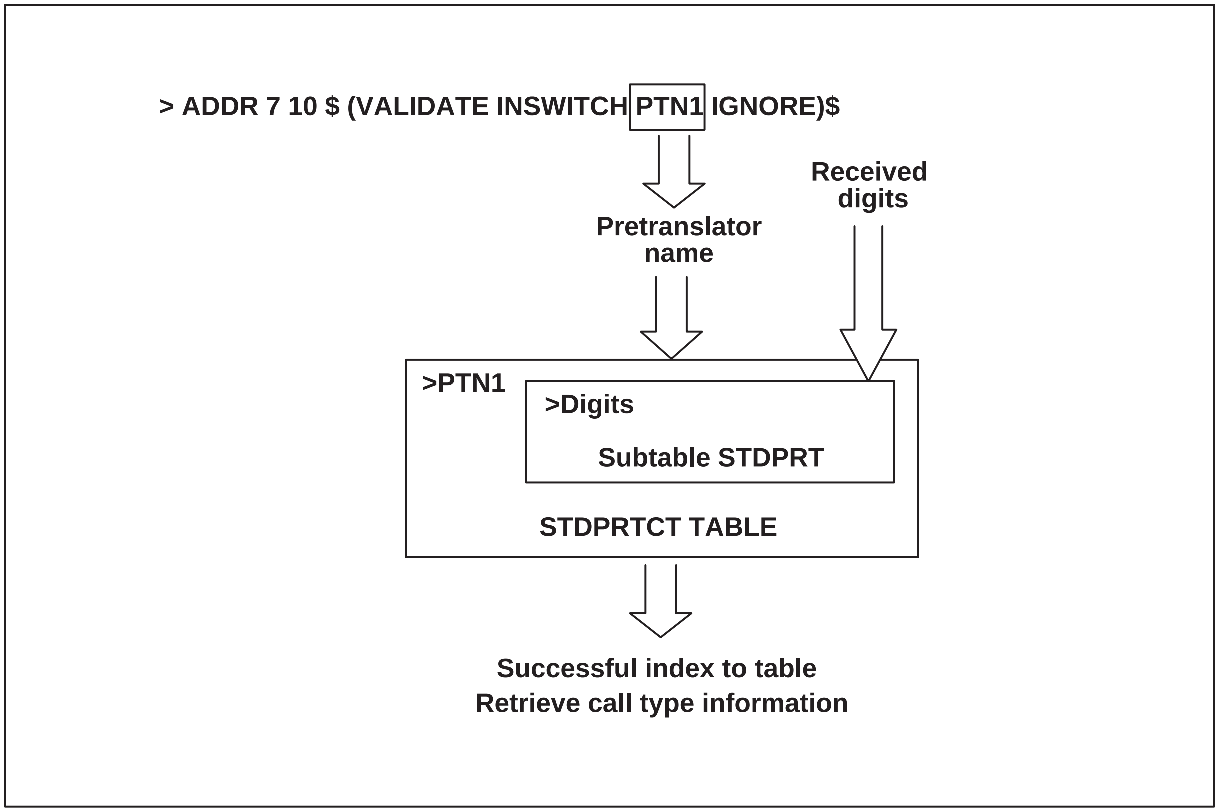
- address number digits

  The digits received for the collectable are also required to perform screening.

All digits received by the collectable are used to index the STDPRT subtable for the pretranslator name.

Figure 16-32 illustrates the STDPRTCT table validation.

Figure 16-32
STDPRTCT table validation


> ADDR 7 10 $ (VALIDATE INSWITCH PTN1 IGNORE)$
Received digits
Pretranslator name
>PTN1
>Digits
Subtable STDPRT
STDPRTCT TABLE
Successful index to table
Retrieve call type information

If the received digits successfully index the subtable associated with the pretranslator name, and a supported selector/call type is retrieved, then screening of the digits is successful.

If the received digits do not successfully index the subtable, then the action identified by the FAILACT VALIDATE option field is performed.

If the received digits identify an unsupported selector/call type in the table, then an exception is generated and FNAL Treatment is set for the call.

STDPRTCT MIN/MAX support

A number of pretranslator selectors support the identification of the minimum number of digits to collect and the maximum number of digits to collect.

When the STDPRTCT table is used for special address MIN/MAX processing, the MIN/MAX values datafilled in STDPRTCT are used to reset the MIN/MAX values for the collectable.

If the STDPRTCT table is not used for special address MIN/MAX processing, then MIN/MAX values provisioned against a pretranslator selector are used as follows:

- If the number of received digits is less than the STDPRT MIN value specified, then a partial dial exception is generated for the call.
- If the number of received digits is greater than the STDPRT MAX value specified, then only the MAX amount of digits are recorded in the CDR and used by call processing. Excess digits that were consumed by from the digit buffer are replaced back into the digit buffer before proceeding with the next collectable. Excess digits that were obtained from the originating agent or from filed sources are consumed.

Special Address MIN/MAX Processing.

Typically the MIN/MAX values for the address collectable must be set to 1 and 18, respectively, to account for all the variations of address digits (from zero-minus calls to 18-digit international calls). For out-of-band collection and MF pulse type inband collection, this is not an issue since all the address digits are provided at once to the call processing application.

For DTMF inband pulse type collection, digits are collected individually. For receipt of address digit streams less than 18 digits, this requires that a terminating digit or a timeout is received in order to identify the end of the address digit stream. Special address MIN/MAX processing is provided for the address collectable.

Activation

Activation of the special address MIN/MAX processing occurs when the provisioned MAX value is equal to 18 and the VALIDATE option is provisioned for the address collectable. If the MAX value is not set to 18 or the VALIDATE option is not provisioned, then special MIN/MAX processing does not occur.

Alternatively, the MSGCTR ADDR MINMAX message can be used to specify a MAX value of 18 for special address MIN/MAX processing.

Processing Flow

Special address MIN/MAX processing performs the following steps to collect and process address digits:

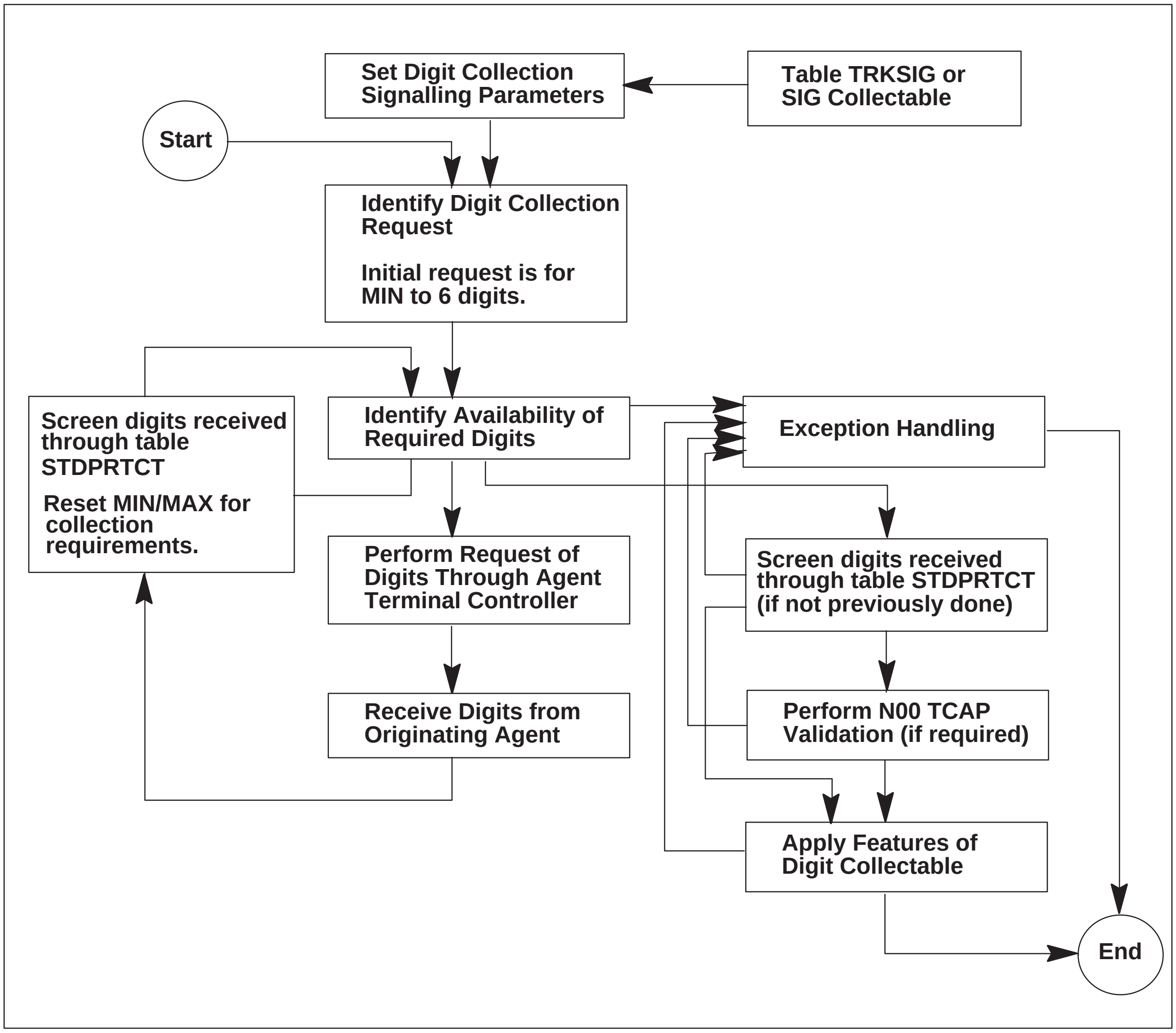
- Request a minimal number of digits from the originating agent. When the MIN is less than or equal six (6), a request for a minimum of MIN and a maximum of six digits is made. When the MIN is greater than six, a request for exactly MIN digits is made.
- Index table STDPRTCT with the received six digits. Reset the MIN/MAX values being used for address collectable processing with values provisioned in table STDPRTCT.

  If the received six digits do not index the STDPRT subtable of table STDPRTCT, or the selector used does not contain provisioned MIN/MAX values, then an internal mechanism is used to identify the true MIN/MAX values for the call.

- Collect the remaining digits required from the originating agent based on the revised MIN/MAX values.
- Re-screen the received digits in table STDPRTCT (if necessary), and apply the features of the address collectable.

Figure 16-33 shows the modified inband digit collection process flow for special address MIN/MAX processing.

Figure 16-33
Modified inband digit collection process flow for special address MIN/MAX processing


Set Digit Collection Signalling Parameters
Table TRKSIG or SIG Collectable
Start
Identify Digit Collection Request
Initial request is for MIN to 6 digits.
Screen digits received through table STDPRTCT
Reset MIN/MAX for collection requirements.
Identify Availability of Required Digits
Exception Handling
Perform Request of Digits Through Agent Terminal Controller
Screen digits received through table STDPRTCT (if not previously done)
Receive Digits from Originating Agent
Perform N00 TCAP Validation (if required)
Apply Features of Digit Collectable
End

Essentially the request for address digits is broken up into two distinct digit collection requests, which must be logically viewed and perform as a single request.

In order to support the correct timer values during the ADDR smart MIN/MAX collection processes, the following rules are defined:

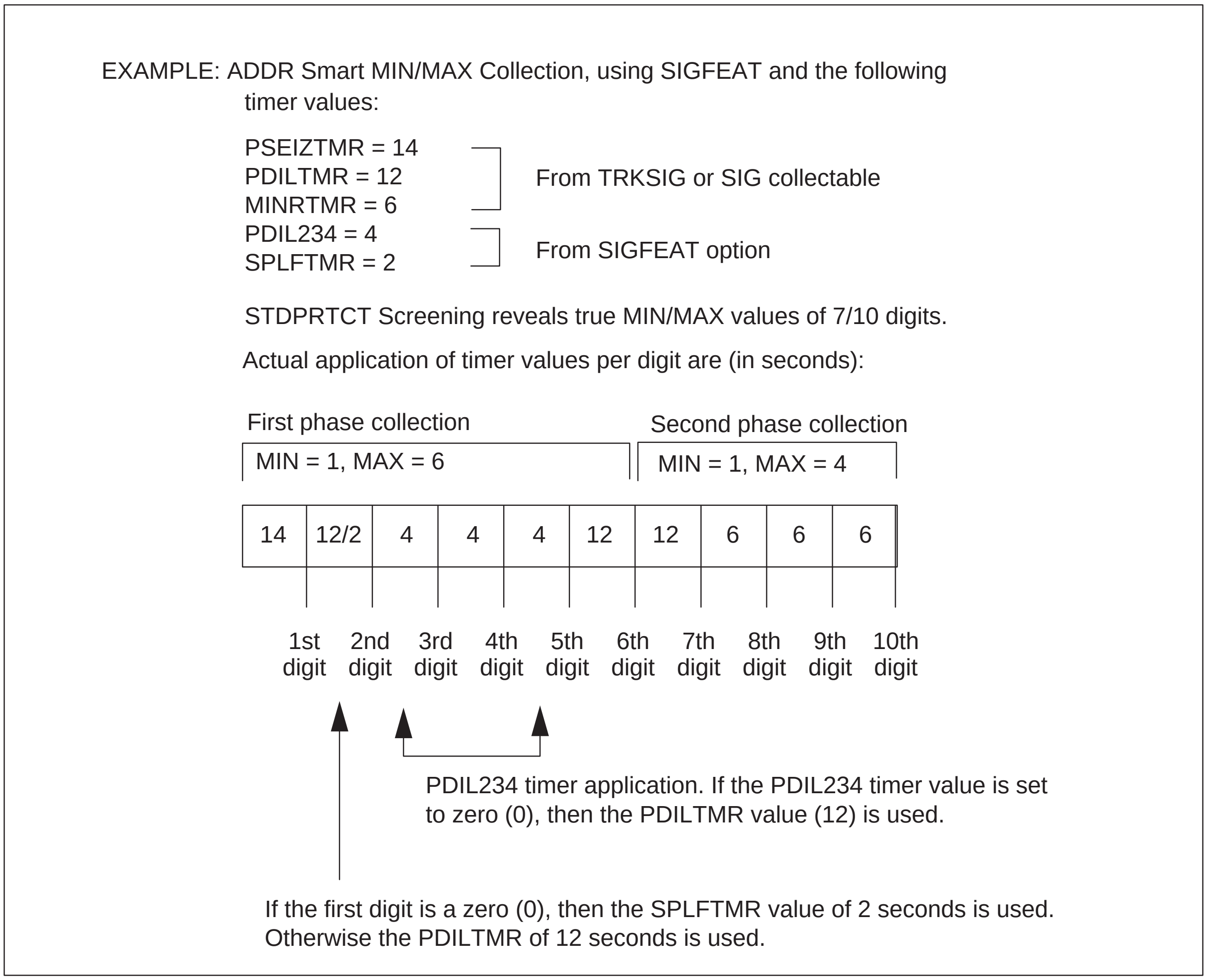
- During the first phase of the collection, the MAX value is set to the larger of 6 or the MIN value, one (1) and six (6) respectively, the use of the MINRTMR timer value is blocked, and exclusive use is made of the PSEIZTMR (for prior to receiving the first digit), and the PDILTMR (for prior to all digits up to the sixth digit).

- During the second phase of the collection (MIN/MAX based on STDPRTCT provisioning), use of the PSEIZTMR timer value is blocked, and exclusive use is made of the PDILTMR (for prior to receiving up to the MIN number of digits), and the MINRTMR (for receiving up to the MAX number of digits).

When the SIGFEAT option is used with the ADDR collectable, then the application of the PDIL234 and SPLFTMR timer values occurs for the first phase of the smart ADDR MIN/MAX collection process, and are ignored for the second phase of the collection process.

Figure 16-34 shows ADDR smart MIN/MAX collection timer use.

Figure 16-34
ADDR smart MIN/MAX collection timer use



EXAMPLE: ADDR Smart MIN/MAX Collection, using SIGFEAT and the following timer values:

PSEIZTMR = 14
PDILTMR = 12
MINRTMR = 6
PDIL234 = 4
SPLFTMR = 2

STDPRTCT Screening reveals true MIN/MAX values of 7/10 digits.

Actual application of timer values per digit are (in seconds):

PDIL234 timer application. If the PDIL234 timer value is set to zero (0), then the PDILTMR value (12) is used.

If the first digit is a zero (0), then the SPLFTMR value of 2 seconds is used. Otherwise the PDILTMR of 12 seconds is used.

Note that the two phase collection approach for Smart ADDR MIN/MAX Collection can be seen when using FLEXSIM to simulate call processing for an Axxess agent.

Interactions

Interactions with address collectable options are not affected by special address MIN/MAX processing. The SIGFEAT, PROMPT, FILED, and RESET options function as specified in previous sections.

If a minimum of six digits are not collected with the initial request, then a subsequent request is not performed. If it is identified that more digits are required, then a partial dial exception is generated. Otherwise the call is processed with the digits received from the initial request.

For pretranslator selectors that do not contain provisioned MIN/MAX values in table STDPRTCT, or when the six digits initially processed do not successfully index the STDPRT subtable, the following logic is used to reset the MIN/MAX values for address collectable processing:

- For international identified calls (01+, 011+), the MIN/MAX values are reset to 7/18.
- For national identified calls without a prefix (NPA+), the MIN/MAX values are reset to 7/10.
- For national identified calls with a prefix (0+NPA+, 1+NPA+), the MIN/MAX values are reset to 8/11.
- Otherwise the MIN/MAX values are reset to 7/11.

Due to current implementation restrictions, CCS7 Axxess agents are limited to the use of just one timer value from the set of MINRTMR, SPLFTMR, and PDIL234. PDIL234 is given the greatest precedence, while MINRTMR is given the least precedence. For Smart ADDR MIN/MAX collection, concurrent activation of SPLFTMR and PDIL234 via the SIGFEAT option results in the sole use of the PDIL234 timer value. If a “0” digit is received as the first digit, then the PDIL234 timer value is used as the special first digit timer duration and not the actual SPLFTMR value.

Note: Interactions with address collectable options are not affected by special address MIN/MAX processing. The SIGFEAT, PROMPT, FILED, and RESET options function as currently defined. The SIGFEAT PDIL234 and SPLFTMR values may supplement the collection process for the first half of the special ADDR MIN/MAX collection only.

Exceptions and Limitations

The use of SUFFIX filed digits (through either the FILED option or a MSGCTR ADDR FILED message) disables special address MIN/MAX processing. The address digits consist of any possible digits retrieved from the digit buffer appended by the filed digits. No request for digits is performed (since the MIN = 1 and there is at least one filed digit).

Note: It is recommended to either not use special address MIN/MAX processing when specifying a FILED option with suffixed digits, or use the MSGCTR ADDR MINMAX message in conjunction with a MSGCTR ADDR FILED message that contains suffixed digits.

Apply digit collectable features

The following STDPRT selectors are supported for address collectable processing:

- NT

  This selector identifies the basic call type and translation type.

- D

  This selector identifies a treatment to apply to the call.

- S

  The standard selector specifies a CLLI or POSITION route for the call.

- T

  The table selector indicates a table index that identifies the route for the call.

- CT

  The call feature selector identifies a number of supported call features for ADDR/ADDRPARM collectable processing:

  - ONNET—identifies ONNET call
  - OFFNET—identifies OFFNET call
  - CROSSON—identifies a cross network ONNET call
  - PUBSPD—public speed number call
  - PRVSPD—private speed number call
  - HTLSPD—hotline speed number call
  - ZPLUS—0+ call requiring COSUS screening

The ACCT, TOLLFR, and AUTHREQ call feature types are not supported for ADDR/ADDRPARM collectable processing and their use results in FNAL treatment being applied to the call.

- P

  The position selector identifies a POSITION route for the call.

- R

  The replace selector identifies digits used to replace called party address digits.

- ES

  The 800 services selector provides support for N00 type numbers, including:

  The 800 services selector provides support for N00 type numbers, including:

  — SACREMOT, which initiates a TCAP query for the N00 application

  — INWFEAT and INWTRANS, which contain the NXX call blocking option, NXXBLOCK

  Note 1: To use the NXXBLOCK option you must order and activate software optionality control (SOC) NXXR0001.

  Note 2: For more information on NXX call blocking, see the *UCS DMS-250 FGD Application Guide*.

  Note: For more information on tables SACREMOT, see the *UCS DMS-250 Data Schema Reference Manual*.

- UAX

  Similar to the UA selector, this selector identifies a FLEXDIAL index to be processed for the call, identifying further interaction for the originating agent.

  The FLEXDIAL table index from the UAX selector identifies an alteration in the interaction with the originating agent. The list of collectables identified by the FLEXDIAL table index may either:

  — be inserted into the current list of collectables being executed

  — be appended to the current list of collectables being executed

  — replace the current list of collectables being executed.

  — execute a new list as a sub-list in the current list.

See “Collectable list management” on page 15-9 for more specific details about the INSERT, APPEND, REPLACE, and EXEC collection list management methods.

When the UAX selector is used, the digits processed by the address collectable are sent to a SUBR collectable using the SUBR FILED message format (DOMINANT=N, REPOST=N). The subscriber number type for the message is identified by the FLEXDIAL_N00_FLEXTYPE office parameter.

The following selectors are not supported for address collectable processing, and their use results in call treatment:

- CT

  The ACCT, TOLLFR, and AUTHREQ call features are not supported for ADDR/ADDRPARM collectable processing, and their use results in FNAL treatment being immediately applied to the call.

- CS

  The CS selector is used only for COSUS table screening. Use of the CS selector for address digit screening results in feature not allowed (FNAL) treatment being set for the call.

- EA

  The equal access selector is not supported for the UCS DMS-250 switch. Use of this selector results in FNAL treatment being applied to the call.

- L

  The local selector requires directory number translations, which is not supported on the UCS DMS-250 switch.

- F

  This selector is not supported on the UCS DMS-250 switch.

- X, V, Z

  The address regenerate and route selectors are not supported on the UCS DMS-250 switch.

- N

  The N selector is an older version of the NT selector. Table control restricts its use.

- UA—Universal access—replaced by the UAX selector for AXXESS agents. Use of the UA selector for address collectable processing results in FNAL Treatment being applied to the call.
- ID—Information digit screening selector. Used only for originating line information (OLI) digits. Use of this selector for address digit screening results in FNAL Treatment being set for the call.

In addition to STDPRTCT selector processing, other features set for the call may be applicable during address collectable processing. These features include:

- multiple COS screening (FLEXFEAT MLTCOSID option)
- private speed dial number screening (FLEXFEAT PVSPDIDX option)
- on-network call identification (FLEXFEAT ONNET option)
- TCAPANNC table processing for the N00 TCAP application (FLEXFEAT TCAPANNC option)

See individual feature descriptions in Chapter 8 for more details.

Results of digit collection

The received digits are consumed by the address collectable. The called party NOA value is also set.

If the address collectable receives all required digits through the FILED option or the FILED MSGCTR message, and the call is identified as a calling party billed call, then the call is also handled as a hotline call. If the address collectable retrieves just one digit from the digit buffer, then the call is not processed as a hotline call.

The following fields may be captured in the CDR through address collectable processing:

- DIALEDNO

  The identified address digits before validation occurs are recorded in the DIALEDNO field.

- DIALNOA

  This field captures the NOA value associated with the DIALEDNO digits.

- PREDIG

  This field contains the dialed number prefix digits indicator.

- CALLEDNO

  The CALLEDNO field contains the identified address digits after validation and applicable address collectable features have been applied.

- CLDNOA

  This field captures the NOA associated with the CALLEDNO digits.

- CNPREDIG

  This field contains the called number prefix digits indicator.

- DNIS

  The DNIS field contains DNIS digits received in the TCAP response message received through use of the N00 application for validation of the address digits.

- DNISNOA

  This field captures the NOA value associated with the DNIS digits.

- ACG

  Associated with the N00 TCAP application, this field identifies if automatic call gapping occurred on the call.

- ADDRNUM

  If multiple routing numbers are received in the TCAP response message for the N00 application, this field is incremented for each address collectable executed.

- BILLTYPE

  Associated with the N00 TCAP application, this field is captured from information returned in the TCAP response message.

- COSOVE

  When COS screening is performed on the address digits received, this field is captured if COSUS screening fails and the COS screening override capability is activated for the call.

- NUMADDRS

  If multiple routing numbers are received in the TCAP response message for the N00 application, this field is captured and identifies how many routing numbers were received.

- UNIVACC

  For address collectable processing only, the dialed digits received are captured in this field when the UAX STDPRT selector is used.

Exception processing

Treatments or other actions set through multiple COS screening (see "MLTCOSID call processing application" on page 18-22) are immediately applied to the call, and are not delayed actions.

Use case scenario

The collectable is used in the following call scenario:

- collect and process called party address digits

  Called party address digits are collected and potentially validated using the address collectables.

Restrictions and limitations

A number of STDPRTCT selectors are not supported for address collectable processing.

Originating Line Information digit collectable (OLI/OLIPARM)

The Originating Line Information (OLI/OLIPARM) collectables provide the ability to collect information digits from the originating agent. The originating line information collectable collects only digits that are processed as information digits.

Application

The OLI digit collectable follows the general application of inband digit collection.

The OLIPARM digit collectable follows the general application of out-of-band digit collection.

Identify digit collection request

Messages received from the message center can alter the request parameters for the originating line information collectable.

The FILED message identifies filed digits for the collectable.

The PRTNM message identifies the pretranslator name used for OLI digit validation in table STDPRTCT, overriding the pretranslator name provisioned in the VALIDATE option. If the VALIDATE option is not provisioned, then the message is consumed but not used by the collectable.

Validate applicable digits

The OLI/OLIPARM collectable supports validation of the received information digits through table STDPRTCT. The identification of table STDPRTCT screening occurs through the VALIDATE option for the OLI collectable. STDPRTCT table screening requires the following information to screen the OLI digits:

- pretranslator name

  The pretranslator name is identified by the PRTNM field of the VALIDATE option.

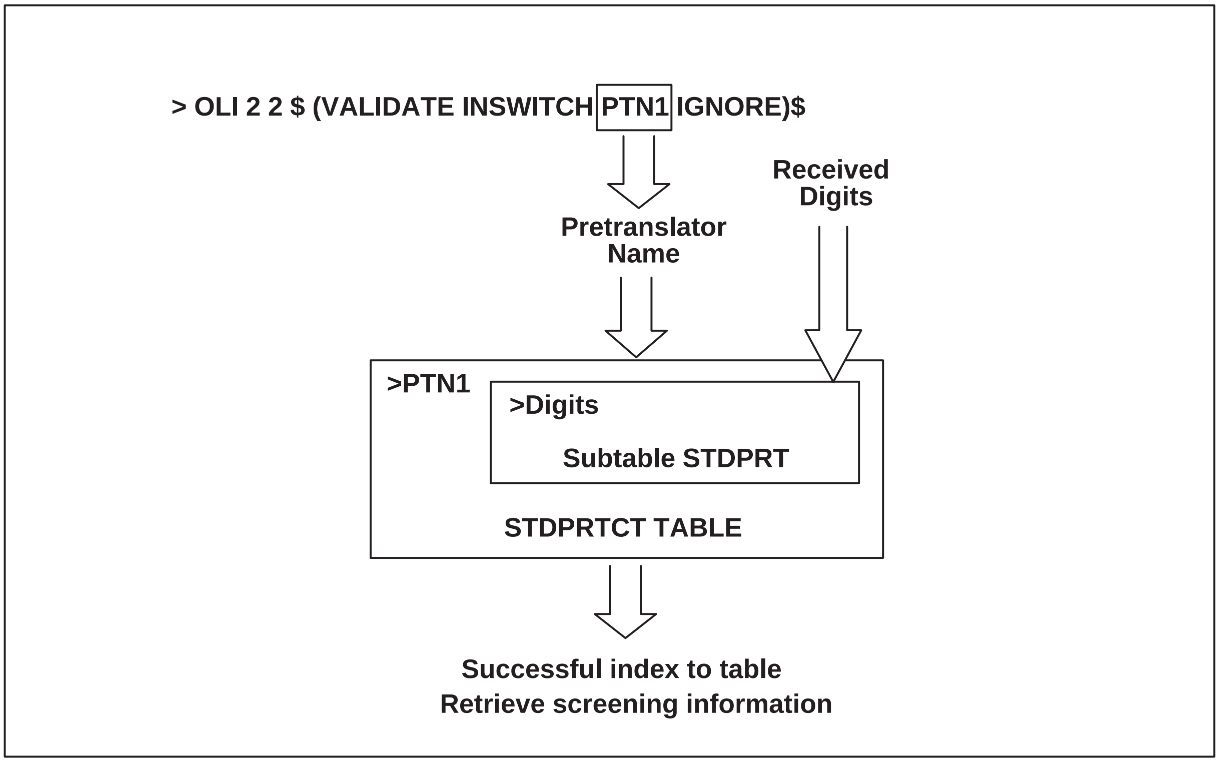
- originating line information digits

  The digits received for the collectable are also required to perform screening.

All digits received by the collectable are used to index the STDPRT subtable for the pretranslator name.

Figure 16-35 shows the STDPRTCT table validation.

Figure 16-35
STDPRTCT table validation


> OLI 2 2 $ (VALIDATE INSWITCH PTN1 IGNORE)$
Received Digits
Pretranslator Name
>PTN1
>Digits
Subtable STDPRT
STDPRTCT TABLE
Successful index to table
Retrieve screening information

If the received digits successfully index the subtable associated with the pretranslator name, and a supported selector/call type is retrieved, then screening of the digits is successful.

If the received digits do not successfully index the subtable, then the action identified by the FAILACT VALIDATE option field is performed. See “Validate applicable digits” on page 16-105 for more information on FAILACT field processing.

If the received digits identify an unsupported selector/call type in the table, then an exception is generated and FNAL treatment is set for the call.

Apply digit collectable features

The following STDPRT selectors are supported for originating line information collectable processing:

- ID

  The information digit selector supports a number of call condition values:

  - CONT—continue processing, screening successful
  - STOP—stop processing, screening unsuccessful
  - TEST—identifies a test call
  - INTDD—identifies an international direct dialed call
  - INTOA—identifies an international operator assisted call
  - CELLULAR—indicates a cellular originated call.

- S

  The standard selector specifies a CLLI or POSITION route to use for the call.

- T

  The table selector indicates a table index that identifies the route for the call.

- UAX

  The UAX selector may be used for information digit screening to identify a FLEXDIAL table index and consequently alter the interaction with the originating agent.

All other selectors are not supported including a number of call conditions for the ID selector:

- ID

  The following call conditions are not supported for OLI collectable processing:

  - AUTHREQ
  - CONDAUTH
  - CONDANI
  - VPNSNC
  - PUBLIC

  The use of these values results in FNAL treatment being immediately applied to the call.

Results of digit collection

The received digits are consumed from the digit buffer.

The originating line information digits received are recorded in the INFODIGS field of the CDR.

Use of the INTOA or INTDD call condition when validating the information digits in table STDPRTCT sets the NOA value for the called party address. A called party NOA value set by the INTOA or INTDD call condition cannot be overwritten through called party address digit processing (specifically, through COLDIG [where PROCNOA = CALLED] or ADDR/ADDRPARM collectable processing).

Use case scenario

The collectable is used in the following call scenario:

- FGD protocol

  Originating line information digits are collected and potentially validated using the OLI/OLIPARM collectables for information digit processing.

Restrictions and limitations

A number of STDPRTCT selectors are not supported for originating line information collectable processing.

Carrier Identification Code digit collectable (CIC/CICPARM)

The Carrier Identification Code (CIC) collectable provides the ability to collect carrier identification digits from the originating agent. The CIC collectable collects only digits that are processed as CICs.

Application

The CIC collectable follows the general application of inband digit collection.

The CICPARM digit collectable follows the general application of out-of-band digit collection.

Validate applicable digits

The digits received by the CIC/CICPARM collectable are not validated, where validation specifies that receiving “incorrect” digits results in a treatment or exception being generated for the call. The received digits are used to index other tables as the feature application of CIC digits.

Apply digit collectable features

The CIC digits are processed as outlined by the Enhanced CIC Support and CIC-Based Routing chapters in the *UCS DMS-250 CIC Routing Application Guide.*

The functionality provided by the Enhanced CIC Support and CIC-Based Routing is controlled through a software optionality control (SOC). The SOC UTRS0001 must be activated (ON state) for the CIC collectable to use this functionality. If the SOC is in an IDLE state, then table CICROUTE is not accessed.

These features identify two tables that apply to CIC processing:

- OPERRTE

  This table is index by the received CIC digits to identify a route for the call.

- CICROUTE

  This table is indexed by the received CIC digits and identifies information used to translate and process the address digits received.

If the CICROUTE table is properly indexed with the processed CIC digits, then a pretranslator name (PRTNM) for address and information digit processing is identified. These pretranslator names are messaged to the ADDR and OLI (ADDRPARM and OLIPARM) collectables through MSGCTR PRTNM messages, with non-dominant indicators.

Received CIC digits override digits specified in the CARRNUM field of the OPER message (see Table 4-15).

Results of digit collection

The received digits are consumed from the digit buffer.

In addition to the results from the application of the CICs, the received digits are captured in the CIC field of the CDR.

Use case scenario

The collectable is used in the following call scenario:

- FGD protocol

  CIC digits are collected and processed using the CIC/CICPARM collectables in accordance with TR394 and TR444 standards.

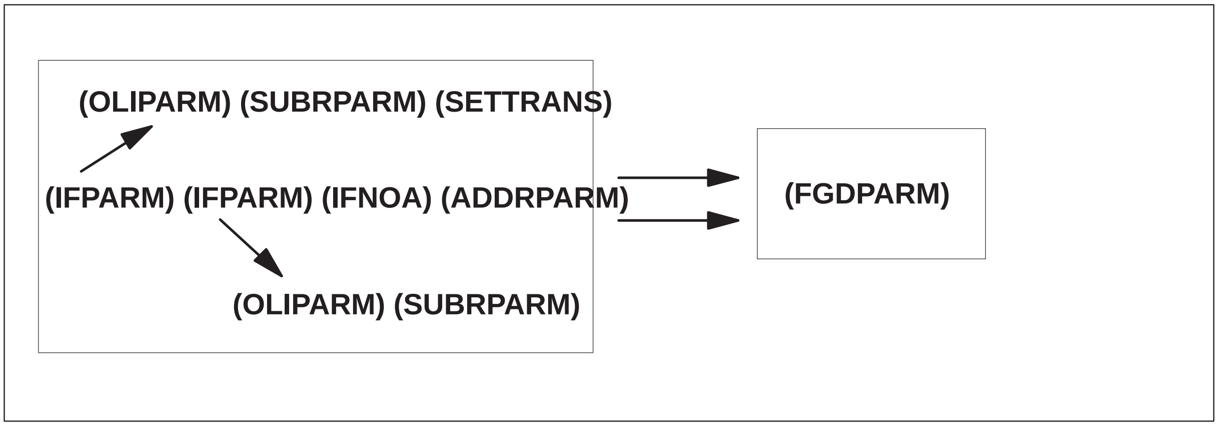
Restrictions and limitations

Received CIC digits are not validated (there is no failure case for incorrect digits received). As a part of feature application for the CIC collectables, the digits index a number of tables.

SS7 FGD digit collectable (FGDPARM)

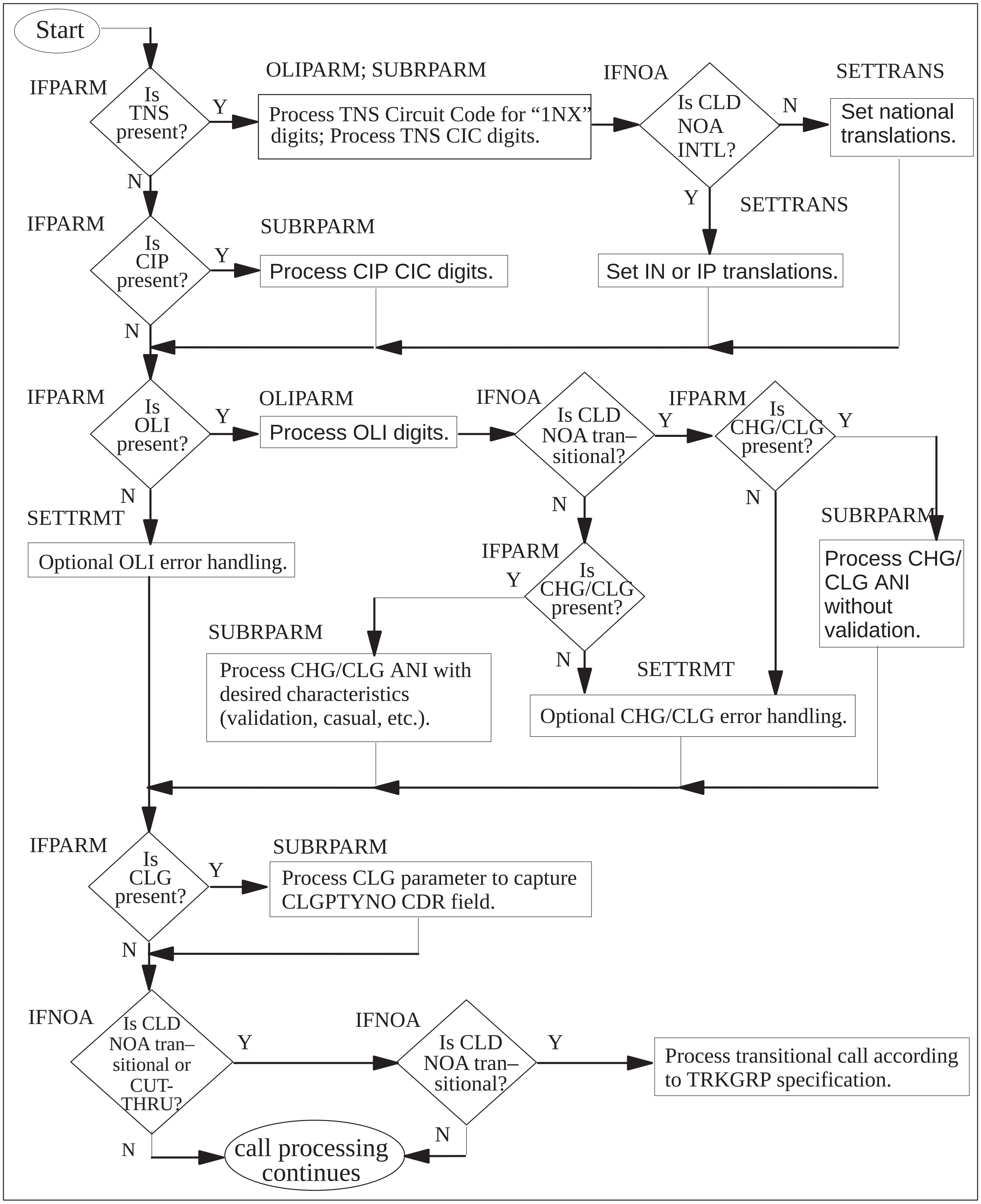
The SS7 FGD (FGDPARM) digit collectable performs, within one collectable, the SS7 FGD signaling protocol. To do so, it consolidates multiple collectables, such as OLIPARM, SUBRPARM, and ADDRPARM, into one collectable. See figure 16-36, Multiple collectables consolidated into one FGDPARM collectable.

Figure 16-36
Multiple collectables consolidated into one FGDPARM collectable


(OLIPARM) (SUBRPARM) (SETTRANS)
(IFPARM) (IFPARM) (IFNOA) (ADDRPARM)
(OLIPARM) (SUBRPARM)
(FGDPARM)

The FGDPARM collectable extracts and processes, in sequence, the parameter digits received in the IAM message. FGDPARM references the SS7 digit collectable information in table FlexDial. FGDPARM processes the call and makes decisions, internally, based on the provisioning data. Figure 16-37, FGDPARM collectable’s functionality, shows the processing the FGDPARM executes within one collectable.

Figure 16-37
FGDPARM collectable's functionality


Start
IFPARM
Is TNS present?
Y
N
OLIPARM; SUBRPARM
Process TNS Circuit Code for "1NX" digits; Process TNS CIC digits.
IFNOA
Is CLD NOA INTL?
N
Y
SETTRANS
Set national translations.
SETTRANS
Set IN or IP translations.
IFPARM
Is CIP present?
Y
N
SUBRPARM
Process CIP CIC digits.
IFPARM
Is OLI present?
Y
N
OLIPARM
Process OLI digits.
IFNOA
Is CLD NOA tran–sitional?
Y
N
IFPARM
Is CHG/CLG present?
Y
N
SUBRPARM
Process CHG/CLG ANI without validation.
SETTRMT
Optional OLI error handling.
IFPARM
Is CHG/CLG present?
Y
N
SUBRPARM
Process CHG/CLG ANI with desired characteristics (validation, casual, etc.).
SETTRMT
Optional CHG/CLG error handling.
IFPARM
Is CLG present?
Y
N
SUBRPARM
Process CLG parameter to capture CLGPTYNO CDR field.
IFNOA
Is CLD NOA tran–sitional or CUT-THRU?
Y
N
IFNOA
Is CLD NOA tran–sitional?
Y
N
Process transitional call according to TRKGRP specification.
call processing continues

Interactions

The following interactions apply to the FGDPARM collectable:

- The FGDPARM collectable reuses the functionality of the existing SS7 parameter digit collectables. For example, when the following message is posted in MSGCTR, the ADDRPARM collectable of FGDPARM consumes and processes the message:

  ADDR N Y PRTNM AXX

- The FGDPARM collectable is designed to be a stand-alone collectable to process the SS7 FGD protocol, but it may be provisioned with other collectables.
- The FGDPARM collectable supports three types of SS7 FGD calls:
  - Pure SS7 FGD calls
  - Cutthru SS7 FGD calls
  - Transitional SS7 FGD calls

  FGDPARM processes Pure SS7 FGD calls. For Cutthru and Transitional SS7 FGD calls executes indexes to process the remainder of the calls.

Replace Data digit collectable (REPLDIG)

The Replace Digits (REPLDIG) collectable provides the ability to collect digit information from the originating agent and replace digits in the digit buffer with the received information. The collect digits collectables do not process the received information, but leave processing of the information to other digit or sequence collectables. The collect digits collectables do not perform any validation or processing on the digits received.

Application

The REPLDIG collectable follows the general application of inband digit collection as described in “General application of inband digit collection” on page 16-51.

Identify digit collection request

The messages received from the message center may alter the request parameters for the collect digits collectable.

The FILED message identifies filed digits for the collectable. See “Application of filed digits” on page 16-60 for more information on the handling of filed digits.

The MINMAX message identifies the minimum and maximum values for the digit collection request, overriding the values provisioned in the MIN and MAX fields of the collectable.

Perform digit collection request

The REPLDIG collectable follows the general application of requesting digits from the agent terminal controller as described in “Perform digit collection request” on page 16-80, with the addition of the SIGFEAT capability.

SIGFEAT capability

The handling of the SIGFEAT option provisioned with the ADDR collectable is identical to SIGFEAT processing for the COLDIG collectable. See “SIGFEAT capability” on page 16-100.

Validate applicable digits

The REPLDIG collectable does not perform any validation on the digits received. Digits collected by the collectable are not stored in any CDR field.

Apply digit collectable features

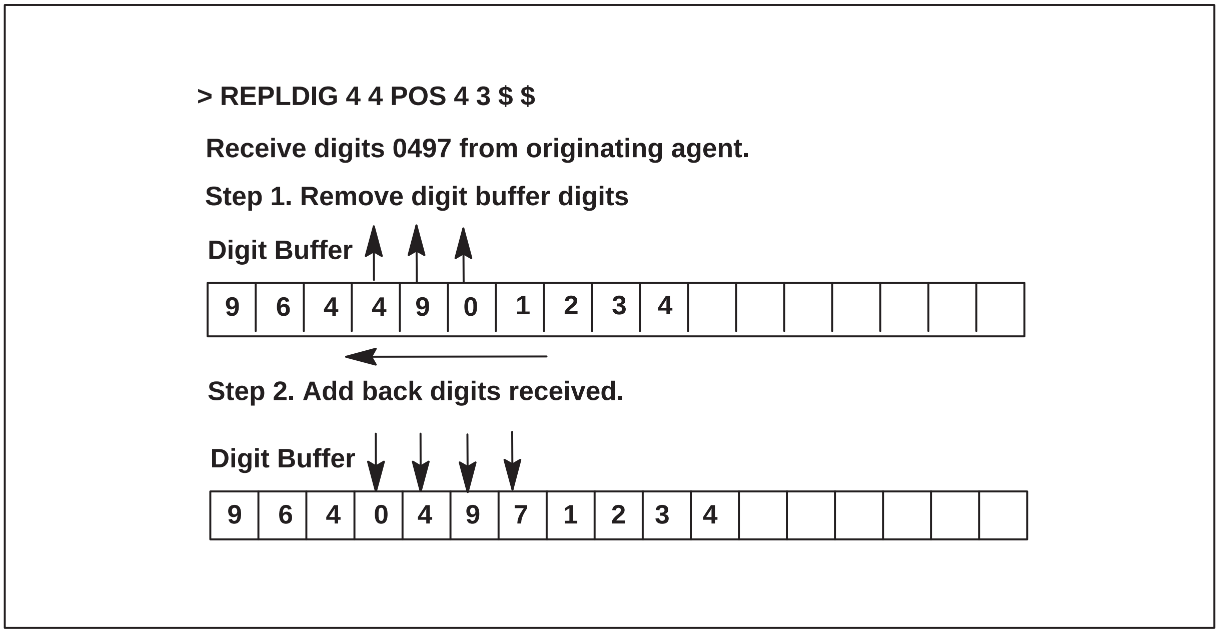
The REPLDIG digit collectable collects digits from the originating agent and uses them to replace digits currently in the digit buffer. No other processing is performed on the digits received.

The digits being replaced are first removed from the digit buffer. Similar to the DELDIGS collectable, there are no possible error scenarios: the buffer is simply shifted out by the remove operation.

The digits received by the REPLDIG collectable are added back to the digit buffer at the same location. More or less digits than the number of digits removed may be added back to the digit buffer based on the MIN/MAX values of the collectable, and remaining digits in the buffer are shifted appropriately.

Figures 16-38 and 16-39 illustrate the REPLDIG application.

Figure 16-38
RELPLDIG application



> REPLDIG 4 4 POS 4 3 $ $

Receive digits 0497 from originating agent.

Step 1. Remove digit buffer digits

Digit Buffer

9	6	4	4	9	0	1	2	3	4							

Step 2. Add back digits received.

Digit Buffer

9	6	4	0	4	9	7	1	2	3	4						

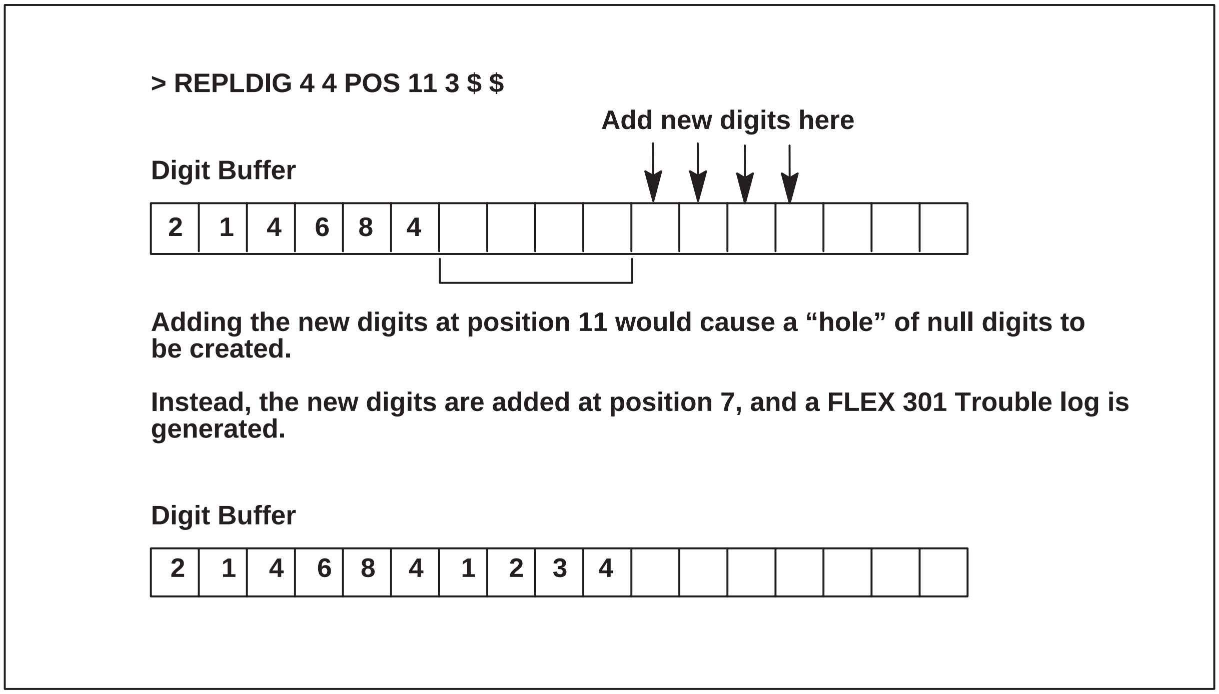
Holes of null digits may not be created in the digit buffer through use of the REPLDIG collectable. If such an attempt is made, then the digits are added at the closest possible location, and a FLEX 301 trouble log is generated with the following trouble code value:

```
Digits force appended to buffer
```

The Flextype field in the log is set to REPLDIG, and the digits added to the buffer by the REPLDIG collectable are included in the report.

This sort of error may only occur when attempting to replace null digits.

Figure 16-39
REPLDIG application



> REPLDIG 4 4 POS 11 3 $ $

Digit Buffer

2	1	4	6	8	4											

Adding the new digits at position 11 would cause a “hole” of null digits to be created.

Instead, the new digits are added at position 7, and a FLEX 301 Trouble log is generated.

Digit Buffer

2	1	4	6	8	4	1	2	3	4							

Results of digit collection

Existing digits in the digit buffer are replaced with digits collected by the REPLDIG collectable.

Use case scenario

The collectable is used in the following call scenario:

- interactive digit manipulation

  Using the REPLDIG collectable, previously received digits can be modified by requesting new digits from the originating agent.

Restrictions and limitations

The following restrictions and limitations apply to the REPLDIG collectable:

- If an attempt is made to copy digits in the digit buffer that exceeds the 64 digit capacity of the buffer, then the excess digits either being appended to the buffer or pushed out of the buffer due to the insert action of the REPLDIGS collectable are discarded and not added to the buffer. In addition, a FLEX 301 trouble log is generated with the following trouble code value:

  ```
  Digits truncated from buffer
  ```

The Flextype field in the log is set to REPLDIG and the digits truncated from the buffer are included in the digits field of the report.

- Holes of null digits may not be created in the digit buffer through use of the REPLDIG collectable. If an attempt to create a hole is performed, then the digits are appended where a hole is not formed, and a FLEX 301 trouble log is generated.

CALLTYPE collectable

The CALLTYPE collectable identifies features, call type characteristics, and billing information applicable to the call. This information is saved by call processing and affects execution and completion of the call as applicable.

The CALLTYPE collectable also identifies access related features and information, and can mark or capture information on a specific path of FLEXDIAL table processing.

Application

The CALLTYPE collectable contains indexes to two tables that are processed:

- table FLEXTYPE
- table FLEXFEAT

The FLEXTYPE specific OM CALLTYPE counter is incremented when a particular FLEXTYPE is processed by a CALLTYPE collectable.

The FLEXTYPE table identifies the following options that are applicable for the CALLTYPE collectable:

- ANSCDR

  identifies that a CDR is generated when answer occurs for the call, and identifies the CDRTMPLT table index to use for formatting purposes

- BILLFLGS

  captures specific information in the CALLTYPE field of the call detail record

- CDRTMPLT

  identifies the CDRTMPLT table index used for end-of-call CDR formatting purposes

With the exception of the REVALIDATE option, the attempted use of other FLEXTYPE table options by the CALLTYPE collectable are ignored. Use of the REVALIDATE FLEXTYPE option by a call type collectable results in the generation of a FLEX 301 trouble log with the following trouble reason:

```
Cannot apply REVALIDATE option
```

The Flextype field in the log is set to CALLTYPE and no digits are included in the report.

The FLEXFEAT table identifies the following options that are applicable for the CALLTYPE collectable:

- ANSCDR

  This option identifies that a CDR is generated when answer occurs for the call, and identifies the CDRTMPLT table index to use for formatting purposes.

- BCCOMPAT

  This option identifies bearer capability for the originating agent of the call.

- CAINGRP

  This option specifies an index into the CAINGRP table for AIN processing.

- CDRTMPLT

  This option identifies the CDRTMPLT table index used for end-of-call CDR formatting purposes.

- CITYCODE

  This option may identify a city code value that indexes table CITYCODE for public speed dial number processing.

- CLDPBILL

  This option identifies the call as a called party billed call.

- DELIVER

  This option identifies the calling party subscriber number delivery status for the call.

- DPIDX

  This option identifies a FLEXDIAL table index for processing.

- FAILVAL

  This option identifies a treatment set for the call.

- GENLOG

  This option generates a FLEX 601 information log.

- MLTCOSID

  This option contains a MULTICOS table index to use for COS screening.

- MSGCTR

  This option specifies a MSGCTR table index that outlines messages that are to be posted at the message center for processing.

- NOANSDUR

  This option identifies a timer value for the no answer duration feature.

- ONNET

  This option identifies the call as an on-network call.

- PVSPDIDX

  This option identifies a private speed dial index to use for the call.

- REORGACT/REORGTYP

  These options outline reorigination capability for the call.

- SPLASHBK

  This option identifies an index for the SPLASHID table for splash back tone application.

- TCAPANNC

  This option identifies an index into the TCAPANNC table for N00 TCAP application processing.

- TRANSTS

  This option identifies the STS for translations.

When the FLEXFEAT TRANSTS option is used, then the following fields in the CDR are also captured:

- STS

  The STS field contains the serving translation scheme used for call translations. This field is overwritten each time an STS is identified through use of the TRANSTS option.

- ORIGSTS

  The ORIGSTS field contains the first STS set for the call. If the field has not been captured yet when the TRANSTS option is processed, then the STS value set is captured. The ORIGSTS CDR field is not overwritten on processing of multiple TRANSTS options.

All of these features and capabilities can be overridden later in the call process through subsequent accesses to tables FLEXTYPE and FLEXFEAT, or through use of the CLRFTRS collectable.

Messages received from the message center may alter processing of the CALLTYPE collectable. The only message received by the CALLTYPE collectable is the NOANSDUR message.

The NOANSDUR message can be identified through processing of the feature bytes parameter during N00 TCAP application processing. If the PROCESS NOANSDUR option is used in table FEATBYTE, then the feature byte value is included in the NOANSDUR message center message, and is used as the timer value for the no answer duration feature. For more information, see Appendix D, ”FlexDial interactions” and the *UCS DMS-250 Transaction Capabilities Application Part (TCAP) Application Guide*.

The NOANSDUR message received by the CALLTYPE collectable includes the timer value for the feature and identifies that the NOANSDUR feature must be active. If the NOANSDUR option is not provisioned against the FLEXFEAT table entry referenced by the CALLTYPE collectable, then the no answer duration feature is activated using the timer value supplied by the message. If the timer expires, then no terminal responding (NTRS) treatment is applied to the call.

Note: The N00 TCAP application may be processed during ADDR or ADDRPARM collectable processing.

If the option is present in the FLEXFEAT table entry referenced by the CALLTYPE collectable, then the feature is activated using the timer value from the message center message instead of the value provisioned with the option. The action provisioned against the option is used if the timer expires.

Use case scenario

The collectable is used in the following call scenario:

- identify specific billing information for the call in progress

  Through use of the BILLFLGS FLEXTYPE table option, certain access, service, or call type values may be captured in the billing record for the call. The information captured is essentially based on the path of collectable execution in the FLEXDIAL table.

Restrictions and limitations

The following restrictions and limitations apply to the CALLTYPE collectable:

- Use of the REVALIDATE FLEXTYPE or FLEXFEAT table option by a CALLTYPE collectable results in the generation of a FLEX 301 trouble log with the following trouble reason:

  ```
  Cannot apply REVALIDATE option
  ```

  The Flextype field in the log is set to CALLTYPE and no digits are included in the report.

- Use of the FAILVAL FLEXFEAT table option by a CALLTYPE collectable when the failure action identified is not a TRMT action results in the generation of FLEX 301 trouble log with the following trouble reason:

  ```
  FAILVAL ignored. Can only apply TRMT action.
  ```

  The Flextype field in the log is set to CALLTYPE and no digits are included in the report.

- The following FLEXFEAT table options are ignored during CALLTYPE collectable processing:
  - CASUAL
  - FLDONLY

Clear Call Features collectable (CLRFTRS)

The Clear Features And Characteristics (CLRFTRS) collectable sets the identified features and characteristic to their off or default values. All of the features identified to be cleared are enabled for the call through FLEXFEAT table processing, although some of the features and characteristics may have alternatively been set through TRKFEAT table provisioning. (Some features, such as the CDRTMPLT option, can be provisioned in both tables.)

Application

The clear features collectable sets each feature identified to its off or default value. See Table 16-11 for per option applications of the CLRFTRS collectable.

Table 16-11
Per option application of the CLRFTRS collectable

Option	Off or default value description
ANSCDR	Generation of an answer CDR is disabled.
BCCOMPAT	Defaulted to 3_1KHz.
CAINGRP	Identified CAINGRP index is cleared disabling indexing into the CAINGRP table for AIN purposes.
CASUAL	No applicable default. No processing is performed for this option.
CDRTMPLT	Clears CDRTMPLT table index identified for CDR formatting.
CITYCODE	The CITYCODE value stored for the subscriber is cleared.
CITYVAL	No processing is performed for this option.
CLDPBILL	Call is reset to default of calling party billed. Additionally, the “calling party” billing number digits are restored.
DELIVER	The calling party number deliver value used for the call is reset to ALWAYS.
DPIDX	All history of previously executed collectables is cleared. This does not affect normal collectable processing, but does affect reset functionality.

—continued—

Table 16-11
Per option application of the CLRFTRS collectable (continued)

Option	Off or default value description
FAILVAL	Any delayed treatment set by digit collectable processing is cleared.
FLDONLY	No applicable default. No processing is performed for this option.
GENLOG	No applicable default. No processing is performed for this option.
MLTCOSID	The multiple COS screening index is set to zero, effectively disabling class of service screening for the call.
MSGCTR	All messages currently posted at the message center from provisioned sources are removed. ***Note:*** The provisioned source for messages is table MSGCTR, and are typically posted through processing the MSGCTR option of table TRKFEAT or table FLEXFEAT. Messages that are posted through call processing sources are not removed.
NOANSDUR	The duration timer value is set to zero, effectively disabling the no answer duration feature.
ONNET	The call is reset to a default of off network.
PVSPDIDX	The private speed dial index is set to zero, defaulting the value used.
REORGPRC	If either this option or REORGTYP is included in the FEATLIST vector, then reorigination capability is disabled for the call.
REORGTYP	If either this option or REORGPRC is included in the FEATLIST vector, then reorigination capability is disabled for the call.
REVALIDATE	The list of all processed subscriber number collectables identified to be revalidated on a reoriginated call is cleared.
SPLASHBK	The splash back index is set to zero, effectively disabling this feature.
TCAPANNC	The TCAPANNC index is defaulted to zero.
TRANSTS	The STS is set to the STS value identified by the DEFAULT_STS office parameter.

—continued—

Table 16-11
Per option application of the CLRFTRS collectable (continued)

Option	Off or default value description
TRANSNUM	No CLRFTRS processing is performed related to the TRANSNUM option.
TRANSYS	The translations system is set to NP, the nil translations system. Upon reset of the CLRFTRS collectable, the previous translation system is restored.
IEXCLINX	The incoming exclusion screening index is set to zero (0). Upon reset of the CLRFTRS collectable, the previous incoming exclusion screening index value is restored.
FEATVAR	All FLEXFEAT variables activated for the call are deactivated. Upon reset of the CLRFTRS collectable, the FEATVAR variables are re-activated.
CPACTVAL	All call processing active subscriber numbers and calltypes are deactivated for the call. Upon reset of the CLRFTRS collectable, the active validation status for the subscriber numbers and calltypes is re-activated.
—end—	

The reprocessing of these features and characteristics later in the processing of the call re-enables them as applicable.

Use case scenario

The collectable is used in the following call scenario:

- clear features set for DPIDX reset processing

  Use of the DPIDX action for reset on inband digit collectables results in the execution of a new collectable list. The CLRFTRS collectable erases what happened in previous collectable processing.

Restrictions and limitations

The following restrictions and limitations apply to the CLRFTRS collectable:

- Only options provisioned in table FLEXFEAT are applicable for the CLRFTRS collectable.
- Overlapping features and characteristics set by other sources (such as table TRKFEAT) are also handled by the CLRFTRS collectable. (Also included are other things such as messages left at the message center by non-subscriber number processing.)

Set Translation System collectable (SETTRANS)

The Set Translation System (SETTRANS) collectable sets the translation system used for called party address translations to the specified value. Typically this is used when pretranslation of the called party address digits is not performed, and the default setting of national translations is inappropriate for the call.

Application

The Set Translation System collectable sets the translation system used for the call to the provisioned value. The following values may be used and indicate the appropriate tables are incorporated for called party address translations:

- NA—for national translation purposes using tables HNPACONT or FNPACONT
- IN—for international translation purposes using table CCTR
- IP—for international partitioned translation purposes using tables CTHEAD, CTCODE, and CTRTE
- NO—specifies a nil translation system

Performing pretranslation of called party address digits also typically sets the translation system used for the call, and may overwrite the value set through the SETTRANS collectable. Likewise, use of the SETTRANS collectable may overwrite a value set during pretranslations.

Use case scenario

The collectable is used in the following call scenario:

- set the translation system for FGD protocol three-stage MF dialing

  For three-stage FGD protocol interactions, pretranslation of the address digits does not occur. Therefore, the SETTRANS collectable is used to specify the appropriate translation system for the call.

Restrictions and limitations

The following restrictions and limitations apply to the SETTRANS collectable:

- Execution of SETTRANS may overwrite a translation system value set through pretranslations.
- Executing pretranslations of the called party address digits may overwrite a translation system set by the SETTRANS collectable.

Set Treatment collectable (SETTRMT)

The Set Treatment (SETTRMT) collectable sets a specific treatment in the FLEXDIAL call processing, as a result of following a particular path of collectable execution.

Application

The SETTRMT collectable sets the specified treatment for the call (immediately applied treatments, such as FNAL, result in suspension of processing and routing of the call to treatment processing). The treatment can be datafilled to either override, or not override, any previously set treatment for the call.

Use case scenario

The SETTRMT collectable is used in the following scenario:

- apply call treatment after further subscriber action

  For security reasons, it is sometimes desirable to have the subscriber continue to enter digit information after the entry of an invalid subscriber number. This reduces the possibility of an individual identifying security related numbers (such as authcodes and/or PIN numbers) using trial and error. The SETTRMT collectable can be strategically placed in FLEXDIAL call processing to force application of a set treatment at specific points in the digit collection interaction.

Restrictions and limitations

Use of SETTRMT to apply any of the immediate call treatments (FNAL, PSIG, PDIL, RESU) results in the immediate application of treatment to the call. If a treatment was already set for the call, and override was not indicated by APRESET, then immediate application of these treatments does not occur because the treatment is not set.

Apply Reset collectable (APRESET)

The Apply Reset (APRSET) collectable applies a deferred reset in the FLEXDIAL collectable processing.

Application

When a deferred reset is indicated earlier in the call, APRESET rewinds the collectable list until the collectable that indicated the deferred reset is encountered. Along the way, all collectables perform their specific reset functions. When the collectable that indicated deferred reset is encountered, the specific reset action of that collectable is taken.

Use case scenario

The APRESET collectable is used in the following scenario:

- apply reset after collection of address digits

  In the FGD MCCS with PIN digit collection scenario, it may be desirable to collect PIN digit information, regardless of whether the MCCS number was successfully validated. This is accomplished by using a deferred action on validation failure, in conjunction with the APRESET collectable.

Restrictions and limitations

The APRESET has no call processing effect if a deferred reset was not indicated at some point prior to the call.

Operation Measurement collectable (OM)

The operational measurement (OM) collectable increments a user-defined operational measurement as a result of following along a particular path of FLEXDIAL execution.

Application

The OM collectable, when datafilled in table FLEXDIAL, creates a new OM type (or references an existing OM). The OM collectable increments the datafilled OM.

Use case scenario

The OM collectable is used in the following call scenario:

- count subscriber number validation failures

  On validation failure of a subscriber number, an EXEC action could be used to cause execution of a FLEXDIAL index containing an OM collectable that increments an ”authcode_failure” measurement. This OM could be monitored to indicate fraud attempts within a particular trunk group.

Restrictions and limitations

There are no itemized restrictions for use of the OM collectable in FLEXDIAL call processing.

Variable Operation collectable (VAROP)

The VAROP collectable is constructed of three components:

- The left side operator
- The operation
- The right side operator.

For VAROP collectable application, the left side operator is modified according to the operation which incorporates the right side operator, the operation updating the value of the left side operator.

The FlexDial variables (AVAR, BVAR, CVAR, and DVAR) as well as the FLEXTYPE FEATVAR variable are supported as left side operators. If a FLEXTYPE FEATVAR is used, then the variable must be activated for the call prior to execution of the VAROP collectable. If the variable has not been activated, then a FLEX 307 log is generated with a trouble reason of “LEFT VAR FEATVAR Not Yet Identified”, and “feature not allowed” (FNAL) treatment is immediately set and applied for the call.

The operations supported include basic assignment of the value of the right side operator or simple mathematical operations using the right side operator. The result of the operation is assigned to the left side variable operator. Overflows or operation errors are not permitted for VAROP operations. If an overflow error (such as incrementing beyond the possible maximum value for the variable or decrementing beyond the possible minimum value) or an operational error (such as attempting to divide by zero) occurs, then a FLEX 307 log is generated with a trouble reason of “Results exceeded max allowed” or “Attempt to divide by zero”, and “feature not allowed” (FNAL) treatment is immediately set and applied for the call.

The right side operator supports three distinct types:

- VARIABLE – identifies a FlexDial variable or FEATVAR variable
- INTEGER – specifies use of a specific integer value
- CALLDUR – indicates use of the current call duration as the right side operator value.

For a right side operator type of VARIABLE, the FlexDial variables (AVAR, BVAR, CVAR, and DVAR) as well as the FLEXTYPE FEATVAR variable are supported. If a FLEXTYPE FEATVAR is used, then the variable must be activated for the call prior to execution of the VAROP collectable. If the variable has not been activated, then a FLEX 307 log is generated with a trouble reason of “RIGHT VAR FEATVAR Not Yet Identified”, and “feature not allowed” (FNAL) treatment is immediately set for the call.

For a right side operator type of INTEGER, the right operator value is taken directly from the provisioning.

For a right side operator type of CALLDUR, the current call duration (from the point of answer) is calculated in either units of minutes or seconds and becomes the value of the right side operator. If the call has not yet been answered, then the right side operator is simply set to a value of zero.

The VAROP collectable may also be the recipient of a VAROP MSGCTR message, identifying the variable for the left side operator.

Usecase Scenarios

The VAROP collectable provides a generic mechanism for FlexDial call processing whereby an allocated variable can be modified and compared to control the interaction with the originating agent.

Interactions

The VAROP collectable requires the use of the FLEXFEAT FEATVAR table option in order to perform FEATVAR variable related operations. Additionally, the IFVAR collectable is used to perform unique variable comparisons.

Restrictions and limitations

The following call processing restrictions or limitations exist for the VAROP collectable:

- A referenced FLEXTYPE FEATVAR must be activated prior to use of an VAROP collectable containing the FEATVAR within an expression. If the proper FLEXTYPE FEATVAR has not been activated prior to execution of the VAROP collectable, then “feature not allowed” (FNAL) treatment is immediately applied to the call, and a FLEX 307 log is generated with a reason of “Uninitialized variable accessed.”

The Variable Operation (VAROP) collectable performs mathematical operations involving any of the four pre-defined FLEXDIAL variables (AVAR, BVAR, CVAR, and DVAR).

Variable Comparison Conditional Branch collectable (IFVAR)

The Variable Comparison Conditional Branch (IFVAR) collectable allows dynamic branching of collectable processing, based on the value of one of the defined FLEXDIAL variables (AVAR, BVAR, CVAR, DVAR, FLEATVAR). FLEATVAR represents use of the FLEXFEAT FEATVAR table option.

The IFVAR collectable modifies the interaction with the subscriber, according to the value of any of the FLEXDIAL variables. These variables allow the operating company personnel to count events or set indications during FLEXDIAL processing. This collectable is provisioned with the following four fields of call processing information:

- The method of comparison to use
- A vector of up to four variable comparisons
- The FLEXDIAL IFTRUE and IFFALSE indices to use
- The FLEXDIAL action to take

Call Processing Application

The IFVAR collectable does not modify a FlexDial variable identified in the IFVAR expressions vector, but bases comparisons against what is currently set for the FlexDial variables. This collectable is provisioned with two key elements of information needed for processing:

- The expressions involving FlexDial variables.
- The FLEXDIAL table index to branch to based on comparison results, and the appropriate action for application of this index.

If the result of the expression evaluation according to the comparison method is true, then the IFTRUE FLEXDIAL table index is used. Otherwise, the IFFALSE FLEXDIAL table index is used. The action field identifies how the collectables identified by the FLEXDIAL table index are interworked with the current collectable list.

Variables AVAR, BVAR, CVAR, and DVAR are defaulted to zero (0) for the call, and can only be assigned different values through use of the VAROP collectable. There is no penalty for attempting to use these variables through an IFVAR collectable prior to explicitly setting them through VAROP.

For FEATVAR use, the identified FLEXTYPE FEATVAR must be currently active for the call for successful IFVAR comparisons. If the IFVAR expression identifies a FLEXTYPE FEATVAR which has not been activated for the call, then a FLEX 307 log is generated with the trouble reason of “Uninitialized variable accessed” and “feature not allowed” (FNAL) treatment is immediately applied to the call.

Usecase

The IFVAR collectable provides a generic mechanism for FlexDial call processing whereby an allocated variable can be modified and compared to control the interaction with the originating agent.

Interactions

The IFVAR collectable requires the use of the FLEXFEAT FEATVAR table option in order to perform FEATVAR variable related comparisons. Additionally, the VAROP collectable is used to set values for FlexDial variables.

Call processing restrictions and limitations

The following call processing restrictions or limitations exist for the IFVAR collectable:

- A referenced FLEXTYPE FEATVAR must be activated prior to use of an IFVAR collectable containing the FEATVAR within an expression. If the proper FLEXTYPE FEATVAR has not been activated prior to execution of the IFVAR, then “feature not allowed” (FNAL) treatment is immediately applied to the call, and a FLEX 307 log is generated with a reason of “Uninitialized variable accessed.”

For the “IS” comparison method, if the comparison is successful (just one of the identified variable comparisons is true), then the IFTRUE index is selected. Otherwise, the IFFALSE index is selected.

For the “NOT” comparisons method, if none of the variable comparisons are true, then the IFTRUE index is selected. Otherwise, the IFFALSE index is selected.

The FLEXDIAL action taken by the collectable (INSERT, APPEND, REPLACE, or EXEC) is applied using the FLEXDIAL index supplied by the indicated IFTRUE or IFFALSE index, according to the results of the comparison. If the NIL index is indicated, then the collectable takes no action.

The IFVAR collectable may be executed multiple time during the processing of a call.

FLEXTYPE features and characteristics applications

Table FLEXTYPE provides the specification of subscriber number types and call type definitions within the FlexDial framework. Through the subscriber number collectables (SUBR, SUBRPARM) and the call type collectable (CALLTYPE), entries in the FLEXTYPE table are processed for features and characteristics that apply to the call.

Supported agents

FLEXTYPE table processing occurs only during collectable processing within the FlexDial framework. Execution of collectables in the framework is applicable only when using the AXXESS trunk group type originating agent.

Activation

Processing of FLEXTYPE table entries occurs during the “collect information” point in the call model. Entries in table FLEXTYPE are processed by call processing through two methods:

- execution of subscriber number digit collectables (SUBR, SUBRPARM)
- execution of call type collectables (CALLTYPE)

Features and characteristics provisioned in table FLEXTYPE differ from those provisioned in table FLEXFEAT in that the options provisioned in table FLEXTYPE are on a per subscriber number type basis, and not on a more granular subscriber number basis.

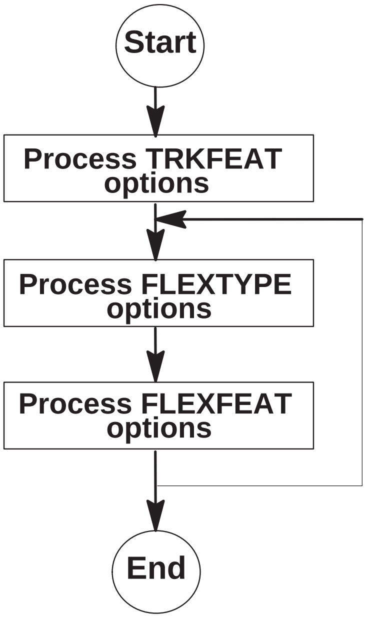
Some features and characteristics have representation in two or more of the following tables:

- FLEXTYPE
- FLEXFEAT
- TRKFEAT

If an option is provisioned in multiple tables, the call processing uses the values set by the last option processed.

Figure 17-1 shows features and characteristics run time processing.

Figure 17-1
Features and characteristics run time processing



Each set of options processed may overwrite values previously set.

Process TRKFEAT options once up front.

For each subscriber number call type collectable, process the FLEXTYPE options first. Then process the FLEXFEAT options.

The execution of multiple subscriber number or call type collectables causes the identified FLEXTYPE and FLEXFEAT indexes to be processed, potentially overwriting previous values set.

For the subscriber number digit collectables, processing of FLEXTYPE entries occurs during the "apply collectable features" portion of collectable processing.

For both the subscriber number collectables (SUBR, SUBRPARM) and the call type collectable (CALLTYPE), table FLEXTYPE options are processed before table FLEXFEAT options.

Call processing description

The FLEXTYPE table provides the following features and capabilities for call processing purposes:

- ANSCDR
- BILLFLD
- BILLFLGS
- CALLING
- CDRTMPLT
- EMPTYIDX
- FLEXLOG
- OPERDISP
- CAINFLG
- REVALIDATE

ANSCDR call processing application

The Answer Call Detail Record (CDR) (ANSCDR) option enables the generation of a CDR when answer occurs for the call, and is applicable for both subscriber number collectable and call type collectable processing.

When answer occurs for a call with this feature active, the per-call recording unit for the call is copied and delivered to the billing system for formatting. Upon disconnect, a completed CDR record is also formatted for the call.

The billing record is delivered to the device independent recording package (DIRP) stream identified through the FCDR_ANSCDR_CDT office parameter (see the *UCS DMS-250 Billing Records Application Guide*). This office parameter identifies the call data type that indexes table CRSMAP. The DIRP stream name is retrieved from the CRSMAP table entry.

Call processing actions

Processing the ANSCDR option identifies the following information for call processing:

- the CDRTMPLT table index to use for answer CDR formatting
- an indicator identifying that the editing versus active template is to be used to format the CDR for the DIRP stream

The ANSCDR option can be processed multiple times throughout the call. Each subsequent handling of the ANSCDR option overwrites the previous feature information stored for the call. The ANSCDR feature may only be disabled through the CLRFTRS collectable.

In addition to the FLEXTYPE table, the ANSCDR option can be provisioned in the FLEXFEAT and TRKFEAT tables. The order in which the tables are processed identifies the precedence for setting the ANSCDR feature information:

1 The TRKFEAT table ANSCDR option is processed before any collectables are executed. The TRKFEAT options are only processed once during the duration of a call.

   Note: The TRKFEAT options information is stored in per-call data structures, and can trigger the execution of other features at a later point in the call.

2 During execution of a subscriber number or call type collectable, the FLEXTYPE table ANSCDR option is processed, overwriting information set by the TRKFEAT option or by a previously executed subscriber number or call type collectable.

3 During execution of a subscriber number or call type collectable, the FLEXFEAT table ANSCDR option is processed after the FLEXTYPE table option, overwriting any feature information previously set.

Once set, the ANSCDR feature information is retained in call processing data structures until answer occurs for the call. When answer occurs for the call:

1 An additional recording unit is allocated.

2 Information from the per-call recording unit is copied to the additional recording unit.

3 The additional recording unit is delivered to the formatter and formatted according to the template definition in the CDRTMPLT table entry.

4 The formatted CDR is delivered to the DIRP file system.

After the answer CDR is delivered to the billing system, a value of 1 is captured in the ANSCDR field of the CDR.

FlexDial generate CDR upon answer feature

The “FlexDial generate CDR upon answer” feature generates a log for an originating AXXESS agency, or generates a CDR instead of a log upon answer for originating PRI, DAL, FGD and AXXESS agencies.

For originating PRI, DAL and FGD agencies, a log or a CDR is generated based on the datafill of the switch and the office parameter NETWORK_SECURITY_GEN_CDR.

If the office parameter NETWORK_SECURITY_GEN_CDR is set to N, a log is generated. If the office parameter NETWORK_SECURITY_GEN_CDR is set to Y, a CDR is generated. The CDR template information is identified in office parameter NETSEC_CDR_TMPLT.

For the originating AXXESS agency, if the ANSCDR option is present in any of the tables FLEXFEAT, TRKFEAT, FLEXTYPE, or RTEATTR, a log or a CDR is generated.

If the office parameter NETWORK_SECURITY_GEN_CDR is set to N, a log is generated. If the office parameter NETWORK_SECURITY_GEN_CDR is set to Y, a CDR is generated. The CDR template information is identified in the ANSCDR option.

If the ANSCDR option is not present in any of the table FLEXFEAT, TRKFEAT, FLEXTYPE, or RTEATTR, no log or CDR is generated.

For more information, see the *UCS DMS-250 Office Parameters Reference Manual*, *UCS DMS-250 Logs Reference Manual,* and *UCS DMS-250 Billing Records Application Guide.*

Restrictions and limitations

The following restrictions and limitations apply to the ANSCDR call processing application:

- Because the CDR is generated upon answer of the call, it is only complete with information captured to that point in the call. The template identified for formatting purposes must take into consideration that the following fields have not been captured at that point in the call:
  - CALLDUR
  - CDRALGOR
  - COMPCODE
  - DISCDATE
  - DISCAMPM
  - DISCTIME
  - DISCTYPE
- If an additional recording unit is not available when answer occurs for the call, then no answer CDR is generated.

BILLFLD call processing application

The Billing Record Field Identification (BILLFLD) option specifies the field in the CDR where the received subscriber number digits are to be stored. This field has no application for call type collectable processing and is subsequently ignored.

Call processing actions

The billing record field identified is only used to store the subscriber number digits if the BILLCAPT field of the subscriber number collectable is set to Y.

One of a number of billing record fields may be used by the BILLFLD option. See “BILLFLD option” on page 5-4 or “Apply digit collectable features” on page 16-118. See the *UCS DMS-250 Billing Records Application Guide* for a description of the billing record fields.

The received subscriber number digits are appended to digits already captured in the identified recording unit field. This occurs if two or more subscriber number types processed for a call identify the same field for capturing the received subscriber number digits.

If there is not enough room in the identified recording unit field to store all of the received digits, the excess digits are truncated and cannot be captured. This results in the generation of a FLEX 304 subscriber number capture failure log.

Interactions

For subscriber number collectable interactions, see “Subscriber Number collectable (SUBR/SUBRPARM)” on page 16-104.

Restrictions and limitations

This option is only applicable for subscriber number collectable processing. This option is ignored in call type collectable processing.

BILLFLGS call processing application

The Billing Record Flags (BILLFLGS) option identifies changes to information captured in the CALLTYPE field of the CDR, and is applicable for both subscriber number collectable and call type collectable processing.

Call processing actions

When processed, the information contained within the BILLFLGS option is immediately captured in the CALLTYPE field of the CDR. The following information is contained within the CALLTYPE field:

- ACCESS value—range 0 to 31

- SERVICE value—range 0 to 63
- CALLTYPE value—range 0 to 31

If the sub-field selector is set to NOCHANGE, then the value captured in the CALLTYPE CDR field is not altered. The value stored in the CDR is only overwritten if the subfield selector is set to CHANGE.

Interactions

Through the CALLTYPE collectable, this option can be used to store billing information related to the processing of FLEXDIAL table collectables.

This option relies on the CALLTYPE CDR field definition as outlined in the *UCS DMS-250 Billing Records Application Guide.*

Restrictions and limitations

This option immediately updates the CALLTYPE field in the CDR. These changes may only be undone through a reset action on the collectable that use the BILLFLGS option, or by executing another collectable that would overwrite the set values.

CALLING call processing application

The Calling Party Number (CALLING) option identifies the received subscriber number as the calling party address for the call. This field has no application for call type collectable processing and is subsequently ignored.

As the identified calling party address, the subscriber number received is involved in multiple areas of processing:

- in determining the NOA for the calling party number and capturing the value in the CLGNOA field of the CDR billing record
- in outpulsing, where the calling party address digits are delivered to the next network switch
- in TCAP applications requiring calling party address digits to be present in the query to the SCP
- in logs and other applications requiring the calling party address digits

Call processing actions

When processed, the calling party tag is immediately applied to the subscriber number collectable and the received digits. The NOA of the received digits is captured in the CLGNOA billing record field.

If another subscriber number type identified as the calling party is subsequently processed, the received digits replace the originally identified calling party address digits.

Interactions

For subscriber number collectable interactions, see “Subscriber Number collectable (SUBR/SUBRPARM)” on page 16-104.

Restrictions and limitations

This option is only applicable for subscriber number collectable processing. This option is ignored in call type collectable processing.

CDRTMPLT call processing application

The CDR Template (CDRTMPLT) option identifies the entry in table CDRTMPLT that specifies the layout of the CDR generated for the call, and is applicable for both subscriber number collectable and call type collectable processing.

The billing record is delivered to the DIRP stream identified by the OCC call data type. This call data type indexes table CRSMAP. The DIRP stream name as defined by table CRSFMT is retrieved from the CRSMAP table entry.

Call processing actions

If the BILLACT field in the CDRTMPLT option is set to Y, then billing is active for the call. (For more information on the BILLACT field, see “CDRTMPLT option” section on page 5-9 .)

Processing the CDRTMPLT option identifies the following information for call processing:

- the CDRTMPLT table index to use for end of call formatting
- an indicator identifying that the editing versus active template is to be used to format the CDR for the DIRP stream

The CDRTMPLT option may be processed multiple times throughout the call. Each subsequent handling of the CDRTMPLT option overwrites the previous information stored for the call. The effects of the CDRTMPLT option can be cleared only through use of the CLRFTRS collectable.

In addition to the FLEXTYPE table, the CDRTMPLT option can be provisioned in the FLEXFEAT and TRKFEAT tables. The order in which the tables are processed identifies the precedence for setting the CDRTMPLT call information.

- The TRKFEAT table CDRTMPLT option is processed before any collectables are executed. The TRKFEAT options are only processed once during the life of a call.

  Note: The TRKFEAT options information is stored in per-call data structures, and can trigger the execution of other features at a later point in the call.

- During execution of a subscriber number or call type collectable, the FLEXTYPE table CDRTMPLT option is processed, overwriting information set by the TRKFEAT option or by a previously executed subscriber number or call type collectable.
- During execution of a subscriber number or call type collectable, the FLEXFEAT table CDRTMPLT option is processed after the FLEXTYPE table option, overwriting any feature information previously set.

At the end of the call, if the BILLACT field’s value is Y, the CDR is generated according to the specifications in the CDRTMPLT option. But, if the BILLACT field’s value is N, then no CDR is generated for the call.

Once set, the CDRTMPLT call information is retained in call processing data structures until the call is taken down (disconnect occurs). At disconnect, the CDRALGOR field is captured with a value from 0 to 7 indicating how call processing determined the template to use for formatting. The record is then formatted according to the template definition in the CDRTMPLT table entry, and delivered to the billing system.

Interactions

For more information on the flexible CDR application for the FlexDial framework, see the *UCS DMS-250 Billing Records Application Guide.*

Restrictions and limitations

If the CDRTMPLT option is not used for a call, the format for the call detail record may be identified through other methods. See the *UCS DMS-250 Billing Records Application Guide.*

EMPTYIDX call processing application

The Empty Screening Index (EMPTYIDX) option identifies that table FLEXVAL validation of a subscriber number is handled as a successful validation attempt if the subscriber number of this type (meaning the specified type provisioned with the EMPTYIDX option) is screened against an empty numeric index. An empty numeric index is identified as a numeric index value not used in any entries to the FLEXVAL table for the subscriber number type (FLEXTYPE).

This field has no application for call type collectable processing and is subsequently ignored.

Call processing actions

When a FLEXVAL validation attempt is performed on the received subscriber number digits, the validation attempt requires the following information to index the FLEXVAL table:

- subscriber number type (identified in the collectable FLEXTYPE field)
- numeric index (identified in the collectable VALIDATE option or through the message center)
- received subscriber number digits

For normal validation attempts, if the three articles of provisioned and call information (received digits) do not match an existing table entry, validation is considered unsuccessful.

If validation is unsuccessful because there are no table entries provisioned using the identified numeric index for the specified subscriber number type (FLEXTYPE), the use of the EMPTYIDX option for the subscriber number type results in a successful validation outcome.

Interactions

The following interactions apply for the EMPTYIDX call processing application:

- For information on FLEXVAL table provisioning, see “Table FLEXVAL” in Chapter 6..
- For subscriber number collectable interactions, see “Subscriber Number collectable (SUBR/SUBRPARM)” on page 16-104.

Restrictions and limitations

The following restrictions and limitations apply to the EMPTYIDX call processing application:

- This option is only applicable for subscriber number collectable processing. This option is ignored in call type collectable processing.
- If the subscriber number type used does not have any entries provisioned against it in table FLEXVAL, then empty index screening also applies to the subscriber number validation attempt.
- Although validation is considered successful when allowing empty index screening, there is no FLEXFEAT table index retrieved and therefore no FLEXFEAT features and characteristics are processed for the subscriber number received.

FLEXLOG call processing application

The Generate Flex 302 Log (FLEXLOG) option indicates that for validation failures occurring for the subscriber number type, a FLEX 302 validation failure log is to be generated.

This field has no application for call type collectable processing and is subsequently ignored.

Call processing actions

When the FLEXLOG option is provisioned for the subscriber number type in table FLEXTYPE, then on validation failures, a FLEX 302 validation failure log is generated by the subscriber number collectable. If the FLEXLOG option is not provisioned, then a FLEX 302 log is not generated.

Interactions

For subscriber number collectable interactions, see “Subscriber number collectable (SUBR/SUBRPARM)” on page 16-104.

Restrictions and limitations

The following restrictions and limitations apply to the FLEXLOG call processing application:

- This option is only applicable for subscriber number collectable processing. This option is ignored in call type collectable processing.
- All causes for failure of the validation attempt generate a FLEX 302 log. There are no exceptions.

CAINFLG call processing application

CAIN can only interpret the syntax and semantics of ANIs, authcodes, account codes, MCCS/TNS, and PIN digits as found on non-AXXESS agents. To interact with the FlexDial framework, these subscriber number types need to be mapped from FlexDial abstract entities into entities which CAIN understands and can use in its decision-making. See *UCS DMS-250 CAIN/FlexDial Interactions* or ”FlexDial Interactions” appendix for more information.

The CAINFLG option indicates to CAIN how a subscriber number maps into the digit types understood by CAIN. The CAINFLG option appends a parameter that maps the FlexDial subscriber number to the CAIN digit type. The allowed subscriber number types and their interpretation are shown in Table 5-2.

Through table FLEXTYPE, FlexDial provides several options which give specific meaning to subscriber number types defined in that table. For example:

- The BILLFLD option indicates to the SUBR collectable which CDR field should be used to place the collected information.
- The CALLING option indicates to the SUBR collectable that this subscriber number represents the calling party number for call processing, and should be treated as the ANI for the call.

A similar approach is used to indicate to CAIN how a subscriber number maps into the subscriber numbers understood by CAIN. The CAINFLG option appends a parameter that maps the CAIN subscriber number type to the FlexDial subscriber number. The allowed subscriber number types and their interpretation are shown in Table 17-1.

In all cases, the parameters are used for outgoing SSP to SCP message parameter population, including the identification of the digits to send as well as the corresponding CAIN natures of address (vs. NOAs as used by FlexDial) for those digits.

Table 17-1
Table FLEXTYPE CAINFLG option interpretations

Subscriber number type	Interpretation
ACCT	Indicates account code interpretation.
AUTH	Indicates authorization code interpretation.
CLGPTYADD	Indicates interpretation as a calling_party_address parameter of an incoming ISUP IAM message.
MCCS	Indicates MCCS/TCN digits interpretation.
PIN	Indicates PIN digits interpretation.
—end—	

Note: The AUTH and MCCS values are limited to the identification of the nature of address for the ChargeNumber parameter. The digits for the ChargeNumber parameter are identified by the FLEXTYPE BILLFLD option when the billing field is identified as BILLNUM. Presumably, the FLEXTYPE tuple would have both the CAINFLG AUTH and BILLFLD BILLNUM options provisioned.

REVALIDATE call processing application

The Revalidate Call Processing (REVALIDATE) option identifies that received subscriber number digits of this type are revalidated on reoriginated calls.

This field has no application for call type collectable processing and use by a CALLTYPE collectable causes a FLEX 301 trouble log to be generated.

Call processing actions

Use of the REVALIDATE option marks the subscriber number collectable as requiring a revalidation attempt on reoriginated calls.

Note: Reorigination capability is provisioned through the FLEXFEAT table REORGPRC and REORGTYP options, as well as the TRKFEAT table REORIGAL option.

If the call reoriginates, the subscriber number is revalidated according to the VALIDATE option provisioned against the SUBR or SUBRPARM collectable. Any identified table FLEXTYPE and FLEXFEAT features and characteristics are not re-applied to the call, nor is the subscriber number re-captured in the CDR as identified by the FLEXTYPE table BILLFLD option.

The REVALIDATE FLEXTYPE table option works in conjunction with the REORIGAL TRKFEAT table option and the REVALIDATE FLEXFEAT table option to identify the sum of all received subscriber numbers identified by subscriber number type (FLEXTYPE) that need to be revalidated on reoriginated calls.

Initially through the TRKFEAT REORIGAL option, the REVERIFY field identifies a list of subscriber number types that are to be marked for revalidation. If a subscriber number of this type with a VALIDATE option specifying inswitch FLEXVAL table validation (match revalidation would always be successful, and therefore not performed) is processed for the call, then the subscriber number collectable is marked for the revalidation attempt on a reoriginated call.

Note: If the subscriber number collectable does not contain a VALIDATE option with inswitch FLEXVAL table validation, then revalidation of the subscriber number cannot occur.

Through the FLEXTYPE table, all subscriber numbers of a specific subscriber number type are marked for the revalidation attempt. With the FLEXFEAT table, individual subscriber numbers received can be marked for the revalidation attempt.

Interactions

The following interactions apply for the REVALIDATE call processing application:

- For interactions with reorigination, see “REORGACT call processing application” on page 18-29 and “REORGTYP call processing application” on page 18-34.
- This option works in conjunction with the TRKFEAT REORIGAL option (see “REORIGAL call processing application” on page 20-13) and the FLEXFEAT REVALIDATE option to identify all subscriber numbers that need to be revalidated on reoriginated calls.
- For subscriber number validation scenarios and processing, see “Subscriber number collectable (SUBR/SUBRPARM)” on page 16-104.

Restrictions and limitations

The REVALIDATE option has the following restrictions and limitations:

- This option is only applicable for subscriber number collectable processing. If used by a CALLTYPE collectable, this option causes a FLEX 301 trouble log to be generated.

- For CAIN/FlexDial interactions, if any subscriber numbers (ANI, authcodes, etc.) fail validation on a given execution of FLEXDIAL, all re-validation of subscriber numbers is turned off on all re-originations from that point forward for that AXXESS origination. See *UCS DMS-250 CAIN/FlexDial Interactions* for more information.

FLEXFEAT features and characteristics applications

Table FLEXFEAT provides the feature and characteristic specifications for defined subscriber number types and call types within the FlexDial framework. Through the subscriber number collectables (SUBR, SUBRPARM) and the call type collectable (CALLTYPE), entries in the FLEXFEAT table are processed for features and characteristics that apply to the call.

Supported agents

FLEXFEAT table processing occurs only during collectable processing within the FlexDial framework. The execution of collectables in the framework is only applicable when using the AXXESS trunk group type originating agent.

Activation

Processing of FLEXFEAT table entries occurs during the “collect information” point in the call model. Entries in table FLEXFEAT are processed by call processing through two methods:

- execution of subscriber number digit collectables (SUBR, SUBRPARM)
- execution of CALLTYPE collectables (CALLTYPE)

Features and characteristics provisioned in table FLEXFEAT differ from those provisioned in table FLEXTYPE in that the options provisioned in table FLEXFEAT are on a per-subscriber number basis, and not on a subscriber number type basis.

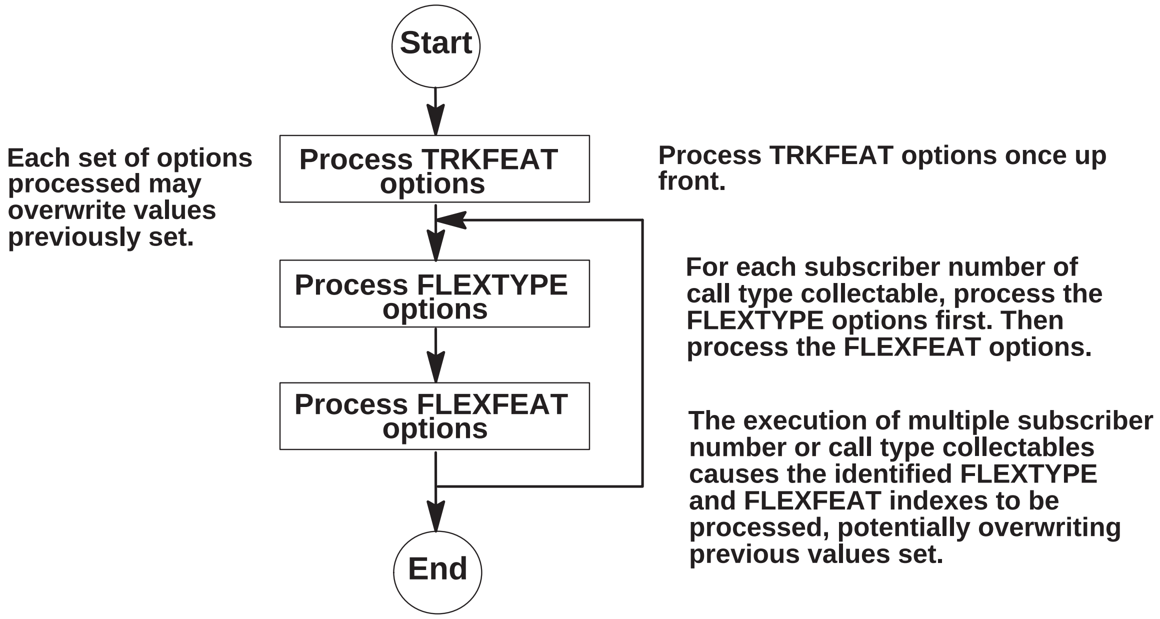
Some features and characteristics have representation in two or more of the following tables:

- FLEXTYPE
- FLEXFEAT
- TRKFEAT

If an option is provisioned in multiple tables, call processing uses the values set by the last option processed.

Figure 18-1 shows features and characteristics run time processing.

Figure 18-1
Features and characteristics run time processing



Each set of options processed may overwrite values previously set.

Process TRKFEAT options once up front.

For each subscriber number of call type collectable, process the FLEXTYPE options first. Then process the FLEXFEAT options.

The execution of multiple subscriber number or call type collectables causes the identified FLEXTYPE and FLEXFEAT indexes to be processed, potentially overwriting previous values set.

For the subscriber number digit collectables, processing of FLEXFEAT table entries occurs during the “apply collectable features” portion of collectable processing.

For both the subscriber number collectables (SUBR, SUBRPARM) and the call type collectable (CALLTYPE), table FLEXFEAT options are processed after table FLEXTYPE options.

Call processing description

The FLEXFEAT table provides the following features and capabilities for call processing purposes:

- ANSCDR
- BCCOMPAT
- CAINGRP
- CASUAL
- CDRTMPLT
- CITYCODE
- CITYVAL
- CLDPBILL
- CPACTVAL
- DELIVER
- DPIDX
- FAILVAL
- FEATVAR
- FLDONLY
- GENLOG
- IEXCLINX
- MLTCOSID
- MSGCTR
- NOANSDUR
- ONNET
- PVSPDIDX
- REORGACT
- REORGTYP
- REVALIDATE
- SPLASHBK
- TCAPANNC
- TRANSNUM
- TRANSTS
- TRANSYS

ANSCDR call processing application

The FLEXFEAT table ANSCDR option application is identical to the description of the ANSCDR option for table FLEXTYPE, except that the option is executed on a subscriber number basis and not on a subscriber number type basis. See “ANSCDR call processing application” on page 17-3.

BCCOMPAT call processing application

The Bearer Capability Compatibility (BCCOMPAT) option identifies the bearer capability associated with the originating agent as identified by the received subscriber number or call type use. This bearer compatibility value is assigned as the bearer compatibility for the call.

Call processing actions

Processing the BCCOMPAT option during subscriber number or call type collectables execution identifies the bearer capability value for the call.

The BCCOMPAT option can be processed multiple times between the authorize origination and collect information points in the call model. Each subsequent handling of the BCCOMPAT option overwrites the previous value set for the call. The BCCOMPAT option may be defaulted through use of the CLRFTRS collectable.

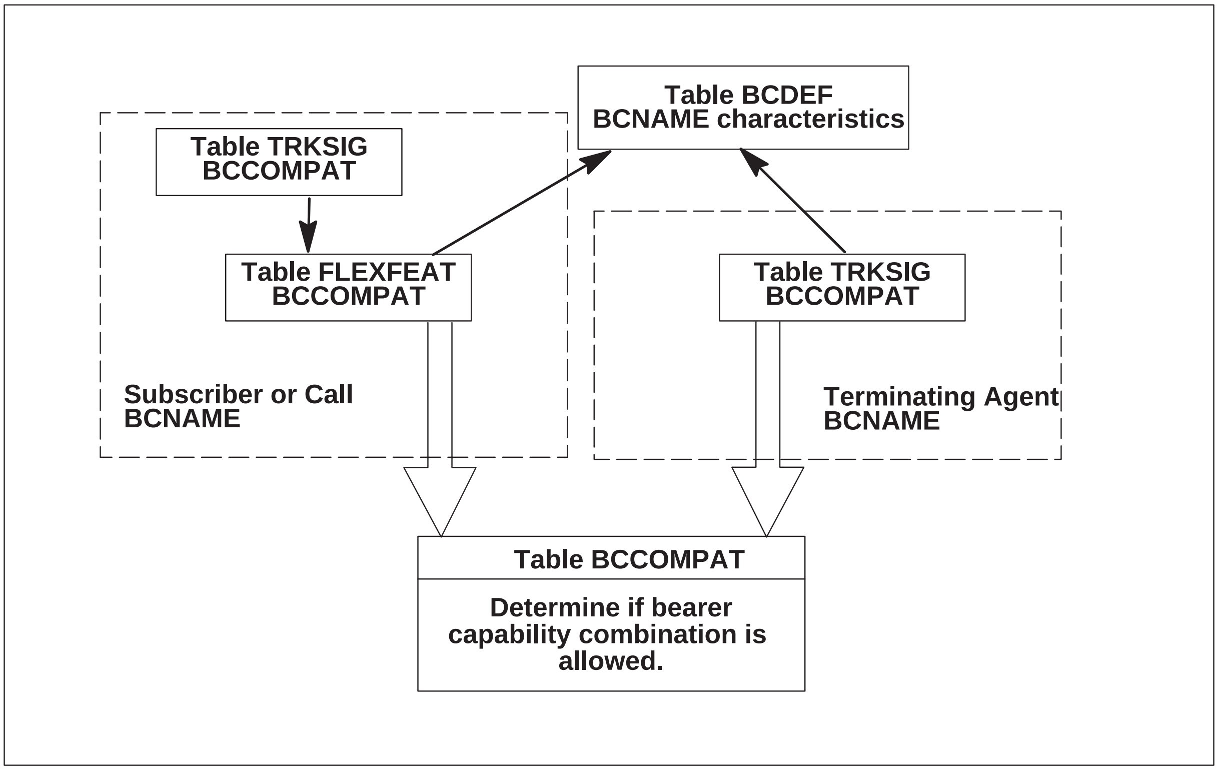
In addition to table FLEXFEAT, the BCCOMPAT option can be provisioned in table TRKSIG against the originating agent. The order in which the tables are processed identifies the precedence for setting the bearer compatibility for the call:

- The TRKSIG table BCCOMPAT option is processed for the originating agent before any collectables are executed. The TRKSIG options are processed once during the life of the call. (The information the TRKSIG options contain is stored in per-call data structures, and can trigger the execution of other features at a later point in the call.)
- During execution of a subscriber number or call type collectable, the FLEXFEAT table BCCOMPAT option is processed, overwriting the bearer compatibility value previously set.

During the “authorize termination” point in the call, the bearer compatibility for the call and the bearer compatibility for the terminating agent are used to index the BCCOMPAT and BCDEF tables to determine if the call is to be blocked due to incompatible bearer channel use.

Figure 18-2 shows bearer compatibility screening.

Figure 18-2
Bearer compatibility screening



If bearer capability screening through table BCCOMPAT is successful, the call continues to allocate a terminating agent member. If bearer capability screening is unsuccessful, a route advance occurs and an attempt is performed to terminate the call to a member of another terminating trunk group.

Interactions

See the *UCS DMS-250 Data Schema Reference Manual* for a description of tables BCCOMPAT and BCDEF.

Restrictions and limitations

The BCCOMPAT option value used for the call is set to the last option processed in FlexDial framework collectable processing. If the BCCOMPAT option is never processed, a default value of 3_1KHz is used.

CAINGRP call processing application

The Carrier Ain Group (CAINGRP) option specifies the CAINGRP table index used to identify AIN service execution for the call. The CAINGRP option is applicable for both subscriber number and call type collectable processing.

Call processing actions

Processing the carrier AIN (CAIN) group option identifies an index into table CAINGRP that identifies AIN call processing triggers for the subscriber or call.

The CAINGRP option can be processed multiple times during FlexDial framework collectable processing. Each subsequent handling of the CAINGRP option overwrites the previous value set for the call. CAIN group processing may be disabled through use of the CLRFTRS collectable.

In addition to table FLEXFEAT, the CAINGRP option can be provisioned in table TRKFEAT against the originating agent. The order in which the tables are processed identifies the precedence for setting the CAINGRP table index used for the call.

- The TRKFEAT table CAINGRP option is processed for the originating agent before any collectables are executed. The TRKFEAT options are processed once during the life of the call. (The information the TRKFEAT options contain is stored in per-call data structures, and can trigger the execution of other features at a later point in the call.)
- During execution of a subscriber number or call type collectable, the FLEXFEAT table CAINGRP option is processed, overwriting the CAINGRP table index previously set.

For more information on CAIN triggering and processing, see the *UCS DMS-250 NetworkBuilder Application Guide*. For more information about FlexDial and CAIN interactions, see *UCS DMS-250 CAIN/FlexDial Interactions* or Appendix D, ”FlexDial Interactions.”

Restrictions and limitations

The CAINGRP table index set through subscriber number or collectable execution is not processed until the collect information point in the call is exited (that is, collectable processing is complete or suspended).

CASUAL call processing application

The Casual Subscriber Number (CASUAL) option identifies the received subscriber number as a casual number. A casual number is identified as a number that can be used by multiple normally independently billable subscribers on a network.

The CASUAL option is not applicable for call type collectable processing, and is subsequently ignored.

Call processing actions

A processed CASUAL option immediately identifies the received subscriber number as a casual subscriber number. The casual indicator only applies to the subscriber number digits received by the SUBR or SUBRPARM collectable being processed.

Casual subscriber numbers may or may not be blocked in subscriber number collectable processing, depending on whether or not the CASUBLK option is provisioned for the collectable. For a description of casual subscriber number blocking, see “Casual number blocking” on page 16-116.

Processing the CASUAL option is not applicable for CALLTYPE collectable execution, and the option is ignored when processed by a CALLTYPE collectable.

Interactions

For interactions with casual number blocking validation, see “Casual number blocking” on page 16-116.

Restrictions and limitations

The following restrictions and limitations apply to the CASUAL call processing application:

- The CASUAL option is only applicable for the received subscriber number digits being processed by the subscriber number collectable, and is not applicable for other subscriber numbers or for the call in general.
- The CLRFTRS collectable has no application for the CASUAL option, as there is no specific call information to reset or default.

CDRTMPLT call processing application

The FLEXFEAT table CDRTMPLT option application is identical to the description of the CDRTMPLT option for table FLEXTYPE, except that the option is executed on a subscriber number basis and not on a subscriber number type basis. See “CDRTMPLT call processing application” on page 17-8.

CITYCODE call processing application

The Citycode (CITYCODE) option is used in conjunction with the subscriber collectable CITYVAL option and the TRKFEAT CITYCODE option to perform citycode screening on the received subscriber number. The city code value is also used to index table CITYCODE to identify the index for public speed dial numbers.

City code screening is not applicable through CALLTYPE collectables.

Call processing actions

A processed CITYCODE option immediately identifies the citycode value (digits) for the subscriber or call after potential citycode validation has been completed.

In addition to being processed by subscriber number collectables, the CITYCODE option can be processed by call type collectables. Each subsequent processing of the CITYCODE option overwrites the previous value stored for the subscriber or call. The citycode value for the call may be cleared through use of the CLRFTRS collectable.

For citycode validation, the FLEXFEAT CITYCODE option is used in conjunction with the current per-call citycode value and the CITYVAL option to perform citycode validation. For a description of citycode validation, see “CITYCODE validation” on page 16-112.

Citycode validation is not applicable for CALLTYPE collectable processing.

The citycode value set for the subscriber or call may also be used to index table CITYCODE to retrieve a speed dial table index for public speed dial number processing.

Interactions

The CITYCODE option is used for subscriber number citycode validation. See “CITYCODE validation” on page 16-112.

Restrictions and limitations

The following restrictions and limitations apply to the CITYCODE call processing application:

- The CITYCODE option is used in conjunction with the TRKFEAT table CITYCODE option and the subscriber number collectable CITYVAL option for subscriber number citycode validation.
- Use of the CITYCODE option for CALLTYPE collectable is not applicable for citycode validation. Citycode validation is only performed by a subscriber number collectable.

CITYVAL call processing application

The Citycode Validation (CITYVAL) option identifies on a per-subscriber number basis that citycode validation is to occur for the subscriber number. Citycode validation compares the CITYCODE of the subscriber number to the CITYCODE value set for the call. If the CITYCODE values are identical, then validation is successful. Otherwise the validation attempt fails.

Citycode validation can only occur for the call if a citycode value is identified for the subscriber number, and a citycode value has been identified for the call.

If both CITYCODE values are not available, then citycode validation does not occur and the validation attempt is successful by default.

Citycode validation may alternatively be triggered through use of the table FLEXDIAL SUBR or SUBRPARM collectable CITYVAL option (see “SUBR CITYVAL option” on page 2-85).

For call type collectables, the application of city code validation does not apply.

Call processing actions

The CITYVAL option is applied during validation of a subscriber number by a SUBR or SUBRPARM collectable.

For citycode validation, the FLEXFEAT CITYCODE option for the subscriber number being processed is used in conjunction with the per-call CITYCODE value set and the CITYVAL option to perform citycode validation. For a description of citycode validation, see “CITYCODE validation” on page 16-112.

Citycode validation is not applicable for CALLTYPE collectable processing.

Interactions

The CITYVAL option is used for subscriber number citycode validation. See "CITYCODE validation" on page 16-112 for a description of citycode validation.

The FLEXFEAT CITYVAL option is one of two triggering mechanisms for citycode validation. The second mechanism is the CITYVAL SUBR collectable option. See "SUBR CITYVAL option" on page 2-85 for more information.

When the FLEXFEAT CITYVAL option is used to trigger citycode validation, the validation is occurring on a per-subscriber number basis. When the SUBR collectable CITYVAL option is used, then citycode validation is occurring on a per-trunk group or per-service basis.

Restrictions and limitations

The following restrictions and limitations apply to the CITYVAL call processing application:

- The citycode value for the subscriber number and the call must be available to perform citycode validation. If neither value is available, then the citycode validation attempt is aborted and the result is successful by default.
- Use of the CITYVAL option for CALLTYPE collectable is not applicable for citycode validation. Citycode validation is only performed by a subscriber number collectable.

CLDPBILL call processing application

The Called Party Billed (CLDPBILL) option identifies the call as a called party billed call instead of a calling party billed call. This option is applicable for both subscriber number and call type collectable processing.

Call processing actions

Processing the called party billed option immediately identifies the call as called party billed versus being calling party billed.

The CLDPBILL option may be processed multiple times during FlexDial framework collectable processing. Each subsequent handling of the CLDPBILL option reiterates that the call is a called party billed call. The call may be reset to calling party billed through use of the CLRFTRS collectable.

If a call is identified as a called party billed call, then the result of subscriber number validation is ignored, in particular, subscriber number validation failures. The application of handling the called party billed option for subscriber number processing involves three areas:

- For previously executed subscriber number collectables, if a delayed treatment was set as a result of the subscriber number validation failure, then the treatment is cleared for the call. Other FAILACT processing due to the validation failure can not be undone.
- For the currently executing subscriber number collectable, the validation attempt is considered successful. In this scenario, the VALIDATE option FAILACT action is not executed and the collectable continues with the application of subscriber number features.

  Note: Only a FLEXVAL table validation attempt is considered successful by default. The CLDPBILL option cannot be processed unless FLEXVAL table validation is used.

- For subscriber number collectables yet to be executed, any validation attempt for the collectable returns a successful result, and the VALIDATE option FAILACT action is not executed for the collectable.

For a called party billed call identified by the CLDPBILL option, the called party digits are captured in the BILLNUM field of the CDR. This occurs through one of two methods:

- If a called party address has already been processed for the call, then the digits captured in the CALLEDNO field of the CDR are copied into the BILLNUM field, overwriting any information previously captured in the BILLNUM field.
- If a called party address had not been processed for the call, then upon completion of ADDR or ADDRPARM execution, the received called party address digits (captured in the CALLEDNO field) are also captured in the BILLNUM CDR field, overwriting any information previously captured in the BILLNUM field.

Interactions

A called party billed call may also be identified through N00 identified address digits. For the N00 application, the N00 number processed is captured in the BILLNUM field of the CDR.

Restrictions and limitations

Use of the CLRFTRS option to reset the call to calling party billed does not cause subscriber number validation failures to be reapplied to the call. Resetting the call to calling party billed only affects subscriber number collectables executed after the CLRFTRS collectable.

CPACTVAL call processing application

Use of the CPACTVAL option causes call processing active validation to occur for the subscriber number or calltype being processed. For subscriber numbers, CPACTVAL validation can only occur in conjunction with FLEXVAL validation, and occurs after all other types of validation have been performed (absence of FAILVAL, FLDONLY validation, CITYCODE validation, and casual number blocking validation).

For calltype use, CPACTVAL validation can only occur if the CALLTYPE collectable is provisioned with a valid and non-NIL FLEXTYPE table index. Otherwise the CPACTVAL option is ignored for CALLTYPE processing. For calltype processing, the CPACTVAL option is processed following processing of a potential FAILVAL option (allowing a possible CPACTVAL set treatment to override a FAILVAL set treatment). If the CPACTVAL option is provisioned with a FAILACT value of “FAILACT”, and call processing active validation is unsuccessful for the calltype, then no treatment is set and a FLEX 301 log is generated with the following trouble reason:

```
Cannot apply CPACTVAL FAILACT option.
```

Upon successful CPACTVAL validation, the CALLPS field counter within the CPACTVAL provisioned option is immediately incremented, identifying to a craftsperson that a call is in progress using the particular subscriber number or calltype. The CALLPS counter is not decremented until call completion occurs, deactivating the subscriber number or calltype for that call.

Unsuccessful CPACTVAL validation does not modify the CALLPS field counter within the CPACTVAL provisioned option.

If the FLEXTYPE table FLEXLOG option is provisioned for the flextype being processed, then upon unsuccessful CPACTVAL validation attempts, a FLEX 302 log is generated with the following reason:

```
Active CALLP Limit Enforced for Subscriber Number.
```

For all call processing active validation attempts, the LVALTOT FLEXTYPE OM register is incremented for the particular FLEXTYPE being processed. For successful CPACTVAL validation attempts, the LVALSUCC register is also incremented. For unsuccessful CPACTVAL validation attempts, the LVALFAIL register is incremented.

Up to three (3) independent active validations may be active for a particular call in progress. If a fourth successful CPACTVAL is achieved for the call, then the first CPACTVAL activated for the call is deactivated (i.e. the CALLPS is decremented), and the fourth successful CPACTVAL remains active. Only the last three successful CPACTVAL occurrences are maintained for the call.

The use of CPACTVAL in a CLRFTRS collectable results in the deactivation of all currently active subscriber numbers or calltypes for the call.

Usecase Scenarios

The CPACTVAL option is used to limit subscriber number fraud by restricting the number of calls that may be active at any one time for the particular subscriber number.

Interactions

For subscriber number validations, CPACTVAL validation occurs after all other types of FLEXVAL / FLEXFEAT validation have been performed.

On system restart COLD and RELOAD, all CPACTVAL CALLPS counters are reset to zero.

Call Processing Restrictions / Limitations

No call processing restrictions and limitations exist for CPACTVAL option use.

DELIVER call processing application

The Calling Party Number Deliver (DELIVER) option specifies the calling party subscriber number (as identified by the FLEXTYPE CALLING option) delivery status for the call, and identifies when or what type of calling party subscriber number is to be delivered (or outpulsed) to the next network switch.

Note: The effects of the DELIVER option may be nullified through use of table RTEATTR. See the *UCS DMS-250 Data Schema Reference Manual.*

Call processing actions

Processing the DELIVER option identifies the calling party number delivery method for delivering the calling party number to the terminating switch. The deliver method can be one of four values:

- ALWAYS—Always deliver the calling party number as it was received.
- CPNONLY—Deliver a received calling party number in the calling party number parameter.

- CGNONLY—Deliver a received calling party number in the charge number parameter.
- NEVER—Never deliver a received calling party number.

Use of the DELIVER option is controlled through software optionality control (SOC) NSER0001. See the *UCS DMS-250 Software Optionality Control (SOC) User's Manual*. This SOC must be in an ON state to use the DELIVER option.

The DELIVER option can be processed multiple times during FlexDial framework collectable processing. The processing of each subsequent DELIVER option overwrites the value previously stored for the call. The calling party number deliver method is reset or defaulted to the value of ALWAYS through use of the CLRFTRS collectable.

In addition to the FLEXFEAT table, the DELIVER option may also be provisioned in table TRKSIG. The order in which the tables are processed identifies the precedence for setting the DELIVER feature information.

- During execution of a subscriber number or call type collectable, the FLEXFEAT table DELIVER option is processed, overwriting any feature value previously set.
- The TRKSIG table DELIVER option is provisioned for the terminating trunk group, and is processed when the terminating agent member is allocated for the call. The TRKSIG options for the terminating trunk group are only processed once during the life of a call.

  Note: Note that the information the TRKSIG options contain is stored in per-call data structures, and can trigger the execution of other features at a later point in the call.

Once set, the DELIVER feature information is retained in call processing data structures until the call is presented to the next network switch at the "present call" point in the call model. The method of delivering a received calling party number is applied when the call is presented to the next network switch.

The DELIVER option is only applicable for subscriber number types identified as the calling party subscriber number. A calling party subscriber number is identified through use of the FLEXTYPE table CALLING option for the subscriber number type specified (see "CALLING option" on page 5-8).

Interactions

The DELIVER option interacts with the following features and capabilities:

- ANI delivery enhancements
- route-based outgoing parameters modification table

For more information on these features, see the *UCS DMS-250 Feature Group D (FGD) Application Guide*.

Restrictions and limitations

The CLRFTRS collectable resets the calling party number deliver method to a value of ALWAYS.

DPIDX call processing application

The FLEXDIAL Table Index (DPIDX) option identifies through the FLEXDIAL table index a list of collectables that must be processed for the call, modifying the interaction with the originating agent due to use of a specific subscriber number or call type.

The DPIDX option is applicable for both subscriber number and CALLTYPE collectable processing.

Call processing actions

Processing a DPIDX option identifies a FLEXDIAL table index that identifies an alteration in the interaction with the originating agent. The list of collectables identified by the FLEXDIAL table index may either:

- be inserted into the current list of collectables being executed
- be appended to the current list of collectables being executed
- replace the current list of collectables being executed.
- be executed as a subroutine

See “Collectable list management” on page 15-9 for more specific details about the INSERT, APPEND, REPLACE, and EXEC collection list management methods.

The DPIDX option can be processed multiple times during FlexDial framework collectable processing. The processing of each subsequent DPIDX option modifies the collectable list being processed for the call. When used with the CLRFTRS collectable, the DPIDX option specifies that all history of collectable execution up to the processing of the CLRFTRS collectable is erased. (This action doesn’t affect the forward progress of collectable execution, but does affect the ability to reset [RESET action on digit collectables] to previously executed collectables.)

Interactions

The DPIDX option modifies the list of collectables being executed for the call. See “Collectable list management” on page 15-9 for a description on how the collectable list modifications occur.

Restrictions and limitations

The DPIDX option only modifies the current list of collectables being executed.

FAILVAL call processing application

The Fail Validation (FAILVAL) option identifies that use of the subscriber number causes a failure result to be returned on the subscriber number validation attempt. Through normal processing, retrieval of the FAILVAL option indicates that the validation attempt of the subscriber number digits was successful. However, the presence of the FAILVAL option overrules the successful result to return an unsuccessful result.

This option can be used by the CALLTYPE collectable to set a delayed treatment for the call.

Call processing actions

Processing the FAILVAL option identifies the failure action to apply for subscriber number or call type collectable. This action is immediately applied for subscriber number or CALLTYPE collectable processing. Two types of actions are supported:

- TRMT—Set the identified treatment for the call. The treatment is applied as a delay treatment.
- FAILACT—Apply the failure action provisioned against the VALIDATE option for the subscriber number collectable.

The FAILVAL option action of TRMT overrides the FAILACT action provisioned in the VALIDATE option of the subscriber number collectable.

Use of the FAILVAL option specifies that the FLEXVAL table validation attempt of the subscriber number is unsuccessful. Without the FAILVAL option, the validation attempt would be successful, as the FLEXVAL table was successfully indexed to retrieve the FLEXFEAT index being processed.

The FAILVAL option is processed during validation of the subscriber number and thus before any other FLEXFEAT table options. When the FAILVAL option is provisioned in a FLEXFEAT table entry retrieved in the validation attempt, then validation failure occurs and the apply features portion of subscriber number collectable processing is not reached.

Therefore none of the other FLEXFEAT options provisioned for the subscriber number are applied to the call.

Only the FAILVAL action of TRMT is supported for CALLTYPE collectable processing. The action of FAILACT is not supported (FAILACT cannot be supported because the CALLTYPE collectable doesn’t support a VALIDATE option with a FAILACT field), and use of this action by a CALLTYPE collectable results in the generation of a FLEX 301 trouble log. No call processing action is performed during FAILVAL option processing for the CALLTYPE collectable with a FAILVAL action of FAILACT. For CALLTYPE collectable processing, remaining FLEXFEAT options continue to be processed.

When used in the CLRFTRS collectable, the FAILACT option causes the treatment set for the call to be cleared if the delayed treatment was set through subscriber number or CALLTYPE collectable processing.

Interactions

For interactions with subscriber number FLEXVAL validation attempts, see “FLEXVAL validation” on page 16-107.

When validation failure occurs, a FLEX 302 validation failure log can be generated if the FLEXLOG option is provisioned for the subscriber number type. See section “FLEXLOG call processing application” on page 17-11 for more information on FLEXLOG processing.

Restrictions and limitations

The following restrictions and limitations apply to the FAILVAL call processing application:

- The FAILVAL action of FAILACT is not supported for CALLTYPE collectable processing. The use of the FAILACT action for CALLTYPE collectable processing results in the generation of a FLEX 301 trouble log. No call processing action is performed for FAILVAL option processing for the CALLTYPE collectable.
- When the FAILVAL option is processed for a subscriber number or CALLTYPE collectable, no other FLEXFEAT options provisioned against the table entry are processed for the call.
- Use of the FAILVAL option in the CLRFTRS collectable only clears a treatment set for the call if the delayed treatment was set through subscriber number or CALLTYPE collectable processing.

FEATVAR call processing application

Use of the FEATVAR option enables this feedback mechanism for FlexDial call processing purposes. The specific meaning of the value of the FEATVAR variable is dictated by its use, enabling it to be used for whatever purpose the service provider desires. The FEATVAR option is linked to the FLEXTYPE of the subscriber number or calltype against which it is provisioned.

Processing of the FEATVAR option activates it for call processing use, and immediately increments the CALLPS counter for the FEATVAR option in the provisioned tuple. The FEATVAR option remains active for the complete duration of the call. Upon call completion, the FEATVAR option is deactivated by decrementing the CALLPS counter within the provisioned FEATVAR option.

For calltype use, FEATVAR activation can only occur if the CALLTYPE collectable is provisioned with a valid and non–NIL FLEXTYPE table index. Otherwise the FEATVAR option is ignored for CALLTYPE processing.

The maximum value for the CALLPS field counter of 255 represents the maximum limit of the number of calls which may activate a singular FEATVAR option. An attempt by call processing to activate a 256th reference to the FEATVAR option results in the generation of a FLEX 301 log with a trouble reason of “FEATVAR Activation Limit Exceeded”, and the FEATVAR option is not activated for the call.

The value of the FEATVAR option may be modified by table control at any time, including when the CALLPS value is greater than zero, indicating active call processing use is also occurring for the option.

Up to four (4) FEATVAR variables may be active at any one time for call processing purposes. Interactive provisioning and call processing identifies a specific FEATVAR option by the associated FLEXTYPE of the subscriber number or calltype that activated the FEATVAR option. Multiple FEATVAR options processed by the same FLEXTYPE override the previous FEATVAR use for that FLEXTYPE, decrementing the CALLPS counter for the previous FEATVAR as it is deactivated. If more than four unique FLEXTYPE FEATVAR options are processed for the call, then only the latest four enabled FEATVAR options remain active and the earlier activated FEATVAR options are deactivated.

The use of FEATVAR in a CLRFTRS collectable results in the deactivation of all currently active subscriber numbers or calltypes for the call.

Usecase Scenarios

The FEATVAR option provides a generic mechanism whereby call processing may feedback information to the provisioned database by updating the FEATVAR value.

Interactions

The FEATVAR option contains linked usage through the VAROP, IFVAR, and TIMER collectables, as well as the VAROP VALUE MSGCTR message. The FEATVAR value may be updated by call processing by using FEATVAR as the assigned to variable within the VAROP collectable. When using the FEATVAR selection within these collectables, an additional field identifying the associated FLEXTYPE must also be input.

Call Processing Restrictions / Limitations

The following processing restrictions and limitations exist for FEATVAR option use:

- A maximum of 255 calls may activate a singular provisioned FEATVAR option. An attempt to exceed this limit results in the generation of a FLEX 301 log with non–activation of the desired FEATVAR option for the call.
- A FLEXTYPE referenced FEATVAR option must be activated prior to use by FlexDial call processing. Attempts by call processing to use an unactivated FEATVAR reference result in the application of “feature not allowed” (FNAL) treatment and generation of a FLEX 307 log.

FLDONLY call processing application

The Filed Digits Only (FLDONLY) option specifies that all digits processed by the subscriber number collectable must be filed (provisioned) in the switch and not received from the originating agent.

The filed digits only option is only applicable for subscriber number collectable processing, and is not applicable for call type collectable processing.

Call processing actions

Processing the FLDONLY option identifies that all of the processed subscriber number digits for the SUBR or SUBRPARM digit collectable must be filed digits. This action is immediately applied for subscriber number collectable processing. Digits may be filed for the collectable through two methods:

- through use of the SUBR/SUBRPARM FILED option in table FLEXDIAL provisioning

- through use of the SUBR FILED message in MSGCTR table provisioning

See “Application of filed digits” on page 16-60 for a description of the application of filed digits for digit collectable processing.

If the filed digits only option is provisioned for a subscriber number and not all of the subscriber number digits consumed by the digit collectable are filed digits, then the validation attempt for the subscriber number is unsuccessful. Generally, the validation attempt of the subscriber number would be successful, as the FLEXVAL table was successfully indexed to retrieve the FLEXFEAT index being processed. However, since all of the processed digits were not filed and the FLDONLY option was present, the validation attempt is unsuccessful.

The FLDONLY option is processed during validation of the subscriber number and hence before any other FLEXFEAT table options (with the exception of FAILVAL that is also processed during validation). When the FLDONLY option is provisioned in a FLEXFEAT table entry retrieved in the validation attempt where not all the digits processed were filed digits, then validation failure occurs and the apply features portion of subscriber number collectable processing is not reached. Therefore none of the other FLEXFEAT options provisioned for the subscriber number are applied to the call.

Processing the FLDONLY option is not applicable for CALLTYPE collectable execution, and is subsequently ignored.

Interactions

For interactions with subscriber number FLEXVAL validation attempts, see “FLEXVAL validation” on page 16-107.

When validation failure occurs, a FLEX 302 validation failure log can be generated if the FLEXLOG option is provisioned for the subscriber number type. See “FLEXLOG call processing application” on page 17-11 for more information on FLEXLOG processing.

Restrictions and limitations

Use of the FLDONLY option in the CLRFTRS collectable does not perform any call processing function, and the option is essentially ignored.

GENLOG call processing application

The Generate Information Log (GENLOG) option indicates that a FLEX 601 information log is to be generated due to the use of the subscriber number or call type within call processing.

Call processing actions

Use of the GENLOG option causes a FLEX 601 information log to be immediately dispatched to the log system with the following report indicator:

```
FLEXTYPE Processed
```

The log identifies the subscriber number type or call type was processed for the call. For subscriber number processing, the received subscriber number digits are included in the log report. For CALLTYPE collectable processing, no digits are included in the report.

When the GENLOG option is used for a subscriber number collectable, and the collectable identifies that a SUBRNUMx billing record field (one of SUBRNUM1, SUBRNUM2, SUBRNUM3, or SUBRNUM4) is to be used to capture the subscriber number digits in the billing record, then the associated SUBRLOGx billing record field is captured with a value of 1 due to use of the GENLOG option.

Interactions

See “FLEX 601 Information Log” on page 22-10 for a description of the FLEX 601 information log. This option does not interact with the FLEXTYPE table FLEXLOG option.

Restrictions and limitations

The SUBRLOGx billing record field is only captured if the associated SUBRNUMx field is identified as the billing record field for the subscriber number type being processed. This field is not captured when the ANISP, ACCTCD, BILLNUM, CLGPTYNO, or PINDIGS billing record field is provisioned to store the subscriber number digits received.

IEXCLINX call processing application

Processing the IEXCLINX option identifies the incoming exclusion screening index to use for the call during called party address digit processing. The IEXCLINX option may be processed multiple times during the call, with each subsequent occurrence overwriting the value set by the previous occurrence. The nature and purpose of the FLEXFEAT IEXCLINX option is identical to the IEXCLINX table TRKFEAT option, and may overwrite the value set by the TRKFEAT option.

The IEXCLINX option must be set before processing of the called party address digits occurs for the call. Setting the incoming exclusion screening index for the call after call translations has occurred has no discernible effect on the call.

The use of IEXCLINX in a CLRFTRS collectable results in the incoming exclusion screening index being set to 0 for the call.

Usecase Scenarios

The IEXCLINX option is used for explicit subscriber number control over incoming exclusion screening.

Interactions

The IEXCLINX option identifies the incoming exclusion screening index for called party address digit processing which occurs through the ADDR and ADDRPARM collectables.

Restrictions and limitations

No call processing restrictions and limitations exist for IEXCLINX option use.

MLTCOSID call processing application

The Multiple Class Of Service Screening Index (MLTCOSID) option enables class of service (COS) screening functionality for called party address digits received by ADDR or ADDRPARM digit collectables. The MLTCOSID option provides an index into table MULTICOS that identifies one or more COSUS table indexes that are used to screen the call.

The multiple COS option is applicable for both subscriber number and call type collectable processing.

Call processing actions

Processing the COS screening index option identifies the MULTICOS table index to use for COS screening.

The COS screening option may be processed multiple times during FlexDial framework collectable processing. How each subsequent MLTCOSID option is handled depends on what stage in the call the option is being processed. A COS screening index not yet processed may be cleared (and screening disabled for the call) through use of the CLRFTRS collectable.

COS screening typically occurs during ADDR or ADDRPARM collectable processing, and only occurs for calling party billed calls where the address digits processed are validated through table STDPRTCT. The value provided by the MLTCOSID option indexes table MULTICOS, which provides one or more indexes to table COSUS. Each table COSUS index processed provides parameters for COS screening of the received address digits.

If COS screening is successful, then processing of the address collectable is completed under normal conditions and processing of FlexDial framework collectables continues.

If COS screening is unsuccessful, then one of two actions occurs for the call:

- If the COSOVE option is provisioned against the originating agent in table TRKFEAT, then the FLEXDIAL table index specified by the COSOVE option is processed and collectable list identified replaces the remainder of the current collectable list.
- If the COSOVE option is not provisioned, then a delayed treatment is set for the call.

MLTCOSID option processing differs if the option is processed before or after ADDR or ADDRPARM collectable processing.

Before address collectable processing

When the MLTCOSID option is processed before an ADDR or ADDRPARM collectable is executed, then the MULTICOS table index is stored in call processing data structures. Each handling of the MLTCOSID option overwrites the previous value stored.

After address collectable processing

When the MLTCOSID option is processed after an ADDR or ADDRPARM collectable is executed, the MULTICOS index is stored in the call processing data structures, overwriting a previous value recorded.

Also, COS screening is immediately executed (or re-executed) using the new index identified (providing the call is a calling party billed call and the address received was validated through table STDPRTCT). If COS screening fails, then the appropriate action identified above is applied to the call.

Interactions

COS screening is only performed for calling party billed calls, and not for called party billed calls. See “CLDPBILL call processing application” on page 18-10 for information on called party billed calls.

Restrictions and limitations

The following restrictions and limitations apply to the MLTCOSID call processing application:

- COS screening is only performed for calling party billed calls, and not called party billed calls.
- COS screening is only performed for calls in which the processed address digits are validated in table STDPRTCT.
- If COS screening is executed through a subscriber number or CALLTYPE collectable (that is, an address collectable has already been executed), then regardless of the outcome of the screening attempt, the remaining subscriber number or calltype features are processed for the call.

MSGCTR call processing application

The Message Center (MSGCTR) option identifies messages provisioned in table MSGCTR that are to be posted for the call. These messages relate subscriber number or call type specific information that is to be used in processing other FlexDial framework collectables.

Call processing actions

Processing the MSGCTR option identifies an index into the MSGCTR table. This index identifies a list of messages that are posted for the call at the message center. The FLEXFEAT table may contain up to eight MSGCTR options provisioned, where each option identifies additional messages to be processed for the call.

The MSGCTR options are processed in the order that they are provisioned within table FLEXFEAT. The messages at each identified MSGCTR table index are posted in the order they are provisioned.

None of the newly posted messages are consumed by the currently processing collectable, even messages identified for the same subscriber number type used.

Interactions

For table MSGCTR specifications, see Chapter 4, “Table MSGCTR.”

Restrictions and limitations

No messages posted are consumed by the currently processing collectable. This includes messages identified for the same subscriber number type used.

NOANSDUR call processing application

The No Answer Duration (NOANSDUR) option enables the no answer duration timer that specifies the allowable amount of time between seizure and answer of the terminating agent. This option is applicable for both subscriber number and call type collectable processing.

Call processing actions

Processing the NOANSDUR option identifies the following information for call processing use:

- the timer value for the duration timer (in seconds)
- the action to take when the timer expires. The action may be one of four values: TRMT, RTEADV, RECOL, or DPIDX.

The NOANSDUR option can be processed multiple times throughout the call. Each subsequent handling of the NOANSDUR option overwrites the previous feature information stored for the call. The NOANSDUR feature can only be disabled through use of the CLRFTRS collectable.

In addition to the FLEXFEAT table, the NOANSDUR option may also be provisioned in table TRKFEAT. The order in which the tables are processed identifies the precedence for setting the NOANSDUR feature information.

- During execution of a subscriber number or call type collectable, the FLEXFEAT table NOANSDUR option is processed, overwriting any feature information previously set.
- The TRKFEAT table NOANSDUR option is provisioned as a terminating trunk group option, and is processed when the terminating agent member is allocated for the call. However, the TRKFEAT table NOANSDUR option cannot override the existing value if the original value was set as a result of N00 TCAP processing. The TRKFEAT options for the terminating trunk group are only processed once during the life of a call.

  Note: Note that the information the TRKFEAT options contain is stored in per-call data structures, and can trigger the execution of other features at a later point in the call.

Once set, the NOANSDUR feature information is retained in call processing data structures until the call is presented to the next network switch at the “present call” point in the call model. When the call is presented to the next network switch, the timer is started.

If the timer expires before answer occurs for the call, then the connection to the terminating agent is broken, the terminating agent is released from the call, and the action identified by the no answer duration action is executed.

If answer occurs before the timer expires, then the timer is cancelled and the call continues.

The actions supported for handling expiration of the no answer duration timer include:

- TRMT

  This action specify a treatment that is set and applied to the call.

- RTEADV

  This action indicates that a route advance should occur in order to attempt to present the call to a different terminating agent.

- RECOL

  This action indicates that the collect information point in the call should be re-executed to continue processing FlexDial framework collectables.

- DPIDX

  Through the FLEXDIAL table index specified, a collectable list is identified and the collect information point in the call model is re-executed to process this collectable list.

The actions RECOL and DPIDX only apply for AXXESS trunk group originations. For non-AXXESS originations, these actions are ignored and are replaced by setting no terminal responding (NTRS) treatment for the call.

Interactions

The timer value for the no answer duration timer may also be identified through N00 TCAP application feature byte processing and subsequently set through execution of the CALLTYPE collectable. See “CALLTYPE collectable” on page 16-153.

Restrictions and limitations

The actions RECOL and DPIDX only apply for AXXESS trunk group originations. When an attempt to use these actions on non-AXXESS originations occurs, the action is ignored and instead no terminal responding (NTRS) treatment is set for the call.

ONNET call processing application

The On-network (ONNET) option identifies the call as an on-network call, which affects handling of the called party address digits by the call process. This option is applicable for both subscriber number and call type collectable processing.

Call processing actions

Use of the ONNET option identifies that the call is an on-network call, and is identical to the use of the CT ONNET selector for address collectable processing.

Processing multiple ONNET options does not alter the on-network indicator for the call. The ONNET indicator can only be cleared through one of the following ways:

- through use of the CLRFTRS collectable that sets the call indicator to OFFNET
- through use of the CT OFFNET selector in table STDPRTCT when screening received address digits during address collectable processing

The on-network indicator is used in multiple areas of call processing:

- Incoming exclusion screening is not performed for ONNET calls.
- ONNET calls are differentiated from other direct dialed calls in table COSUS COS screening.
- Only six or seven address digits are expected for the called party address, and not ten digits.
- Inclusion of OPART digits for IMT originations and terminations for the QS3PAO IMT dialing plan.
- Unique setting of treatments in certain call processing scenarios. (Typically ONNET calls set busy line [BUSY] treatment instead of generalized no circuit [GNCT] treatment.)

Interactions

The on-network indicator interacts with many existing UCS DMS-250 switch features.

Restrictions and limitations

The use of the ONNET option is identical to the use of the CT ONNET selector in table STDPRTCT for called party address screening.

PVSPDIDX call processing application

The Private Speed Number Index (PVSPDIDX) option provides the necessary information to successfully screen subscriber number or call type specific private speed dial numbers.

Call processing actions

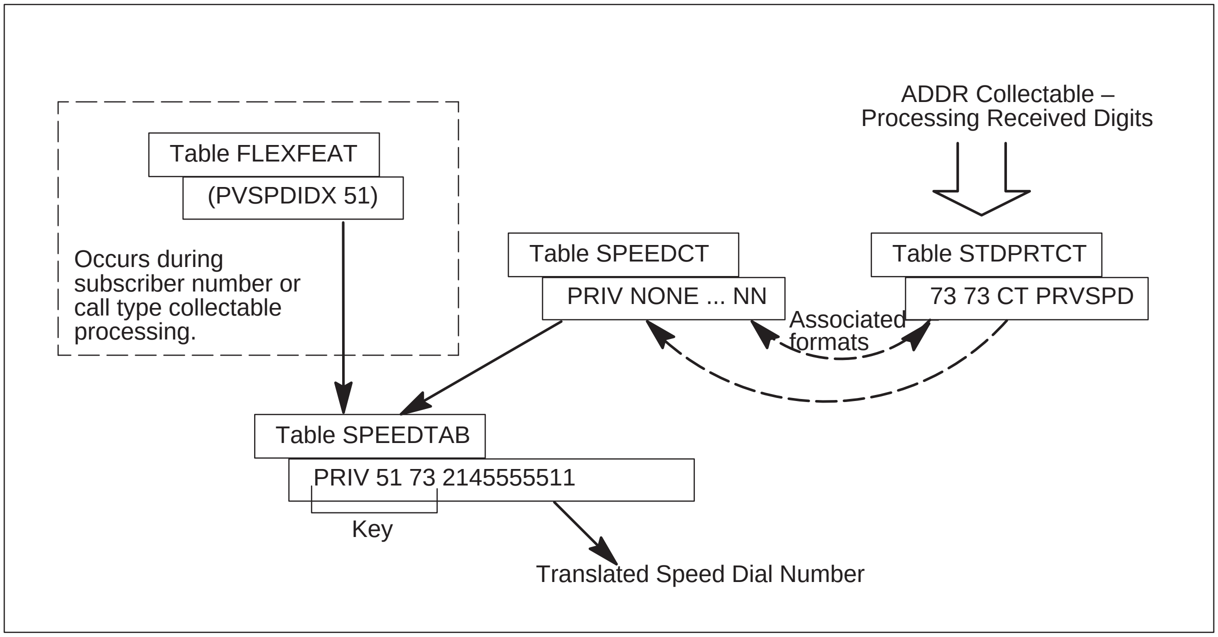
The PVSPDIDX option identifies a numeric index with a range of 0 to 32762 that is to be used as an index into table SPEEDTAB.

The private speed number index option may be processed multiple times during FlexDial framework collectable processing, where each option processed overwrites a value previously set for the call. A private speed number index may be defaulted to zero through use of the CLRFTRS collectable.

The private speed number index for the call is set and processed during the collect information point in the call. The index is set through subscriber number or CALLTYPE collectable processing, and then processed during address collectable execution, when the STDPRTCT table “CT PRVSPD” STDPRT subtable selector is processed for an ADDR or ADDRPARM collectable. The private speed number index is then used in table SPEEDTAB along with the speed number digits to retrieve the called party address for the call.

Figure 18-3 illustrates private speed dial number handling.

Figure 18-3
Private speed dial number handling



Interactions

For a description of the CT PVTSPD selector, see the *UCS DMS-250 Data Schema Reference Manual*.

Restrictions and limitations

Speed dial number capability is enabled through provisioning in table STDPRTCT. The PVSPDIDX option only provides a mechanism to successfully screen private speed dial numbers.

REORGACT call processing application

The Call Reoriginate Actions (REORGACT) option performs a dual role for both subscriber numbers and call types processed:

- Reorigination is enabled for the subscriber number or call type in conjunction with the REORGTYP option.
- How to process the call reorigination attempt is identified.

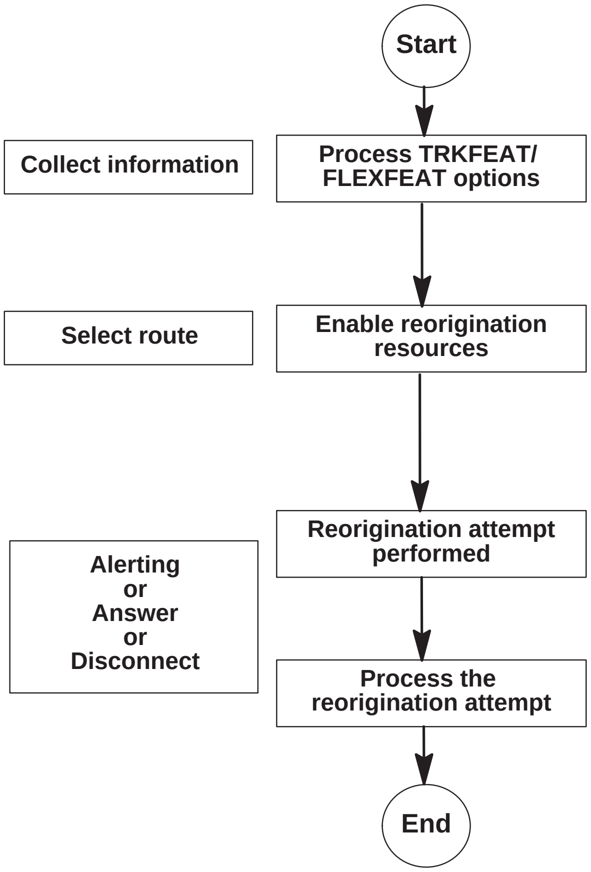
Call processing actions

The REORGACT option identifies the following information for call processing use for reorigination capability:

- the FLEXDIAL table index identifying a list of collectables that are used to identify the required interaction with the originating agent on the reorigination attempt
- the MSGCTR table index that is processed before the identified collectable list to provide required subscriber number or call type specific information for collectable processing
- the disconnect timer value, which is used after called party disconnect to transition the call from ONKEY reorigination access type to ONDISC reorigination access type

Reorigination is a multi-phased feature that contains a number of processing stages, as shown in Figure 18-4.

Figure 18-4
Reorigination processing stages


Start
Collect information
Process TRKFEAT/ FLEXFEAT options
Select route
Enable reorigination resources
Reorigination attempt performed
Alerting or Answer or Disconnect
Process the reorigination attempt
End

The TRKFEAT REORIGAL option must be identified for the originating trunk group, and the REORGACT and REORGTYP options must be provisioned for a processed subscriber number type or calltype in order to enable reorigination.

ONKEY reorigination resources are enabled when the connection to the terminating agent is established and seizure of the terminating agent has occurred. The REORGTYP option identifies the type of reorigination which is to be enabled for the call.

The reorigination attempt is received according to the type of reorigination enabled. This may occur before answer, during the talking state of the call, or after called party disconnect.

When processing the reorigination attempt, the take down of the terminating side of the call is executed as needed, the billing record is delivered to the formatter and a record for the reoriginated call is started. Also, messages identified by the MSGCTR index are posted, and the call returns to the collect information point in the call model where the identified collectable list is processed.

The REORGACT option is used in conjunction with the REORGTYP option and TRKFEAT REORIGAL option to enable reorigination for a call. (The TRKFEAT REORIGAL option is provisioned for the originating trunk group.) All the options must be provisioned in order to enable reorigination for the call. If the REORGACT and REORGTYP options are provisioned for the subscriber number or CALLTYPE collectable, but the REORIGAL option is not provisioned for the trunk group, then reorigination is not enabled for the call (and vice-versa).

The REORGACT option can be processed multiple times during FlexDial framework collectable processing, where each option processed overwrites the reorigination DPIDX, MSGCTR, and DISCTMR values previously set for the call. Reorigination ability for the call may be disabled through use of the CLRFTRS collectable.

In addition to the requirement of provisioning the REORGACT, REORGTYP, and REORIGAL options, additional requirements must be met before reorigination can be enabled for the call:

- A treatment set for the call cannot have been set by a subscriber number collectable.
- The call may not be a hotline-identified call.

During the select route point in the call model, if reorigination ability is enabled for the call, then reorigination is enabled according to the types of reorigination provisioned in the REORGTYP option.

When reorigination occurs for the call (either through the ONDISC or ONKEY reorigination type), then a number of events occur in order to process the reorigination attempt:

- Any resources required to enable reorigination for the call are released, in particular STR resources. UTR resources remain allocated for collectable processing.
- The terminating agent member is released and the network connection is deallocated if needed (if not already performed).
- The recording unit is copied and delivered to the formatter to format the CDR record. The fields copied are based on the REORIGCP field in table CDRTMPLT for the template used for formatting.

  Note: If the CDR record is formatted according to the INTERNAL definition, then the information copied to the new recording unit is based upon non-FlexDial hard-coded standards.

- Messages identified by the MSGCTR table index are posted at the message center.
- Call processing returns to the collect information point in the call model.
- All required subscriber numbers are revalidated according to the subscriber number collectable which originally processed the digits. Features and characteristics of the subscriber number are not re-applied to the call though (see “REVALIDATE call processing application” on page 18-39). For validation failures, the action identified by the FAILACT field is applied to the call.

- After revalidation attempts, the collectable list identified by the FLEXDIAL table index retrieved from the DPIDX field is processed.

Received subscriber numbers can be identified for revalidation through the following methods:

- through provisioning the REVALIDATE option for the subscriber number type in table FLEXTYPE
- through provisioning the REVALIDATE option for the subscriber number in table FLEXFEAT
- through provisioning the REVERIFY field of the TRKFEAT REORIGAL option with the subscriber number type. Multiple subscriber number types may be provisioned in the REVERIFY field.

Interactions

The REORGTYP option identifies the type of reorigination enabled for the call (either ONDISC or ONKEY or a combination). See “REORGTYP call processing application” on page 18-34.

The REVALIDATE options for tables FLEXTYPE and FLEXFEAT identify if a particular subscriber number type or subscriber number is to be revalidated on reoriginated calls. See “REVALIDATE call processing application” on page 18-39.

The REVERIFY option for table TRKFEAT identifies one or more subscriber number types that when processed for the call, must be revalidated on reoriginated calls. See “REORGVAL call processing application” on page 20-13.

Restrictions and limitations

The following restrictions and limitations apply to the REORGACT call processing application:

- Features and characteristics of revalidated subscriber numbers are not applied to the call. Values currently set for the call continue to be used for the reoriginated call.
- The FAILACT action of a revalidated subscriber number collectable applies if the revalidation is unsuccessful.
- A reoriginated call begins with a clean copy of the call processing entities that are essential to collectable processing. No leftover MSGCTR messages or unprocessed collectables are carried over to the reoriginated call.

- If the CDR record is formatted according to the INTERNAL format, then the CDR fields copied for the reoriginated call are based on non-FlexDial hard-coded standards.

REORGTYP call processing application

The Reorigination Enable Type (REORGTYP) option performs a dual role for subscriber numbers and call types processed:

- Reorigination is enabled for the subscriber number or call type in conjunction with the REORGACT option.
- How to enable reorigination for the call is identified.

Call processing actions

Processing the REORGTYP option identifies how reorigination is enabled for the call. Three types of reorigination are available for call processing use:

- ONDISC reorigination type specifies that reorigination occurs at called party disconnect, either immediately or after a period of time identified by the REORGACT option DISCTMR field.
- ONKEY STR reorigination type indicates that reorigination may occur through reception of the identified reorigination digit. This digit can be received before answer, during the talking state of the call, or after called party disconnect until the timer identified by the DISCTMR field expires. Receipt of the reorigination digit is monitored by the STR XPM hardware.
- ONKEY UTR reorigination type indicates that reorigination can occur through reception of the identified reorigination digits. This digit can be received before answer or after called party disconnect until the timer identified by the DISCTMR field expires. Receipt of the reorigination digits is monitored by the UTR XPM hardware.

  Note: Due to limited UTR resources in the XPM, the use of the UTR for reorigination digit monitoring should be limited in order to avoid UTR resource exhaustion for call processing (including call origination) purposes.

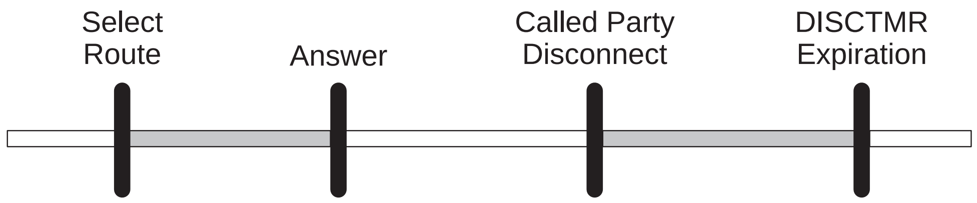
The REORGTYP option is used in conjunction with the REORGACT option and TRKFEAT REORIGAL option to enable reorigination for a call. (The TRKFEAT REORIGAL option is provisioned for the originating trunk group.) All the options must be provisioned in order to enable reorigination for the call. If the REORGACT and REORGTYP options are provisioned for the subscriber number or CALLTYPE collectable, but the REORIGAL option is not provisioned for the trunk group, then reorigination is not enabled for the call (and vice versa).

The REORGTYP option may be processed multiple times during FlexDial framework collectable processing, where each option processed overwrites the reorigination types previously set for the call. Reorigination ability for the call may be disabled through use of the CLRFTRS collectable.

Reorigination ability for the call is identified during the collect information point in the call model, and enabled for the call during either the select route or called party disconnect point in the call model.

Reorigination is enabled according to the reorigination types specified in the REORGTYP option. The available types are processed for the call, as shown in Figures 18-5, 18-6, 18-7, 18-8, 18-9, and 18-10.

Figure 18-5
ONKEY UTR reorgination type support



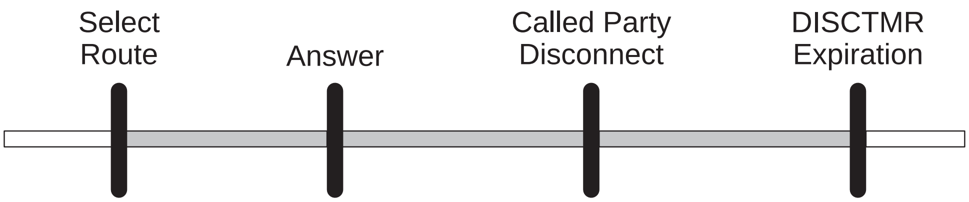
Reorigination is enabled before answer and after called party disconnect.

Supported Reorigination Digits : Single Asterisk (*)
Two Asterisks (**)
Single Octothorpe (#)

For two-asterisk reorigination support, both digits must be received within the DISCTMR timer value identified by the REORGACT option. If one digit but not the other is received, then upon expiration of the DISCTMR timer value, the digit received is discarded. At this point the subscriber must enter two asterisks in order to reoriginate.

UTR reorigination digit reception supports the receipt of digits using XPM UTR hardware.

Figure 18-6
ONKEY STR reorgination type support

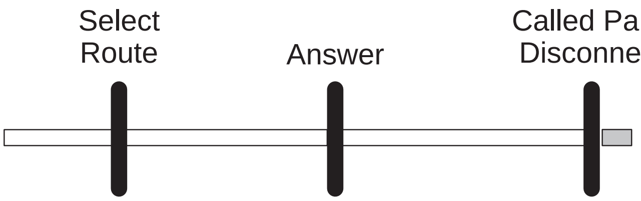


Reorigination is enabled before answer and after called party disconnect.

Supported Reorigination Digits : Single Asterisk (*)
Single Octothorpe (#)

STR reorigination digit reception supports the receipt of digits using XPM STR hardware. The minimum supported digit duration is 500 ms.

Figure 18-7
ONDISC reorigination type support—immediate activation

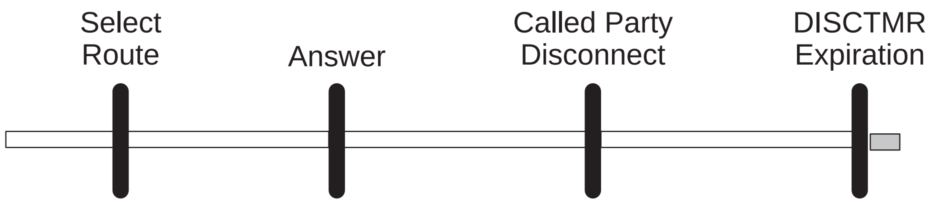


The IMMED field is set to Y.

Reorigination is activated immediately after called party disconnect.

For ONDISC type of reorigination, the activation of reorigination may be immediate or it may be delayed according to the DISCTMR value Y, then the ONDISC reorigination is activated immediately after called party disconnect. If the IMMED field is set to N, then ONDISC reorigination occurs after the expiration of the DISCTMR.

Figure 18-8
ONDISC reorigination type support—delayed DISCTMR activation

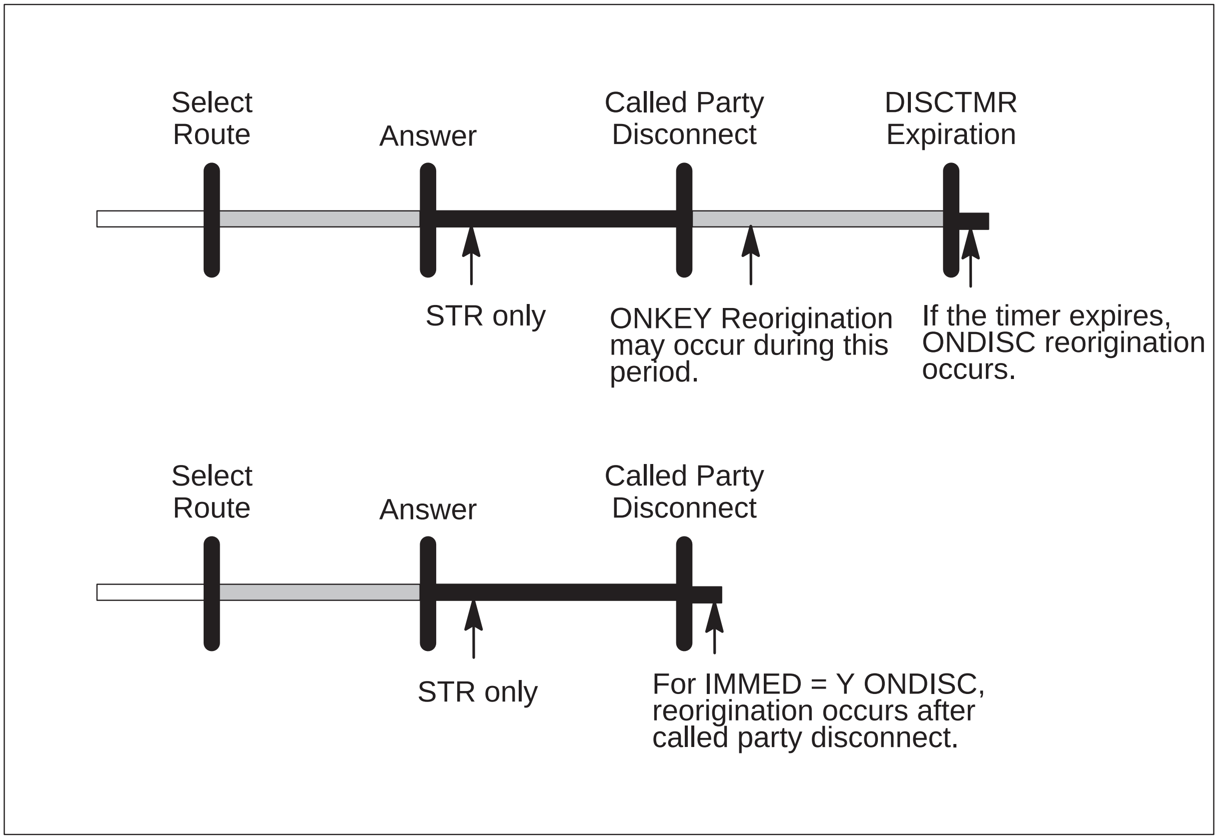


The IMMED field is set to N.

Reorigination is activated immediately after the DISCTMR value timer expires.

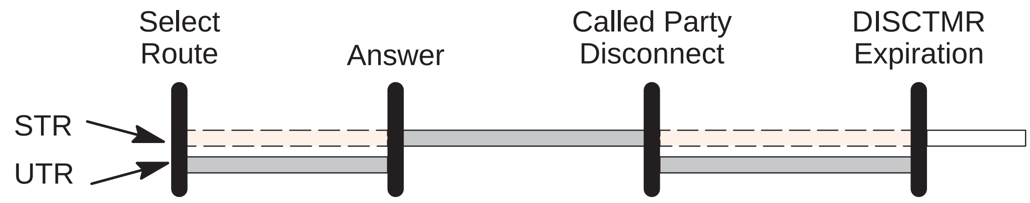
The different types of reorigination may be used in combination for the call. For ONKEY in combination with the ONDISC reorigination type, the DISCTMR value provisioned in the REORGACT option marks the transition from ONKEY reorigination support to ONDISC reorigination support.

Figure 18-9
Combination of ONKEY and ONDISC reorgination support


Select Route
Answer
Called Party Disconnect
DISCTMR Expiration
STR only
ONKEY Reorigination may occur during this period.
If the timer expires, ONDISC reorigination occurs.
Select Route
Answer
Called Party Disconnect
STR only
For IMMED = Y ONDISC, reorigination occurs after called party disconnect.

For ONKEY UTR in combination with ONKEY STR, the two hardware devices may be used independently to receive the reorigination digits. If the identified reorigination digit is the same for both devices, then the UTR device takes precedence for receiving the reorigination digit from the subscriber.

Figure 18-10
Combination of ONKEY UTR and STR reorigination type support



The Reorigination Digit for both STR and UTR either:

Single Asterisk (*)
or
Single Octothorpe (#)

Interactions

The REORGACT option identifies how call processing handles the reoriginated call. See “REORGACT call processing application” on page 18-29.

Restrictions and limitations

For UTR and STR ONKEY types of reorigination, the UTR ONKEY type takes precedence when the reorigination digit for both options is identical.

REVALIDATE call processing application

The FLEXFEAT table REVALIDATE option application is identical to the description of the REVALIDATE option for table FLEXTYPE, except that the option is executed on a subscriber number basis and not on a subscriber number type basis. See “REVALIDATE call processing application” on page 17-13 for more information.

SPLASHBK call processing application

The Splashback (SPLASHBK) option identifies an index into table SPLASHID to provide a special splashback tone before a treatment tone is applied to the call.

Call processing actions

The SPLASHBK option identifies an index into table SPLASHID that is used when the call is terminated to a treatment tone.

The SPLASHBK option may be processed multiple times during FlexDial framework collectable processing, where each option processed overwrites the table index previously set for the call. The SPLASHBK index value for the call is reset to a value of zero through use of the CLRFTRS collectable.

Restrictions and limitations

The SPLASHID table is only used when the call terminates to a treatment tone.

TCAPANNC call processing application

The TCAP Announcement (TCAPANNC) option identifies an index into the Transaction Capabilities Application Part (TCAP) ANNC table that is used in processing standard or custom TCAP announcement parameters in the response message for the N00 TCAP application. This option is applicable for both subscriber number and CALLTYPE collectable processing.

Call processing actions

The TCAPANNC option identifies an index into table TCAPANNC that is used for processing standard or custom TCAP announcement parameters received in the N00 application TCAP response message.

The TCAPANNC option may be processed multiple times during FlexDial framework collectable processing, where each option processed overwrites the table index previously set for the call. The TCAPANNC index value for the call is reset to a value of zero through use of the CLRFTRS collectable.

The execution of the N00 TCAP application occurs within ADDR or ADDRPARM collectable processing, where a received custom announcement parameter can be processed.

Note: For more information on the TCAP feature, see the *UCS DMS-250 Transaction Capabilities Application Part (TCAP) Application Guide.*

Restrictions and limitations

If the identified index is not provisioned in the TCAPANNC table, then the default value of zero is used by call processing.

TRANSNUM option call processing application

The TRANSNUM option is only processed upon successful validation of the “dialed” subscriber number, and replaces the dialed subscriber number for the application of subscriber number features (with the exception of the CLDPBILL and GENLOG option features).

For the GENLOG option, the dialed subscriber number digits are output in the generated FLEX 302 log. For the CLDPBILL option, the dialed subscriber number digits are automatically recorded in the BILLNUM field of the CDR.

The translated subscriber number digits are recorded in the CDR in accordance with the use of the table FLEXTYPE BILLFLD option. The translated subscriber number digits are also used to represent the subscriber number type processed for the remaining of the call, and may be outpulsed or presented to the terminating switch dependent upon the outpulsing scheme used.

The use of TRANSNUM within a CLRFTRS collectable does not have any effect on call processing.

Usecase Scenarios

The TRANSNUM option may be used for pseudo–subscriber number support. The TRANSNUM option would also be used for table STDPRTCT ES FLEXVAL usage for inswitch translations of a called party number.

Interactions

The TRANSNUM option interacts with ADDR/ADDRPARM collectable use of the STDPRTCT ES FLEXVAL selector in translating called party address digits for the call. Typically when incorporated for ES FLEXVAL processing, the TRANSNUM option would also be used with the TRANSYS option.

Restrictions and limitations

The following call processing restrictions and limitations exist for TRANSNUM option use:

- The TRANSNUM option does not apply to CALLTYPE collectable processing.

TRANSTS call processing application

The Translation Serving Translation Scheme (TRANSTS) option identifies either the originating and terminating partition numbers or the serving translation scheme used for translations on the UCS DMS-250 switch. This option is applicable for both subscriber number and call type collectable processing.

Call processing actions

The TRANSTS option identifies the serving translation scheme (STS) used for translations. This option identifies either:

- the STS value directly
- an OPART and TPART value used to index table PARTOSTS to retrieve the STS value

The TRANSTS option can be processed multiple times during FlexDial framework collectable processing, where each option processed overwrites the STS value previously set for the call. The TRANSTS value for the call is reset to the value of the DEFAULT_STS office parameter through use of the CLRFTRS collectable.

Use of the STS value occurs during the analyze information point in the call, when the destination for the call is being determined.

Interactions

See the *UCS DMS-250 CSP Translations Reference Manual* for more information on STS processing.

Restrictions and limitations

If the TRANSTS option is not used to identify the STS for the call, then the value specified by the DEFAULT_STS office parameter is used for the call for translations.

TRANSYS call processing application

Processing the TRANSYS option identifies the translations system for the call. The TRANSYS option may be processed multiple times during the call, with each subsequent occurrence overwriting the value set by the previous occurrence. The translations system for the call may also be set / overwritten through address digits pretranslations in table STDPRTCT, or through use of the SETTRANS collectable.

The TRANSYS option must be used before translations occurs for the call. Setting the translations system for the call after call translations has occurred has no discernible effect on the call.

The use of TRANSYS in a CLRFTRS collectable results in the translations system being set to nil for the call.

Usecase Scenarios

The TRANSYS option would typically be used for table STDPRTCT ES FLEXVAL usage for inswitch translations of a called party number.

Interactions

The TRANSYS option interacts with ADDR/ADDRPARM collectable use of the STDPRTCT ES FLEXVAL selector in setting the translations system for the call.

Restrictions and limitations

The following call processing restrictions and limitations exist for TRANSYS option use:

- The translations system value set by the TRANSYS option may be overwritten by another TRANSYS option, a SETTRANS collectable, or called party address digits pretranslations which occurs later in the call.

TRKGRP features and characteristics applications

Table TRKGRP provides the definition of originating and terminating AXXESS agents used for call processing. This definition includes signaling characteristics, feature characteristics, and originating agent interaction definition.

Supported agents

The TRKGRP definition described in this document applies only to AXXESS trunk group types.

Activation

The definition of the originating agent is applicable throughout the call, and is referenced when the origination attempt is detected.

The definition of the terminating agent is applicable for the call after the route has been selected and the terminating agent is allocated.

Call processing description

The TRKGRP table provides the following features and capabilities for call processing purposes:

- TRAFSNO
- PADGRP
- NCCLS
- SELSEQ
- SIGIDX
- FEATIDX
- DPIDX
- OGRPTYP

The following trunk group fields are not described in this document, but are described in the *UCS DMS-250 Data Schema Reference Manual*:

- TRAFSNO
- PADGRP
- NCCLS
- SELSEQ

SIGIDX field definition

The TRKSIG Table Index (SIGIDX) field contains up to two valid indexes into table TRKSIG. Table TRKSIG identifies the signaling and interface facility characteristics of the AXXESS trunk group. The defined signaling characteristics apply to all members of the trunk group.

The signaling characteristics of the trunk group are referenced throughout call processing. The digit collection signaling characteristics are used during the collect information point in the call model in the processing of FlexDial collectables.

See Chapter 21 for a description of TRKSIG table fields and options.

FEATIDX field definition

The TRKFEAT Table Index (FEATIDX) field contains a valid index into table TRKFEAT. Table TRKFEAT identifies the features and characteristics of the AXXESS trunk group. The defined features and characteristics apply to all members of the trunk group.

The features and characteristics associated with the originating trunk group are accessed and set for the call during the authorize origination point in the call model. Some TRKFEAT features and characteristics set for the call may be overridden with information retrieved from tables FLEXTYPE and FLEXFEAT. (See Chapter 17, “FLEXTYPE application of features and characteristics,” and Chapter 18, “FLEXFEAT application of features and characteristics”.)

The features and characteristics associated with the terminating trunk group are accessed and set for the call during the allocate terminator point in the call model.

See Chapter 20 for a description of TRKFEAT table fields and options.

DPIDX field definition

The FLEXDIAL Table Indexes (DPIDX) field contains a vector of up to four valid indexes into table FLEXDIAL. The TRKGRP identified

FLEXDIAL table indexes identify the initial interaction with the originating agent.

See Chapter 2 for a description of FLEXDIAL table index processing.

OGRPTYP field definition

The Outgoing Group Type (OGRPTYP) field identifies the outpulsing characteristics of the AXXESS trunk group. AXXESS trunk group members may mimic either existing DAL, ONAL, ONAT, or EANT outpulsing capability.

The generic definition of the AXXESS agent does not provide a detailed specification of how the call is to be presented to the terminating switch for AXXESS trunk groups. The OGRPTYP field provides this information by identifying an existing agent type that the AXXESS trunk group mimics for outpulsing purposes.

Supported outpulsing capability for the AXXESS trunk group includes:

- DAL FXS 2-wire seizure
- DAL DS1SIG 4-wire MF and DTMF capability with called party and calling party number delivery
- ONAL FXO 2-wire seizure and DTMF capability with called party number delivery
- ONAL DS1SIG 4-wire DTMF capability with called party number delivery
- ONAT DS1SIG 4-wire MF and DTMF capability with called party number delivery
- EANT DS1SIG 4-wire MF capability with called party number delivery
- multi-stage EANT DS1SIG 4-wire MF capability
- EANT CCS7 capability with IAM generation and delivery

A number of TRKSIG table options relate to the presentation of the call to the terminating switch. These fields include:

- ALTSEIZ PTS option

  The ALTSEIZ option indicates that a different AB-bit seizure configuration is to be used for the FXS or FXO signaling type.

- CPIALLOW PTS option

  The CPIALLOW option identifies that the calling party number is to be outpulsed, and is only applicable for the DAL OGRPTYP.

- DELIVER PTS option

  The DELIVER option identifies if and how the calling party number is to be delivered to the terminating switch, and is only applicable for the EANT OGRPTYP.

- DIGSOUTP PTS option

  The digits to outpulse option identifies the maximum number of called party address digits that are to be outpulsed, and is only applicable for the DAL OGRPTYP.

- MLTSTAGE PTS option

  The multi-stage option indicates that multi-stage outpulsing is to be used, and is only applicable for the EANT OGRPTYP.

Calling party number delivery conditions

The calling party number can only be presented if it has been received for the call. Otherwise, no calling party number digits are delivered, and originating line information digits 02 are outpulsed if applicable.

Note: Originating line information digits are only outpulsed for the EANT (FGD) OGRPTYP.

For AXXESS trunk group originations, the calling party number must be received by the SUBR or SUBRPARM collectable, where the FLEXTYPE used contains the provisioned CALLING FLEXTYPE table option.

Typically, when true calling party number digits are not received for the call, the SNPA and optionally SNXX associated with the originating AXXESS trunk group are used as the calling party number for the call, and are delivered as the calling party number to the terminating switch.

For this functionality to occur for AXXESS originations, the calling party number must be set through FlexDial collectable processing. If the calling party number is not set through FlexDial processing, then no calling party number is available for outpulsing purposes.

Note: The calling party number may be created from the SNPA and SNXX using three collectables: (AGNTDATA PREFIX SNXX) (AGNTDATA PREFIX SNPA) (SUBR 6 6 ANI N N $ $). The ANI FLEXTYPE must be provisioned with the FLEXTYPE table CALLING option. The OLI collectable must also be provisioned with filed digits to identify the originating line information digits.

TRKFEAT features and characteristics application

Table TRKFEAT provides the feature and characteristic specifications for originating and terminating AXXESS agents used for call processing. These interface features and characteristics associated with the agent are retrieved once during the processing of the call (once for the originating agent and once for the terminating agent). They are immediately applied or stored in per-call data structures, where they can be overwritten with other information as applicable.

Supported agents

Table TRKFEAT is only provisioned for AXXESS trunk group types.

Activation

Processing of TRKFEAT table entries for the originating agent occurs during the “authorize origination” point in the call model, which occurs before FlexDial framework collectables are executed.

For the terminating trunk group, entries in these tables are processed during the “allocate terminator” point in the call after selection of the terminating trunk group member occurs.

Terminal interface features and characteristics provisioned in table TRKFEAT differ from those provisioned in tables FLEXTYPE and FLEXFEAT in that the options provisioned in table TRKFEAT are on a per-trunk group basis, and not on a subscriber number (type) basis.

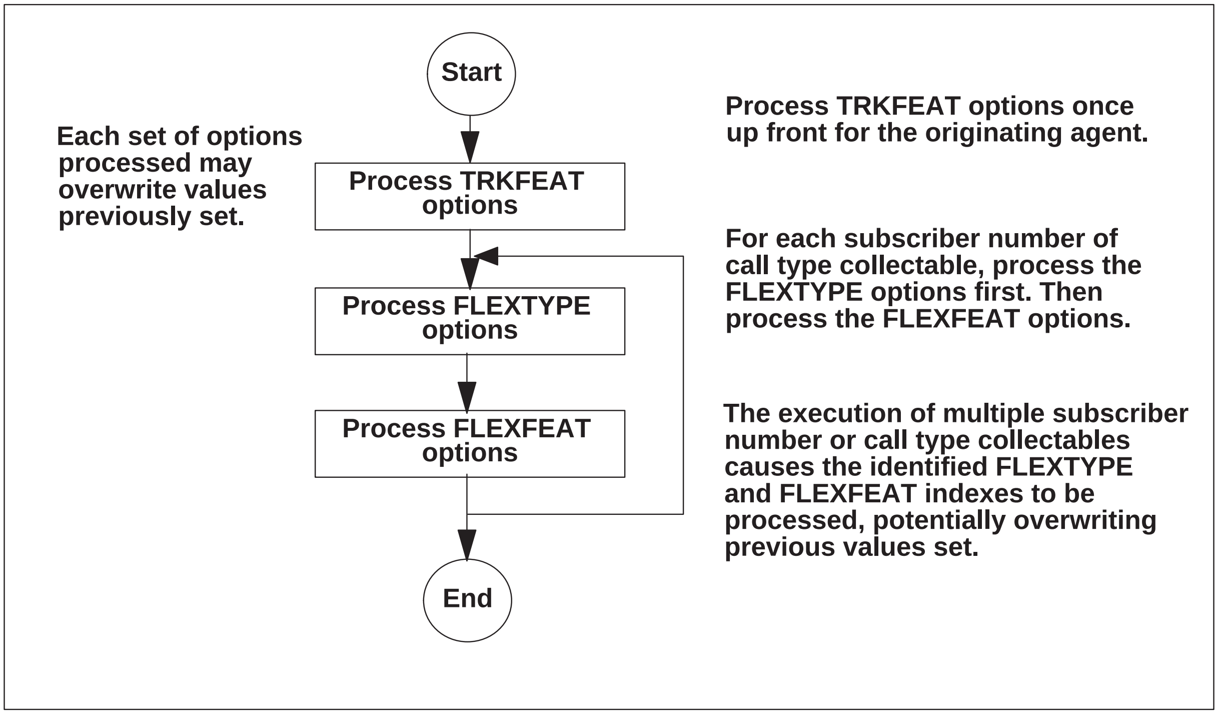
Some features and characteristics for the originating agent have representation in two or more of the following tables:

- FLEXTYPE
- FLEXFEAT
- TRKFEAT

If an option is provisioned in multiple tables, the values used by call processing are the values set by the last option processed.

Figure 20-1 shows common features run time processing for TRKFEAT.

Figure 20-1
Common features and characteristics run time processing


Start
Each set of options processed may overwrite values previously set.
Process TRKFEAT options
Process TRKFEAT options once up front for the originating agent.
For each subscriber number of call type collectable, process the FLEXTYPE options first. Then process the FLEXFEAT options.
Process FLEXTYPE options
Process FLEXFEAT options
The execution of multiple subscriber number or call type collectables causes the identified FLEXTYPE and FLEXFEAT indexes to be processed, potentially overwriting previous values set.
End

TRKFEAT table options are only accessed once for both the originator and terminator, and can be overwritten by other information processed for the call.

Call processing description

The TRKFEAT table provides the following originating features and capabilities for originating agent call processing purposes:

- ALTTRTMT
- ANSCDR
- CAIN
- CAINGRP
- CDRTMPLT
- CICRTE
- CICSIZE
- CITYCODE
- COSOVE

- DEFCIC
- IEXCLINX
- LATA
- MSGCTR
- NETSEC
- OHQ
- REORIGAL
- SNPA
- SNXX
- SUSTMR
- TIMEBIAS
- TRANSTS
- TRKCOS
- ZONE
- JIP

The TRKFEAT table provides the following terminating features and capabilities for terminating agent call processing purposes:

- CAIN
- CAINGRP
- ENFNPAOP
- NETSEC
- NOANSDUR
- OHQTERM
- ONNETTRK
- SNPA
- SNXX
- TRKCOS
- ZONE

ALTTRTMT call processing application

The Alternate Treatment (ALTTRTMT) option identifies that the ALTERNAT subtable within the TMTCNTL table is used to decode a treatment set for the call. The ALTTRTMT option is only applicable for originating AXXESS trunk groups.

Call processing actions

The alternate treatment option does not identify any additional information for call processing purposes.

When a set treatment is applied to the call, the treatment set is first decoded in table TMTCNTL to determine the routing action for the call. Typically the TMTCNTL subtable associated with the originating agent is used (in this case, the AXXESS subtable) for this process.

The ALTTRTMT option identifies that the ALTERNAT subtable is to be used instead of the trunk group subtable.

Interactions

Many types of treatments and delayed treatments may be set in collectable processing. See Chapter 15, “FlexDial call processing applications”.

Restrictions and limitations

If the treatment set is not provisioned within the ALTERNAT subtable, then the treatment set is decoded by the OFFTREAT subtable.

ANSCDR call processing application

The TRKFEAT Table ANSCDR option application is identical to the description of the ANSCDR option for table FLEXTYPE, except that the option is executed on a originating AXXESS trunk group basis and not on a subscriber number type basis. See “ANSCDR call processing application” on page 17-3 for more information.

CAIN call processing application

The Carrier AIN (CAIN) option indicates that CAIN services are available for the originating (option ORIGOPTS) or terminating (option TERMOPTS) AXXESS trunk group.

For more information on CAIN services, see the *UCS DMS-250 NetworkBuilder Application Guide*. For more information on CAIN and FlexDial interactions, see *UCS DMS-250 CAIN/FlexDial Interactions* or ”FlexDial Interactions” appendix.

CAINGRP call processing application

The TRKFEAT Table CAINGRP option application is identical to the description of the CAINGRP option for table FLEXFEAT, except that the option is executed on a originating or terminating AXXESS trunk group basis and not on a subscriber number type basis. See “CAINGRP call processing application” on page 18-6 for more information. For more information on CAIN and FlexDial interactions, see *UCS DMS-250 CAIN/FlexDial Interactions* or ”FlexDial Interactions” appendix.

Restrictions and limitations

Before datafilling the CAINGRP option, datafill the identical CAIN group in table CAINGRP.

CDRTMPLT call processing application

The TRKFEAT Table CDRTMPLT option application is identical to the description of the CDRTMPLT option for table FLEXTYPE, except that the option is executed on a originating AXXESS trunk group basis and not on a subscriber number type basis. See “CDRTMPLT call processing” on page 17-8 for more information.

CICSIZE call processing application

The CICSIZE option is only applicable for originating agents and verifies whether the incoming CIC is three or four digits. The CICSIZE option is for use during the transitional phase from three-digit to four-digit CICs. The CIC4_TRANS_COMP office parameter specifies whether the transitional phase is complete. If the office parameter is set to N, the CICSIZE option defaults to 3DIGS. If it is set to Y, the CICSIZE option defaults to 4DIGS.

Call processing actions

The originating agent provides a CIC. The CICSIZE option specifies whether the switch expects the CIC to have three or four digits. If the CIC’s size is different from the expected CIC size, a log is printed:

- If the CIC4_TRANS_COMP office parameter is set to Y and a three-digit CIC is received, a TRK111 log is generated.
- If the CIC4_TRANS_COMP office parameter is set to N and a four-digit CIC is received, a DFIL300 log is generated.

CICRTE call processing application

The CIC Route Option (CICRTE) is only applicable for originating AXXESS trunk groups and identifies that received CIC digits are to be used to index table CICRTE in order to retrieve routing information for the call.

CITYCODE call processing application

The Citycode (CITYCODE) option identifies the citycode digits associated with the originating AXXESS agent.

The citycode option is only applicable for originating AXXESS trunk groups.

Call processing actions

A processed CITYCODE option immediately identifies the citycode value (digits) for the originating trunk group and the call.

The CITYCODE option is used in conjunction with the CITYVAL option and the FLEXFEAT table CITYCODE option to perform citycode validation on a received subscriber number.

Citycode validation can occur during processing of a SUBR or SUBRPARM collectable. For citycode validation, the citycode value for the call (which may be set by the TRKFEAT CITYCODE option) is used in conjunction with the FLEXFEAT CITYCODE option and the CITYVAL option to perform citycode validation. For a description of citycode validation, see “CITYCODE validation” on page 16-112.

The trunk group CITYCODE option is also used in conjunction with the AGNTDATA collectable, where the collectable can add the digits provisioned in the CITYCODE option to the digit buffer.

Interactions

The CITYCODE option is used for subscriber number citycode validation. See “CITYCODE validation” on page 16-112 for a description of citycode validation.

For interactions with the AGNTDATA collectable, see “Digit manipulation collectable (AGNTDATA)” on page 16-43.

Restrictions and limitations

The following restrictions and limitations apply for the CITYCODE call processing application:

- If the CITYCODE option is not provisioned for the originating trunk group, then the citycode value for the call must be identified through use of the FLEXFEAT CITYCODE option in order to perform meaningful citycode validation attempts.
- If the CITYCODE option is not provisioned for the originating trunk group, then the AGNTDATA collectable attempting to add CITYCODE digits does not add any digits to the digit buffer.

COSOVE call processing application

The Class Of Service (COS) Override (COSOVE) option identifies the FLEXDIAL table index that is applied to the call when COS screening fails for the processed called party address digits. The COSOVE option is only applicable for originating AXXESS trunk groups.

Call processing actions

Processing the COSOVE option specifies a FLEXDIAL table index that identifies an alteration to the interaction with the originating agent. The list of collectables identified by the FLEXDIAL table entry replaces the currently processing collectable list (see “REPLACE list modification action” on page 15-13).

Use of the COSOVE option is only applicable during ADDR or ADDRPARM collectable processing. If COS screening fails for the received called party address digits, and the COSOVE option is provisioned for the originating trunk group, then the COS screening action is overridden and the FLEXDIAL index identified by the COSOVE option is applied to the call. The COSOVE option can only be applied once during processing of the call, and if COS screening fails a second time, then the identified COS screening action is applied to the call.

When the COSOVE option is applied to the call, the FLEXDIAL table index identifies a collectable list that replaces the currently processing collectable list for the call. Then processing of the ADDR or ADDRPARM collectable is stopped, and the next collectable (the first one in the new list) is executed.

Restrictions and limitations

The following restrictions and limitations apply for the COSOVE call processing application:

- The COSOVE option is only applicable during ADDR or ADDRPARM collectable processing, and only if COS screening fails for the called party address digits.
- The COSOVE option may only be used once during the call process.

DEFCIC call processing application

The default CIC (DEFCIC) option applies only to originating agents and defines a default CIC for the call when no CIC was received.

Call processing actions

For information on when the switch outpulses the default CIC, see the section “PTS signaling type OUTCIC option” on page 10-17.

When the CIC, CICPARM, SUBR, or SUBRPARM collectable processes a default CIC with the OPERRTE option, an ADDR OPER OPERRTE message is formatted for all ADDR or ADDRPARM collectables instructing the collectables to route operator calls using table OPERRTE, indexed by the DEFCIC. The message will not be marked dominant and may be overridden by other ADDR OPER messages.

CAIN calls may supply the incoming CIC in the carrier parameter of an outgoing TCAP query so it may be used by downstream processing to make decisions about the call. If no incoming CIC is present, a DEFCIC option is present in table TRKFEAT, and the CAINCIC suboption is present in the DEFCIC option, the DEFCIC will be used in the carrier parameter of any outgoing TCAP query.

For information on the AGNTDATA option, see the section “AGNTDATA sequence collectable” on page 2-63.

ENFNPAOP call processing application

The Enforce NPA Outpulsing (ENFNPAOP) option applies only to terminating agents and disallows the UCS DMS-250 switch from stripping the NPA from the outpulsed number.

IEXCLINX call processing application

The Incoming Exclusion Index (IEXCLINX) option identifies an index into table IEXCLUDE which provides for NPA-NXX blocking for certain call types. The IEXCLINX option is only applicable for originating AXXESS trunk groups.

JIP call processing application

A Jurisdiction Information Parameter (JIP) is an SS7 IAM parameter that carries geographic and service provider information that is associated with the calling party. The parameter contains the originating switch’s Location Routing Number (LRN) and is used to enhance Local Number Portability (LNP) capabilities. JIP applies to originating FGD and SS7 IMT trunks (excluding global).

The TRKFEAT JIP option field allows customers to designate a JIP on a per-trunk group basis when one is not present on the incoming call. The JIP is sent to the Switching Control Point (SCP) in a query message (for NetworkBuilder), as well as built and sent out on the outgoing IAM if the terminator is SS7. When the JIP is sent, the ORIGLRN CDR field is populated with the JIP.

Note 1: If a default JIP is provisioned on the originating trunk, then the building and sending of the JIP takes place for all calls, not just for NetworkBuilder calls.

Note 2: For more information on NetworkBuilder and LNP, see the *UCS DMS-250 NetworkBuilder Application Guide* and the *UCS DMS-250 Local Number Portability Feature Application Guide*.

MSGCTR call processing application

The Message Center (MSGCTR) option identifies messages provisioned in table MSGCTR that are to be posted for the call. These messages relate originating AXXESS agent specific information that is to be used in processing FlexDial framework collectables.

Call processing actions

Processing the MSGCTR option identifies an index into the MSGCTR table. This index identifies a list of messages that are posted for the call at the message center. The TRKFEAT table may contain up to eight MSGCTR options provisioned, where each option identifies additional messages that are to be processed for the call.

The MSGCTR options for the originating AXXESS agent are processed during the authorize origination point in the call, before the first collectable is processed. This guarantees that MSGCTR messages associated with the originating trunk group are processed first by executing collectables. MSGCTR messages associated with the originating agent are only processed once during the call.

The MSGCTR options are processed in the order that they are provisioned within table TRKFEAT. The messages at each identified MSGCTR table index are posted in the order they are provisioned.

Note: It is recommended that messages be provisioned in the same order. Do not provision “(ADDR FILED)(ADDR OPER)” one time and “(ADDR OPER)(ADDR FILED)” another, as undesirable results may occur.

Interactions

For table MSGCTR specifications, see Chapter 4, “Table MSGCTR”.

Restrictions and limitations

Any unused messages are ignored and discarded by call processing when answer occurs for the call.

NETSEC call processing application

The NETSEC option applies to both originating and terminating agents and specifies if the UCS DMS-250 switch must generate a network security (NETS) log report or a CDR when the call is answered. The NETS logs pertaining to this option are NETS601 through NETS604.

The NETWORK_SECURITY_GEN_CDR office parameter specifies whether the UCS DMS-250 generates a NETS log or a CDR when the NETSEC option is set. If the office parameter is set to Y, a CDR is generated. The CDR template is specified in the NETSEC option. If the office parameter is set to N, a NETS log report is generated.

For more information, see the *UCS DMS-250 Logs Reference Manual* and the *UCS DMS-250 Office Parameters Reference Manual*.

Call processing actions

Processing the NETSEC option identifies the following:

- the CDR template to be used for the call (when a CDR is generated)
- an index into table NETSPROF

Table NETSPROF, for fraud detection, performs profile screening. A tuple in this table contains the screening criteria. For more information on table NETSPROF, see the *UCS DMS-250 Data Schema Reference Manual*.

Restrictions and limitations

If both the NETSEC and ANSCDR options contain datafill, the ANSCDR option takes precedence over the NETSEC option.

The NETSEC CI command is not supported for AXXESS agencies.

If the SOC UBFR0003 is not in the ON state, the UCS DMS-250 switch does not perform profile screening.

If the NETSEC option’s PROFIDX field’s value equals 0, the UCS DMS-250 switch does not perform profile screening.

NOANSDUR call processing application

The TRKFEAT table NOANSDUR option application is identical to the description of the NOANSDUR option for table FLEXFEAT, except that the option is executed for a terminating AXXESS trunk group basis and not on a subscriber number basis. See “NOANSDUR call processing application” on page 18-25 for more information.

OHQ call processing application

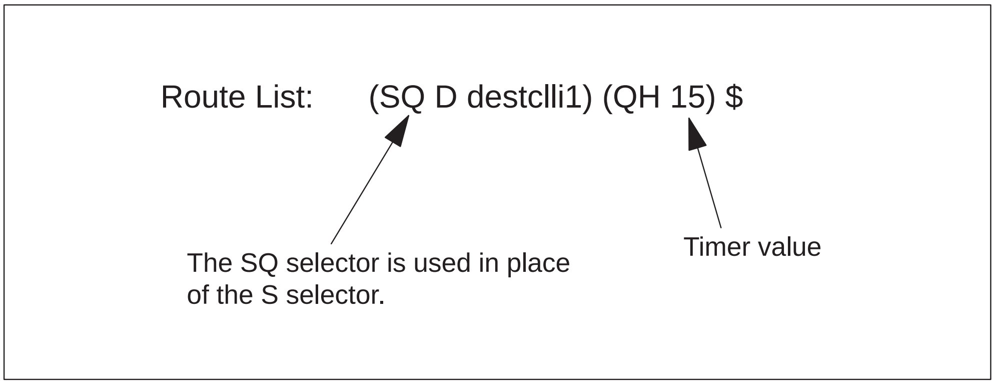
The Off-hook Queuing (OHQ) option indicates that the off-hook queuing routing capability is available for members of the AXXESS trunk group. Off-hook queuing capability is only applicable for originating agents.

Call processing actions

Processing the OHQ option does not identify any additional information for call processing use.

The off-hook queuing option identifies that the originating agent is allowed to queue for the desired trunk group destination. The triggering of off-hook queuing capability occurs during the select route point in the call, through processing of the SQ and QH route selectors, as shown in Figure 20-2 .

Figure 20-2
Off-hook queueing example



```
Route List:     (SQ D destclli1) (QH 15) $
```

The SQ selector is used in place of the S route selector. If an idle member of the destination trunk group is not available, then the trunk group is marked for off-hook queuing (providing that the OHQTERM option in the next section is provisioned against the destination trunk group).

When the QH selector is processed and at least one destination trunk group is marked for queuing, then the call is queued until a destination trunk group member is available or the amount of time indicated expires. A route advance action is performed when the timer expires.

If a destination trunk group member is available before expiration of the OHQ timer, then the timer is cancelled and the call completes to the available member.

Interactions

The OHQTERM option must be provisioned against the destination trunk group. See “OHQTERM call processing application” on page 20-12.

Restrictions and limitations

The OHQ option only identifies that off-hook queuing capability is available for the originating AXXESS trunk group, and does not by itself trigger off-hook queuing functionality. The SQ and QH route selectors must be used to trigger off-hook queuing functionality.

OHQTERM call processing application

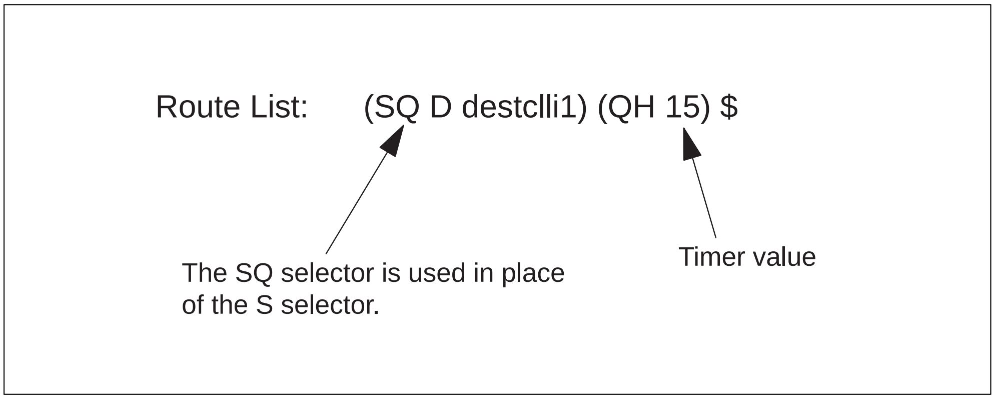
The Off-hook Queuing Terminator (OHQTERM) option indicates that off-hook queuing may occur on the terminating AXXESS trunk group.

Call processing actions

Processing the OHQTERM option does not identify any additional information for call processing use.

The off-hook queuing terminator option identifies that the terminating agent is available for off-hook queuing purposes. The triggering of off-hook queuing capability though occurs during the select route point in the call, through processing of the SQ and QH route selectors, as shown in Figure 20-3.

Figure 20-3
Off-hook queueing example


Route List: (SQ D destclli1) (QH 15) $
The SQ selector is used in place of the S selector.
Timer value

The SQ selector is used in place of the S route selector. If an idle member of the destination trunk group is not available, then use of the OHQTERM option enables the trunk group to be marked for off-hook queuing ability.

When the QH selector is processed and at least one destination trunk group is marked for queuing, then the call is queued until a destination trunk group member is available or the amount of time indicated expires. A route advance action is performed when the timer expires.

If a destination trunk group member is available before expiration of the OHQ timer, then the timer is cancelled and the call completes to the available member.

Interactions

The OHQ option must be provisioned against the originating trunk group. See “OHQ call processing application” on page 20-11.

Restrictions and limitations

The OHQTERM option only identifies that off-hook queuing capability is available for the terminating AXXESS trunk group, and does not by itself trigger off-hook queuing functionality. The SQ and QH route selectors must be used to trigger off-hook queuing functionality.

ONNETTRK call processing application

The On-network Trunk (ONNETTRK) option identifies the AXXESS trunk group as an on-network destination for call terminations.

Call processing actions

Use of the ONNETTRK option identifies that the AXXESS trunk group destination is an on-network destination, versus being an off-network destination. The ONNETTRK option is not applicable for AXXESS trunk groups as the originating agent in the call.

The on-network trunk indicator is used in multiple areas of call processing:

- An onnet answer indicator is returned in ANM messages when answer occurs for interworking with ISUP originators.
- Busy line (BUSY) treatment is set instead of generalized no circuit (GNCT) treatment when a route cannot be successfully terminated to.

Interactions

The on-network trunk indicator interacts with existing UCS DMS-250 switch features.

Restrictions and limitations

The ONNETTRK option is not identical in functionality to the ONNET FLEXFEAT table option or the CT ONNET STDPRTCT table call type.

REORIGAL call processing application

The Allow Reorigination (REORIGAL) option identifies that reorigination capability is available for originations on the AXXESS trunk group members. The REORIGAL option is only applicable for originating AXXESS agents.

Call processing actions

Processing the REORIGAL option identifies two items for call processing purposes:

- Reorigination capability is allowed for call originations on members of the AXXESS trunk group.
- The REVERIFY field identifies a list of subscriber number types, identifying subscriber numbers that must be revalidated on a reoriginated call.

The REORIGAL option only indicates that reorigination capability is allowed for originations on the AXXESS trunk group, and does not enable reorigination capability for the call. Reorigination capability is enabled for the call through use of the FLEXFEAT table REORGACT and REORGTYP options.

If a subscriber number matching a subscriber number type provisioned in the REVERIFY field is processed for the call, then the subscriber number is also marked to be revalidated on reoriginated calls. This functionality is identical to the REVALIDATE option (see “REVALIDATE call processing application” on page 18-39) for tables FLEXTYPE and FLEXFEAT, though the REVERIFY field allows the originating trunk group to identify subscriber number types that are to be revalidated on reoriginated calls.

Subscriber number types in the REVERIFY list not processed for the call are ignored by call processing. If a subscriber number type contained in the REVERIFY list is not collected on the original call, but is collected on a reoriginated call, then the subscriber number type is revalidated on future reoriginated calls in accordance with the REVERIFY subscriber number type list.

Interactions

The REORIGAL is used in conjunction with the FLEXFEAT table REORGACT and REORGTYP options to enable reorigination. For a description of reorigination functionality, see “REORGTYP call processing application” on page 18-34 and “REORGACT call processing application” on page 18-29.

See “REVALIDATE call processing application” on page 18-39 for more information on the REVALIDATE option.

Restrictions and limitations

The following restrictions applies to the REORIGAL call processing application:

- The REORIGAL option only indicates that reorigination capability is allowed for originations on the AXXESS trunk group, and does not enable reorigination capability for the call. Reorigination capability is enabled for the call through use of the FLEXFEAT table REORGACT and REORGTYP options.
- If a subscriber number type in the REVERIFY list is not processed on the original call, but is processed on a reoriginated call, then the received subscriber number is revalidated on future reoriginated call in accordance with the REVERIFY list.

SNPA call processing application

The Serving Numbering Plan (SNPA) option identifies the numbering plan area of the originating or terminating AXXESS trunk group.

Call processing actions

Processing the SNPA option identifies the numbering plan area of the originating or terminating agent. As an option for the originating agent, the SNPA field represents the NPA of the originator or subscriber.

As an option for the terminating agent, the SNPA field represents the connecting NPA of the terminating switch or called party.

The SNPA agent characteristic is typically used in conjunction with the SNXX option to form a six-digit calling party number when a true calling party number is not received for the call. This is performed through use of the AGNTDATA collectable to add the provisioned SNPA digits to the digit buffer, where they can later be processed by a SUBR collectable.

The SNPA agent characteristic is also used during the present call point in the call model, where the SNPA field may be used to determine if the NPA in the called party address number should be outpulsed or not.

For the terminating AXXESS trunk group, the SNPA represents the destination NPA and is also used to determine if the NPA in the called party address number is to be outpulsed or not.

Interactions

For interactions with the AGNTDATA collectable, see “Digit manipulation collectable (AGNTDATA)” on page 16-48.

The SNPA option interacts with a number of existing UCS DMS-250 switch features.

Restrictions and limitations

The following restrictions and limitations apply to the SNPA call processing application:

- If the SNPA option is not provisioned for the originating trunk group, then the AGNTDATA collectable attempting to add SNPA digits adds digits 000 to the digit buffer.
- In order to be used as a part of the identified calling party address for the call, the SNPA option must be used in conjunction with the AGNTDATA collectable to include the SNPA digits in the digit buffer. These digits must then be processed by a SUBR collectable, where the subscriber number type is defined with the FLEXTYPE table CALLING option.

SNXX call processing application

The Serving Numbering Exchange (SNXX) option identifies the exchange of the originating or terminating AXXESS trunk group.

Call processing actions

Processing the SNXX option identifies the exchange of the originating or terminating agent. As an option for the originating agent, the SNXX field represents the exchange area of the originator or subscriber.

As an option for the terminating agent, the SNXX field represents the connecting exchange of the terminating switch or called party.

The SNXX agent characteristic is typically used in conjunction with the SNPA option to form a six-digit calling party number when a true calling party number is not received for the call. This is performed through use of the AGNTDATA collectable to add the provisioned SNXX digits to the digit buffer, where they can later be processed by a SUBR collectable.

Interactions

For interactions with the AGNTDATA collectable, see “Digit manipulation collectable (AGNTDATA)” on page 16-48.

The SNXX option interacts with a number of existing UCS DMS-250 switch features.

Restrictions and limitations

The following restrictions and limitations apply to the SNXX call processing application:

- If the SNXX option is not provisioned for the originating trunk group, then the AGNTDATA collectable attempting to add SNXX digits adds digits 000 to the digit buffer.
- In order to be used as a part of the identified calling party address for the call, the SNXX option must be used in conjunction with the AGNTDATA collectable to include the SNXX digits in the digit buffer. These digits must then be processed by a SUBR collectable, where the subscriber number type is defined with the FLEXTYPE table CALLING option.

SUSTMR call processing application

The Suspend Timer (SUSTMR) option identifies the CCS7 suspend and resume timer value for connections to CCS7 IMT agents. The SUSTMR option is only applicable for the call when the originating AXXESS trunk group is connected to a terminating CCS7 IMT agent.

TIMEBIAS call processing application

The Time Bias (TIMEBIAS) option identifies that the trunk group destination and the switch are in different time zones, by identifying the time zone delta between the destination and the switch. The time bias option is only applicable for originating AXXESS trunk groups.

Call processing actions

Processing the TIMEBIAS option identifies the time zone delta between the trunk group destination and the switch.

This time zone delta information is used in performing time of day restriction COS screening, as provisioned in tables COSUS, UNRESDAT, UNRESDAY, and UNRESTIM. The delta is applied to the current switch time in order that time of day restriction screening is properly applied according to the time zone of the subscriber.

For information on COS screening, see “MLTCOSID call processing application” on page 18-22.

Restrictions and limitations

There are no identified restrictions for time bias call processing.

TRANSTS call processing application

The TRKFEAT Table TRANSTS option application is identical to the description of the TRANSTS option for table FLEXFEAT, except that the option is executed for an originating AXXESS trunk group basis and not on a subscriber number basis. See “TRANSTS call processing application” on page 18-42 for more information.

TRKCOS call processing application

The Trunk COS (TRKCOS) option identifies the value to use for trunk to trunk connection screening in table TRKCOS. The TRKCOS option is applicable for both originating and terminating AXXESS trunk groups.

Call processing actions

The TRKCOS option identifies a table TRKCOS index associated with the AXXESS trunk group.

During authorization of the call termination (just before the call is presented to the terminating trunk member selected), TRKCOS table screening is performed. The TRKCOS index value for the originating agent and the terminating agent are used to index table TRKCOS (originating value first). If a Y is retrieved from the table, then the call is allowed to proceed. If an N is retrieved from the table, then the call cannot be terminated to the selected member and a route advance occurs.

For AXXESS agents, if the TRKCOS option is not provisioned in table TRKFEAT. then no TRKCOS screening is performed. TRKCOS screening is only performed if the TRKCOS option is only provisioned for both an originating or terminating AXXESS agent.

Restrictions and limitations

If the TRKCOS option is not provisioned for an AXXESS trunk group, then TRKCOS screening is not performed for calls either originating on or terminating to that AXXESS trunk group.

ZONE call processing application

The Zone (ZONE) option identifies when loop-around IMT trunks equipped with external echo cancellers need to be used between the originating and terminating agents of the call. The zone option is applicable for both originating and terminating AXXESS trunk groups.

Call processing actions

The zone option identifies a zone value associated with the AXXESS trunk group.

Just before the call is presented to the terminating agent selected, the zone values of the originating and terminating trunk groups are accessed. If the sum of the zone values exceed the limit specified by the ECHO_DELAY_THRESHOLD office parameter, then an IMT loop-around route as identified by the ECHOCAN_IMT_OFRT_INDEX office parameter is inserted as the destination for the call. This IMT loop-around route is then equipped with external echo cancellers in order to apply echo cancellation for the call.

For more information on IMT loop-around functionality for external echo canceller usage, see the *UCS DMS-250 NT6X50 EC Integrated Echo Canceller Application Guide.*

Restrictions and limitations

Through available 6X50EC XPM hardware capability, external echo cancellers loops are no longer required on UCS DMS-250 switches. See the *UCS DMS-250 NT6X50EC Integrated Echo Canceller Application Guide.*

TRKSIG features and characteristics applications

Table TRKSIG provides signaling feature and characteristic specifications for originating and terminating AXXESS agents used for call processing. Most of the interface or signaling features and characteristics associated with the agent are used during the life of the call to properly execute call events in accordance to the specified agent terminal interface in use. Some feature and characteristics are stored in per-call data structures where they may be overwritten with other information as applicable.

Table TRKSIG replaces use of table TRKSGRP for AXXESS trunk group types.

Note: The Spectrum feature is a multi-application, high-speed platform that provides an Optical Carrier (OC)-3 interface to the DMS SuperNode. The first application of the Spectrum is trunk call processing. The Spectrum feature uses datafill in table TRKSIG to pass information to Spectrum for trunk applications such as echo cancellation and trunk maintenance. For more information on how Spectrum supports AXXESS trunks, see the *UCS DMS-250 Spectrum Reference Manual.*

Supported agents

Table TRKSIG is only provisioned for AXXESS trunk group types (both PTS and CCS7 signaling types).

Activation

Processing of TRKSIG table entries for the originating agent occurs during the “authorize origination” point in the call model, which occurs before FlexDial framework collectables are executed.

For the terminating trunk group, entries in these tables are processed during the “allocate terminator” point in the call after selection of the terminating trunk group member occurs.

For both the originating and terminating agent, the information in TRKSIG entries processed is retained for processing throughout the remainder of the call.

TRKSIG entries are for either PTS type signaling interfaces, or CCS7 type signaling interfaces.

Similar to TRKFEAT processing, certain TRKSIG options of the originating agent may be overwritten by subscriber number or CALLTYPE collectable processing. In contrast, because a number of terminating features can be specified by collectable processing before the terminating agent is allocated, these options (when retrieved from the TRKSIG table for terminating agents) can overwrite information previously set for the call by subscriber number or CALLTYPE collectable processing.

The focus of table TRKSIG is on the signaling characteristics of the agent. Therefore, most fields and options deal exclusively with agent signaling characteristics and not call features.

SIGTYPE call processing description

The signaling type field (SIGTYPE) identifies the type of signaling associated with the TRKSIG entry. Four types of signaling are supported for the SIGTYPE field:

- DS1
- FXS
- FXO
- CCS7

The first three types (DS1, FXS, FXO) refer to per-trunk signaling (PTS) type agents. For PTS agents, the bearer channel of the agent member also serves as the signaling channel, with signals send as A or AB bit transitions, and digits are sent and received as audible tones pulses.

The CCS7 signaling type refers to common channel signaling (CCS) type agents. For these agents, the signaling channel and bearer channel are two distinct physical resources. The signaling channel consists of a digital connection to a service switching point (SSP) or a service transfer point (STP), and signals are delivered as binary messages to the connecting switch. All information (including digit information) is delivered as binary data on the signaling channel (except where inband digit collection is specifically requested and required in order to collect information directly from the subscriber).

DS1 signaling refers to 4-wire A-bit signaling for PTS type agents. Application of the DS1 option is identical to use of the DS1SIG cardcode and STD signaling data in table TRKSGRP.

FXS signaling refers to 2-wire AB-bit signaling for PTS type agents, and may be either ground start (GS) or loop start (LS) associated signaling. Application of the FXS option is identical to the use of the FXSGS cardcode and STD signaling data for a ground start agent in table TRKSGRP, and the FXSLS cardcode and STD signaling data for a loop start agent. The ground start or loop start indicator is identified by the IPULSTYP and OPULSTYP fields.

FXO signaling refers to 2-wire AB-bit signaling for PTS type agents, and may be either GS or LS associated signaling. Application of the FXO option is identical to the use of the FXOGS cardcode and STD signaling data for a ground start agent in table TRKSGRP, and the FXOLS cardcode and STD signaling data for a loop start agent. The ground start or loop start indicator is identified by the IPULSTYP and OPULSTYP fields.

CCS7 signaling type identifies that the agent is using CCS signaling #7. Application of the CCS7 signaling type is identical to the use of the DS1SIG cardcode and C7UP signaling data type in table TRKSGRP.

PTS signaling call processing description

The TRKSIG table provides the following PTS signaling features and capabilities for call processing purposes:

- terminal signaling parameters
  - ISTARTSIG
  - DIALMODE
  - OSTARTSG
  - OPULSTYP
  - IODGTMR
  - TRKGRDTM
- digit Collection signaling parameters
  - IPULSTYP
  - PSEIZTMR
  - PDILTMR
  - MINRTMR
  - FDIGMASK
  - LDIGMASK

  - DIGMASK
  - TRMDIGIT
- signaling options
  - ACKWINK
  - ALTSEIX
  - ANSWFLTR
  - ATDANS
  - BCCOMPAT
  - CPIALLOW
  - DELIVER
  - DETDIAL
  - DIGSOUTP
  - ECSTAT
  - ESUPR
  - GLAREYD
  - IRINGCHK
  - MLTSTAGE
  - ODSCFLTR
  - ORIGFLTR
  - REMBSY
  - RETOFFHK
  - TDSCFLTR
  - SPMECIDX

CCS7 signaling call processing description

The TRKSIG table provides the following originating CCS7 features and capabilities for call processing purposes:

- terminal signaling parameters
  - PROTOCOL
  - ADJNODE
  - ISUPIDX
- signaling options
  - ABCNTRL
  - BCCOMPAT
  - COT
  - DELIVER
  - ECSTAT
  - ESUPR
  - GLARE
  - SPMECIDX

PTS terminal signaling call processing application

The purpose of the terminal signaling parameters for PTS agents is to define the basic signaling characteristics of the agent terminal.

Call processing actions

The basic terminal signaling characteristics of the agent are applied during various phases during the call model. The parameters and their meanings include:

- ISTARTSG

  The incoming start signal parameter identifies an acknowledgment signal returned in response to the incoming seizure signal, and is applicable when an origination attempt occurs on the terminal.

  For WK and DD start signal types which involve transmission of a wink signal, the timer values used in the generation of the wink signal are taken from the following office parameters:

  - PRE_SND_WK_DD_TIME

    Found in table OFCSTD, this office parameter identifies the pre-wink duration that must expire before the on-hook to off-hook transition beginning the wink signal is performed.

  - — SND_MF_WK_TIME

    Found in table OFCSTD, this office parameter identifies the duration of the wink signal. At the expiration of this timer, the off-hook to on-hook transition occurs completing the wink signal.

- DIALMODE

  The dialing mode field identifies if digits received are customer dialed or machine-dialed, but provides no other useful function for call processing purposes.

- OSTARTSG

  The outgoing start signal parameter identifies the type of acknowledgment signal that is expected to be received in response to an outgoing seizure signal. The OSTARTSG parameter is applicable when the call is being presented to the terminating switch.

- OPULSTYP

  The outpulse type parameter identifies the tone pulsing type used for outpulsing digits to the terminating switch. The OPULSTYP parameter is applicable when the call is being presented to the terminating switch.

- OIDGTMR

  The outgoing interdigit timer identifies the amount of time to wait between outpulsing of each digit. The OIDGTMR parameter is applicable when the call is being presented to the terminating switch.

- TRKGRDTM

  The trunk guard time is used in two areas of processing for the terminal:

  - — The guard time identifies the amount of time within which the identified outgoing start signal must be received from the terminating switch. If the start signal is not received within the time period, then an event such as glare or a call failure may be occurring on the agent. In this manner the parameter is applicable when the call is being presented to the terminating switch.
  - — The guard time also identifies the amount of time after the agent is deallocated from a call before it may be seized for another call. The value provisioned in this parameter is designed to allow the disconnect to be propagated to all network switches before an attempt to originate the call on the resources is performed. In this manner the parameter is applicable when the call is disconnected.

Interactions

The terminal signaling characteristics are used to control bearer channel facility messaging for the agent. This involves handling originations as well as outpulsing.

Restrictions and limitations

The following restrictions and limitations apply to the PTS terminal signaling call processing application:

- The terminal signaling characteristics as defined in table TRKSIG are only applicable for AXXESS trunk groups.
- TRKSIG signaling characteristics are identical in functionality to table TRKSGRP defined terminal signaling parameters. For a field-to-field comparison, see “Table TRKGRP table mapping” on page 12-1.

PTS digit collection signaling call processing application

The purpose of the digit collection signaling parameters for PTS agents is to define the digit collection signaling characteristics of the agent terminal.

Call processing actions

The digit collection signaling characteristics of the agent are applicable for the agent during the collect information point in the call model, as collectables (in particular, digit collectables) are being executed. The parameters and their meanings include:

- IPULSTYP

  The incoming pulse type identifies the pulsing method for receiving inband digits. This method may either be MF or DTMF.

- PSEIZTMR

  The permanent signal timer identifies the amount of time within which the first digit of a request must be received.

- PDILTMR

  The partial dial timer identifies the amount of time within which each digit must be received, up until the minimum number of digits in the request have been received.

- MINRTMR

  The minimum digits received timer identifies the amount of time within which each digit must be received after the minimum number of digits in the request have been received.

- FDIGMASK

  The first digit mask set identifies the allowable first digits for MF pulse type signaling, and marks the beginning of the MF digit stream. Any MF digits received before a digit identified by the first digit mask are ignored.

- LDIGMASK

  The last digit mask identifies the digit which is considered the last digit to be received in the digit stream, and marks the end of the MF stream. Once the last digit is received, all digits received since the first digit are reported to the CM.

- DIGMASK

  The DTMF digit mask identifies allowable DTMF digits that may be received from the subscriber. Any digit received that is not included in the digit mask is ignored and not processed by the digit collectable.

- TRMDIGIT

  The terminating digit field identifies the digit that indicates that the subscriber has entered all the digits that they are going to enter in order to fulfil the collection request.

Interactions

The digit collection signaling parameters can be modified through use of the Signaling (SIG) collectable. See “Signaling collectable (SIG)” on page 16-1 for a description of the functionality of the SIG collectable.

The general application of inband digit collection requires the specification of the digit collection signaling parameters. See “General application of inband digit collection” on page 16-51 for more information on the general application of inband digit collection.

Restrictions and limitations

Inband digit collection signaling parameters do not exist in TRKSIG table provisioning for CCS7 or out-of-band agents. Therefore, a SIG collectable must be used to specify the inband digit collection signaling parameters before inband digit collection can be performed for the agent.

PTS ACKWINK call processing application

The Acknowledgment Wink (ACKWINK) option for DS1 signaling agents indicates that an acknowledgment wink is to be transmitted upon completion of the interaction with the originating agent.

Note: This happens with special consideration. For instance, using a SIG collectable to switch from MF to DTMF collection causes the acknowledgment wink to be transmitted. Overall the interaction has not yet finished, but the MF portion of the interaction has completed, and therefore the wink is transmitted.

Call processing actions

The acknowledgment wink may be transmitted only once for the call through execution of one of the following call processing triggers:

- during execution of a SIG collectable, where the IPULSTYP is transitioning from MF to DTMF tone collection on the agent
- after completion of the FlexDial interaction, prior to terminating the call to a set treatment
- after completion of the FlexDial interaction, following allocation and seizure of the terminating trunk agent member

The pre-wink duration timer value used for the wink signal transmitted is taken from the PRE_SND_WK_DD_TIME office parameter (table OFCSTD), and the wink duration timer value used is taken from one of two office parameters:

- SND_MF_WK_TIME (table OFCSTD) for national calls
- EA_INT_WINK_DUR (table OFCVAR) for international calls

The ACKWINK option must be used for DS1 AXXESS agents executing a FGD protocol agent interaction.

Interactions

The ACKWINK option is only applicable for DS1 type signaling. See “SIGTYPE call processing description” on page 21-2.

Restrictions and limitations

The following restrictions and limitations apply to the PTS ACKWINK call processing application:

- The ACKWINK option is only applicable for DS1 type signaling.
- The ACKWINK option must be used for DS1 AXXESS agents executing a FGD protocol agent interaction.

PTS ALTSEIZ call processing application

The Alternate Seizure (ALTSEIZ) option indicates that a different AB bit seizure configuration is used for the originating or terminating trunk group.

Call processing actions

Use of the ALTSEIZ option only applies for FXS and FXO type signaling, and identifies that a different AB bit seizure configuration is used for incoming and outgoing seizures. With the ALTSEIZ option, the AB bit pattern used for seizure is AB=10, where normally the seizure bit pattern is AB=00.

The ALTSEIZ option is therefore applicable to the call during origination of the call (for originating agents), and when the call is presented to the next network switch (for terminating agents).

Interactions

The ALTSEIZ option is only applicable for FXS and FXO type signaling. See “SIGTYPE call processing application description” on page 21-2.

For incoming seizure related options, see a description of the IRINGCHK option on page 21-19.

Restrictions and limitations

The following restrictions and limitations apply to the PTS ALTSEIZ call processing application:

- The ALTSEIZ option identifies that a different AB-bit seizure configuration is used for the agent. Since DS1 signaling only involves A-bit signaling, the ALTSEIZ option is not applicable for DS1 signaling type agents.
- The ALTSEIZ option is only applicable for DAL and ONAL type terminations.

PTS ANSWFLTR call processing application

The Answer Filter (ANSWFLTR) option identifies the amount of time that the off-hook signal must be received on the terminating agent before the signal is accepted as an answer indicator.

Call processing actions

Processing the answer filter option identifies a timer value in 10 ms increments that filters the answer signal from the terminating agent. The off-hook signal must be received for the identified period of time before answer is recognized and processed for the call.

If the off-hook signal does not remain set for the amount of time specified, then answer is not processed for the call.

When answer occurs, the answer point in the call is processed, and the answer signal is propagated to the originating agent.

Interactions

There are no specific interactions for the ANSWFLTR option.

Restrictions and limitations

There are no specific call processing restrictions for the answer filter option.

PTS ATDANS call processing application

The Audio Tone Detector Answer (ATDANS) option identifies that an audio tone detector (ATD) is allocated to detect answer for the call. The ATD receiver is programmed to detect audio (voice) signals, and when the audio signals are detected, answer is reported for the call.

An answer signal detected using the ATD receiver is termed a software answer. In contrast, answer detected through a hook state change (such as going off-hook) is termed a hardware answer. ATD receivers are typically only used when the receipt of a hardware answer signal is not available or not reliable.

Call processing actions

Processing the ATD answer option identifies a timer value that delays activation of the ATD receiver. This timer value is specified by the DELAYTMR field that contains a value between 100 ms and 16 seconds.

When the call is presented to the next network switch and the ATDANS option is present for the terminating trunk group, then an ATD receiver is allocated and connected to the selected member of the terminating trunk group. After a period of time identified by the DELAYTMR field, the ATD receiver is activated to detect software answer.

After a specified period of time as identified by the office parameter HI_AND_DRY_TIMEOUT, the ATD receiver either reports answer detected or no answer detected. At this point in time, the ATD receiver is deallocated from the call and the network connection to the ATD receiver is released.

If answer is detected, then the ANSTYPE field is the CDR is captured with a value indicating software answer, and the appropriate answer signal is propagated to the originating agent.

If answer is not detected, then the terminating agent member is released from the call and ATD timeout treatment is set and applied to the call.

Interactions

For information on the ANSTYPE CDR field, see the *UCS DMS-250 Billing Records Application Guide.*

Restrictions and limitations

ATD answer capability requires available MTM or STM ATD receiver resources.

PTS BCCOMPAT call processing application

The Bearer Compatibility (BCCOMPAT) option identifies the bearer compatibility for the trunk group. See “PTS BCCOMPAT call processing application” on page 18-4 for a description of the call processing application of the BCCOMPAT option.

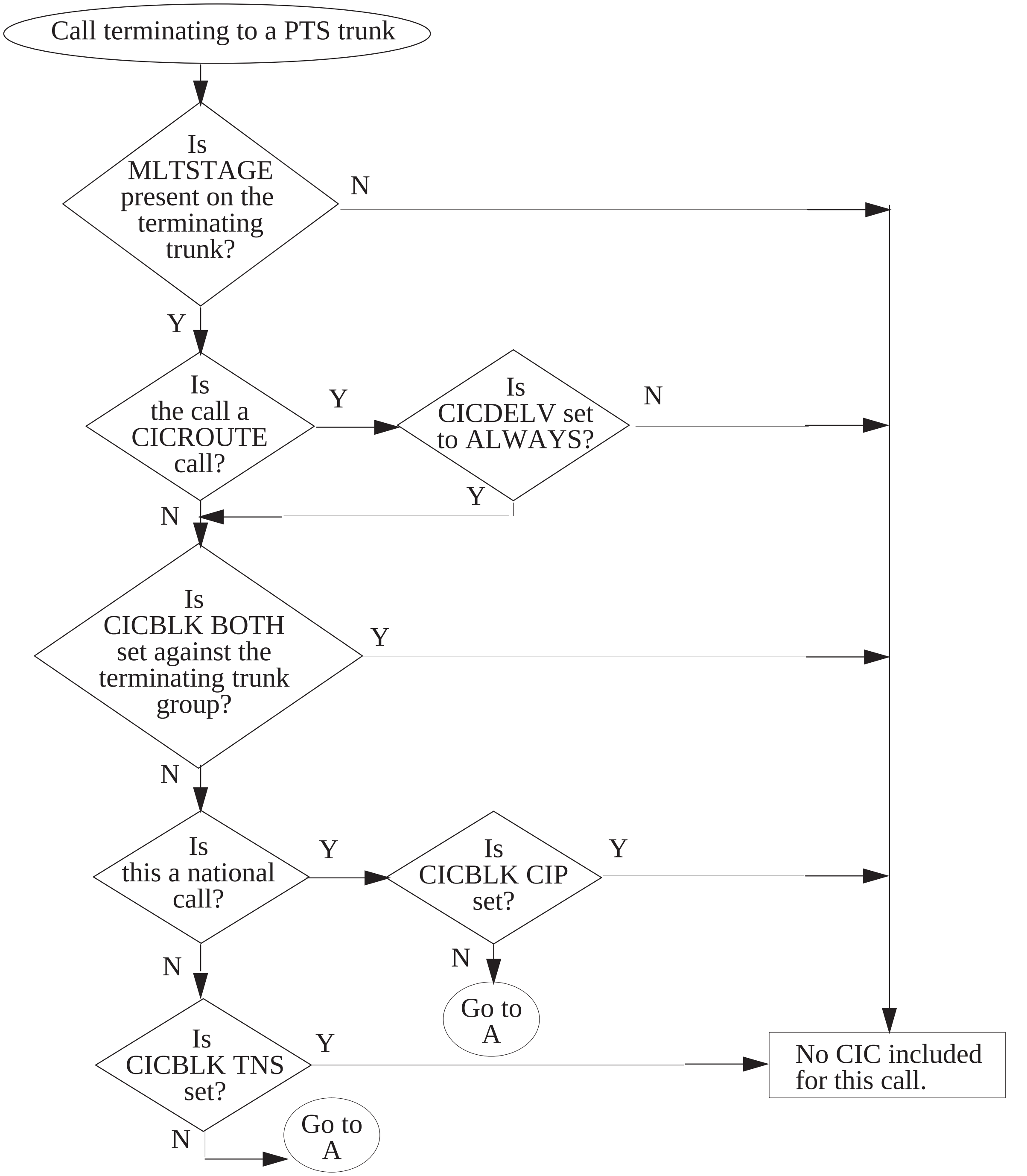
PTS CICBLK call processing application

The CICBLK option indicates that the terminating trunk does not deliver CIC information. If the CICBLK option is present, the UCS DMS-250 switch blocks the delivery of the CIC on the terminating trunk.

Call processing actions

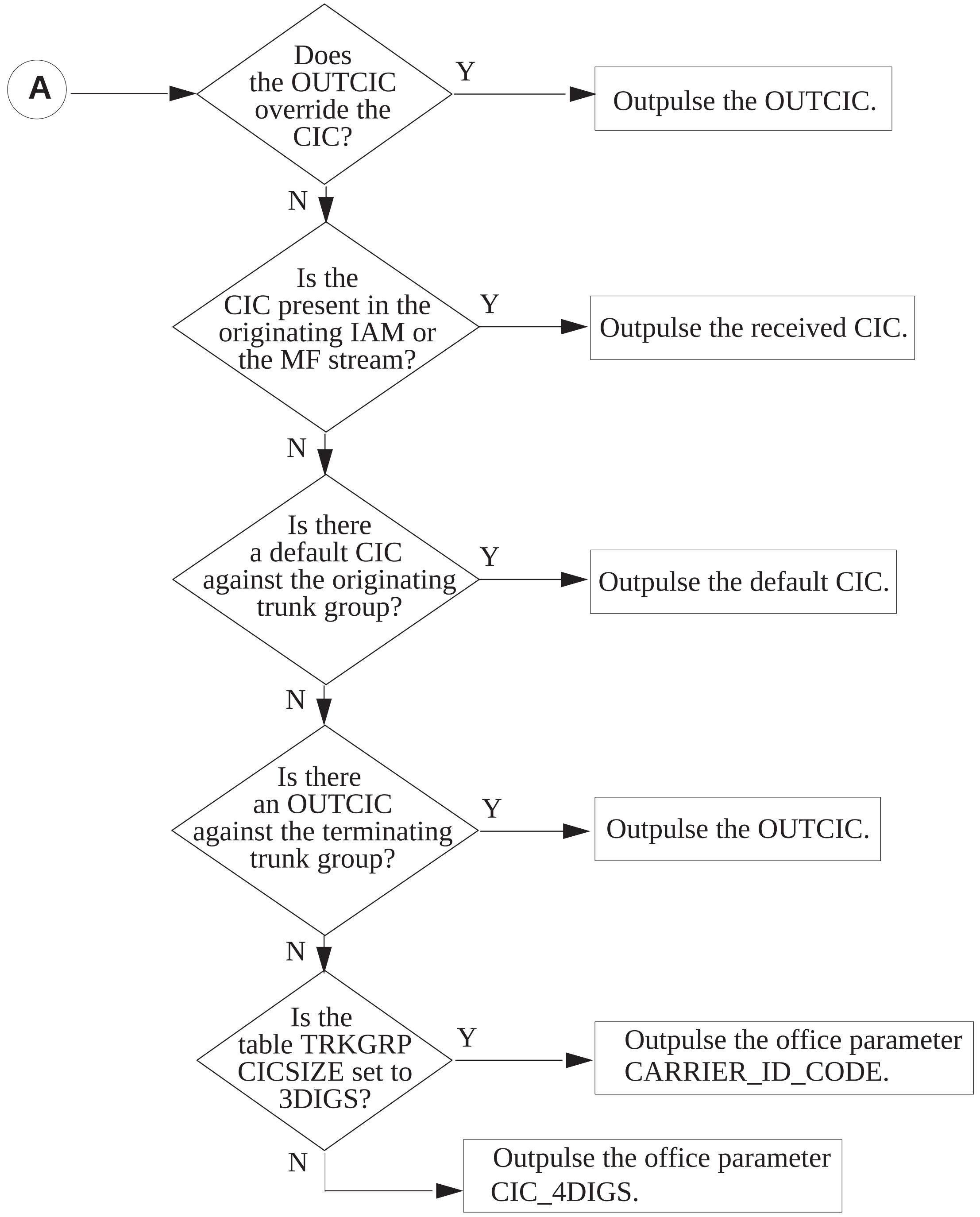
Figure 21-1, PTS CICBLK call processing actions, shows the flow of a call when the CICBLK option is present on a terminating AXXESS agent that is using PTS signaling.

Figure 21-1
PTS CICBLK call processing actions


Call terminating to a PTS trunk
Is MLTSTAGE present on the terminating trunk?
N
Y
Is the call a CICROUTE call?
Y
Is CICDELV set to ALWAYS?
N
Y
N
Is CICBLK BOTH set against the terminating trunk group?
Y
N
Is this a national call?
Y
Is CICBLK CIP set?
Y
N
Go to A
N
Is CICBLK TNS set?
Y
No CIC included for this call.
N
Go to A

—continued—

Figure 21-1 (continued)
PTS CICBLK call processing actions


A
Does the OUTCIC override the CIC?
Y
Outpulse the OUTCIC.
N
Is the CIC present in the originating IAM or the MF stream?
Y
Outpulse the received CIC.
N
Is there a default CIC against the originating trunk group?
Y
Outpulse the default CIC.
N
Is there an OUTCIC against the terminating trunk group?
Y
Outpulse the OUTCIC.
N
Is the table TRKGRP CICSIZE set to 3DIGS?
Y
Outpulse the office parameter CARRIER_ID_CODE.
N
Outpulse the office parameter CIC_4DIGS.

–end–

PTS CICSIZE call processing application

The CICSIZE option specifies the number of digits to outpulse to the terminating trunk agency.

PTS CPIALLOW call processing application

The Calling Party Identification Delivery (CPIALLOW) option specifies for the terminating agent that the calling party number is to be delivered to the next network switch. The CPIALLOW option is only applicable for DAL type terminations (the TRKGRP OGRPTYP field must be set to DAL).

Note: For AXXESS agent originations, the calling party number must be set for the call through execution of a SUBR collectable. When the calling party number is not identified as thus, no CPI digits are outpulsed.

When the call is presented to the next network switch for DAL type terminations and the CPIALLOW option is provisioned for the terminating trunk group, then the calling party number is included in the digits outpulsed.

For interactions with AXXESS agent originations, see “OGRPTYP field definition” on page 19-3.

PTS DELIVER call processing application

The Calling Party Number Deliver (DELIVER) option specifies the calling party subscriber number delivery status or method for the call. See “DELIVER call processing application” on page 18-13 for a description of the call processing application of the DELIVER option.

PTS DETDIAL call processing application

The Detect Dialtone (DETDIAL) option indicates that dialtone must be detected as the proceed-to-send signal in order to outpulse digits on the terminating agent. The DETDIAL option is only applicable for DAL and ONAL type terminations (the TRKGRP OGRPTYP field must be set to DAL or ONAL).

Call processing actions

The DETDIAL option simply identifies that dialtone must be detected as a proceed-to-send signal in order to outpulse digits on the terminating agent.

When the call is presented to the next network switch and the DETDIAL option is provisioned for the terminating trunk group member, then an ATD receiver is allocated and connected to the selected member of the terminating trunk group. The ATD receiver is then activated to detect the dialtone proceed-to-send signal.

The dialtone proceed-to-send signal must be received by the ATD receiver within the timeout value specified by the FXO_DIALTONE_DETECTION_TIMEOUT office parameter. If dialtone is detected, then the identified digits are outpulsed to the next network switch.

If dialtone is not detected within the timeout specified, then the terminating trunk member is released from the call, and either a member advance or route advance occurs as appropriate.

Interactions

The DETDIAL option may be used in conjunction with the ATDANS option in using an ATD receiver during the present call event processing. See “PTS ATDANS call processing application” on page 21-11.

Restrictions and limitations

DETDIAL capability requires available MTM or STM ATD receiver resources.

PTS DIGSOUTP call processing application

The Digits To Outpulse (DIGSOUTP) option identifies the number of called party address digits that are outpulsed as the call is presented to the next network switch. The DIGSOUTP option is only applicable for DAL type terminations (the TRKGRP OGRPTYP must be set to DAL).

Call processing actions

Processing the DIGSOUTP option identifies the allowable maximum number of digits that may be outpulsed to the next network switch.

In presenting the call to the next network switch, the called party address digits are first formatted for delivery. For DAL type terminations, the number of called party address digits are chopped to fit within the maximum limit identified by the DIGSOUTP field. Digits are removed from the beginning of the digit register. Once the proper number of digits have been identified to be delivered, they are then presented to the next network switch.

Figure 21-2 illustrates a DIGSOUTP option example.

Figure 21-2
DIGSOUTP option example

Called Party Address Digits for Call

2	1	4	5	5	5	5	5	1	1

Apply DIGSOUTP option where DIGSOUTP = 7.

Called Party Address Digits Delivered

5	5	5	5	5	1	1

Interactions

The address collectable receives and processes called party address digits. See “Address digit collectable (ADDR/ADDRPRM)” on page 16-127.

Restrictions and limitations

The application of the DIGSOUTP option truncates the number of available called party address digits in order to fit within the allowable maximum. Digits are removed from the beginning of the digit register.

PTS ECSTAT call processing application

The echo canceller status option (ECSTAT) identifies the availability of echo cancellers for members of the trunk group, and provides the following information for call processing purposes:

- status

  Identifies if echo cancellers are unequipped (UNEQ), equipped on the XPM (INTERNAL), equipped through external hardware (EXTERNAL), or equipped on the XPM but without 2100Hz tone control (INNOTONE).

- NSMATCH

  For INTERNAL or INNOTONE, this boolean field identifies if noise matching is activated.

- AUTOON

  For INTERNAL, this boolean field indicates that automatic re-enabling control is activated.

For more information on the application of echo cancellers, see the *UCS DMS-250 NT6X50EC Integrated Echo Canceller Application Guide*.

PTS ESUPR call processing application

The Echo Suppressor (ESUPR) option indicates that members of the trunk group have XPM hardware echo suppressors activated.

Processing the ESUPR option identifies if the echo suppression is set to half or full suppression. Half suppression identifies that just one echo suppressor is located at the near end of the trunk group (for either the transmit or receive side). Full suppression that both echo suppressors (for both transmit and receive channels) are located at the near end of the trunk group.

PTS GLAREYD call processing application

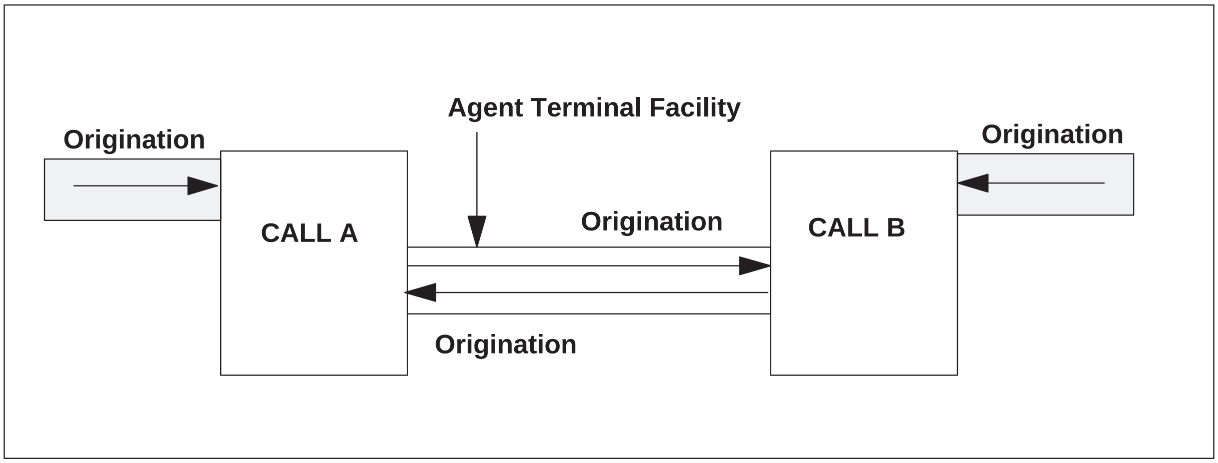
The Glare Yield (GLAREYD) option indicates that members of this trunk group yield (instead of standing) to glare situations. The GLAREYD option is only applicable for terminations on members of the trunk group.

Call processing actions

Processing the GLAREYD option identifies that the members of the trunk group yield in glare situations.

When the call is presented to the next network switch, the terminating agent member is allocated and a seizure signal is delivered to the connected switch through the terminating agent facility. Glare occurs when the terminating agent facility is seized simultaneously by each connected switch, as shown in Figure 21-3.

Figure 21-3
Glare Situation Example



A trunk group member yielding to glare removes the seizure request signal and is deallocated from the call. The incoming seizure request signal is then available to proceed as a new call origination on the switch.

For the call of the yielding trunk group member, after the yielding member is deallocated a member advance or route advance occurs as appropriate.

A trunk group member standing to glare does not remove the seizure request signal, but instead waits as directed for the appropriate seizure acknowledgment signal.

Interactions

There are no specific call processing interactions for the GLAREYD option.

Restrictions and limitations

The following restrictions and limitations apply to the PTS GLAREYD call processing application:

- In glare situations, a trunk group member yielding to glare is deallocated from the current call and either a member advance or route advance occurs as appropriate.
- If both switches are instructed to stand in glare situations, then seize failure occurs the calls of each switch, due to timeout of the required seizure acknowledgment signal.

PTS IRINGCHK call processing application

The Incoming Ringing Check (IRINGCHK) option identifies that an incoming ringing signal must also be received in order for an incoming seizure signal to be properly recognized. The IRINGCHK option is only applicable for ONAL type terminations (the TRKGRP OGRPTYP field must be set to ONAL).

Call processing actions

The IRINGCHK option identifies that the incoming ringing signal must be received on origination attempts.

The IRINGCHK option is applicable during the origination attempt on the trunk group member. If the IRINGCHK option is active for the trunk group members, then the proper AB-bit signal identifying seizure and ringing must be received on the originating agent facility. If the proper signal is not received, then the origination attempt is not recognized.

Interactions

The IRINGCHK option is only applicable for FXO type signaling. See “SIGTYPE call processing description” on page 21-2.

For outgoing seizure related options, see a description of the ALTSEIZ option in “PTS ALTSEIZ call processing application” on page 21-10.

Restrictions and limitations

The following restrictions and limitations apply to the PTS IRINGCHK call processing application:

- If the ringing signal is not received in addition to the seizure signal, then seizure of the agent facility is not recognized and processed.
- The IRINGCHK option is only applicable for ONAL type terminations.

PTS MLTSTAGE call processing application

The Multi-stage (MLTSTAGE) option indicates that multiple stage outpulsing is to be used in presenting the call to the next network switch. Multiple stage capability is only supported for PTS EANT type terminations (the TRKGRP OGRPTYP field must be set to EANT).

For a description of multi-stage outpulsing functionality, see the *UCS DMS-250 Feature Group D (FGD) Application Guide.*

PTS ODSCFLTR call processing application

The Originating Disconnect Filter (ODSCFLTR) option identifies the amount of time that the on-hook signal must be received on the originating agent before the signal is accepted as a calling party disconnect indicator.

Call processing actions

Processing the originating disconnect filter option identifies a timer value in 10 ms increments that filters the disconnect signal from the originating agent. The on-hook signal must be received for the identified period of time before originating party disconnect is recognized and processed for the call.

If the on-hook signal does not remain set for the amount of time specified, then a disconnect signal is not processed for the call.

When calling party disconnect occurs, the billing record is generated and the call is released.

Interactions

There are no specific interactions for the ODSCFLTR option.

Restrictions and limitations

There are no specific call processing restrictions for the originating disconnect filter option.

PTS ORIGFLTR call processing application

The Origination Filter (ORIGFLTR) option identifies the amount of time that the off-hook signal must be received on the originating agent before the signal is accepted as an origination attempt indicator.

Call processing actions

Processing the origination filter option identifies a timer value in 10 ms increments that filters the off-hook signal from the originating agent. The off-hook signal must be received for the identified period of time before an origination attempt is recognized and processed for the call.

If the off-hook signal does not remain set for the amount of time specified, then no origination attempt is processed for the agent.

Interactions

There are no specific interactions for the ORIGFLTR option.

Restrictions and limitations

There are no specific call processing restrictions for the origination filter option.

PTS OUTCIC call processing application

The OUTCIC option determines the CIC digits to outpulse over the terminating trunk.

Call processing actions

The UCS DMS-250 switch determines which CIC value to outpulse based on the following precedences:

1. The switch outpulses the OUTCIC value. (The OUTCIC OVERRIDE option is set to Y.)
2. The switch outpulses the received CIC value.
3. The switch outpulses the default CIC (DEFCIC) value. (The OUTPULSE option is set to Y.)
4. The switch outpulses the OUTCIC value. (The OUTCIC OVERRIDE is set to N, otherwise the switch would have already outpulsed the OUTCIC value.)
5. The switch outpulses the three-digit or four-digit CIC in the CARRIER_ID_CODE or CIC_4DIGS office parameters, respectively.

PTS REMBSY maintenance application

The Remote Make Busy (REMBSY) option indicates if members of the trunk group provide RMB (remote man busy) display status. The REMBSY option is a maintenance application and not a call processing application.

For REMBSY capable agents, a trunk group member for the switch enters the RMB state if the trunk group member on the connected switch enters a man busy (MB) state.

AXXESS agents with CCS7 type signaling are always REMBSY capable.

PTS RETOFFHK call processing application

The Return Off-hook (RETOFFHK) option identifies when an answer signal is to be propagated to the originating agent.

Call processing actions

Processing the RETOFFHK option identifies the following available answer propagation indicators for the terminating agent.

- NIL

  The NIL option identifies that an answer signal is never propagated to the originating agent for the call.

- After Outgoing Seizure

  During the present call state, the answer signal is propagated to the originating agent immediately after seizure of the terminating agent.

- After Outpulsing

  During the present call state, the answer signal is propagated to the originating agent immediately after outpulsing of all digits to the next network switch has occurred.

When the RETOFFHK option is not used, then an answer signal is propagated under normal circumstances (after answer has been detected on the terminating agent). If answer has already been propagated for the call (potentially by the SNDSIG collectable), then any value identified by the RETOFFHK option is ignored for the call.

The SNDSIG collectable must be used when it is desirable to return an answer signal at some point during the setup of the call. Use of the SNDSIG collectable is not affected by any setting of the RETOFFHK option, including the NIL value.

Interactions

The SNDSIG collectable may be used to deliver an answer signal to the originating agent. For a description of the SNDSIG collectable, see “Send Signal collectable (SNDSIG)” on page 16-5.

Restrictions and limitations

Any value set by the RETOFFHK option is not applicable for the call if an answer signal has already been delivered to the originating agent (potentially through use of the SNDSIG collectable).

PTS TDSCFLTR call processing application

The Terminating Disconnect Filter (TDSCFLTR) option identifies the amount of time that the on-hook signal must be received on the terminating agent before the signal is accepted as a disconnect indicator.

Call processing actions

Processing the terminating disconnect filter option identifies a timer value in 10 ms increments that filters the disconnect signal from the terminating agent. The on-hook signal must be received for the identified period of time before disconnect is recognized and processed for the call.

If the on-hook signal does not remain set for the amount of time specified, then disconnect is not processed for the call.

When disconnect occurs, the disconnect point in the call is processed, and the release signal is propagated to the originating agent.

Interactions

There are no specific interactions for the TDSCFLTR option.

Restrictions and limitations

There are no specific call processing restrictions for the terminating disconnect filter option.

PTS SPMECIDX call processing application

The Spectrum echo canceller (ECAN) Index (SPMECIDX) option provisions a Spectrum trunk with a Spectrum ECAN. The SPMECIDX option is associated with an integer value that indexes table Spectrum ECAN (SPMECAN) for the selection of control parameters.

If the AXXESS agency is serviced by an XPM, the presence of an external ECAN is indicated by setting the TRKSIG ECSTAT to “EXTERNAL,” and the presence of an internal ECAN is indicated by setting the TRKSIG ECSTST to “INTERNAL” or “INNOTONE.” If the agency is serviced by Spectrum, the TRKSIG SPMECIDX option is used to index the desired control parameters in table SPMECAN, and thus associates control parameters with Spectrum ECAN.

Note: The Spectrum feature is a multi-application, high-speed platform that provides an Optical Carrier (OC)-3 interface to the DMS SuperNode. The first application of the Spectrum is trunk call processing. The Spectrum feature uses datafill in table TRKSIG to pass information to Spectrum for trunk applications such as echo cancellation and trunk maintenance. For more information on how Spectrum supports AXXESS trunks, see the *UCS DMS-250 Spectrum Reference Manual.*

CCS7 Terminal Signaling call processing application

The purpose of the terminal signaling parameters for CCS7 agents is to define the basic signaling characteristics of the agent terminal.

Call processing actions

The basic terminal signaling characteristics of the agent are applied during various phases during the call model. The three parameters and their meanings include:

- PROTOCOL

  The protocol parameter identifies the supported type of Q.764 protocol for the SS7 agent. This protocol value specifies formatting characteristics for the Q.764 messages. Currently only the UCP protocol is supported for AXXESS agents.

- ADJNODE

  The adjacent node field identifies the type of node the SS7 agent is connected to.

- ISUPIDX

  The ISUP index parameter identifies an index value which is used in the control and formatting of ISUP Q.764 messages.

Interactions

The CCS7 terminal signaling characteristics affect the format and control of Q.764 messages created and delivered for the SS7 agent.

Restrictions and limitations

The following restrictions and limitations apply to the CCS7 terminal signaling call processing application:

- The terminal signaling characteristics as defined in table TRKSIG are only applicable for AXXESS trunk groups.
- TRKSIG signaling characteristics are identical in functionality to table TRKSGRP defined terminal signaling parameters. For a field to field comparison, see “TRKSGRP table mapping” on page 12-8.

CCS7 ABCTRL signaling application

The A-bit Control (ABCTRL) option identifies if A-bit bearer channel signaling is active for the trunk group members. A-bit bearer channel signaling is performed in addition to the Q.764 message based signaling that occurs for the agent.

The ABCTRL option performs no specific function for call processing purposes. When set, the agent terminal controller (XPM) maintains proper delivery of the A-bit bearer channel signal as the Q.764 messages are sent and received.

CCS7 BCCOMPAT call processing application

The Bearer Compatibility (BCCOMPAT) option identifies the bearer compatibility for the trunk group. See “BCCOMPAT call processing application” on page 18-4 for a description of the call processing application of the BCCOMPAT option.

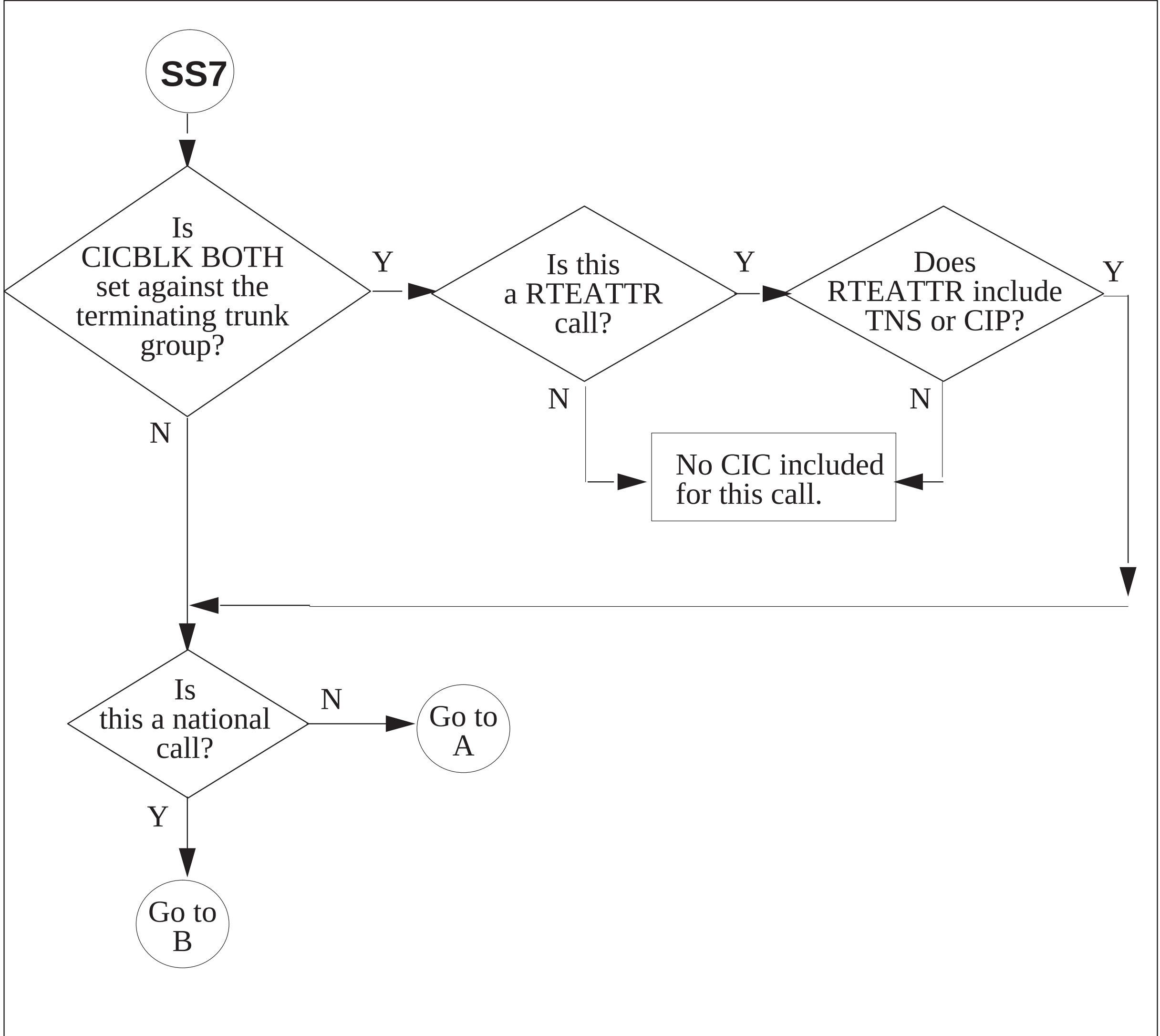
CCS7 CICBLK call processing application

The CICBLK option indicates that the CCS7 terminating trunk does not deliver CIC information.

Call processing actions

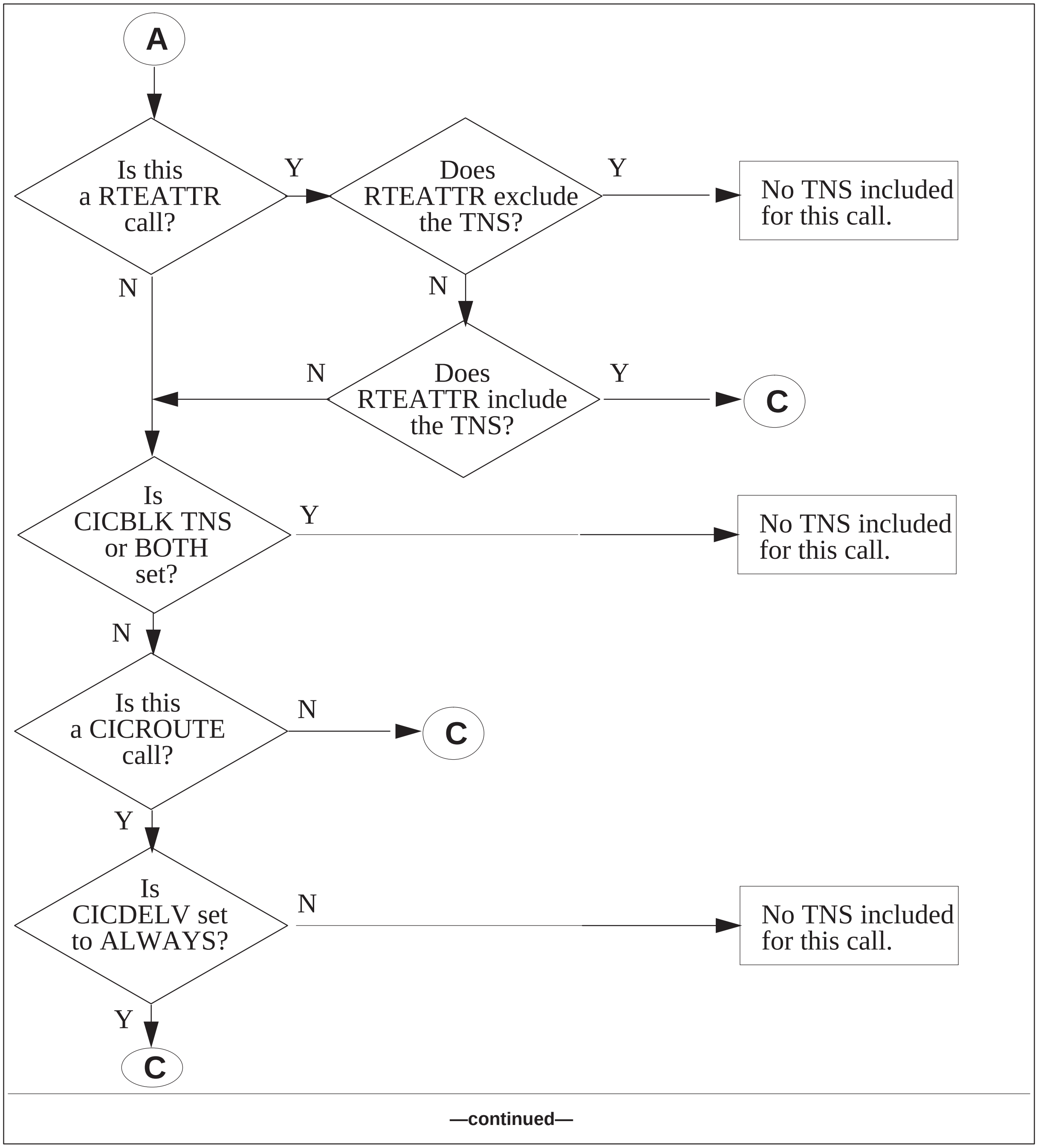
Figure 21-4, CCS7 CICBLK call processing actions, shows the flow of a call when the CICBLK option is present on a terminating AXXESS agent that is using CCS7 signaling.

Figure 21-4
CCS7 CICBLK call processing actions


SS7
Is CICBLK BOTH set against the terminating trunk group?
Y
Is this a RTEATTR call?
Y
Does RTEATTR include TNS or CIP?
Y
N
N
N
No CIC included for this call.
Is this a national call?
N
Go to A
Y
Go to B

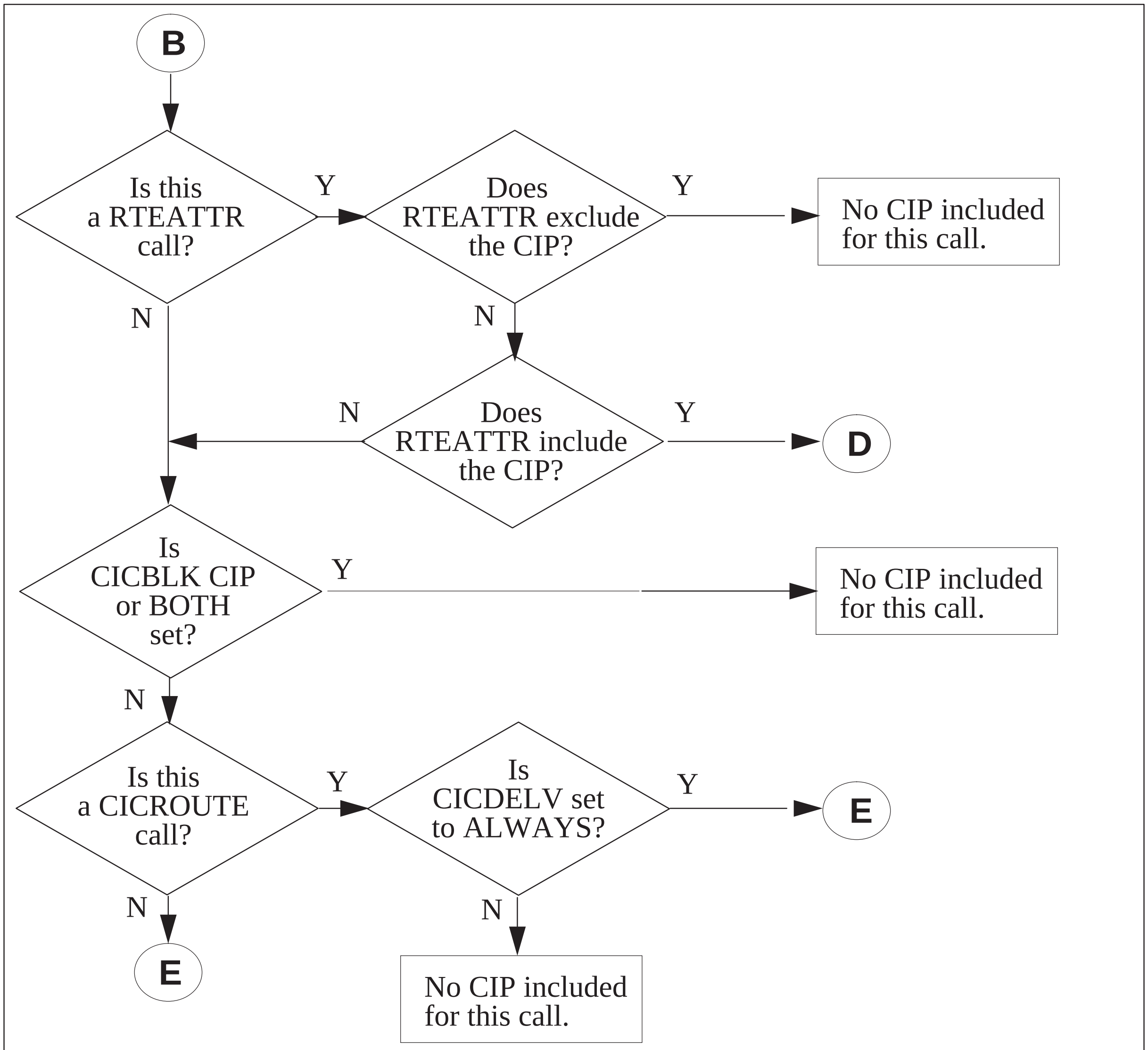
—continued—

Figure 21-4 (continued)
CCS7 CICBLK call processing actions


A
Is this a RTEATTR call?
Y
Does RTEATTR exclude the TNS?
Y
No TNS included for this call.
N
N
N
Does RTEATTR include the TNS?
Y
C
Is CICBLK TNS or BOTH set?
Y
No TNS included for this call.
N
Is this a CICROUTE call?
N
C
Y
Is CICDELV set to ALWAYS?
N
No TNS included for this call.
Y
C

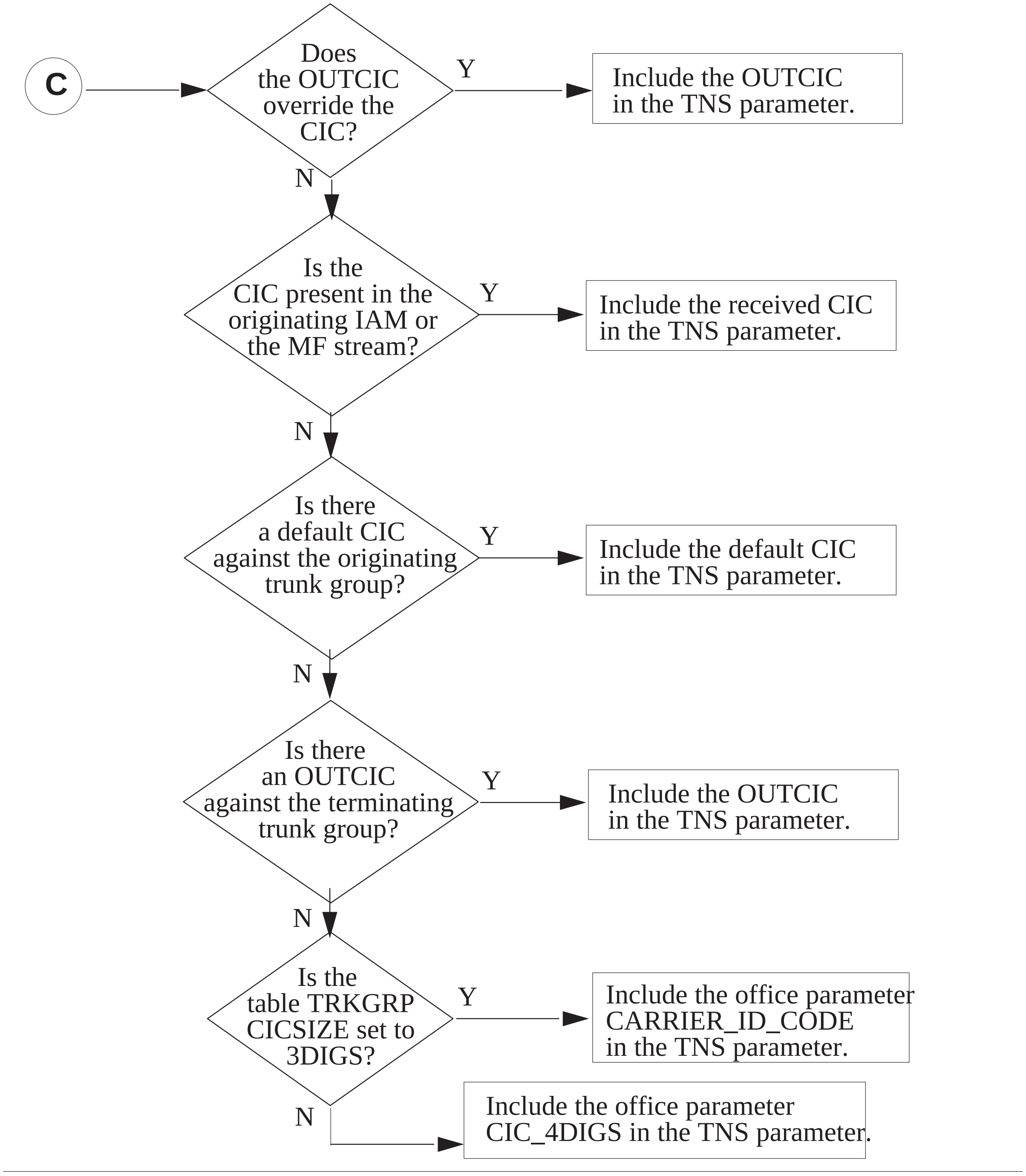
—continued—

Figure 21-4 (continued)
CCS7 CICBLK call processing actions


B
Is this a RTEATTR call?
Y
Does RTEATTR exclude the CIP?
Y
No CIP included for this call.
N
N
N
Does RTEATTR include the CIP?
Y
D
Is CICBLK CIP or BOTH set?
Y
No CIP included for this call.
N
Is this a CICROUTE call?
Y
Is CICDELV set to ALWAYS?
Y
E
N
N
E
No CIP included for this call.

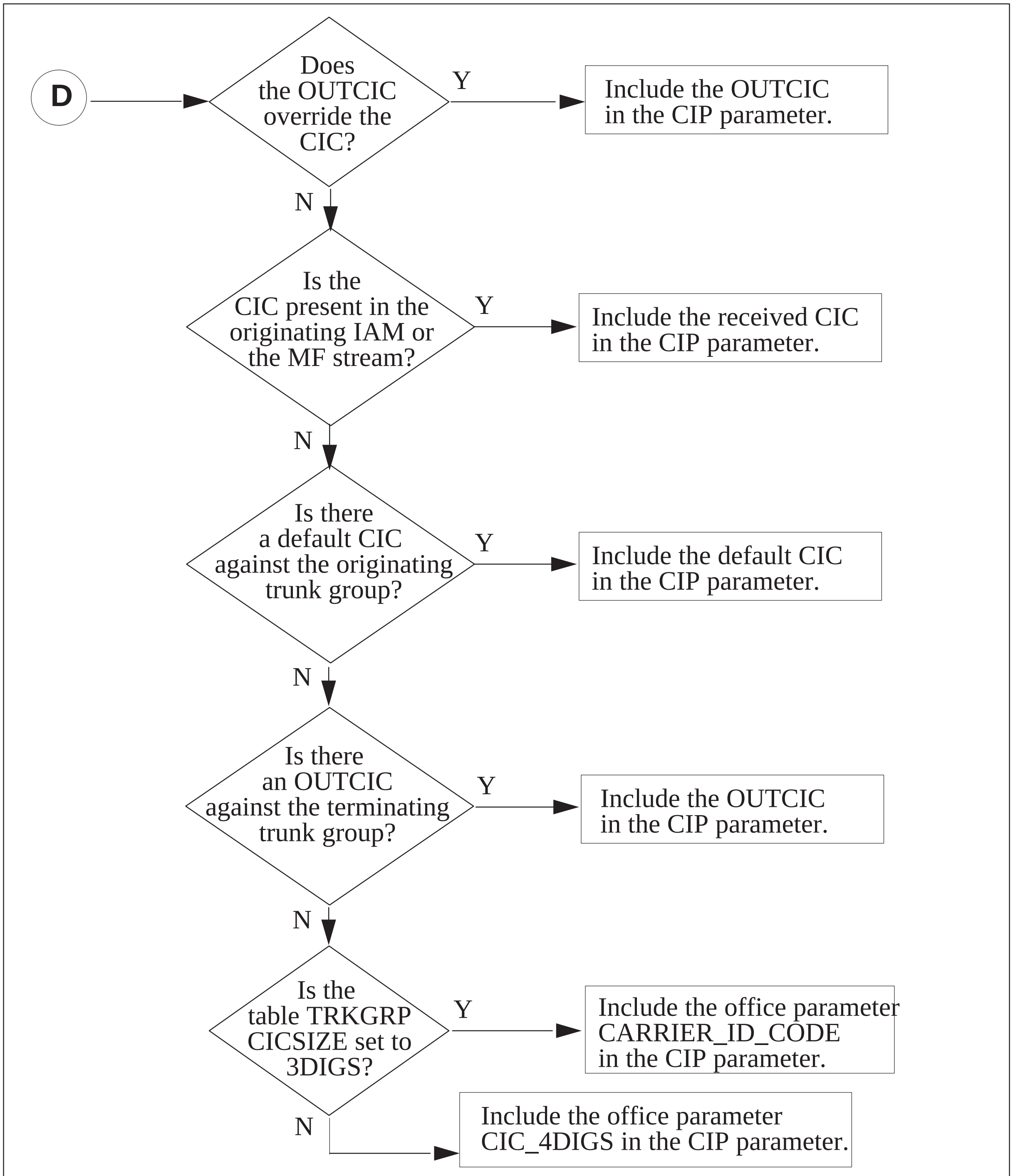
—continued—

Figure 21-4 (continued)
CCS7 CICBLK call processing actions


C
Does the OUTCIC override the CIC?
Y
Include the OUTCIC in the TNS parameter.
N
Is the CIC present in the originating IAM or the MF stream?
Y
Include the received CIC in the TNS parameter.
N
Is there a default CIC against the originating trunk group?
Y
Include the default CIC in the TNS parameter.
N
Is there an OUTCIC against the terminating trunk group?
Y
Include the OUTCIC in the TNS parameter.
N
Is the table TRKGRP CICSIZE set to 3DIGS?
Y
Include the office parameter CARRIER_ID_CODE in the TNS parameter.
N
Include the office parameter CIC_4DIGS in the TNS parameter.

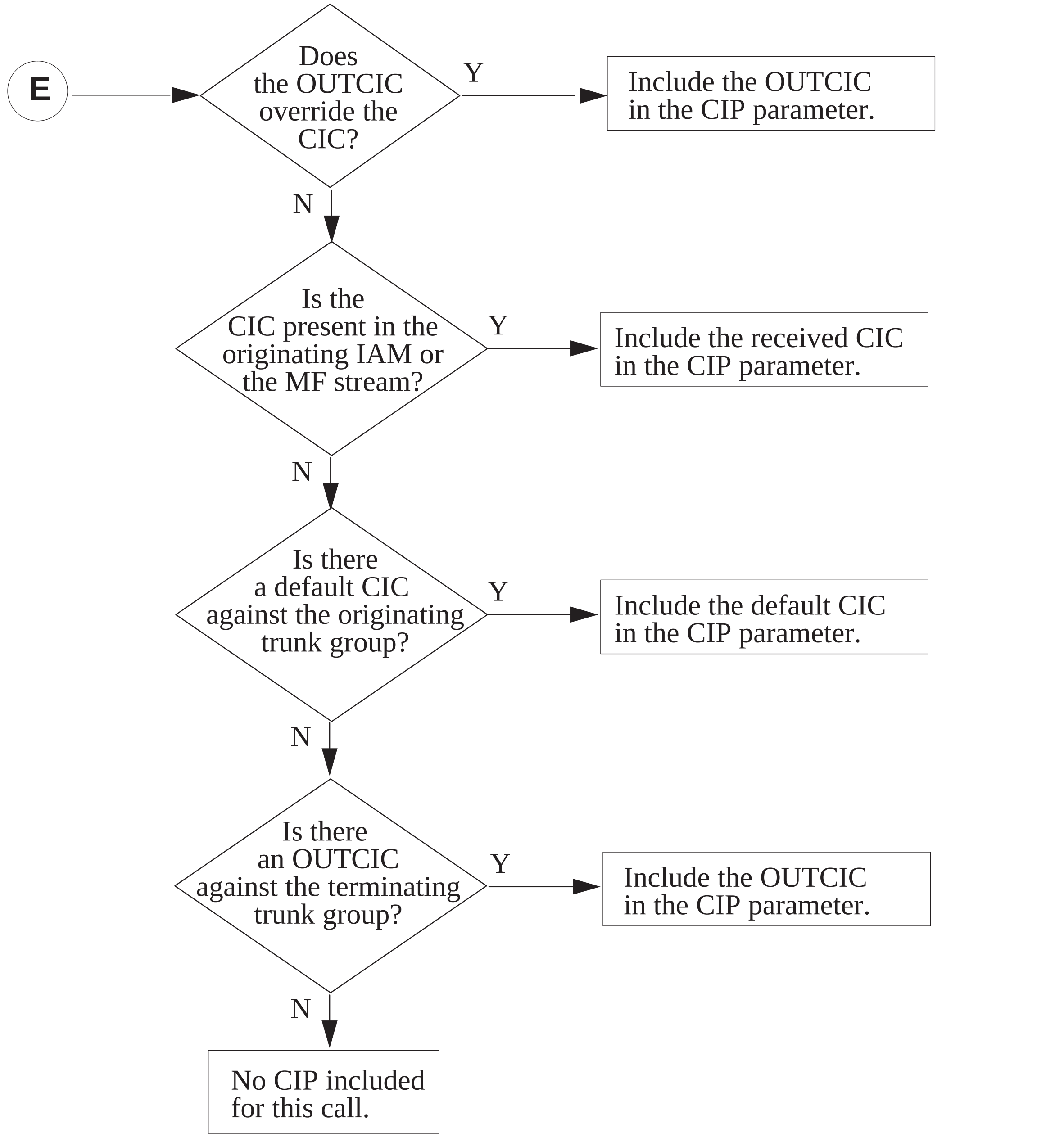
—continued—

Figure 21-4 (continued)
CCS7 CICBLK call processing actions


D
Does the OUTCIC override the CIC?
Y
Include the OUTCIC in the CIP parameter.
N
Is the CIC present in the originating IAM or the MF stream?
Y
Include the received CIC in the CIP parameter.
N
Is there a default CIC against the originating trunk group?
Y
Include the default CIC in the CIP parameter.
N
Is there an OUTCIC against the terminating trunk group?
Y
Include the OUTCIC in the CIP parameter.
N
Is the table TRKGRP CICSIZE set to 3DIGS?
Y
Include the office parameter CARRIER_ID_CODE in the CIP parameter.
N
Include the office parameter CIC_4DIGS in the CIP parameter.

—continued—

Figure 21-4 (continued)
CCS7 CICBLK call processing actions


E
Does the OUTCIC override the CIC?
Y
Include the OUTCIC in the CIP parameter.
N
Is the CIC present in the originating IAM or the MF stream?
Y
Include the received CIC in the CIP parameter.
N
Is there a default CIC against the originating trunk group?
Y
Include the default CIC in the CIP parameter.
N
Is there an OUTCIC against the terminating trunk group?
Y
Include the OUTCIC in the CIP parameter.
N
No CIP included for this call.

–end–

CCS7 CICSIZE call processing application

The CICSIZE option specifies the number of digits to outpulse to the terminating trunk agency.

CCS7 COT maintenance application

The Continuity Test (COT) option identifies a percentage of calls on the trunk group that request a continuity test to be performed during call setup.

Call processing actions

The information in the COT option is analyzed when the call is being presented to the next network switch. The PERCENT field value is compared to the number of calls terminated to members of the trunk group. If the continuity test is to be performed as a result of this comparison, the information in the CHKTYPE field is used in performing the COT test. The purpose of the continuity test is to verify the sanity of the bearer channel for the call.

The COT test is initiated after the seizure signal is delivered to the terminating switch. If continuity testing is successful, then the call continues and continuity testing resources are deallocated.

If the continuity signal is not properly received by the agent member, then the terminating agent is deallocated and released from the call. Then a member advance or route advance occurs as appropriate in order to select another trunk group member for the call.

Interactions

There are no specific interactions for the COT option.

Restrictions and limitations

There are no specific call processing restrictions for the COT option.

CCS7 DELIVER call processing application

The Calling Party Number Deliver (DELIVER) option specifies the calling party subscriber number delivery status or method for the call. See “DELIVER call processing application” on page 18-13 for a description of the call processing application of the DELIVER option.

CCS7 ECSTAT call processing application

The application of the CCS7 echo canceller status option is identical to the description of the ECSTAT option for PTS agents.

CCS7 ESUPR call processing application

The application of the CCS7 echo suppressor option is identical to the description of the ECSTAT option for PTS agents. See “PTS ECSTAT call processing application” on page 21-17.

CCS7 GLARE call processing application

The GLARE option identifies how glare situations are handled for members of the trunk group.

Call processing actions

The GLARE option for CCS7 agents is similar to the GLAREYD option for PTS agents. See “PTS GLAREYD call processing application” on on page 21-18.

For CCS7 agents, glare situations may be handled in one of three ways:

- The agent member may yield to glare (default if option not provisioned).
- The agent member may stand to glare (STAND ACTION value).
- The agent member may either stand or yield to glare based on the circuit identification code of the agent and the point code of the switch (CIC ACTION value).

For a description of standing versus yielding to glare, see the description of the PTS GLAREYD option.

For the CIC glare action, the following algorithm determines if the trunk group member stands or yields in glare situations:

- Even-numbered trunk group members stand to glare for the switch with the higher point code. Even-numbered circuits on the switch with the lower point code yield to glare.
- Odd-numbered trunk group members stand to glare for the switch with the lower point code. Odd-numbered circuits on the switch with the higher point code yield to glare.

The trunk group member circuit number (circuit identification code) is identified by the C7TRKMEM table.

For a description of the call processing associated with glare handling for the call, see the description of the PTS GLAREYD option.

Interactions

There are no specific call processing interactions for the GLARE option.

Restrictions and limitations

The following restrictions and limitations apply to the CCS7 glare call processing application.

- In glare situations, a trunk group member yielding to glare is deallocated from the current call and either a member advance or route advance occurs as appropriate.
- If both switches are instructed to stand in glare situations, then seize failure occurs the calls of each switch. This is due to timeout of the required seizure acknowledgment signal.

CCS7 NO_HOP call processing application

The presence of a NO_HOP option indicates the hop counter is not modified. The hop counter (HC) parameter is passed to the outgoing CCS7 IAM message when the call terminates.

Note: If Table RTEATTR exclude controls are used to block the hop counter (HC) parameter’s delivery, then the parameter is not passed to the outgoing CCS7 IAM message.

CCS7 OUTCIC call processing application

The OUTCIC option determines the CIC digits to outpulse over the terminating trunk.

Call processing actions

The UCS DMS-250 switch determines which CIC value to outpulse in the TNS parameter using the following precedences:

1. The switch outpulses the OUTCIC value. (The OUTCIC OVERRIDE option is set to Y.)
2. The switch outpulses the received CIC value.
3. The switch outpulses the default CIC (DEFCIC) value. (The OUTPULSE option is set to Y.)
4. The switch outpulses the OUTCIC value. (The OUTCIC OVERRIDE is set to N, otherwise the switch would have already outpulsed the OUTCIC value.)
5. The switch outpulses the three-digit or four-digit CIC in the CARRIER_ID_CODE or CIC_4DIGS office parameters, respectively.

The UCS DMS-250 switch determines which CIC value to outpulse in the CIP parameter to an SS7 trunk,that does not have the RTEATTR INC CIP set, using the following precedences:

1 The switch outpulses the OUTCIC value. (The OUTCIC OVERRIDE option is set to Y.)

2 The switch outpulses the received CIC value.

3 The switch outpulses the default CIC (DEFCIC) value. (The OUTPULSE option is set to Y.)

4 The switch outpulses the OUTCIC value. (The OUTCIC OVERRIDE is set to N, otherwise the switch would have already outpulsed the OUTCIC value.)

Note: If a DEFCIC, OUTCIC, or received CIC does not exist for the call, a CIP is not built.

The UCS DMS-250 switch determines which CIC value to outpulse in the CIP parameter to an SS7 trunk,that has the RTEATTR INC CIP set, using the following precedences:

1 The switch outpulses the OUTCIC value. (The OUTCIC OVERRIDE option is set to Y.)

2 The switch outpulses the received CIC value.

3 The switch outpulses the default CIC (DEFCIC) value. (The OUTPULSE option is set to Y.)

4 The switch outpulses the OUTCIC value. (The OUTCIC OVERRIDE is set to N, otherwise the switch would have already outpulsed the OUTCIC value.)

5 The switch outpulses the three-digit or four-digit CIC in the CARRIER_ID_CODE or CIC_4DIGS office parameters, respectively.

CCS7 SPMECIDX call processing application

The application of the CCS7 Spectrum echo canceller index option is identical to the description of the SPMECIDX option for PTS agents.

FlexDial logs and OMs

This chapter provides information on FlexDial logs and Operational Measurements (OMs).

FlexDial logs

Logs are generated during call processing for a number of purposes:

- an unexpected or abnormal event occurs
- requested information about the call is to be output

The FlexDial framework defines a new log class to deal with these scenarios as they arise in the execution of collectables within the framework. This new log class is the FLEX log class.

A number of logs within the FLEX log class are defined:

- FLEX 301 trouble log
- FLEX 302 subscriber validation failure log
- FLEX 303 execution failure log
- FLEX 304 CDR capture failure log
- FLEX 306 trouble in collectable list log
- FLEX 307 trouble in variable operation log
- FLEX 308 resource manager audit log
- FLEX 309 TRK4CIC ONP trouble log
- FLEX 401 FEATBYTE access error log
- FLEX 601 information log

FLEX 301 trouble log

The FLEX 301 trouble log is generated when a unexpected or abnormal event related to the processing of collectables occurs for the call.

Figure 22-1 shows the format of the FLEX 301 log.

Figure 22-1
FLEX 301 log format

```
FLEX 301 APR15 20:09:23 FLEXDIAL Trouble Log
   Agent : CKT <clli> <member number>
   Trouble : <string>
   Flextype : <string>
   Flexdial : <string>
   Digits : <digit vector>
```

The Agent field contains the call processing identifier of the originating agent involved in the call.

The Trouble field contains a brief description of the abnormal condition that occurred.

The Flextype field contains the FLEXTYPE subscriber number or call type definition associated with the event. If an associated FLEXTYPE is not available, then the identity of the collectable is output in the field.

The Flexdial field identifies the FLEXDIAL table index that contains the executing collectable.

The Digits field is used by sequence and digit collectables to output the digits being processed at the time the abnormal condition occurred.

FLEX 302 validation failure log

The FLEX 302 validation failure log is generated when subscriber number validation failure occurs for a SUBR or SUBRPARM collectable. An example is that a FLEX 302 log is reported when an authcode with OAC functionality is used for more than the maximum number of calls simultaneously for an AXXESS agent.

Figure 22-2 shows the format of the FLEX 302 log.

Figure 22-2
FLEX 302 log format

```
FLEX 302 APR15 20:09:23 Subscriber Validation Failure
   Agent : CKT <clli> <member number>
   Reason : <string>
   Flexdial : <string>
   Flextype : <string>
   Flexval Index : <decimal>
   Validation Count : <decimal>
   Digits : <digit vector>
```

The Agent field contains the call processing identifier of the originating agent involved in the call.

The Failure Reason field identifies the specific reason for the validation failure (for example, “Active CALLP Limit Enforced for Subscriber Number”).

The Flexdial field identifies the FLEXDIAL table index that contains the executing collectable.

The Flextype field contains the subscriber number FLEXTYPE value used in the SUBR or SUBRPARM collectable.

The Flexval Index field contains the numeric index used for the FLEXVAL table validation attempt.

The Validation Count field identifies the number of processed digits that were actually used in the validation attempt.

The Digits field contains the digits that were processed by the subscriber number collectable.

FLEX 303 execution failure log

The FLEX 303 execution failure log is generated when a collectable fails to execute for call processing.

Figure 22-3 shows the format of the FLEX 303 log.

Figure 22-3
FLEX 303 log format

```
FLEX 303 APR15 20:09:23 Collectable Execution Failure
   Agent : CKT <clli> <member number>
   Reason : <string>
   Flexdial : <string>
   Collectable : <string>
   Flextype : <string>
```

The Agent field contains the call processing identifier of the originating agent involved in the call.

The Reason field contains a brief description outlining the event triggering the generation of the log.

The Flexdial field identifies the FLEXDIAL table index that contains the collectable which failed to execute.

The Collectable field contains the identity of the collectable that failed to execute.

If the collectable that failed to execute is a subscriber number or CALLTYPE collectable, the Flextype field contains the FLEXTYPE value provisioned against the collectable.

FLEX 304 CDR capture failure log

The FLEX 304 CDR capture failure log is generated when a subscriber number is not captured correctly in the CDR field identified by the BILLFLD option of table FLEXTYPE.

Figure 22-4 shows the format of the FLEX 304 log.

Figure 22-4
FLEX 304 log format

```
FLEX 304 APR15 20:09:23 CDR Capture Failure Log
   Agent : CKT <clli> <member number>
   Flexdial : <string>
   Flextype : <string>
   CDR Field : <string>
   Digits : <digit vector>
```

The Agent field contains the call processing identifier of the originating agent involved in the call.

The Flexdial field identifies the FLEXDIAL table index that contains the executing collectable.

The Flextype field contains the FLEXTYPE defined subscriber number type associated with the event.

The CDR field identifies the field in the CDR where all the subscriber number digits were to be placed.

The Digits field contains two items of information delimited by a space:

- The number of digits that were truncated is identified (either one or two digits).
- All of the subscriber number digits processed by the SUBR or SUBRPARM collectable are output.

FLEX 306 trouble in collectable list log

The FLEX 306 trouble in collectable list log is generated in the following two cases:

- The number of FLEXDIAL lists built exceeds the office parameter FLEXDIAL_MAX_LIST_BUILT, which resides in table OFCENG.
- The number of collectables executed in one diaplan exceeded the office parameter FLEXDIAL_MAX_LIST_EXEC, which resides in table OFCENG.

Figure 22-5 shows the format of the FLEX 306 log.

Figure 22-5
FLEX 306 log format

```
FLEX 306 <date><time> FLEXDIAL Trouble in Collectable List
  CLLI: <agent where trouble occurred>
  Trouble: <reason>
  Flexdial : <current flexdial>
  Digits: <known collected digits>
  Last 5 Collectable: <last 5 collectables>
```

The CLLI field contains the call processing identifier of the originating agent involved in the call.

The Trouble field contains a brief description of the abnormal condition that occurred.

The Flexdial field identifies the FLEXDIAL table index that contains the executing collectable.

The Digits field contains those digits that have been collected so far.

The Last 5 Collectables are the 5 collectables built with the last collectable shown first.

Figure 22-6 shows an example of the FLEX 306 log.

Figure 22-6
FLEX 306 log example

```
FLEX 306 apr15 20:09:23 FLEXDIAL Trouble in Collectable List
   CLLI: AXXESS
   Trouble: Exceed max number of executed collectables.
   Flexdial : CURRENT_DPIDX
   Digits: 2146845511
   Last 5 Collectable: <IFCNT> <APTRMT> <ADDR> <SUBR> <SUBR>
```

FLEX 307 trouble in variable operation log

The FLEX 307 trouble in variable operation log is generated when an illegal use of a FLEXDIAL variable collectable, either VAROPT or IFVAR occurs for the call. Illegal uses are listed below:

- referencing an uninitialized FLEXDIAL variable
- overflow during an operation on a FLEXDIAL variable
- divide by zero

Figure 22-7 shows the format of the FLEX 307 log.

Figure 22-7
FLEX 307 Log Format

```
FLEX 307 <date><time> FLEXDIAL Trouble in Variable Operation
   CLLI: <agent where trouble occurred>
   Trouble: <reason>
   Flexdial : <current flexdial>
   Variable: <variable enumeration>
   Operation: <operator enumeration>
   Digits: <known collected digits>
```

The CLLI field contains the call processing identifier of the originating agent involved in the call.

The Trouble field contains a brief description of the abnormal condition that occurred.

The Flexdial field identifies the FLEXDIAL table index that contains the executing collectable.

The Variable field is the identifier to cause the error.

The Operation field is the operation to cause the error.

The Digits field contains those digits that have been collected so far.

Figure 22-8 shows an example of the FLEX 307 log.

Figure 22-8
FLEX 307 log example

```
FLEX 307 APR15 20:09:23 FLEXDIAL Trouble in Variable Operation
   CLLI: AXXESS
   Trouble: Reference uninitialize variable
   Flexdial : CURRENT_DPIDX
   Variable: AVAR
   Operation: MULT
   Digits: 2146845511
```

FLEX 308 resource manager audit log

For information on FLEX 308 log, refer to the “Resource Audit and FLEX 308 Log Generation” section in Chapter “System Requirements.”

FLEX 309 TRK4CIC ONP trouble log

The FLEX 309 TRK4CIC ONP trouble log is generated when an attempt to modify a table TRKSIG or table TRKFEAT tuple is made after the tuple has already been modified by the one night process (ONP) of table TRK4CIC.

Figure 22-9 shows the format of the FLEX 309 log.

Figure 22-9
FLEX 309 log format

```
FLEX 309  <date><time><nnnn> TRK4CIC ONP Trouble Log
   AGENT: <CLLI>
   TABLE: <string>
   INDEX: <string>
   OPTION(S): <string>
```

The AGENT field identifies which trunk group in table TRK4CIC attempted to write to a previously-written-to tuple in table TRKSIG or TRKFEAT. The valid value is an AXXESS agent defined in table CLLI.

The TABLE field indicates which table could not be written to. The valid values for this field are TRKSIG and TRKFEAT.

The INDEX field identifies the table TRKSIG or TRKFEAT index that could not be written to. The valid values are any table TRKSIG or TRKFEAT index.

The OPTION field identifies options that could not be written to the table TRKSIG or TRKFEAT tuple. The valid values are CICSIZE, DEFCIC, and OUTCIC.

Figure 22-10 shows an example of the FLEX 309 log.

Figure 22-10
FLEX 309 log example

```
FLEX 309 APR15 20:09:23 TRK4CIC ONP Trouble Log
   AGENT: CKT AXEAN626TWMFWK
   Table: TRKFEAT
   Index: TRKFEAT_0008
   Option(s): CICSIZE DEFCIC
```

FLEX 401 FEATBYTE access error log

The FLEX 401 FEATBYTE access error log is generated when non-fatal application errors are encountered while accessing the FEATBYTE table. Specific error details are given in the log text. Call processing proceeds with normal in-switch routing when these errors occur.

Figure 22-11 shows the format of the FLEX 401 log.

Figure 22-11
FLEX 401 log format

```
FLEX 401 <date><time><nnnn> INFO Featbyte Key Not Found
   AGENT: <TRKGRP><CLLI><TRKMEM#>
   UNIQUE_NUM: <feature byte unique number>
   FEATB_NUM: <feature byte number>
   FEATB_VAL: <feature byte value>
   FIRST_MASK: <feature byte mask value>
   N00: <the dialed N00 number digits>
```

The Agent field contains the trunk group, CLLI, and trunk subgroup number for the call.

The Unique Number field contains the unique number portion of the FEATBYTE key, for the key that returned ”no data found.”

The Feature Byte Number field contains the Featb_Num portion of the FEATBYTE key, for the key that returned ”no data found.”

The Feature Byte Value field contains the Featb_Val portion of the FEATBYTE key, for the key that returned ”no data found.”

The First Feature Byte Mask field contains the value stored in the office parameter, FEATBYTE_FIRST_MASK. This value is applied to the Feature Byte value returned from the TCAP Response message, to determine the ”real” Feature Byte value with which to query the table.

The N00 field contains the dialed N00 number digits.

FLEX 601 information log

The FLEX 601 information log is generated due to a request by the service provider to output certain information about a call in progress.

Figure 22-12 shows the format of the FLEX 601 log.

Figure 22-12
FLEX 601 log format

```
FLEX 601 NOV04 14:14:53 FLEXDIAL Information
   Agent : CKT <clli> <member number>
   Report : <string>
   Flextype : <string>
   Digits : <digit vector>
```

The Agent field contains the call processing identifier of the originating agent involved in the call.

The Report field contains a brief description of the specific report being generated.

The Flextype field contains the FLEXTYPE subscriber number or call type definition associated with the event. If an associated FLEXTYPE is not available, then the identity of the collectable is output in the field.

The Digits field contains received digits available for the information report.

FlexDial Operational Measurements

The FlexDial framework defines the following operational measurement (OM) groups:

- FLEXTYPE Specific Group

  The FLEXTYPE OM group contains counters that are incremented during the processing of subscriber number or call types defined in table FLEXTYPE.

- FLEXDIAL OM Group

  The FLEXDIAL OM Group combines a set of user-definable OM tuples that assist operating company personnel in measuring the frequency of dialplan usage.

FLEXTYPE OM Group

The FLEXTYPE OM group identifies a number of registers that may be incremented with the use of a particular subscriber number or call type defined in table FLEXTYPE.

This group supports up to 1024 tuples, where the creation of a new entry in table FLEXTYPE defines the additional tuple for the FLEXTYPE OM group. The deletion of the entry in table FLEXTYPE removes the associated tuple from the OM group.

For each tuple, a number of registers are defined:

- NOVAL Register

  Size : 32-bit unsigned double integer counter

  Description : This register is incremented when a subscriber number is processed but not validated.

- VALSUCC Register

  Size : 32-bit unsigned double integer counter

  Description : This register is incremented when a subscriber number is processed and successfully validated.

- VALFAIL Register

  Size : 32-bit unsigned double integer counter

  Description : This register is incremented when a subscriber number is processed and not successfully validated.

- VALEMPTY Register

  Size : 32-bit unsigned double integer counter

  Description : This register is incremented when a subscriber number is processed and successfully validated using the EMPTYIDX option.

- TVALSUCC Register

  Size : 32-bit unsigned double integer counter

  Description : This register is incremented when a subscriber number is processed and citycode validation is also performed and is successful.

- TVALFAIL Register

  Size : 32-bit unsigned double integer counter

  Description : This register is incremented when a subscriber number is processed and citycode validation is also performed and is not successful.

- CVALSUCC Register

  Size : 32-bit unsigned double integer counter

  Description : This register is incremented when a subscriber number is processed and casual blocking validation is also performed and is successful (that is, the call is not blocked).

- CVALFAIL Register

  Size : 32-bit unsigned double integer counter

  Description : This register is incremented when a subscriber number is processed and casual blocking validation is also performed and is unsuccessful (that is, the call is blocked).

- CALLTYPE Register

  Size : 32-bit unsigned double integer counter

  Description : This register is incremented when the FLEXTYPE is used by an executing CALLTYPE collectable.

- VALTOT Register

  Size : 16-bit unsigned double integer counter

Description : This register is incremented when a subscriber number is processed and a validation attempt is performed. This register has the following relationship with other FLEXTYPE registers:

VALTOT = VALSUCC + VALFAIL + VALEMPTY

VALTOT + NOVAL = Total of subscriber numbers processed

- TVALTOT Register

  Size : 16-bit unsigned double integer counter

  Description : This register is incremented when a citycode validation attempt is performed for a subscriber number being processed. This register has the following relationship with other FLEXTYPE registers:

  TVALTOT = TVALSUCC + TVALFAIL

- CVALTOT Register

  Size : 16-bit unsigned double integer counter

  Description : This register is incremented when a casual number blocking validation attempt is performed for a subscriber number being processed. This register has the following relationship with other FLEXTYPE registers:

  CVALTOT = CVALSUCC + CVALFAIL

- LVALTOT Register

  Size : 16-bit unsigned double integer counter

  Description : This register is incremented when a call processing active validation attempt is performed for a subscriber number being processed. This register has the following relationship with other FLEXTYPE registers:

  LVALTOT = LVALSUCC + LVALFAIL

- LVALSUCC Register

  Size : 16-bit unsigned double integer counter

  Description : This register is incremented when a successful call processing active validation attempt is performed.

- LVALFAIL Register

Size : 16-bit unsigned double integer counter

Description : This register is incremented when an unsuccessful call processing active validation attempt is performed.

Note: Note that multiple forms of validation may occur for the processing of a single subscriber number. Therefore while no particular relationship exists between the TVALTOT, CVALTOT, and LVALTOT registers, each of these registers has an independent percentage relationship with the VALTOT register. For example, of VALTOT number of total subscriber validation attempts, a TVALTOT amount also performed citycode validation.

FLEXDIAL OM group

The FLEXDIAL OM group combines a set of user-definable OM tuples that assist operating company personnel in measuring the frequency of dialplan usage. The FLEXDIAL OM counts up to 2 billion plus.

The FLEXDIAL OM has one field, INFO, which is a 32-bit counter. Its value is determined by combining the OM_LSIG register and the OM_MSIG register, which are pegged when the respective OM framework collectable executes. The INFO field shows the combined total of the two registers.

The OM collectable inside table FLEXDIAL defines the tuple at datafill time and increments the counter when it is encountered during call processing. This tuple is deleted when no other OM collectable references it.

See Chapter 2, ”FLEXDIAL table,” for more information on the OM framework collectable.

The number of tuples is 1024; for each tuple, two registers are defined:

- OM_LSIG Register

  Size : 16-bit integer

  Description : This register contains the 16 least significant bits of the 32-bit counter. When OM_LSIG overflows, the OM_MSIG is incremented and OM_LSIG is set back to zero.

- OM_MSIG Register

  Size : 16-bit integer

  Description : This register contains the 16 most significant bits of the 32-bit counter.

System requirements

This chapter provides information on the system requirements for FlexDial framework.

Pooled system resources

The FlexDial call processing system requires FlexDial framework specific system resources in order to execute provisioned collectables. Table 23-1 identifies the different pooled resources defined exclusively for the FlexDial framework.

Table 23-1
FlexDial pooled resource allocation

Resource pool	Description	Duration allocated per call	Size of elements (in bits)	Maximum defined pool size ratio (multiplied by the defined number of CCBs)
DS1 Agent	An agent class instance is used to represent the originating Axxess agent and terminating Axxess agent in the call.	Up to two agents are allocated for entire call. The number of Axxess agents involved in the call identifies the number of agent resources allocated.	1432	0.3
FXS Agent	An agent class instance is used to represent the originating Axxess agent and terminating Axxess agent in the call.	Up to two agents are allocated for entire call.	1432	0.1

—continued—

Table 23-1
FlexDial pooled resource allocation (continued)

Resource pool	Description	Duration allocated per call	Size of elements (in bits)	Maximum defined pool size ratio (multiplied by the defined number of CCBs)
FXO Agent	An agent class instance is used to represent the originating Axxess agent and terminating Axxess agent in the call.	Up to two agents are allocated for entire call.	1432	0.1
SS7 Agent	An agent class instance is used to represent the originating Axxess agent and terminating Axxess agent in the call.	Up to two agents are allocated for entire call.	1369	0.3
Q764 Agent	An agent class instance is used to represent the originating Axxess agent and terminating Axxess agent in the call.	Up to two agents are allocated for entire call.	1465	0.3
FS DS1 Agent	An CI agent class instance used to represent a DS1 agent for FLEXSIM.	Duration of the FLEXSIM call.	1432	0.0
FS FXS Agent	An CI agent class instance used to represent an FXS agent for FLEXSIM.	Duration of the FLEXSIM call.	1432	0.0
FS FXO Agent	An CI agent class instance used to represent an FXO agent for FLEXSIM.	Duration of the FLEXSIM call.	1432	0.0
FS SS7 Agent	An CI agent class instance used to represent a CCS7 agent for FLEXSIM.	Duration of the FLEXSIM call.	1369	0.0
FS Q764 Agent	An CI agent class instance used to represent a Q 764 agent for FLEXSIM.	Duration of the FLEXSIM call.	1465	0.0
Setup Collector	The setup collector is the driver for executing collectables, and contains the collectable manager and digit buffer.	From Origination through Answer.	1344	0.6

—continued—

Table 23-1
FlexDial pooled resource allocation (continued)

Resource pool	Description	Duration allocated per call	Size of elements (in bits)	Maximum defined pool size ratio (multiplied by the defined number of CCBs)
ADDDIGS	This is the pool resource for ADDDIGS collectables.	Collect Info. Point in Call Only.	935	0.1
ADDR	This is the pool resource for ADDR collectables.	Collect Info. Point in Call Only.	2930	0.5
ADDRBOOM	This pool resource is used to process address digits for boomerang reoriginations.	Collect Info. Point in Call Only.	3013	0.1
ADDRSD	This pool resource is used to manage shared data information for CICPARM collectable instances.	Life-span of cached provisioned instance	483	0.1
ADDRPARM	This is the pool resource for ADDRPARM collectables.	Collect Info. Point in Call Only.	2661	0.5
ADRPRMSD	This pool resource is used to manage shared data information for ADDRPARM collectable instances.	Life-span of cached provisioned instance	371	0.1
AGNTDATA	This is the pool resource for AGNTDATA collectables.	Collect Info. Point in Call Only.	691	0.1
APRESET	This is the pool resource for APRESET collectables.	Collect Info. Point in Call Only.	240	0.1
APTRMT	This is the pool resource for APTRMT collectables.	Collect Info. Point in Call Only.	241	0.1
CALLCOND	This is the pool resource for CALLCOND collectables.	Collect Info. Point in Call Only.	287	0.1
CALLTYPE	This is the pool resource for CALLTYPE collectables.	Collect Info. Point in Call Only.	2288	0.1

—continued—

Table 23-1
FlexDial pooled resource allocation (continued)

Resource pool	Description	Duration allocated per call	Size of elements (in bits)	Maximum defined pool size ratio (multiplied by the defined number of CCBs)
CIC	This is the pool resource for CIC collectables.	Collect Info. Point in Call Only.	1369	0.1
CICSD	This pool resource is used to manage shared data information for CIC collectable instances.	Life-span of cached provisioned instance	356	0.1
CICPARM	This is the pool resource for CICPARM collectables.	Collect Info. Point in Call Only.	1225	0.2
CICPRMSD	This pool resource is used to manage shared data information for CICPARM collectable instances.	Life-span of cached provisioned instance	274	0.1
CLRFTRS	This is the pool resource for CLRFTRS collectables.	Collect Info. Point in Call Only.	2253	0.1
COLDIG	This is the pool resource for COLDIG collectables.	Collect Info. Point in Call Only.	1804	0.5
COLDIGSD	This pool resource is used to manage shared data information for COLDIG collectable instances.	Life-span of cached provisioned instance	396	0.1
COLPARM	This is the pool resource for the COLPARM collectables.	Collect Info. Point in Call Only.	1533	0.5
COLPRMSD	This pool resource is used to manage shared data information for COLPARM collectable instances.	Life-span of cached provisioned instance	278	0.1
COPYDIGS	This is the pool resource for COPYDIGS collectables.	Collect Info. Point in Call Only.	679	0.1
CPRCPTCH	This pool resource is used as an additional storage resource for patching purposes.	Life of CPRC involvement in the call.	800	0.1

—continued—

Table 23-1
FlexDial pooled resource allocation (continued)

Resource pool	Description	Duration allocated per call	Size of elements (in bits)	Maximum defined pool size ratio (multiplied by the defined number of CCBs)
DELDIGS	This is the pool resource for DELDIGS collectables.	Collect Info. Point in Call Only.	667	0.1
DO	This is the pool resource for DO collectables.	Collect Info. Point in Call Only.	255	0.3
FGDPARM	This is the pool resource for FGDPARM collectables.	Collect Info. Point in Call Only.	1424	0.6
FGDPRMSD	This pool resource is used to manage shared data information for FDGPARM collectable instances.	Life-span of cached provisioned instance	5587	1.0
FLXCRANK	This pool resource is used to cache FlexDial table entries in the format for CALLP use. Only entries that are accessed by call processing are cached for future CALLP use.	Life-span of cached provisioned FLEXDIAL table index	1016	1.0
GOTO	This is the pool resource for GOTO collectables	Collect Info. Point in Call Only.	255	0.1
IFCNT	This is the pool resource for IFCNT collectables.	Collect Info. Point in Call Only.	302	0.4
IFDIGS	This is the pool resource for IFDIGS collectables.	Collect Info. Point in Call Only.	819	0.4
IFNOA	This is the pool resource for IFNOA collectables.	Collect Info. Point in Call Only.	355	0.4
IFPARM	This is the pool resource for IFPARM collectables.	Collect Info. Point in Call Only.	338	0.4
IFPRMT	This is the pool resource for IFPRMT collectables.	Collect Info. Point in Call Only.	500	0.1
IFTOD	This is the pool resource for IFTOD collectables.	Collect Info. Point in Call Only.	436	0.1

—continued—

Table 23-1
FlexDial pooled resource allocation (continued)

Resource pool	Description	Duration allocated per call	Size of elements (in bits)	Maximum defined pool size ratio (multiplied by the defined number of CCBs)
IFTRMT	This is the pool resource for IFTRMT collectables.	Collect Info. Point in Call Only.	340	0.1
IFVAR	This is the pool resource for IFVAR collectables.	Collect Info. Point in Call Only.	564	0.1
INCLUDE	This is the pool resource for INCLUDE collectables.	Collect Info. Point in Call Only.	255	0.4
KILLTMR	This is the pool resource for KILLTMR collectables.	Collect Info. Point in Call Only.	272	0.1
MODDIGS	This is the pool resource for MODDIGS collectables.	Collect Info. Point in Call Only.	695	0.1
MODNOA	This is the pool resource for MODNOA collectables.	Collect Info. Point in Call Only.	315	0.1
MSGCTR	This is the pool resource for MSGCTR messages. One element can hold up to 20 messages for a call.	Orig. Auth. through Answer	320	1.0
NOOP	This is the pool resource for NOOP collectables.	Collect Info. Point in Call Only.	240	0.1
NOTIFY	This is the pool resource for NOTIFY collectables.	Collect Info. Point in Call Only.	304	0.1
OLI	This is the pool resource for OLI collectables.	Collect Info. Point in Call Only.	1587	0.3
OLISD	This pool resource is used to manage shared data information for OLI collectable instances.	Life-span of cached provisioned instance	451	0.1
OLIPARM	This is the pool resource for OLIPARM collectables.	Collect Info. Point in Call Only.	1443	0.2
OLIPRMSD	This pool resource is used to manage shared data information for OLIPARM collectable instances.	Life-span of cached provisioned instance	371	0.1

—continued—

Table 23-1
FlexDial pooled resource allocation (continued)

Resource pool	Description	Duration allocated per call	Size of elements (in bits)	Maximum defined pool size ratio (multiplied by the defined number of CCBs)
OM	This is the pool resource for OM collectables.	Collect Info. Point in Call Only.	485	0.1
OPERDISP CLG	This is the pool resource for OPERDISP FLEXTYPE option CLG value.	Collect Info. through Answer	232	0.1
OPERDISP SPL	This is the pool resource for OPERDISP FLEXTYPE option SPL value.	Collect Info. through Answer	252	0.1
RCVSIG	This is the pool resource for receive signalling collectables.	Collect Info. Point in Call Only.	264	0.1
REPLDIG	This is the pool resource for REPLDIG collectables.	Collect Info. Point in Call Only.	1416	0.1
REPLDGSD	This pool resource is used to manage shared data information for REPLDIG collectable instances.	Life-span of cached provisioned instance	415	0.1
RETRIEVE	This is the pool resource for RETRIEVE collectables.	Collect Info. Point in Call Only.	725	0.1
ROUTE	This is the pool resource for ROUTE collectables.	Collect Info. Point in Call Only.	369	0.1
SETTRANS	This is the pool resource for SETTRANS collectables.	Collect Info. Point in Call Only.	244	0.1
SETTRMT	This is the pool resource for SETTRMT collectables.	Collect Info. Point in Call Only.	258	0.1
SIG	This is the pool resource for signalling collectables with NP pulsetype.	Collect Info. Point in Call Only.	336	0.1

—continued—

Table 23-1
FlexDial pooled resource allocation (continued)

Resource pool	Description	Duration allocated per call	Size of elements (in bits)	Maximum defined pool size ratio (multiplied by the defined number of CCBs)
SIGDTMF	This is the pool resource for signalling collectables with DTMF pulsetype.	Collect Info. Point in Call Only.	361	0.3
SIGMF	This is the pool resource for signalling collectables with MF pulsetype.	Collect Info. Point in Call Only.	342	0.1
SNDSIG	This is the pool resource for send signalling collectables	Collect Info. Point in Call Only.	266	0.2
SUBR	This is the pool resource for SUBR collectables.	Collect Info. Point in Call Only.	4197	0.6
SUBRSD	This pool resource is used to manage shared data information for SUBR collectable instances.	Life-span of cached provisioned instance	602	0.1
SUBRPARM	This is the pool resource for SUBRPARM collectables.	Collect Info. Point in Call Only.	4037	0.6
SUBRPRSD	This pool resource is used to manage shared data information for SUBRPARM collectable instances.	Life-span of cached provisioned instance	500	0.1
SUBRRVAL	This pool resource is used for SUBR instances used during revalidation upon reorigination.	Collect Info. Point in Call Only.	4277	0.1
TERMINATE	This is the pool resource for TERMINATE collectables.	Collect Info. Point in Call Only.	255	0.1

—continued—

Table 23-1
FlexDial pooled resource allocation (continued)

Resource pool	Description	Duration allocated per call	Size of elements (in bits)	Maximum defined pool size ratio (multiplied by the defined number of CCBs)
TIMER	This is the pool resource for TIMER collectables.	Collect Info. Point in Call Only.	320	0.1
VAROP	This is the pool resource for VAROP collectables.	Collect Info. Point in Call Only.	384	0.1

—end—

Much of the data store used by collectables is used for call information that can be restored during a RESET digit collectable operation.

Call processing for MSGCTR messaging no longer uses an independent pooled element resource, but has been migrated to use the FlexDial shared pool resources system. Therefore the FLEXDIAL_MSGCTR_MULT office parameter is no longer used by the FlexDial framework.

A number of factors contribute to the required size of pool resources that are needed for call processing purposes:

- the number of originations that are AXXESS trunk originations versus other trunk type originations (such as IMT)
- the number of calls that are in a setup state versus a talking state
- the provisioning scheme for the different FlexDial-defined interactions, and the number of originations occurring on each particular scheme

The currently specified maximum defined pool size ratios are derived through analysis of the above factors.

Sharing pool resources

The FlexDial framework pool resources are built upon a sophisticated pooling system, and the individual resource pools defined above are actually grouped into a single shared pool resource based upon compatibility between pool element sizes. Individual resource pools then allocate required pool elements from the shared pool resource. This enables the different individual pool resources to “take” elements from another individual pool when their resource availability reaches a specific threshold amount.

- When an individual pool reaches 97% of consumed capacity and the overall shared pool is at less than 95% consumed capacity, then elements from other individual pools within the shared group are migrated to the pool in need.
- When the shared pool is reaches 95% consumed capacity, then an additional 10% is allocated based upon the “maximum defined pool size ratios” identified and made available to the shared pool resource.

The individual pools are grouped into the shared pool resources as shown in Table 23-2, which are managed by the FlexDial Resource Distributor (FRED).

The individual pools are grouped into the following shared pool resources, which are managed by the FlexDial Resource Distributor (FRED).

Table 23-2
FlexDial shared pool resource allocation

Shared resource pool	Individual pools included	Size of shared resource elements (in bits)	Maximum defined pool size ratio (multiplied by the defined number of CCBs)
ELF	SETTRMT, TERMINATE, DO, GOTO, INCLUDE, NOOP, SETTRANS, APTRMT, APRESET, OPERD CLG, OPERD SPL, SETTRANS	260	1.6
DASHER	MSGCTR, MODNOA, KILLTMR, TIMER, SNDSIG, CALLCOND, NOTIFY, IFCNT, RCVSIG, CICPRMSD, COLPRMSD	320	2.4
DANCER	ROUTE, SIG, IFNOA, SIGDTMF, SIGMF, VAROP, IFTRMT, IFPARM, ADRPRMSD, CICSD, COLDIGSD, OLIPRMSD	400	2.0

—continued—

Table 23-2
FlexDial shared pool resource allocation (continued)

Shared resource pool	Individual pools included	Size of shared resource elements (in bits)	Maximum defined pool size ratio (multiplied by the defined number of CCBs)
PRANCER	IFTOD, IFPRMT, OM, SUBRPRSD, ADDRSD, OLISD, REPLDGSD	560	.07
VIXEN	CPRCPTCH, DELDIGS, MODDIGS, COPYDIGS, AGNTDATA, RETRIEVE, IFVAR, IFDIGS, SUBRSD	880	1.2
COMET	ADDDIGS, CICPARM, FLXCRANK	1264	1.3
CUPID	CIC, REPLDIG, SETUP COL, FGDPARM	1424	1.4
DONNER	COLDIG, COLPARM, OLI, OLIPARM, FSQ764 AGNT, Q764 AGNT, FSDS1 AGNT, FSFXS AGNT, FSFXO AGNT, FXO AGNT, FXS AGNT, DS1 AGNT,	1904	2.3
BLITZEN	ADDR, ADDRPARM, CLRFTRS, CALLTYPE, ADDRBOOM	3792	1.3
RUDOLPH	SUBR, SUBRPARM, SUBRRVAL	4928	1.3
SANTA	FGDPRMSD	6000	1.0

—end—

Information on the status of the shared pool system resources and the individual pool resources may be obtained from the UCS DMS-250 switch through execution of the QFRED (Query FlexDial Resource Distributor) CI command:

```
CI:
> flextest pool;qfred
```

The information output from the QFRED command consists of a shared resource summary report and an individual manager/client summary report. These reports identify current allocation values for the FlexDial resources, and show actual ratio values currently in use versus the defined ratio values from the above tables.

An audit process is used to increase required shared pool resource sizes. This process is triggered through two events:

- During pool migrations, it can be determined that the shared pool resource threshold has been reached. The migrator then causes the audit process to run which increases the pool resources.
- The audit process runs periodically to verify the integrity of the pools and increase pool resources as needed.

The FlexDial resource audit process (module FLXINFRM) runs on a pre-defined schedule, and is used to verify the integrity of the resource pools and perform general auditing of the allocated memory used by FRED. If this resource manager audit detects the need to increase the size of a shared pool resource, then it sends a message triggering execution of the FlexDial pool audit, which then expands the shared pool resources as needed.

The office parameter, FLEXDIAL_AUDIT_INTERVAL, also controls execution of the FlexDial pool audit. However, since the pool audit is automatically triggered through an office parameter change (changes to FLEXDIAL_MIN_CALLP_ALLOC or NUM_CPRC_EXT_BLK) and by the resource manager audit, there is currently no need to enact the audit on a periodic basis. Therefore the FLEXDIAL_AUDIT_INTERVAL office parameter may be set to zero (0) for UCS08.

Shared pool resource allocation

The shared pool resource elements are allocated in blocks of 64K bytes according to the following equation:

Elements = (FLEXDIAL_MIN_CALLP_ALLOC/100) ∗ MaxRatioValue ∗ NUMCPRCEXTBLK

Since the number of elements allocated is rounded up to fill all allocated 64K byte blocks of store, the number of 64K blocks of store allocated is derived from the following equations:

PoolMemoryReqd = {[(FLEXDIAL_MIN_CALLP_ALLOC/100) ∗ MaxRatioValue ∗ NCCBS] ∗ (PoolBitSize/8)} bytes

Num of 64K blocks = [PoolMemoryReqd MOD(64K)] + 1

Resources can be immediately allocated when events occur, such as the audit process, or changing the FLEXDIAL_MIN_CALLP_ALLOC or NCCBS office parameters. However, a restart cold must be performed to reduce system resource requirements and release allocated memory.

For the MSGCTR pool resource equation, replace the “MaxRatioValue” in the above equations with the value from the FLEXDIAL_MSGCTR_MULT office parameter. The pool bit size for the message center pool resource is 160 bits.

Expansion and contraction of pool resources

Due to the variable allocation requirements for the system, the pool resources are designed to grow as needed in order to fill call processing needs, and can potentially grow beyond the maximum defined pool size. The “maximum defined pool size” therefore provides two real purposes:

- The size defines a ratio of the number of elements in the pool resource to the number of CPRCs. The number of available CPRCs represents the number of calls that involve an Axxess agent (as either the originator or terminator) at any one time. This provides a mechanism to “favor” some pooled elements over others (i.e. allocate more of one resource and less of another) while basing the overall strategy on the number of Axxess agent calls possible.
- Though the ratio, a means of allocating a percentage of FLEXDIAL call processing resources is possible using the FLEXDIAL_MIN_CALLP_ALLOC office parameter.

Because the resource pools can grow automatically as needed by call processing, three possible events can occur when modifying the FLEXDIAL_MIN_CALLP_ALLOC office parameter:

- The new office parameter value is less than the currently set value.

  In this case, the pool resource sizes are decreased in order to match the office parameter value. The actual releasing of the pool memory does not occur until a cold or reload restart is performed on the switch.

- The new office parameter value is greater than the currently set value, but less than the actual pool sizes (due to audit process expansion of pool sizes).

  In this case no changes are performed to the pool resources.

- The new office parameter value is greater than the currently set value, and also greater than the actual pool sizes.

  In this case the pool resource sizes are immediately increased to the percentage identified by the office parameter.

Restart Behavior

During cold and reload restarts, all FlexDial framework pool resources are re-initialized and pool sizes are adjusted appropriately. Restarts perform the following functions for pool resources:

- Resize pools when the FLEXDIAL_MIN_CALLP_ALLOC is adjusted lower.

  Pools can only be shrunk on restarts.

- Clean up allocated store when memory corruption occurs for pooled elements.

  Memory corruption can only be cleaned up by performing a cold or reload restart.

Resources Used for Data Caching

Some of the individual pool resources allocated are used by the FlexDial framework for caching data in a format optimal for call processing use. Currently, the FLEXDIAL table is the only recipient of the use of pool resource caching.

For table FLEXDIAL, table entries that are “hit” during call processing are then cached for future call processing use, resulting in faster execution times as more calls are performed and the set of inuse FLEXDIAL table entries becomes cached.

The caching method is to convert the format of the provisioned collectables into a structure organized for optimal call processing access and use. The FLXCRANK pool resource is used for this purpose, as well as the shared data resources defined for the individual digit collectable instances.

Resource Audit and FLEX 308 Log Generation

Two unique audits exist for monitoring the use of memory for the FlexDial framework:

- FlexDial Resource Manager Audit
- FlexDial Pool Audit

The FlexDial Resource Manager audit runs on a pre-defined schedule and performs the following tasks:

- Verifies the integrity of the pooled resources
- Verifies that clients are actively maintaining control over their allocated resources, and that resources are not getting “dangled”.
- Clean up dangled resources and resolve mis-matches between the pool managers, pool migrators, and FRED.
- Determine if the availability of shared pool resources is at a critical level, and trigger execution of the FlexDial Pool Audit if so.

The FlexDial Pool Audit runs as explicitly requested and performs the following task:

- Increase the sizes of shared pool resources.

The FLEX 308 log is used to inform the service provide when errors are detected by the resource manager audit, or when resources are allocated and deallocated by call processing clients.

It is expected that FLEX 308 logs would not be generated under normal operating conditions, as generation of a FLEX 308 log indicates that an abnormal error condition has occurred within the FlexDial shared pool resources.

The FlexDial resource manager audit is capable of resolving most errors which may occur, however Nortel Networks ETAS personnel should be contacted immediately upon the generation of a FLEX 308 log. A worst case scenario would require a system restart cold in order to resolve FlexDial resource issues.

Figure 22-1 shows the format of the FLEX 308 log.

Figure 22-1
FLEX 308 log example

```
FLEX 308 FEB 05 12:22:47 INFO
     Trouble :              <string>
     Resource ID:           <integer>
     Item count:            <integer>
     Item limit:            <integer>
     Manager count:         <integer>
```

Multiple scenarios exist where the FLEX 308 log may be generated.

FLEX 308 Log Generation Scenarios

The following scenarios identify when a FLEX 308 log could be generated, for what reason, and the severity of the possible problem.

1

```
FLEX 308 FEB 05 14:47:15 INFO
    Trouble:           Pool resource auto added by system
    Resource ID:       <always 0>
    Item count:        <always 0>
    Item limit:        <resource max items allowed>
    Manager Count:     <always 0>
```

This log could only appear as a result of a patch that requests a new pool manager, and is not a severe issue.

```
2  FLEX 308 FEB 05 14:47:15 INFO
       Trouble:          Could not add pool resource for new requestor
       Resource ID:      <always 0>
       Item count:       <always 0>
       Item limit:       <always 0>
       Manager Count:    <always 0>
```

This log indicates that a new shared resource could not be created to meet a new request. This is a severe issue and swerr logs should be checked.

```
3  FLEX 308 FEB 05 14:47:15 INFO
       Trouble:          Dangling pool item found
       Resource ID:      <resource id>
       Item count:       <resource total items>
       Item limit:       <resource max items allowed>
       Manager Count:    <pool manager id>
```

This log indicates that a pool element was not properly managed by a client process, possibly lost by the client process due to abnormal process termination, client audit failure, or a general software memory leak. This is not a severe issue and is handled by the resource manager audit.

```
4  FLEX 308 FEB 05 14:47:15 INFO
       Trouble:          Pool item count mismatch detected
       Resource ID:      <pool resource id>
       Item count:       <item count mismatch>
       Item limit:       <resource max items allowed>
       Manager Count:    <managers in resource>
```

This log indicates the resource involved and the item count mismatch between the total number of used items in the resource minus the total number of items in the resource’s pool managers (which should always be equal). Any mismatch results in a purge operation that recovers missing items and restores proper counts. This log should be associated with a “dangling pool item found” log, and is a severe issue - though handled by the resource manager audit.

```
5  FLEX 308 FEB 05 14:47:15 INFO
       Trouble:          Pool allocations are currently blocked
       Resource ID:      <resource id>
       Item count:       <resource total item count>
       Item limit:       <resource max items allowed>
       Manager Count:    <managers in resource>
```

This log indicates that an internal flag is set to block memory allocations by a resource, and would only be observed if a patch were applied to block resource expansion. A conflict arises where the system requires additional resources to meet increasing call processing demands, but the applied patch is blocking the allocation request. This is a severe issue.

6

```
FLEX 308 FEB 05 14:47:15 INFO
    Trouble:          Max resource item count exceeded
    Resource ID:      <resource id>
    Item count:       <resource total item count>
    Item limit:       <resource max items allowed>
    Manager Count:    <managers in resource>
```

This log indicates that a shared resource has reached its allocation limit and cannot expand to meet new requests. This is a critical issue.

7

```
FLEX 308 FEB 05 14:47:15 INFO
    Trouble:          Resource block-memory allocation attempt failed
    Resource ID:      <resource id>
    Item count:       <resource total item count>
    Item limit:       <resource max items allowed>
    Manager Count:    <managers in resource>
```

This log indicates a resource could not allocate more memory due to memory exhaustion on the UCS DMS-250 switch. This is a critical issue.

8

```
FLEX 308 FEB 05 14:47:15 INFO
    Trouble:          Pool resource is thrashing
    Resource ID:      <resource id>
    Item count:       <resource total item count>
    Item limit:       <resource max items allowed>
    Manager Count:    <managers in resource>
```

This log indicates that a shared pool resource is approaching its capacity and is preparing for an expansion. This is not a severe issue.

9

```
FLEX 308 FEB 05 14:47:15 INFO
    Trouble:          Pool resource allocation pending too long
    Resource ID:      <resource id>
    Item count:       <resource total item count>
    Item limit:       <resource max items allowed>
    Manager Count:    <managers in resource>
```

This log indicates that a resource memory expansion attempt has timed out likely due to a high volume of call processing occurring on the switch. This is a severe issue, though the attempt to allocate memory continues to re-occur until the allocation is successful. The impact is that until the memory is allocated and the shared resource pool expanded, additional FLEX 308 thrashing logs may be output.

10

```
FLEX 308 FEB 05 14:47:15 INFO
    Trouble:          Pool resource expansion due to thrashing
    Resource ID:      <resource id>
    Item count:       <resource total item count>
    Item limit:       <resource max items allowed>
    Manager Count:    <managers in resource>
```

This log indicates that a shared pool resource has expanded itself due to an increasing demand for resource elements. This is not a severe issue.

11

```
FLEX 308 FEB 05 14:47:15 INFO
    Trouble:          Resource memory allocation attempt failure
    Resource ID:      <resource id>
    Item count:       <resource total item count>
    Item limit:       <resource max items allowed>
    Manager Count:    <managers in resource>
```

This log indicates a resource could not allocate enough memory to expand as required due to call processing demands. This is a severe issue.

12

```
FLEX 308 FEB 05 14:47:15 INFO
    Trouble:          Item requested from empty pool
    Resource ID:      <resource id>
    Item count:       <resource total item count>
    Item limit:       <resource max items allowed>
    Manager Count:    <managers in resource>
```

This log indicates that demand for elements of a resource have increased faster than the resource has been able to respond preemptively. The result is single or multiple call failures until the quickly triggered FlexDial pool audit can execute and additional resources can be allocated.

13

```
FLEX 308 FEB 05 14:47:15 INFO
    Trouble:          Item return state invalid
    Resource ID:      <item state>
    Item count:       <pool manager free items>
    Item limit:       <pool manager total items>
    Manager Count:    <pool manager id>
```

This log indicates that an item flagged as something other than “inuse” was returned to the pool manager, and is likely preceded by a “dangling pool item found” FLEX 308 log or a “client audit failure for item” FLEX 308 log. The likely cause of this log is a dangled element, and is not a severe issue.

14

```
FLEX 308 FEB 05 14:47:15 INFO
    Trouble:          Client audit failure for item
    Resource ID:      <item state>
    Item count:       <pool manager free items>
    Item limit:       <pool manager total items>
    Manager Count:    <pool manager id>
```

This log indicates is similar to the “item return state invalid” FLEX 308 log, though more severe. In this case, a recovered dangling item from one process was given to another process, and then returned to the resource manager by both processes. This issue is severe for all circumstances except a typical memory leak situation, where it is not a severe issue.

Appendix A FlexDial simulator

This appendix describes the FlexDial simulation tool, FLEXSIM. It also describes table FLEXSIMT, which defines the Initial Address Message (IAM) for an SS7 simulation.

Overview

The FLEXSIM Command Interpreter (CI) tool provides the ability to execute FlexDial framework collectables and simulate collectables call processing from a MAP terminal. The MAP terminal replaces the agent terminal controller in the basic switch call model.

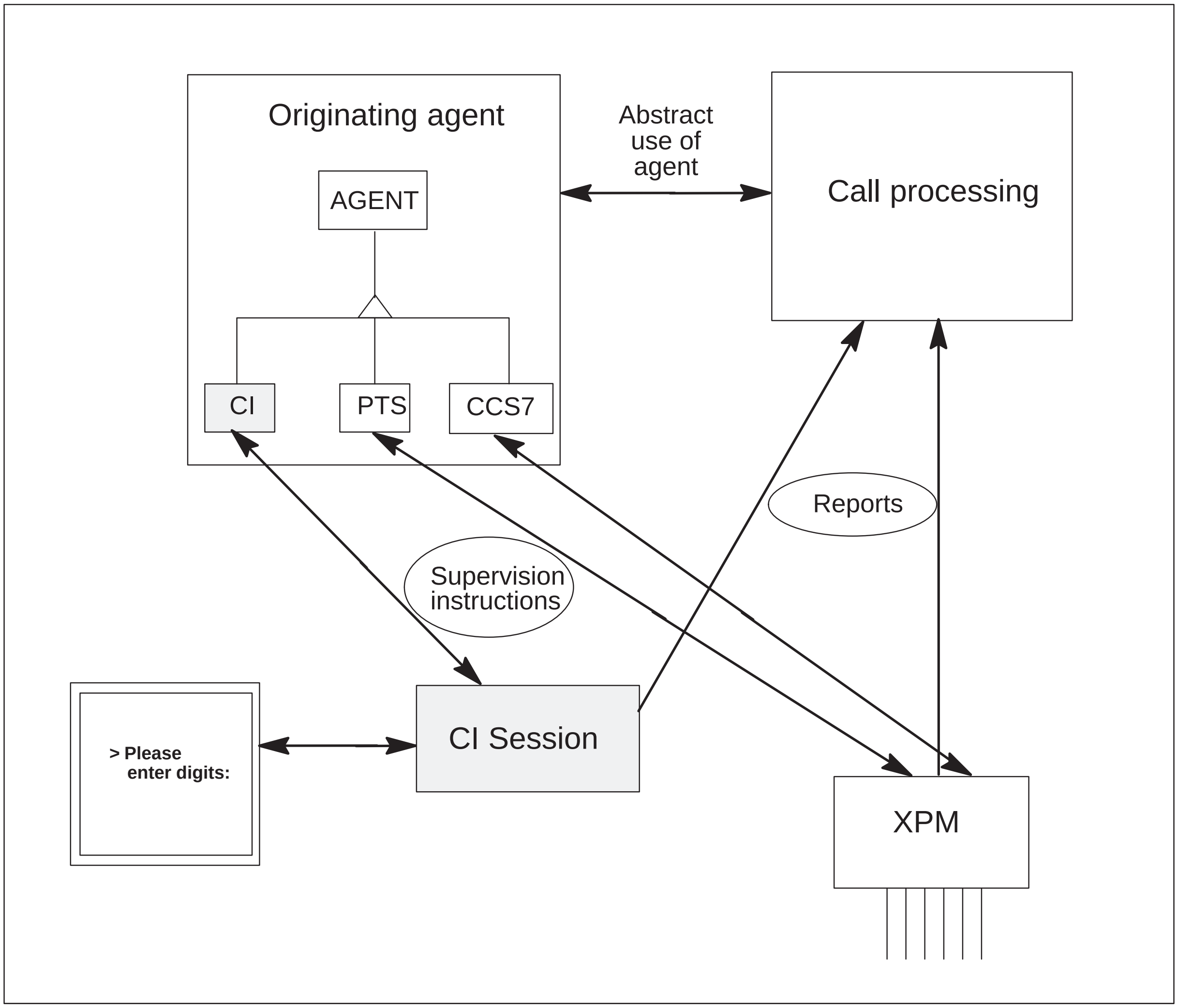
The FLEXSIM command allows access to the following subcommands:

- HELP—displays a list of subcommands with a brief description
- SELECT (SEL)—selects the trunk used for call simulation
- INPUT (IN)—selects inputs from user, file, or preset buffers
- DIGITS (DIGS)—defines digits for call simulation
- ISUPNDX (ISUP)—defines an ISUPTST index for call simulation
- LEVEL (LEV)—defines the level of detail for the report
- FLXREAD—executes FLEXSIM subcommands from a predefined file
- STATUS (STAT)—displays current status of FLEXSIM values
- START (GO)—starts call simulation
- DISPLAY—displays the integrated services user part (ISUP) IAM in table FLEXSIMT
- ABORT—cancels the call simulation in progress
- QUIT—exits the FLEXSIM command

The FlexDial simulator works by serving as the extended multiprocessor system peripheral (XPM) for the AXXESS agent being simulated. For normal trunk group members, interaction with the XPM is required to initiate and complete a call. Through call processing abstraction, FLEXSIM initiates an AXXESS agent call using a “CI agent” instead of an XPM agent member.

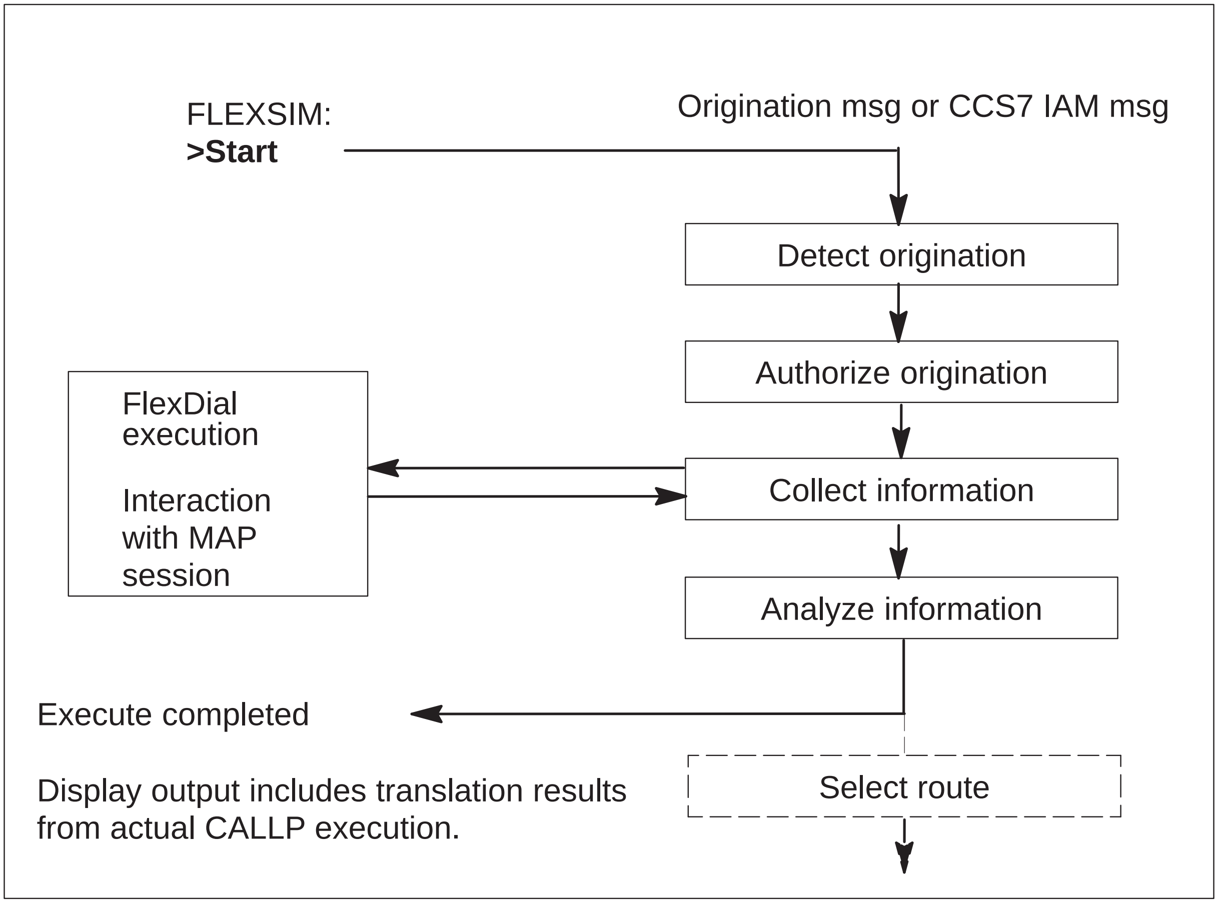
Figure 24-1 provides an overview of the FLEXSIM CI agent.

Figure 24-1
FLEXSIM CI agent



The FLEXSIM tool simulates call processing, with the MAP session identified as the originating agent. Although an actual call process is initiated, only the first four points of the call model are executed before FLEXSIM exits the call process, as shown in Figure 24-2.

Figure 24-2
FLEXSIM call processing execution



FLEXSIM command overview

The FLEXSIM command and its associated subcommands are described in the following sections. Command conventions are also included.

FLEXSIM command and subcommands functionality

The SELECT, DIGITS, ISUPNDX, LEVEL, and INPUT subcommands define values used by call simulation. These commands set or change the values specified and do not interact with the call simulation process. The START subcommand begins the call simulation, which uses the values the user selected with the other subcommands.

In addition to entering the subcommands separately, you can create a file containing subcommands. This file can be executed using the FLXREAD subcommand followed by the file's name. FLXREAD works the same as the system READ command, but displays the commands in the file before they are executed.

The ABORT subcommand ends the call simulation. Because you can only enter input when a user prompt is present, an ABORT subcommand cannot be entered until you are asked for input. This occurs during call simulation when digits or other call processing information are requested. If a call is not being simulated, the ABORT subcommand aborts the command being entered.

While a call is being simulated, the terminal displays the collectables that are processed for the call. The FLEXSIM terminal display shows the switch setting up the call based on processed FLEXSIM subcommand values and FLEXSIM entry, and displays the FLEXDIAL table entries used to process the call. The level of detail displayed is determined by the level set by the LEVEL subcommand.

FLEXSIM command conventions

The following conventions are used to clarify instructions and commands. However, when the commands are entered, the case is not considered and the special characters are not allowed.

- All commands and keywords are in upper case.
- User-supplied values are in lower case.
- Parameters for each subcommand are enclosed in angled brackets (< >).
- Optional parameters are enclosed in brackets ([ ]).
- Default values are in bold type.

FLEXSIM command syntax and examples

The FLEXSIM command and subcommands syntax, default values, valid values, and examples are shown in this section.

FLEXSIM command

FLEXSIM [<TRUNK clli>]

Enter the FLEXSIM command to access the FLEXSIM level of the CI. The FLEXSIM subcommands are available from the FLEXSIM level. A trunk may be entered as an optional parameter to the FLEXSIM command.

```
FLEXSIM
FLEXSIM TRUNK axdal231twdtls
```

Enter the FLEXSIM command, followed by a trunk name, to both access the FLEXSIM level of the CI and to start call simulation using the specified trunk. Entering this command is the same as entering the FLEXSIM command, followed by the SELECT subcommand and the START subcommand. The TRUNK keyword is not needed.

FLEXSIM axdal231twdtls
FLEXSIM TRUNK axdal231twdtls

If the CLLI you selected is not in the CLLI table, you will get the following error message:

```
Clli <clli> is not in table CLLI.
```

If the CLLI you selected is not a trunk group CLLI (provisioned in table TRKGRP), you will still enter the FLEXSIM directory, but you will get the following error message:

```
Clli <clli> is not in TRKGRP. Add clli to table TRKGRP, then
use select again.
```

If the CLLI you selected is defined in table TRKGRP, but is not an AXXESS trunk group type, then you will still enter the FLEXSIM directory, but you will get the following error message:

```
Trunk <clli> is not an axxess trunk.
```

If the CLLI you selected is an SS7 trunk, the call will not be executed until you select an ISUPNDX. You will be prompted to select the ISUPNDX after you enter the FLEXSIM directory.

HELP subcommand

HELP

Enter the HELP subcommand to display a list of subcommands and a brief description of each. If you enter a valid subcommand as a parameter, help is displayed for the specified command.

If you do not specify a parameter, the HELP subcommand provides the following output display:

```
FLEXSIM:
>help
FLEXSIM is a CI command which simulates call
origination with FLEXDIAL.

Optional parameters are:
TRUNK followed by a valid trunk group.

The available subcommands are:

HELP    Displays help; If followed by a valid command
```

```
        help on the specific command is displayed.
SELECT  Selects a trunk clli to be tested.
        Alias is SEL.
INPUT   Selects method of getting digits for call processing.
        PROMPT means the user will enter the data when
        requested.
        BUFFER means the digits will be read from the digits
        buffer or the isupndx buffer.
        Alias is IN.
DIGITS  Holds digits read by call processing when needed.
        Alias is DIGS.
ISUPNDX Holds an isup table index used by call processing
        if needed.
        Alias is ISUP.
DISPLAY Displays the IAM message datafilled in table
        FLEXSIMT.
LEVEL   Specifies the level of detail displayed on the
        report. Level 1 shows the least detail and Level 3
        shows the most detail.
        Alias is LEV.
FLXREAD Selects a file which contains FLEXSIM commands. The
        file is read and executed when the read command is
        entered.
STATUS  Shows the current settings of the commands.
        Alias is STAT.
START   Begins the call processing simulation. You must
        select a trunk clli before start can execute.
        Alias is GO.
ABORT   Aborts the current command in process.
QUIT    Exits the Flexsim command level
```

When executed with a specific command option, HELP provides the following display outputs:

```
FLEXSIM:
>help select
The SELECT subcommand selects the trunk clli used to process
the call simulation.
Parms: <trunk group> STRING

FLEXSIM:
>help input
The INPUT subcommand indicates whether call processing digits
will be input by the user (PROMPT), read from the digits or
isupndx buffers (BUFFER) or read from a file (your file
name).
Parms: <input_key> {PROMPT,
                    BUFFER}
```

```
FLEXSIM:
>help digits
The DIGITS subcommand allows you to put digits in a buffer to
be used by call processing simulation.
Parms: <digits buffer> STRING

FLEXSIM:
>help isupndx
The ISUPNDX subcommand sets an IAM message index to table
FLEXSIMT. This table is used by the call processing
simulation to build the IAM message when an SS7 trunk is
selected.
Parms: <isup index> {1 TO 9}

FLEXSIM:
> help display
The DISPLAY subcommand displays the IAM in TABLE FLEXSIMT
in the tuple indexed by the isupndx.
Parms: <isup index> {1 TO 9}

FLEXSIM:
>help level
The LEVEL subcommand sets the level of detail in the report.
Level 1 displays general call processing. Level 2 adds event
information. Level 3 adds data information.
Parms: <report level> {1 TO 3}

FLEXSIM:
>help flxread
The FLXREAD command selects a file which contains FLEXSIM
commands. The file is read, the commands in the file are
displayed, then the commands are executed.
The file must contain valid flexsim commands with valid
parameters.
The file cannot execute the START or GO command.
Parms: <filename> STRING

FLEXSIM:
>help status
The STATUS subcommand displays the current settings of
flexsim values.

FLEXSIM:
>help start
The START subcommand starts call processing simulation.
Parms: [<read file> STRING]
```

SELECT subcommand

SELECT <clli>
SEL <clli>

Enter the SELECT subcommand to select a trunk to use in the call simulation. The command verifies the trunk when it is entered. If the trunk group does not exist or cannot be processed, the terminal displays an error message and you must select another trunk. The trunk must be an AXXESS trunk.

```
SELECT axdal231twdtls
```

If you select a CLLI that is not in the CLLI table, you will get the following error message:

```
FLEXSIM:
> select invalidclli
CLLI invalidclli is not in the clli table.
```

If you select a CLLI that is not a trunk group CLLI (provisioned in table TRKGRP), you will get the following error message:

```
FLEXSIM:
> select invalidclli
Clli invalidclli is not in TRKGRP. Add clli
to table TRKGRP, then use SELECT again.
```

If you select a CLLI that is defined in table TRKGRP, but is not an AXXESS trunk group type, you will get the following error message:

```
FLEXSIM:
> select nonaxxessclli
Trunk nonaxxessclli is not an axxess trunk.
```

INPUT subcommand

INPUT <PROMPT/BUFFER>
IN <PROMPT/BUFFER>

Enter the INPUT subcommand to define how call simulation will retrieve additional input when it is needed to continue the call. If you want to enter the data through the keyboard, use the following default keyword PROMPT:

```
INPUT PROMPT
```

If you define the data using the DIGITS subcommand, use the following keyword BUFFER:

```
INPUT BUFFER
```

DIGITS subcommand

DIGITS < user input >
DIGS < user input >

Enter the DIGITS subcommand to define call simulation input values in a buffer. If the subcommand INPUT BUFFER has been executed, these values are read when additional data is needed to continue the call processing. Some possible values are ANI digits and AUTHCODES. If additional digits are needed after all values have been read from the buffer, you will be asked to input the digits from the keyboard. Enter all digits needed to simulate the call into the buffer in a continuous string without spaces. The digits buffer holds a maximum of 64 digits.

```
DIGITS 2145552211
DIGITS kp2135555555stp
DIGITS kp2135555555stp6112211
```

If you enter a character that is not a valid digit (0–9,A,B,C,D,S, P, KP, KPP, ST, STP, ST2P, ST3P), you are prompted to re-enter the digit string.

Valid digits for DTMF parts of the buffer are 0–9,A,B,C,D,S, and P.

Valid digits for an MF stream are KP or KPP, followed by digits 0–9, and ending with ST, STP, ST2P, or ST3P.

If you enter invalid digits, you will get the following error message:

```
FLEXSIM:
>digits r0t
MF streams must start with KP or KPP followed by digits 0–9
and end with ST, STP, ST2P, or ST3P.

Valid digits for DTMF streams are:
   0–9,A,B,C,D,S,P

Invalid Digits Entered: <digits buffer> STRING
Enter: <digits buffer>
```

ISUPNDX subcommand

ISUPNDX <index into FLEXSIMT table>
ISUP <index into FLEXSIMT table>

Enter the ISUPNDX subcommand to select the ISUP index into the FLEXSIMT table. This index accesses an ISUP IAM message. The index is used to build the IAM message when an SS7 trunk is selected. Valid values are 1–9. The table must contain an IAM message at the index selected before this command will execute. Refer to the section on table FLEXSIMT (page 24-45) for information on how to datafill the table for use by FLEXSIM.

ISUPNDX 3

If you enter an isupndx value that is not within the allowable range, you are prompted to re-enter the ISUPNDX value with the following message:

```
FLEXSIM:
>isupndx 12
Out of range: <isup index> 1 TO 9
Enter: <isup index>
```

LEVEL subcommand

LEVEL <1–3>
LEV <1–3>

Enter the LEVEL subcommand to set the level of detail displayed in the report. Level 1 shows the translations used to place the call; it is used to confirm the call origination is set up correctly. Level 2 adds event information and includes Level 1. Level 3 adds data flow information and includes Levels 1 and 2.

LEVEL 1

If you enter a level value that is not within the allowable range, you are prompted to re-enter the level value with the following message:

```
FLEXSIM:
>level 4
Out of range: <report level> 1 TO 3
Enter: <report level>
```

FLXREAD subcommand

FLXREAD < file name >

Enter the FLXREAD subcommand to display and execute commands that are defined in a file. Any of the FLEXSIM subcommands can be included in this file, except for the START or GO subcommands. The first word of each line must be a valid subcommand.

FLXREAD MyFile

If MyFile contains the subcommands shown below, the switch selects trunk axdal231twdtls, sets the digits buffer to kp2135555555stp6112211, and directs FLEXSIM to use the digits defined in the digits buffer when digits are requested during call processing.

```
MyFile:
   select axdal231twdtls
   digits KP2135555555stp6112211
   input buffer
```

STATUS subcommand

STATUS

STAT

Enter the STATUS subcommand to show the current status of the FLEXSIM subcommands. This command does not have any parameters. You will get the following output display:

```
FLEXSIM:
>status
Current Status of FLEXSIM Variables
        SELECT  –> NONE
        INPUT   –> PROMPT
        DIGITS  –> NONE
        ISUPNDX –> 0
        LEVEL   –> 1
        VID     –> SELECT TID#00B #0840
```

START subcommand

START

GO

Enter the START subcommand to start the call simulation. The call simulation uses the values entered using other FLEXSIM subcommands. At a minimum, the SELECT subcommand must be executed first so that a trunk is known for the call simulation. If you select an ISUP trunk, you must also enter a valid ISUPNDX. This command does not have any parameters.

After you enter the START subcommand, no other input is accepted until you are prompted for data. Once you are prompted, you can enter the data. You can halt the processing using the ABORT subcommand.

See “FLEXSIM display description” on page 24-15 for an example of the START subcommand display and interaction.

DISPLAY subcommand

DISPLAY <index into FLEXSIMT table>

Enter the DISPLAY subcommand to display the IAM message from the FLEXSIMT table. Valid values are 1–9. The table must contain an IAM message at the index you selected before this command will execute. Refer to the table FLEXSIMT section on page 24-45 for information on how to datafill the table for use by FLEXSIM.

DISPLAY 3

If you enter a display value that is not within the allowable range, you are prompted to re-enter the index, as shown below:

```
FLEXSIM:
>display 12
Out of range: <isup index> 1 TO 9
Enter: <isup index>
```

The following is an example of the output from the DISPLAY subcommand:

```
FLEXSIM:
>display 1
01 10 08 00 00 03 06 0C 03 80 90 A2 06 81 10 22 10 32 04 EB
07 03 10 12 84 17 22 11 EA 01 00 00

CCS7 Message Type: ISUP_IAM
NATURE OF CONNECTION:
                     ISUP_NO_SATELLITE,  NO_ISUP_CONT_CHK
                     ISUP_HALF_ECHO_SUP
FORWARD CALL INDICATORS:
   ISUP_Call Type: ISUP_NATIONAL_CALL
   E_to_E_Method: NO_E_TO_E_METHOD_AVAIL
   ISUP_Interworking: INTERWORKING_ENCOUNTERED
   E_to_E_Info_Check: NO_E_TO_E_INFO
   ISDN_UP_Ind: NOT_ISUP_ALL_THE_WAY
   ISDN_UP_Pref_Ind: ISDN_UP_PREF_ALL_THE_WAY
   ISDN_Access_Ind: ACCESS_NON_ISDN
CALLING PARTY CATEGORY:  ISUP_CPC_UNKNOWN
MAND.VARIABLE PART:
   CALLED PARTY:
   EVEN_ODD_IND  : ODD_NUM_NIBBLES
   NATURE_OF_ADD : SUBSCRIBER_NUMBER
   NUMBERING_PLAN : ISDN_TELEPHONY_NUM_PLAN
   ADDRESS_INFORMATION = 2201234
```

```
OPTIONAL PART:  CHARGE NUMBER : NATURE_OF_ADDR_IND
    CLGP_ANI_NATIONAL_NUMBER
   NUMBERING_PLAN : ISDN_TELEPHONY_NUM_PLAN
   ADDR_SIGNAL = 2145552211
ORIGINATING LINE INFORMATION :
   ORIG_LINE_IND : NORMAL_ANI
```

ABORT subcommand

ABORT

Enter the ABORT subcommand to cancel the simulated call that is in progress. You can only enter this subcommand when data is requested during call simulation. This does not affect the values you selected, and it does not exit the FLEXSIM level. This command does not have parameters.

QUIT subcommand

QUIT

Enter the QUIT subcommand to exit the FLEXSIM level of the CI. This command does not have parameters.

FLEXSIM peripheral interaction

To interact with call processing “realistically”, the FLEXSIM map interface duplicates the methodology of the XPM as much as possible with gathering of inband digits for digit requests. When inband digits are required, a “Peripheral Interaction” banner is output, and you are requested to enter digits on the command line.

A peripheral interaction may or may not actually request you to enter digit information, depending on the availability of buffered digits.

Non-buffered digit peripheral interaction

A non-buffered digit peripheral interaction requires you to enter digit information. The following is an example of the peripheral interaction banner and output:

```
********************************************************
Peripheral Interaction
********************************************************
Prompt : Tone—DIAL_TONE_350_440HZ
Digits Required: 1 (min) , 6 (max)
Digit Set: NONE
Signaling: DTMF
(Use $ to terminate input without entering digits.)
Next par is: <Digits> STRING
```

```
Enter: <Digits>
>
```

The output lines give you the following information about the interaction:

- Prompt

  The prompt field identifies the audible proceed to send the tone that is being applied for the interaction. This field may also specify a tone or announcement CLLI.

- Digits Required

  The digits required field identifies the number of digits being requested by the call process.

- Digit Set

  The digit set field identifies the available digits that FLEXSIM has available to report to the call process.

- Signaling

  The signaling field identifies the pulse type for the inband collection. This field contains a value of MF or DTMF.

Enter the digits by typing them in at the MAP display. For DTMF digit entry, you can only use the characters 0–9, S (asterisk), P (octothorpe), and A, B, C, D (4th column DTMF digits).

For MF digit entry, use only the characters 0–9, and KP, KPP, ST, STP, ST2P, and ST3P. Delimit MF digit streams by KP and ST digits. For example, you can enter the following: KP2145551212ST.

You are prompted to re-enter the digits until only valid digit characters have been received. To enter no digits, use the $ character. Alternatively, you can use the key word ABORT to abort the FlexDial simulation.

After you enter the desired digits, the peripheral interaction completes with a display such as the following:

```
Enter: <Digits>
>2135556789

Received Digit Set: 2135556789

Reporting: 2
Remaining: 135556789
********************************************************
 End Peripheral Interaction
```

```
********************************************************
```

The remaining interaction display provides the following information for the user:

- Received Digit Set

  The received digit set field acknowledges the digits you have entered.

- Reporting

  The reporting field identifies the digits being reported to the call process for processing.

- Remaining

  The remaining field identifies digits you entered on the command line, but are being held by FLEXSIM to be reported in subsequent peripheral interactions.

Buffered digit peripheral interaction

A buffered digit peripheral interaction does not require you to enter digit information, as FLEXSIM is currently storing enough digits to satisfy the requirements of the call processing request. The peripheral interaction banner and output look like the following example:

```
********************************************************
Peripheral Interaction
********************************************************
Digits Required: 5(min) , 5(max)
Digit Set: 132436789
Signaling: DTMF

Reporting: 13243
Remaining: 6789
********************************************************
 End Peripheral Interaction
********************************************************
```

The field information carries an identical meaning to the fields for the non-buffered digit peripheral interaction. In this case, FLEXSIM contains a number of digits identified by the Digit Set field that are used to satisfy the call processing request.

FLEXSIM display

For FLEXSIM, the user screen displays reports generated while the call is being simulated. After you enter the START subcommand, the details from the call simulation are displayed sequentially as different events occur. If additional values are needed to process the call, the terminal displays an input request. At this time, you may either enter the next values or the values are read from the DIGITS buffer, the ISUPNDX buffer, or a file.

Display descriptions

Level 1

Level 1 shows the routing of the call with the collectables being executed and FlexDial indexes being accessed. The digits buffer is also displayed, before and after digits are processed. Validation is shown when the collectable does validation.

In general, the FLEXSIM output shows the execution of each collectable for the call interaction. The information displayed contains the following highlights:

- The currently executing collectable in the collectable list is identified by “>> <<” brackets.
- The FLEXDIAL table index containing the currently executing collectable is displayed.
- Before and after peripheral interactions, the contents of the call processing digit buffer are displayed.
- The results of validation attempts for digit collectables is displayed.
- The results of translations are provided once the interaction is complete.

Level 2

Level 2 displays the Level 1 information and adds information about events which drive FLEXDIAL processing.

Level 3

Level 3 displays Level 1 and Level 2 information and adds information about data processing, features, and messages from MSGCTR.

FLEXDIAL entity outputs

The following tables show the collectables and events printed at each FLEXSIM level. Level 2 also includes everything printed at Level 1; Level 3 includes everything printed at Level 2. NA means that there is no additional data printed at this level.

Each event also calls the current collectable print object to print collectable specific data.

Collectable print entities

The tables in this section show the Protocol, Framework, Digit, Sequence, Call Type, and Variable collectables printed at each FLEXSIM level.

Table 24-1 lists the protocol collectables printed at Levels 1, 2, and 3.

Table 24-1
FLEXSIMT print entities for FLEXDIAL protocol collectables

Print entity	Level 1	Level 2	Level 3
SIG	NA	NA	MF signaling info DTMF signaling info
SIGDTMF	NA	NA	DTMF signaling info
RCVSIG	NA	NA	NA
SNDSIG	NA	NA	Signaling information
—end—			

Table 24-2 lists the framework collectables printed at Levels 1, 2, and 3.

Table 24-2
FLEXSIMT print entities for FLEXDIAL framework collectables

Print entity	Level 1	Level 2	Level 3
TERMINATE	Processing direction processing options	NA	NA
APTRMT	NA	NA	Treatment set
GOTO	Flexdial index	NA	NA
DO	Flexdial index	NA	NA
INCLUDE	Flexdial index	NA	NA
ROUTE	Route information	NA	NA
NOOP	NA	NA	NA
IFTRMT	NA	NA	Treatment information Comparison results
IFPROMPT	NA	NA	Prompt information Comparison results
IFTOD	NA	NA	Time of day options Comparison results
—end—			

Table 24-3 lists the sequence collectables printed at Levels 1, 2, and 3.

Table 24-3
FLEXSIMT print entities for FLEXDIAL sequence collectables

Print entity	Level 1	Level 2	Level 3
IFDIGS	FLEXDIAL index	NA	Digit information Comparison results
IFCNT	FLEXDIAL index	NA	Digit information Comparison results
IFNOA	FLEXDIAL index	NA	NOA information Comparison results
DELDIGS	NA	NA	Digits options Digits before and after apply features
ADDIGS	NA	NA	Digits options Digits before and after apply features
MODDIGS	NA	NA	Digits options Digits before and after apply features
COPYDIGS	NA	NA	Digits options Digits before and after apply features
MODNOA	NA	NA	NOA info before and after apply features
AGNTDATA	NA	Digits options	Digits before and after apply features
—end—			

Table 24-4 lists the framework collectables printed at Levels 1, 2, and 3.

Table 24-4
FLEXSIMT print entities for FLEXDIAL framework collectables

Print entity	Level 1	Level 2	Level 3
COLDIG	NA	NA	Digits options Digits before and after apply features
SUBR	General validation info FLEXDIAL information	FLEXTYPE	SUBR options Digits information Detailed validation

Table 24-4
FLEXSIMT print entities for FLEXDIAL framework collectables (continued)

Print entity	Level 1	Level 2	Level 3
ADDR	General validation info FLEXDIAL information	NA	Digits options Digits before and after apply features Detailed validation
OLI	General validation info	NA	Digits options Digits before and after apply features Detailed validation
CIC	NA	NA	Digits options Digits before and after apply features
REPLDIG	NA	NA	Digits options Digits before and after apply features
COLPARM	NA	NA	Digits options Digits before and after apply features
SUBPARM	General validation info	NA	SUBRPARM options Digits information Detailed validation
ADDRPARM	General validation info	NA	ADDRPARM options Digits information Detailed validation
OLIPARM	General validation info	NA	OLIPARM options Digits information Detailed validation
CICPARM	NA	NA	CICPARM options Digits information Detailed validation

—end—

Table 24-5 lists the Call Type collectables printed at Levels 1, 2, and 3.

Table 24-5
FLEXSIMT print entities for FLEXDIAL Call Type collectables

Print entity	Level 1	Level 2	Level 3
CALLTYPE	NA	Flextype Flexfeat	NA
CLRFTRS	NA	Featlist	NA
SETTRANS	NA	NA	SETTRANS options apply features
OM	NA	NA	OM options OM count
APRESET	NA	Reset options	NA
SETTRMT	Treatment set	NA	NA
—end—			

Table 24-6 lists the variable collectables printed at Levels 1, 2, and 3.

Table 24-6
FLEXSIMT print entities for FLEXDIAL variable collectables

Print entity	Level 1	Level 2	Level 3
VAROP	NA	NA	Var options
IFVAR	NA	NA	Var options Comparison results
—end—			

Event print entities

Table 24-7 shows the output for each event.

Table 24-7
FLEXSIMT event print entities

Print entity	Level 1	Level 2	Level 3
Initial processing	Initial FLEXDIAL index list Executing clli name	NA	NA
MsgAnalysis Event	NA	Event name Next state	NA
Setup Event	NA	Event name Next state	NA
Start event	Collectable list FLEXDIAL index	Event name Next state	NA
Collect Event	NA	Event name Next state	NA
Parse Event	NA	Event name Digits	NA
Prompt Event	Digit buffer Peripheral interaction banner Prompt type / name	Event name	Collectable prompt details
Free Prompt Event	NA	Event name	NA
Load Event	NA	Event name Digits loaded	NA
Reset Event	Reset information	Event name	NA
Validate Event	Validate banner Current collection print	Event name	NA
Rdb Query Event	NA	Event name Type of Query	NA
Rdb Response Event	Rdb response banner Rdb response info	Event name	NA
Apply Ftrs Event	NA	Event name Collectable basic apply feature information	Collectable detailed apply feature information
Exception Event	Current collectable print	Event name	NA
CAIN	CAIN information	NA	NA

—continued—

Table 24-7
FLEXSIMT event print entities (continued)

Print entity	Level 1	Level 2	Level 3
Begin Peripheral Interaction	Digit collection info	NA	NA
End Peripheral Interaction	Digits to be sent Digits remaining End banner	NA	NA
Translations	Routing information	NA	NA
Collectable manager	List of collectables with current collectable marked	NA	NA

—end—

Sample display output

This section includes a sample of a general report for Levels 1, 2, and 3.

Levels 1–3 sample display output

A brief example of the general report format for Levels 1, 2, and 3 follows. This example shows commands and output, starting with entering the FLEXSIM command from the CI prompt.

QTRK is shown for the trunk selected, although this is not part of FLEXSIM. The QTRK CI command displays all provisioning information for an AXXESS trunk group. The associated datafill is displayed for the specified AXXESS trunk group CLLI.

See Chapter 14 for more information on the QTRK tool.

```
>qtrk axdal231twdtls

Axxess Trunk Group Display for CLLI: AXDAL231TWDTLS
TRKGRP:
    AXDAL231TWDTLS AXXESS 127 NPDGP NCON MIDL FXSLSDTLSDT_0003 TRKFEAT_0073
         (SNPA_SNXX_ANI) (I_AD_SD_AU_HSD) $ DAL
TRKSIG Index: FXSLSDTLSDT_0003
   FXSLSDTLSDT_0003 FXS LS DTMF 15 6 5 S1234567890P P M LS DTMF 6 160 UNDEF
         (BCCOMPAT VOICE_DATA) (ORIGFLTR 7) (ANSWFLTR 16) (RETOFFHK NIL)
         (DIGSOUTP 10) $
TRKFEAT Index: TRKFEAT_0073
    TRKFEAT_0073 (CITYCODE 111)
         (REORIGAL ( AUTH) $) (SNPA 214) (TRANSTS PARTITION 111 0)
         (MSGCTR 2060) $ (OHQTERM ) (ONNETTRK ) (SNPA 214) (TRKCOS 1) $
```

```
MSGCTR Indices: 2060
    2060 (ADDR N Y PRTNM DAL) (ADDR N Y OPER NORMAL OFRT 1001 DAL NONE ) $

FLEXDIAL Indices: SNPA_SNXX_ANI
                   I_AD_SD_AU_HSD
    SNPA_SNXX_ANI (AGNTDATA PREFIX SNXX) (AGNTDATA PREFIX SNPA)
         (SUBR 6 6 ANI N N $ $)
         (OLI 2 2 $ ( FILED PREFIX 02) $)$ N
    I_AD_SD_AU_HSD (INCLUDE C_ADDR_SD) $ Y I_AUTH_HSD
TRKMEM Member List: 2 trunk members
    AXDAL231TWDTLS 1 0 DTC 1 5 2
    AXDAL231TWDTLS 2 0 DTC 3 16 2

CI:
```

> flexsim

```
FLEXSIM:
```

> select axdal231twdtls

> status

```
Current Status of FLEXSIM Variables
    SELECT  -> AXDAL231TWDTLS
    INPUT   -> PROMPT
    DIGITS  -> NONE
    ISUPNDX -> 0
    LEVEL   -> 1
    VID     -> SELECT TID #00B #83D
```

> start

```
Executing FLEXDIAL Call Processing Simulation for: AXDAL231TWDTLS
Initial FLEXDIAL Index(s): (SNPA_SNXX_ANI) (I_AD_SD_AU_HSD)
Digits Buffer   Digits :
                Reset  : NO   Term : NO   Timeout : NO   CSP: NO
                NOA    : UNKNOWN_UNIQUENESS
                CarrSel: NO_INDICATION

Flexdial Index: SNPA_SNXX_ANI
Collectable List: (AGNTDATA) >>AGNTDATA<< (SUBR) (OLI) (INCLUDE)
                  (SUBR)

Flexdial Index: SNPA_SNXX_ANI
Collectable List: (AGNTDATA) (AGNTDATA) >>SUBR<< (OLI) (INCLUDE)
                  (SUBR)

Flexdial Index: SNPA_SNXX_ANI
Collectable List: (AGNTDATA) (AGNTDATA) (SUBR) >>OLI<< (INCLUDE)
                  (SUBR)
```

```
Flexdial Index: I_AD_SD_AU_HSD
Collectable List: (AGNTDATA) (AGNTDATA) (SUBR) (OLI) >>INCLUDE<<
                  (SUBR)

Flexdial Index: C_ADDR_SD
Collectable List: (AGNTDATA) (AGNTDATA) (SUBR) (OLI) (INCLUDE)
                  >>ADDR<< (SUBR)

Digits Buffer   Digits :
                Reset  : NO   Term : NO   Timeout : NO   CSP: NO
                NOA    : UNKNOWN_UNIQUENESS
                CarrSel: NO_INDICATION
********************************************************
 Peripheral Interaction
********************************************************
Prompt : Tone – DIAL_TONE_350_440HZ
Digits Required  : 1 (min), 6 (max)
Digit Set  : NONE
Signalling  : DTMF
(Use $ to terminate input without entering digits.)
Invalid Digits Entered: <Digits> STRING
Enter: <Digits>
```

>2142201234

```
Received Digit Set : 2142201234
Reporting:  2
Remaining:  142201234
********************************************************
 End Peripheral Interaction
********************************************************
Digits Buffer   Digits :
                Reset  : NO   Term : NO   Timeout : NO   CSP: NO
                NOA    : UNKNOWN_UNIQUENESS
                CarrSel: NO_INDICATION
Collectable List: (AGNTDATA) (AGNTDATA) (SUBR) (OLI) (INCLUDE)
                  >>ADDR<< (SUBR)

********************************************************
Peripheral Interaction
********************************************************
Digits Required  : 5 (min), 5 (max)
Digit Set  : 142201234
Signalling  : DTMF
Reporting:  14220
Remaining:  1234
********************************************************
Peripheral Interaction
********************************************************
```

```
Digits Required  : 5 (min), 5 (max)
Digit Set  : 142201234
Signalling  : DTMF
Reporting:  14220
Remaining:  1234
********************************************************
 End Peripheral Interaction
********************************************************
Digits Buffer   Digits : 2
                Reset  : NO   Term : NO   Timeout : NO   CSP: NO
                NOA    : UNKNOWN_UNIQUENESS
                CarrSel: NO_INDICATION
Collectable List: (AGNTDATA) (AGNTDATA) (SUBR) (OLI) (INCLUDE)
                  >>ADDR<< (SUBR)

******Validate state data******
pretranslator name:                DAL
pretranslation digits:             214220
pretranslator route selector:      CT
call_feature:                      OFFNET
********************************************************
Peripheral Interaction
********************************************************
Digits Required  : 1 (min), 4 (max)
Digit Set  : 1234
Signalling  : DTMF
Reporting:  1
Remaining:  234
********************************************************
 End Peripheral Interaction
********************************************************
Digits Buffer   Digits :
                Reset  : NO   Term : NO   Timeout : NO   CSP: NO
                NOA    : UNKNOWN_UNIQUENESS
                CarrSel: NO_INDICATION
Collectable List: (AGNTDATA) (AGNTDATA) (SUBR) (OLI) (INCLUDE)
                  >>ADDR<< (SUBR)

********************************************************
Peripheral Interaction
********************************************************
Digits Required  : 3 (min), 3 (max)
Digit Set  : 234
Signalling  : DTMF
Reporting:  234
Remaining:  NONE
********************************************************
 End Peripheral Interaction
********************************************************
```

```
Digits Buffer   Digits : 1
                Reset  : NO   Term : NO   Timeout : NO   CSP: NO
                NOA    : UNKNOWN_UNIQUENESS
                CarrSel: NO_INDICATION
Collectable List: (AGNTDATA) (AGNTDATA) (SUBR) (OLI) (INCLUDE)
                  >>ADDR<< (SUBR)

******Validate state data******
pretranslator name:                DAL
pretranslation digits:             2142201234
pretranslator route selector:      CT
call_feature:                      OFFNET
called_noa:                        NATL
Flexdial Index: I_AUTH_HSD
Collectable List: (AGNTDATA) (AGNTDATA) (SUBR) (OLI) (INCLUDE)
                  (ADDR) >>SUBR<<

Digits Buffer   Digits :
                Reset  : NO   Term : NO   Timeout : NO   CSP: NO
                NOA    : UNKNOWN_UNIQUENESS
                CarrSel: NO_INDICATION
********************************************************
 Peripheral Interaction
********************************************************
Prompt : Tone – PROMPT_TONE
Digits Required  : 5 (min), 7 (max)
Digit Set  : NONE
Signalling  : DTMF
(Use $ to terminate input without entering digits.)
Next par is: <Digits> STRING
Enter: <Digits>
```

>6112211

```
Received Digit Set : 6112211
Reporting:  61122
Remaining:  11
********************************************************
 End Peripheral Interaction
********************************************************
Digits Buffer   Digits :
                Reset  : NO   Term : NO   Timeout : NO   CSP: NO
                NOA    : UNKNOWN_UNIQUENESS
                CarrSel: NO_INDICATION
Collectable List: (AGNTDATA) (AGNTDATA) (SUBR) (OLI) (INCLUDE)
                  (ADDR) >>SUBR<<

********************************************************
Peripheral Interaction
********************************************************
```

```
Digits Required  : 2 (min), 2 (max)
Digit Set  : 11
Signalling  : DTMF
Reporting:  11
Remaining:  NONE
********************************************************
 End Peripheral Interaction
********************************************************
Digits Buffer   Digits : 61122
                Reset  : NO   Term : NO   Timeout : NO   CSP: NO
                NOA    : UNKNOWN_UNIQUENESS
                CarrSel: NO_INDICATION
Collectable List: (AGNTDATA) (AGNTDATA) (SUBR) (OLI) (INCLUDE)
                  (ADDR) >>SUBR<<

****** Validate ******
Subscriber Type  : AUTH
Collected Digits : 6112211
Validation Type  : FLEXVAL
Validation Status: No Failure
FLEXVAL Index    :       1,   FLEXFEAT Index:   110111
Flexdial Index: I_AUTH_HSD
Collectable List: (AGNTDATA) (AGNTDATA) (SUBR) (OLI) (INCLUDE)
                  (ADDR) >>SUBR<<

***********************************************
TRANSLATIONS
***********************************************
  TRANSLATE
    HNPA_RTER 1A00DC
      Dialed    : 2142201234
      STS       : 611
      TRANS SYS : NA
      CALL TYPE : NP
      ROUTE     : CKT    DAL220TWDTGS    1
********************************************************
 Flexsim Execution Complete for Trunk: AXDAL231TWDTLS
********************************************************
```

>level 2

>status

```
  Current Status of FLEXSIM Variables
    SELECT  –> AXDAL231TWDTLS
    INPUT   –> PROMPT
    DIGITS  –> NONE
    ISUPNDX –> 0
    LEVEL   –> 2
    VID     –> SELECT TID #00B #83D
```

>start

```
Executing FLEXDIAL Call Processing Simulation for: AXDAL231TWDTLS
Initial FLEXDIAL Index(s): (SNPA_SNXX_ANI) (I_AD_SD_AU_HSD)
Digits Buffer   Digits :
                Reset  : NO   Term : NO   Timeout : NO   CSP: NO
                NOA    : UNKNOWN_UNIQUENESS
                CarrSel: NO_INDICATION

--------------------------------------
 MsgAnalysis Event
--------------------------------------
  Next State:Setup State
--------------------------------------
--------------------------------------
 Setup Event
--------------------------------------
  Processing Collectable:AGNTDATA
--------------------------------------
 Start Event
--------------------------------------
  Processing Collectable:AGNTDATA
--------------------------------------
 ApplyFtrs Event
--------------------------------------
***** Collectable: AgntDataCol *****
Loc: PREFIX
digType: SNXX
agntDgt: 000
***** Digit Buffer *****
Digits Buffer   Digits : 000
                Reset  : NO   Term : NO   Timeout : NO   CSP: NO
                NOA    : UNKNOWN_UNIQUENESS
                CarrSel: NO_INDICATION
--------------------------------------
 Collect Event
--------------------------------------
Flexdial Index: SNPA_SNXX_ANI
Collectable List: (AGNTDATA) >>AGNTDATA<< (SUBR) (OLI) (INCLUDE)
                  (SUBR)
--------------------------------------
 Start Event
--------------------------------------
  Processing Collectable:AGNTDATA
--------------------------------------
 ApplyFtrs Event
--------------------------------------
```

```
***** Collectable: AgntDataCol *****
Loc: PREFIX
digType: SNPA
agntDgt: 214
***** Digit Buffer *****
Digits Buffer   Digits : 214000
                Reset  : NO   Term : NO   Timeout : NO   CSP: NO
                NOA    : UNKNOWN_UNIQUENESS
                CarrSel: NO_INDICATION
--------------------------------------
 Collect Event
--------------------------------------
Flexdial Index: SNPA_SNXX_ANI
Collectable List: (AGNTDATA) (AGNTDATA) >>SUBR<< (OLI) (INCLUDE)
                  (SUBR)

--------------------------------------
 Start Event
--------------------------------------
  Processing Collectable:SUBR
--------------------------------------
 Parse Event
--------------------------------------
--------------------------------------
 ApplyFtrs Event
--------------------------------------
--------------------------------------
 Collect Event
--------------------------------------
Flexdial Index: SNPA_SNXX_ANI
Collectable List: (AGNTDATA) (AGNTDATA) (SUBR) >>OLI<< (INCLUDE)
                  (SUBR)

--------------------------------------
 Start Event
--------------------------------------
  Processing Collectable:OLI
--------------------------------------
 Parse Event
--------------------------------------
--------------------------------------
 ApplyFtrs Event
--------------------------------------
--------------------------------------
 Collect Event
--------------------------------------
Flexdial Index: I_AD_SD_AU_HSD
Collectable List: (AGNTDATA) (AGNTDATA) (SUBR) (OLI) >>INCLUDE<<
                  (SUBR)
```

```
***** Collectable: INCLUDE *****
--------------------------------------
 Start Event
--------------------------------------
  Processing Collectable:INCLUDE
***** Collectable: INCLUDE *****
INCLUDE Flexdial index: C_ADDR_SD
--------------------------------------
 ApplyFtrs Event
--------------------------------------
***** Collectable: INCLUDE *****
INCLUDE Flexdial index: C_ADDR_SD
--------------------------------------
 Collect Event
--------------------------------------
Flexdial Index: C_ADDR_SD
Collectable List: (AGNTDATA) (AGNTDATA) (SUBR) (OLI) (INCLUDE)
                  >>ADDR<< (SUBR)

--------------------------------------
 Start Event
--------------------------------------
  Processing Collectable:ADDR
--------------------------------------
 Parse Event
--------------------------------------
--------------------------------------
 Prompt Event
--------------------------------------
Digits Buffer   Digits :
                Reset  : NO   Term : NO   Timeout : NO   CSP: NO
                NOA    : UNKNOWN_UNIQUENESS
                CarrSel: NO_INDICATION
********************************************************
 Peripheral Interaction
********************************************************
Prompt : Tone – DIAL_TONE_350_440HZ
Digits Required  : 1 (min), 6 (max)
Digit Set  : NONE
Signalling  : DTMF
(Use $ to terminate input without entering digits.)
Invalid Digits Entered: <Digits> STRING
Enter: <Digits>
```

>2142201234

```
Received Digit Set : 2142201234
Reporting:  2
Remaining:  142201234
********************************************************
 End Peripheral Interaction
********************************************************
Digits Buffer   Digits :
                Reset  : NO   Term : NO   Timeout : NO   CSP: NO
                NOA    : UNKNOWN_UNIQUENESS
                CarrSel: NO_INDICATION
Collectable List: (AGNTDATA) (AGNTDATA) (SUBR) (OLI) (INCLUDE)
                  >>ADDR<< (SUBR)

--------------------------------------
 MsgAnalysis Event
--------------------------------------
  Next State:Load State
--------------------------------------
--------------------------------------
 Load Event
--------------------------------------
********************************************************
Peripheral Interaction
********************************************************
Digits Required  : 5 (min), 5 (max)
Digit Set  : 142201234
Signalling  : DTMF
Reporting:  14220
Remaining:  1234
********************************************************
 End Peripheral Interaction
********************************************************
Digits Buffer   Digits : 2
                Reset  : NO   Term : NO   Timeout : NO   CSP: NO
                NOA    : UNKNOWN_UNIQUENESS
                CarrSel: NO_INDICATION
Collectable List: (AGNTDATA) (AGNTDATA) (SUBR) (OLI) (INCLUDE)
                  >>ADDR<< (SUBR)
--------------------------------------
 MsgAnalysis Event
--------------------------------------
  Next State:Load State
--------------------------------------
--------------------------------------
 Load Event
--------------------------------------
--------------------------------------
 Parse Event
--------------------------------------
```

```
--------------------------------------
 Validate Event
--------------------------------------
******Validate state data******
pretranslator name:                DAL
pretranslation digits:             214220
pretranslator route selector:      CT
call_feature:                      OFFNET
--------------------------------------
 Request Event
--------------------------------------
********************************************************
Peripheral Interaction
********************************************************
Digits Required  : 1 (min), 4 (max)
Digit Set  : 1234
Signalling  : DTMF
Reporting:  1
Remaining:  234
********************************************************
 End Peripheral Interaction
********************************************************
Digits Buffer   Digits :
                Reset  : NO   Term : NO   Timeout : NO   CSP: NO
                NOA    : UNKNOWN_UNIQUENESS
                CarrSel: NO_INDICATION
Collectable List: (AGNTDATA) (AGNTDATA) (SUBR) (OLI) (INCLUDE)
                  >>ADDR<< (SUBR)

--------------------------------------
 MsgAnalysis Event
--------------------------------------
  Next State:Load State
--------------------------------------
--------------------------------------
 Load Event
--------------------------------------
********************************************************
Peripheral Interaction
********************************************************
Digits Required  : 3 (min), 3 (max)
Digit Set  : 234
Signalling  : DTMF
Reporting:  234
Remaining:  NONE
********************************************************
 End Peripheral Interaction
********************************************************
```

```
Digits Buffer   Digits : 1
                Reset  : NO   Term : NO   Timeout : NO   CSP: NO
                NOA    : UNKNOWN_UNIQUENESS
                CarrSel: NO_INDICATION
Collectable List: (AGNTDATA) (AGNTDATA) (SUBR) (OLI) (INCLUDE)
                  >>ADDR<< (SUBR)

--------------------------------------
 MsgAnalysis Event
--------------------------------------
  Next State:Load State
--------------------------------------
--------------------------------------
 Load Event
--------------------------------------
--------------------------------------
 Parse Event
--------------------------------------
--------------------------------------
 Validate Event
--------------------------------------
******Validate state data******
pretranslator name:                DAL
pretranslation digits:             2142201234
pretranslator route selector:      CT
call_feature:                      OFFNET
--------------------------------------
 ApplyFtrs Event
--------------------------------------
******ApplyFtrs state data******
called_noa:                        NATL
--------------------------------------
 Collect Event
--------------------------------------
Flexdial Index: I_AUTH_HSD
Collectable List: (AGNTDATA) (AGNTDATA) (SUBR) (OLI) (INCLUDE)
                  (ADDR) >>SUBR<<

--------------------------------------
 Start Event
--------------------------------------
  Processing Collectable:SUBR
--------------------------------------
 Parse Event
--------------------------------------
--------------------------------------
 Prompt Event
--------------------------------------
```

```
Digits Buffer   Digits :
                Reset  : NO   Term : NO   Timeout : NO   CSP: NO
                NOA    : UNKNOWN_UNIQUENESS
                CarrSel: NO_INDICATION
********************************************************
 Peripheral Interaction
********************************************************
Prompt : Tone – PROMPT_TONE
Digits Required  : 5 (min), 7 (max)
Digit Set  : NONE
Signalling  : DTMF
(Use $ to terminate input without entering digits.)
Next par is: <Digits> STRING
Enter: <Digits>
```

>6112211

```
Received Digit Set : 6112211
Reporting:  61122
Remaining:  11
********************************************************
 End Peripheral Interaction
********************************************************
Digits Buffer   Digits :
                Reset  : NO   Term : NO   Timeout : NO   CSP: NO
                NOA    : UNKNOWN_UNIQUENESS
                CarrSel: NO_INDICATION
Collectable List: (AGNTDATA) (AGNTDATA) (SUBR) (OLI) (INCLUDE)
                  (ADDR) >>SUBR<<

--------------------------------------
 MsgAnalysis Event
--------------------------------------
  Next State:Load State
--------------------------------------
--------------------------------------
 Load Event
--------------------------------------
********************************************************
Peripheral Interaction
********************************************************
Digits Required  : 2 (min), 2 (max)
Digit Set  : 11
Signalling  : DTMF
Reporting:  11
Remaining:  NONE
********************************************************
 End Peripheral Interaction
********************************************************
```

```
Digits Buffer   Digits : 61122
                Reset  : NO   Term : NO   Timeout : NO   CSP: NO
                NOA    : UNKNOWN_UNIQUENESS
                CarrSel: NO_INDICATION
Collectable List: (AGNTDATA) (AGNTDATA) (SUBR) (OLI) (INCLUDE)
                  (ADDR) >>SUBR<<

--------------------------------------
 MsgAnalysis Event
--------------------------------------
  Next State:Load State
--------------------------------------
--------------------------------------
 Load Event
--------------------------------------
--------------------------------------
 Parse Event
--------------------------------------
--------------------------------------
 Validate Event
--------------------------------------
****** Validate ******
Subscriber Type  : AUTH
Collected Digits : 6112211
Validation Type  : FLEXVAL
Validation Status: No Failure
FLEXVAL Index    :       1,   FLEXFEAT Index:   110111
--------------------------------------
 ApplyFtrs Event
--------------------------------------
--------------------------------------
 Collect Event
--------------------------------------
Flexdial Index: I_AUTH_HSD
Collectable List: (AGNTDATA) (AGNTDATA) (SUBR) (OLI) (INCLUDE)
                  (ADDR) >>SUBR<<

***********************************************
TRANSLATIONS
***********************************************
  TRANSLATE
    HNPA_RTER 1A00DC
      Dialed    : 2142201234
      STS       : 611
      TRANS SYS : NA
      CALL TYPE : NP
      ROUTE     : CKT    DAL220TWDTGS    1
********************************************************
 Flexsim Execution Complete for Trunk: AXDAL231TWDTLS
********************************************************
```

>level 3

>status

```
  Current Status of FLEXSIM Variables
    SELECT  -> AXDAL231TWDTLS
    INPUT   -> PROMPT
    DIGITS  -> NONE
    ISUPNDX -> 0
    LEVEL   -> 3
    VID     -> SELECT TID #00B #83D
```

>start

```
Executing FLEXDIAL Call Processing Simulation for: AXDAL231TWDTLS
Initial FLEXDIAL Index(s): (SNPA_SNXX_ANI) (I_AD_SD_AU_HSD)
Digits Buffer   Digits :
                Reset  : NO   Term : NO   Timeout : NO   CSP: NO
                NOA    : UNKNOWN_UNIQUENESS
                CarrSel: NO_INDICATION

--------------------------------------
 MsgAnalysis Event
--------------------------------------
  Next State:Setup State
--------------------------------------
--------------------------------------
 Setup Event
--------------------------------------
  Processing Collectable:AGNTDATA
--------------------------------------
 Start Event
--------------------------------------
  Processing Collectable:AGNTDATA
***** Collectable: AgntDataCol *****
Loc: PREFIX
digType: SNXX
agntDgt: 214
***** Digit Buffer *****
Digits Buffer   Digits :
                Reset  : NO   Term : NO   Timeout : NO   CSP: NO
                NOA    : UNKNOWN_UNIQUENESS
                CarrSel: NO_INDICATION
Current State : 13
--------------------------------------
 ApplyFtrs Event
--------------------------------------
```

```
***** Collectable: AgntDataCol *****
Loc: PREFIX
digType: SNXX
agntDgt: 000
***** Digit Buffer *****
Digits Buffer   Digits : 000
                Reset  : NO   Term : NO   Timeout : NO   CSP: NO
                NOA    : UNKNOWN_UNIQUENESS
                CarrSel: NO_INDICATION
Current State : 13
--------------------------------------
 Collect Event
--------------------------------------
Flexdial Index: SNPA_SNXX_ANI
Collectable List: (AGNTDATA) >>AGNTDATA<< (SUBR) (OLI) (INCLUDE)
                  (SUBR)

--------------------------------------
 Start Event
--------------------------------------
  Processing Collectable:AGNTDATA
***** Collectable: AgntDataCol *****
Loc: PREFIX
digType: SNPA
agntDgt: 000
***** Digit Buffer *****
Digits Buffer   Digits : 000
                Reset  : NO   Term : NO   Timeout : NO   CSP: NO
                NOA    : UNKNOWN_UNIQUENESS
                CarrSel: NO_INDICATION
Current State : 13
--------------------------------------
 ApplyFtrs Event
--------------------------------------
***** Collectable: AgntDataCol *****
Loc: PREFIX
digType: SNPA
agntDgt: 214
***** Digit Buffer *****
Digits Buffer   Digits : 214000
                Reset  : NO   Term : NO   Timeout : NO   CSP: NO
                NOA    : UNKNOWN_UNIQUENESS
                CarrSel: NO_INDICATION
Current State : 13
--------------------------------------
 Collect Event
--------------------------------------
Flexdial Index: SNPA_SNXX_ANI
Collectable List: (AGNTDATA) (AGNTDATA) >>SUBR<< (OLI) (INCLUDE)
                  (SUBR)
```

```
--------------------------------------
 Start Event
--------------------------------------
  Processing Collectable:SUBR
--------------------------------------
 Parse Event
--------------------------------------
--------------------------------------
 ApplyFtrs Event
--------------------------------------
--------------------------------------
 Collect Event
--------------------------------------
Flexdial Index: SNPA_SNXX_ANI
Collectable List: (AGNTDATA) (AGNTDATA) (SUBR) >>OLI<< (INCLUDE)
                  (SUBR)

Name: OLI
Print Data:
UNKNOWN event
--------------------------------------
 Start Event
--------------------------------------
  Processing Collectable:OLI
Name: OLI
Print Data:
Filed: POS PREFIX  DIGITS:              02
State: start
--------------------------------------
 Parse Event
--------------------------------------
Name: OLI
Print Data:
State: parse
--------------------------------------
 ApplyFtrs Event
--------------------------------------
Name: OLI
Print Data:
Captured INFODIGS in CDR:
State: applyftrs
--------------------------------------
 Collect Event
--------------------------------------
Flexdial Index: I_AD_SD_AU_HSD
Collectable List: (AGNTDATA) (AGNTDATA) (SUBR) (OLI) >>INCLUDE<<
                  (SUBR)

***** Collectable: INCLUDE *****
Current State : 13
```

```
--------------------------------------
 Start Event
--------------------------------------
  Processing Collectable:INCLUDE
***** Collectable: INCLUDE *****
INCLUDE Flexdial index: C_ADDR_SD
Current State : 13
--------------------------------------
 ApplyFtrs Event
--------------------------------------
***** Collectable: INCLUDE *****
INCLUDE Flexdial index: C_ADDR_SD
Current State : 13
--------------------------------------
 Collect Event
--------------------------------------
Flexdial Index: C_ADDR_SD
Collectable List: (AGNTDATA) (AGNTDATA) (SUBR) (OLI) (INCLUDE)
                  >>ADDR<< (SUBR)

--------------------------------------
 Start Event
--------------------------------------
  Processing Collectable:ADDR
--------------------------------------
 Parse Event
--------------------------------------
--------------------------------------
 Prompt Event
--------------------------------------
Digits Buffer   Digits :
                Reset  : NO   Term : NO   Timeout : NO   CSP: NO
                NOA    : UNKNOWN_UNIQUENESS
                CarrSel: NO_INDICATION
********************************************************
 Peripheral Interaction
********************************************************
Prompt : Tone – DIAL_TONE_350_440HZ
Digits Required  : 1 (min), 6 (max)
Digit Set  : NONE
Signalling  : DTMF
(Use $ to terminate input without entering digits.)
Invalid Digits Entered: <Digits> STRING
Enter: <Digits>
```

>2142201234

```
Received Digit Set : 2142201234
Reporting:  2
Remaining:  142201234
```

```
********************************************************
 End Peripheral Interaction
********************************************************
Digits Buffer   Digits :
                Reset  : NO   Term : NO   Timeout : NO   CSP: NO
                NOA    : UNKNOWN_UNIQUENESS
                CarrSel: NO_INDICATION
Collectable List: (AGNTDATA) (AGNTDATA) (SUBR) (OLI) (INCLUDE)
                  >>ADDR<< (SUBR)

--------------------------------------
 MsgAnalysis Event
--------------------------------------
  Next State:Load State
--------------------------------------
--------------------------------------
 Load Event
--------------------------------------
********************************************************
Peripheral Interaction
********************************************************
Digits Required  : 5 (min), 5 (max)
Digit Set  : 142201234
Signalling  : DTMF
Reporting:  14220
Remaining:  1234
********************************************************
 End Peripheral Interaction
********************************************************
Digits Buffer   Digits : 2
                Reset  : NO   Term : NO   Timeout : NO   CSP: NO
                NOA    : UNKNOWN_UNIQUENESS
                CarrSel: NO_INDICATION
Collectable List: (AGNTDATA) (AGNTDATA) (SUBR) (OLI) (INCLUDE)
                  >>ADDR<< (SUBR)

--------------------------------------
 MsgAnalysis Event
--------------------------------------
  Next State:Load State
--------------------------------------
--------------------------------------
 Load Event
--------------------------------------
Digits Buffer   Digits : 214220
                Reset  : NO   Term : NO   Timeout : NO   CSP: NO
                NOA    : UNKNOWN_UNIQUENESS
                CarrSel: NO_INDICATION
--------------------------------------
 Parse Event
--------------------------------------
```

```
--------------------------------------
 Validate Event
--------------------------------------
******Validate state data******
pretranslator name:                DAL
pretranslation digits:             214220
pretranslator route selector:      CT
call_feature:                      OFFNET
--------------------------------------
 Request Event
--------------------------------------

********************************************************
Peripheral Interaction
********************************************************
Digits Required  : 1 (min), 4 (max)
Digit Set  : 1234
Signalling  : DTMF
Reporting:  1
Remaining:  234
********************************************************
 End Peripheral Interaction
********************************************************
Digits Buffer   Digits :
                Reset  : NO   Term : NO   Timeout : NO   CSP: NO
                NOA    : UNKNOWN_UNIQUENESS
                CarrSel: NO_INDICATION
Collectable List: (AGNTDATA) (AGNTDATA) (SUBR) (OLI) (INCLUDE)
                  >>ADDR<< (SUBR)

--------------------------------------
 MsgAnalysis Event
--------------------------------------
  Next State:Load State
--------------------------------------
--------------------------------------
 Load Event
--------------------------------------
********************************************************
Peripheral Interaction
********************************************************
Digits Required  : 3 (min), 3 (max)
Digit Set  : 234
Signalling  : DTMF
Reporting:  234
Remaining:  NONE
********************************************************
 End Peripheral Interaction
********************************************************
```

```
Digits Buffer   Digits : 1
                Reset  : NO   Term : NO   Timeout : NO   CSP: NO
                NOA    : UNKNOWN_UNIQUENESS
                CarrSel: NO_INDICATION
Collectable List: (AGNTDATA) (AGNTDATA) (SUBR) (OLI) (INCLUDE)
                  >>ADDR<< (SUBR)

--------------------------------------
 MsgAnalysis Event
--------------------------------------
  Next State:Load State
--------------------------------------
--------------------------------------
 Load Event
--------------------------------------
Digits Buffer   Digits : 1234
                Reset  : NO   Term : NO   Timeout : NO   CSP: NO
                NOA    : UNKNOWN_UNIQUENESS
                CarrSel: NO_INDICATION
--------------------------------------
 Parse Event
--------------------------------------
--------------------------------------
 Validate Event
--------------------------------------
******Validate state data******
pretranslator name:                DAL
pretranslation digits:             2142201234
pretranslator route selector:      CT
call_feature:                      OFFNET
--------------------------------------
 ApplyFtrs Event
--------------------------------------
******ApplyFtrs state data******
called_noa:                        NATL
--------------------------------------
 Collect Event
--------------------------------------
Flexdial Index: I_AUTH_HSD
Collectable List: (AGNTDATA) (AGNTDATA) (SUBR) (OLI) (INCLUDE)
                  (ADDR) >>SUBR<<

--------------------------------------
 Start Event
--------------------------------------
  Processing Collectable:SUBR
--------------------------------------
 Parse Event
--------------------------------------
```

```
--------------------------------------
 Prompt Event
--------------------------------------
Digits Buffer   Digits :
                Reset  : NO   Term : NO   Timeout : NO   CSP: NO
                NOA    : UNKNOWN_UNIQUENESS
                CarrSel: NO_INDICATION
********************************************************
 Peripheral Interaction
********************************************************
Prompt : Tone – PROMPT_TONE
Digits Required  : 5 (min), 7 (max)
Digit Set  : NONE
Signalling  : DTMF
(Use $ to terminate input without entering digits.)
Next par is: <Digits> STRING
Enter: <Digits>

>6112211

Received Digit Set : 6112211
Reporting:  61122
Remaining:  11
********************************************************
 End Peripheral Interaction
********************************************************
Digits Buffer   Digits :
                Reset  : NO   Term : NO   Timeout : NO   CSP: NO
                NOA    : UNKNOWN_UNIQUENESS
                CarrSel: NO_INDICATION
Collectable List: (AGNTDATA) (AGNTDATA) (SUBR) (OLI) (INCLUDE)
                  (ADDR) >>SUBR<<

--------------------------------------
 MsgAnalysis Event
--------------------------------------
  Next State:Load State
--------------------------------------
--------------------------------------
 Load Event
--------------------------------------
********************************************************
Peripheral Interaction
********************************************************
Digits Required  : 2 (min), 2 (max)
Digit Set  : 11
Signalling  : DTMF
Reporting:  11
Remaining:  NONE
```

```
********************************************************
 End Peripheral Interaction
********************************************************
Digits Buffer   Digits : 61122
                Reset  : NO   Term : NO   Timeout : NO   CSP: NO
                NOA    : UNKNOWN_UNIQUENESS
                CarrSel: NO_INDICATION
Collectable List: (AGNTDATA) (AGNTDATA) (SUBR) (OLI) (INCLUDE)
                  (ADDR) >>SUBR<<

--------------------------------------
 MsgAnalysis Event
--------------------------------------
  Next State:Load State
--------------------------------------
--------------------------------------
 Load Event
--------------------------------------
Digits Buffer   Digits : 6112211
                Reset  : NO   Term : NO   Timeout : NO   CSP: NO
                NOA    : UNKNOWN_UNIQUENESS
                CarrSel: NO_INDICATION
--------------------------------------
 Parse Event
--------------------------------------
--------------------------------------
 Validate Event
--------------------------------------
****** Validate ******
Subscriber Type  : AUTH
Collected Digits : 6112211
Validation Type  : FLEXVAL
Validation Status: No Failure
FLEXVAL Index    :       1,   FLEXFEAT Index:   110111
--------------------------------------
 ApplyFtrs Event
--------------------------------------
--------------------------------------
 Collect Event
--------------------------------------
Flexdial Index: I_AUTH_HSD
Collectable List: (AGNTDATA) (AGNTDATA) (SUBR) (OLI) (INCLUDE)
                  (ADDR) >>SUBR<<

***********************************************
TRANSLATIONS
***********************************************
```

```
  TRANSLATE
    HNPA_RTER 1A00DC
      Dialed    : 2142201234
      STS       : 611
      TRANS SYS : NA
      CALL TYPE : NP
      ROUTE     : CKT    DAL220TWDTGS    1
********************************************************
 Flexsim Execution Complete for Trunk: AXDAL231TWDTLS
********************************************************
```

Table FLEXSIMT

The FLEXSIMT table allows you to provision up to ten initial address messages (IAMs) that are used for FLEXSIM CCS7 type originations. The messages are provisioned with explicit parameter information, allowing for complete control over the call process being simulated.

The FLEXSIM CI process takes the IAM message provisioned in table FLEXSIMT and delivers it to the CALLP process in order to initiate the call process simulation.

The parameter information provisioned in the table is included in the IAM message delivered to the call process. Additionally, the FLEXSIM display command can be used to provide a symbolic display of the message provisioned. Only IAM messages can be provisioned within the FLEXSIMT table.

General layout

This table is indexed by a numeric value in the range of 1 to 9 to retrieve a tuple consisting of two data fields: the message type, and a vector of ISUP mandatory and optional parameters. Five required parameters and up to eight of the optional parameters can be entered for each index.

Overview of ISUP parameters

Currently the only supported message type for table FLEXSIMT is the IAM message. For an IAM message, the NOC, FCI, CPC, USI, and CPA parameters (see definitions in next section) are required and all other parameters are optional. There are no requirements on the order the parameters are provisioned within the table. If a required parameter is not provisioned within the table entry, an error message is displayed.

Required parameters

The following ISUP parameters are required in the IAM message. They can be entered in any order, but are saved in the order shown below.

- NOC—Nature of Connection

- FCI—Forward Call Indicators
- CPC—Calling Party Category
- USI—User Services Information
- CPA—Called Party Address

Optional parameters

The following ISUP parameters are optional IAM message parameters that can be provisioned in table FLEXSIMT:

- CAM—Channel Assignment Map (IAM)
- CGN—Charge Number (IAM)
- CIP—Carrier Identification Parameter
- CLG—Calling Party Address
- CSI—Carrier Selection Information
- GA—Generic Address
- GD—Generic Digits
- MBG—Multiple Business Group
- NSF—Network Specific Facilities
- NSI—Network Specific IAM
- OCN—Original Called Number
- OI—Operator Information
- OLI—Origination Line Information
- OSI—Operator Services Indicator
- SLI—Supplementary Line Information
- TNS—Transit Network Selection

Key

This table is a singly-indexed table with index values in the range of 1 to 9, as shown in Table 24-8.

Table 24-8
FLEXSIM table key

Key field	Description	Values
KEY	The key consists of the FLEXSIM ISUP message index.	1 to 9

Fields

The FLEXSIMT table tuple consists of the message type and a vector of parameters. The parameter vector can be provisioned with any of the identified parameter names. Each parameter name provides its own data refinement.

Table 24-2 contains the FLEXSIMT fields.

Table 24-9
FLEXSIMT table fields

Field	Description	Values	Default
MSGTYPE	ISUP Message type. The only valid value for FLEXSIM is IAM.	IAM Initial Address Message	NA
PARM	This field consists of a parameter vector of up to 13 ISUP parameter names. Each parameter name identifies the ISUP parameter that is being provisioned.	Vector of up to 13 of {NOC, FCI, CPC, USI, CPA, CAM, CGN, CSI, CIP, CLG, GA, GD, MBG, NSF, NSI, OCN, OI, OLI, OSI, SLI, TNS}	NA

Each ISUP parameter has its own field refinements and is presented independently. The parameter values are based on ANSI Q.764 messaging standards.

NOC parameter

The Nature of Connection parameter (NOC) is a mandatory, fixed-length parameter in the IAM and CRA messages. It contains information on use of satellites, echo suppression, and continuity checking.

Definition

The field refinements for the NOC ISUP parameter are shown in Table 24-10.

Table 24-10
NOC PARM fields

Field	Description	Values	Default
SAT	This field identifies the satellite indicator type.	NONE, ONE, TWO, SPARE NONE—no previous satellite hops ONE—one previous satellite hop TWO—two previous satellite hops SPARE—spare	NA
COT	Continuity Check Indicator	NONE,REQD,PREV,SPARE NONE—no continuity check required REQD—continuity check required PREV—previous continuity check SPARE—spare	NA
ECHO	Echo Suppressor Indicator	NONE or HALF NONE—no half echo suppressor HALF—half echo suppressor	NA

Example

An example of the NOC parameter is:

```
NOC NONE NONE NONE
```

This example identifies that no satellite hops, continuity checking, or echo suppression is applicable for the connection.

NOC parameter restrictions and limitations

This parameter is required in the IAM message type.

FCI Parameter

The Forward Call Indicator parameter (FCI) is a mandatory, fixed-length parameter in the IAM. It contains forward call information such as interworking that has been encountered, or whether ISUP is required all the way.

Definition

The field refinements for the FCI ISUP parameter are shown in Table 24-11.

Table 24-11
FCI PARM fields

Field	Description	Values	Default
CALLTYPE	ISUP Call Type	NATL or INTL	NA
EE_METHOD	End-to-End Method	NONE,PASS,SCCP,PASS_SCCP NONE—no method available PASS—pass along method available SCCP—SCCP methods available PASS_SCCP—pass along and SCCP methods available	NA
INTERWORK	ISUP Interworking Indicator	NO_IW, IW	NA
EE_INFO_CHK	End-to-End Information Indicator	NO_EE, EE	NA
ISUP	ISDN User Part Indicator	NO_ATW, ATW	NA
ISUP_PREF	ISDN User Part Preferred Indicator	PREF,NOT_REQD,REQD,SPARE PREF—preferred ATW NOT_REQD—not required ATW REQD—required ATW SPARE—spare	NA
ISDN_ACCESS	ISDN Access Indicator	NON_ISDN or ISDN NON_ISDN—non ISDN access ISDN—ISDN access	NA

Example

An example of the FCI parameter is:

```
(FCI NATL NONE IW NO_EE NO_ATW PREF NON_ISDN)
```

FCI restrictions and limitations

The FCI parameter is required in the IAM message.

CPC Parameter

The Calling Party Category (CPC) parameter is a mandatory, fixed-length parameter in the IAM that indicates the category of the calling party.

Definition

The field refinement for the CPC ISUP parameter is shown in Table 24-12.

Table 24-12
CPC PARM fields

Field	Description	Values	Default
CPC	Calling party category	UNKNOWN, OP_FRENCH, OP_ENGLISH, OP_GERMAN, OP_RUSSIAN, OP_SPANISH, OP_NATIONAL, SUBSCRIBER1, PRIORITY, DATA, TEST, NON_VOICE, PAYPHONE UNKNOWN—unknown OP_ENGLISH—operator–English speaking OP_GERMAN—operator–German speaking OP_RUSSIAN—operator–Russian speaking OP_SPANISH—operator–Spanish speaking OP_NATIONAL—operator–national PRIORITY—priority DATA—data TEST—test NON_VOICE—non voice PAYPHONE—payphone	NA

Example

An example of the CPC ISUP parameter is:

```
CPC UNKNOWN
```

CPC ISUP Parameter restrictions and limitations

The CPC parameter is required in the IAM message.

USI Parameter

The User Service Information parameter (USI) is a mandatory, variable-length parameter in the IAM. It defines the calling party's request for channel bearer capability.

Definition

The field refinements for the USI ISUP parameter are shown in Table 24-13.

Table 24-13
USI PARM fields

Field	Description	Values	Default
TRANS_CAP	Bearer channel transfer capability	SPEECH, UNRES_DIG, RES_DIG, 3_1_KHZ, 7_KHZ, 15_KHZ, VIDEO SPEECH—speech UNRES_DIG—unrestricted digital RES_DIG—restricted digital 3_1_KHZ—3.1 KHz 7_KHZ—7 KHz 15_KHZ—15 KHz VIDEO—video	NA
CODING_STD	Coding standard	CCITT, RSVD, NATL, RSVD2 CCITT—CCITT RSVD—reserved NATL—national RSVD2—reserved	NA

—continued—

Table 24-13
USI PARM fields (continued)

Field	Description	Values	Default
TRANS_RATE	Bearer channel transfer rate	INFO, 192KBS, 256KBS, 320KBS, 448KBS, 512KBS, 576KBS, 640KBS, 704KBS, 768KBS, 832KBS, 896KBS, 960KBS, 1024KBS, 1088KBS, 1152KBS, 64KBS, 128KBS, AT&T2, 384KBS, 1472KBS, 1536KBS, 1600KBS, 1920KBS, BASE_RATE, 1280KBS, 1344KBS, 1408KBS, 1664KBS, 1728KBS, 1792KBS, 1856KBS	NA
TRANS_MODE	Information Transfer Mode	CIRCUIT,SPARE,PACKET, SPARE2 CIRCUIT—circuit mode PACKET—packet mode	NA
OCTET2A	Refined with the three fields below if OCTET2A is available	N or Y N—OCTET2A not available Y—OCTET2A available	NA
ESTAB	Establishment	DEMAND, SPARE, SPARE, SPARE	NA
CONFIG	Configuration	PT_TO_PT, SPARE1, MULTIPT, SPARE2	NA
STRUCT	Structure	DEFAULT,INTEG_8KHZ,SPARE1, SPARE2,SDU_INTEG,SPARE3, SPARE4,UNSTRUCT INTEG_8KHZ—integrity 8 KHz SDU_INTEG—service data unit integrity UNSTRUCT—unstructured	NA
OCTET2B	Refined with the two fields below if OCTET2B is available	N or Y	NA

—continued—

Table 24-13
USI PARM fields (continued)

Field	Description	Values	Default
TRANS_RATE	Information Transfer Rate	INFO, 192KBS, 256KBS, 320KBS, 448KBS, 512KBS, 576KBS, 640KBS, 704KBS, 768KBS, 832KBS, 896KBS, 960KBS, 1024KBS, 1088KBS, 1152KBS, 64KBS, 128KBS, AT&T2, 384KBS, 1472KBS, 1536KBS, 1600KBS, 1920KBS, BASE_RATE, 1280KBS, 1344KBS, 1408KBS, 1664KBS, 1728KBS, 1792KBS, 1856KBS	NA
SYMMETRY		SYM_2W,ASYM_2W,ORIGDEST, DESTORIG SYM_2W—symmetric two way ASYM_2W—asymmetric two way ORIGDEST—origination to destination DESTORIG—destination to origination	NA
OCTET2C	Refined with RATE_MULT field below if OCTET2C is available	N or Y	NA
RATE_MULT	Rate Multiplier	0 to 31	NA
OCTET3	Refined with LAYTER1 field below if OCTET3 is available	N or Y	NA
LAYER1	Layer 1 Protocol	I412, RATE, G711_MULAW, G711_ALAW, G721 I412—CCITT I 412 RATE—rate adaption G711_MULAW—CCITT G 711 MU Law G711_ALAW—CCITT G 711 A Law G721—CCITT G 721	NA

—continued—

Table 24-13
USI PARM fields (continued)

Field	Description	Values	Default
OCTET3A	Refined with RATE field below if OCTET3A is available	N or Y	NA
RATE	Rate adaption	E_BITS,6_KBS,1_2_KBS,2_4_KBS,3_6_KBS,4_8_KBS,7_2_KBS,8_0_KBS,9_6_KBS,14_4_KBS,16_KBS,19_2_KBS,32_KBS,48_KBS,56_KBS,134_5_BS,100_BS,75_FBS,1_2_FKBS,50_BS,75_BS,110_BS,150_BS,200_BS,300_BS,12_KBS	NA
—end—			

Example

An example of the USI ISUP parameter is:

```
USI SPEECH CCITT 64KBS CIRCUIT N N N Y G711_MULAW N
```

USI ISUP parameter restrictions and limitations

The USI parameter is required in the IAM message.

CPA Parameter

The Called Party Number (CPA) parameter is a mandatory, variable-length parameter in the IAM. This parameter can also be found as an optional parameter in the FAR message. In both the IAM and FAR, this parameter provides the number to use to translate and route the call (the initial called address in the case of the IAM, and the redirected address in the case of the FAR).

Definition

The field refinements for the CPA ISUP parameter are shown in Table 24-14.

Table 24-14
CPA PARM fields

Field	Description	Values	Default
NOA	Nature of address	UNKNOWN, SUBR, VPN, NATL, INTL, TREAT_OP, SUBR_OP, NATL_OP, INTL_OP, NO_NUM_OP, NO_NUM_CT, 950_CT, TEST, PANI, INTL_INB_OP, INTL_OP_WZ1	NA
NUMPLAN	Numbering plan	UNKNOWN, ISDN, RSVD0, RSVD1, RSVD2, PRIV, RSVD4, RSVD5	NA
ADDRESS	Called party address digits	Vector of up to 24 of {0 to 9}	NA

Example

An example of the CPA parameter is:

```
CPA NATL UNKNOWN 5551234
```

CPA ISUP parameter restrictions and limitations

The CPA parameter is required in the IAM message.

CLG Parameter

The Calling Party Address (CLG) parameter is optional in the IAM and FAR messages. It identifies the calling party for the call. The Calling Party Address or Charge Number is expected on all SS7 FGD originations, and is optional on SS7 IMT originations. The Calling Party Address and Charge number are delivered according to ANI delivery logic.

Definition

The field refinements for the CLG ISUP parameter are shown in Table 24-15.

Table 24-15
CLG PARM fields

Field	Description	Values	Default
NOA	Nature of address	UNKNOWN, SUBR, VPN, NATL, INTL, TREAT_OP, SUBR_OP, NATL_OP, INTL_OP, NO_NUM_OP, NO_NUM_CT, 950_CT, TEST, PANI, INTL_INB_OP, INTL_OP_WZ1	NA
NUMPLAN	Numbering plan	UNKNOWN, ISDN, RSVD0, RSVD1, RSVD2, PRIV, RSVD4, RSVD5	NA
PRESREST	Presentation Indicator	ALLOW, RESTRICT, SPARE1, SPARE2	NA
ADDRESS	Calling party address digits.	Vector of up to 24 of {0 to 9}	NA

Example

An example of the CLG ISUP parameter is:

```
CLG SUBR ISDN RESTRICT 2142221234
```

CLG ISUP parameter restrictions and limitations

There are no identified restrictions for this parameter.

CSI Parameter

The purpose of the Carrier Selection Information (CSI) parameter is to identify the relationship between the subscriber and inter-exchange carrier.

Definition

The field refinement for the CSI ISUP parameter is shown in Table 24-16.

Table 24-16
CSI PARM fields

Field	Description	Values	Default
CARR_SEL	Carrier Selection	NO_IND, PRESUB_NOINP, PRESUB_INP, PRESUB_NOIND, NOT_PRESUB NO_IND—no indication PRESUB_NOINP—pre-subscribed and NO input PRESUB_INP—pre-subscribed and input PRESUB_NOIND—pre-subscribed and No indication NOT_PRESUB—not pre-subscribed	NA

Example

An example of the CSI ISUP parameter is:

```
CSI PRESUB_INP
```

CSI ISUP parameter restriction and limitations

There are no identified restrictions for this parameter.

CAM Parameter

The Channel Assignment Map parameter (CAM) is an optional parameter in the IAM. It is only used for wideband calls using the flexible method of choosing circuits, and is used to identify the circuits involved in the wideband call.

Definition

The field refinements for the CAM ISUP parameter are shown in Table 24-17.

Table 24-17
CAM PARM fields

Field	Description	Values	Default
CHANTYPE	Wideband Channel Type	DS1,PCM30,DS3	NA
SLOTMAP	Slot Map	TABLE OF 32 {N,Y}s	NA

Example

An example of the CAM ISUP parameter is:

```
CAM DS1 YNNYN YYNNN NNNNN YNYNN NNNNN NNNNN NN
```

CAM ISUP parameter restrictions and limitations

There are no identified restrictions for this parameter.

CGN Parameter

The Charge Number parameter (CGN) is an optional parameter in the IAM or FAR message. In the IAM, the Charge Number contains the ANI for the call, and is always sent in conjunction with the Originating Line parameter. In the FAR, the Charge Number contains the special billing number. The Calling party number and Charge number are delivered according to ANI delivery logic.

Definition

The field refinements for the CGN ISUP parameter are shown in Table 24-18.

Table 24-18
CGN PARM fields

Field	Description	Values	Default
NOA	Nature of Address	ANI_SUBR, ANI_NONE, ANI_NATL, OPER_COLLECT ANI_SUBR—subscriber ANI ANI_NONE—no ANI ANI_NATL—national ANI OPER_COLLECT—operator collect	NA
NUMPLAN	Numbering Plan	UNKNOWN, ISDN, RSVD0, RSVD1, RSVD2, PRIV, RSVD4, RSVD5	NA
ADDRESS	Charge number address digits	Vector of up to 24 of {0 to 9}	NA

Example

An example of the CGN ISUP parameter is:

```
CGN ANI_NONE UNKNOWN 2211234
```

CGN ISUP parameter restriction and limitations

There are no identified restrictions for this parameter.

CIP Parameter

The Carrier Identification parameter (CIP) is an optional parameter in the IAM. It identifies the carrier for the call. It is only expected on national calls, and is delivered for national calls according to feature functionality.

Definition

The field refinements for the CIP ISUP parameter are shown in Table 24-19.

Table 24-19
CIP PARM fields

Field	Description	Values	Default
NETWORK_ID	Network identifier	UNKNOWN, 3_DIGIT_CIC, 4_DIGIT_CIC	NA
NET_ID_TYPE	Network type	SPARE, NATL	NA
CIC_CODE	Carrier identification code digits	Table of 4 of {N, 1, 2, 3, 4, 5, 6, 7, 8, 9, 0, B, C, D, E, F}	NA

Example

An example of the CIP ISUP parameter is:

```
CIP 3_DIGIT_CIC NATL N455
```

CIP ISUP parameter restrictions and limitations

There are no identified restrictions for this parameter.

GA Parameter

The Generic Address parameter (GA) is an optional parameter in the IAM. It is used to relay different type of addresses, on a per-feature basis. There may be more than one Generic Address parameter in a single message.

Definition

The field refinements for the GA ISUP parameter are shown in Table 24-20.

Table 24-20
GA PARM fields

Field	Description	Values	Default
GATYPE	Generic Address Type	DIALED_NO, DEST_NO, FAIL_SCRN, NOT_SCRN, COMP_NO, BC_DEST, RSVD_XPN DIALED_NO—dialed number DEST_NO—destination number FAIL_SCRN—supplemental number, failed screening NOT_SCRN—supplemental number, not screened COMP_NO—completion number BC_DEST—BC destination RSVD_XPN—reserved	NA
NOA	Nature Of Address	UNKNOWN, SUBR, VPN, NATL, INTL, TREAT_OP, SUBR_OP, NATL_OP, INTL_OP, NO_NUM_OP, NO_NUM_CT, 950_CT, TEST, PANI, INTL_INB_OP, INTL_OP_WZ1	NA
NUMPLAN	Numbering Plan	UNKNOWN, ISDN, RSVD0, RSVD1, RSVD2, PRIV, RSVD4, RSVD5	NA
PRESREST	Address Presentation Restriction	ALLOW, RESTRICT, SPARE1, SPARE2	NA
ADDRESS	Address	Vector of up to 24 of {0 to 9}	NA

Example

An example of the GA ISUP parameter is:

```
GA DIALED_NO NO_NUM_OP PRIV ALLOW 2145551111
```

GA ISUP parameter restrictions and limitations

There are no identified restrictions for this parameter.

GD Parameter

The Generic Digits parameter (GD) is an optional parameter in the IAM, FAA and FAR. It is used to relay different type of digits, on a per-feature basis. Note that there may be more than one Generic Digits parameter in a single message.

Definition

The field refinements for the GD ISUP parameter are shown in Table 24-21.

Table 24-21
GD PARM fields

Field	Description	Values	Default
GDTYPE	Generic Digits Type	AUTHCODE, CLASSMARK, CLLI_ADMIN, MEM_RG_INF, OSID_OTG, IMT_INFO, HOTEL_ROOM, HOTEL_NAME, DIV_ID, TRK_INFO, BILLNUM, UNIVACC, PINDIGS, ACCTCD AUTHCODE—authorization code digits CLASSMARK—private network traveling classmark CLLI_ADMIN—CLLI administrative number MEM_RG_INF— OSID_OTG originating switch ID and trunk group IMT_INFO—IMT information HOTEL_ROOM—hotel room HOTEL_NAME—hotel name DIV_ID—division identifier TRK_INFO—trunk information BILLNUM—billing number UNIVACC—universal access code PINDIGS—PIN digits ACCTCD—account code digits	NA
ENCODE	Encoding Scheme	BCD_EVEN,BCD_ODD,BCD_IA5,BINARY	NA
DIGITS	Digits	Vector of up to 24 of {0 to 9}	NA

Example

An example of the GD ISUP parameter is:

```
GD AUTHCODE BCD_EVEN 232314
```

GD ISUP parameter restriction and limitations

There are no identified restrictions for this parameter.

MBG Parameter

The Multiple Business Group (MBG—also known as NETINFO) parameter is an optional parameter in the IAM, ACM and ANM. It is included in the IAM when the table TRKGRP option MBGXLA is set. The UCS DMS-250 switch does not build it in the ACM or ANM, but tandems it in these messages.

Definition

The field refinements for the MBG ISUP parameter are shown in Table 24-22.

Table 24-22
MBG PARM fields

Field	Description	Values	Default
CODE	Network Party Selection Code	NO_IND,CLG_PARTY,CLD_PARTY,CON_PARTY,REDIRECTED, REDIRECTION,OVERRIDE NO_IND—no indication CLG_PARTY—calling party CLD_PARTY—called party CON_PARTY—connected party REDIRECTED—redirected information REDIRECTION—redirection information OVERRIDE—override information	NA
NETID	Network Identification	0 to 32767	NA
NETCGID	Network Customer Group Identification	0 to 32767	NA
NETNCOS	Network Class of Service	0 to 255	NA

Example

An example of the MBG ISUP parameter is:

```
MBG CLD_PARTY 45 432 34
```

MBG ISUP parameter restriction and limitations

There are no identified restrictions for this parameter.

NSF Parameter

The Network Specific Facilities (NSF) parameter is used to transport ISDN network information across a network of ISUP trunks.

Definition

The field refinements for the NSF ISUP parameter are shown in Table 24-23.

Table 24-23
NSF PARM fields

Field	Description	Values	Default
ELEM	Element Identifier	INFO	NA
PLAN	Network Identification Plan	UNKNOWN,CARR_ID	NA
NETTYPE	Network Identification Type	NATL	NA
ID	Network Identification Table	TABLE OF 3 bytes 0 to 255s	NA
FAC_CODE	Facility Coding Value	VNET,INWATS,PRISM	NA
		VNET—Virtual Private Network INWATS—INWATS PRISM—PRISM	
FEAT_SERV	Feature of Service Type	FEAT,SERV	NA
PARM_BIN	Parameter Coding Type	PARM,BIN	NA

—continued—

Table 24-23
NSF PARM fields (continued)

Field	Description	Values	Default
PARM	Parameterized	PARM	NA
PARM_FIELD	Parameterized Field	Four Bytes	NA
BIN	Binary	BIN	NA
SID	Service Identification	0 to 32767	NA

—end—

Examples

Examples of the NSF ISUP parameter are:

```
NSF INFO CARR_ID NATL 2 2 2 INWATS FEAT BIN 453
NSF INFO CARR_ID NATL 3 2 4 PRISM SERV PARM 4 4 4 4
```

NSF ISUP parameter restrictions and limitations

There are no identified restrictions for this parameter.

NSI Parameter

The Network Specific IAM (NSI) parameter is an optional parameter that is sent to, or received from, a DEX switch. It contains dialing plan information such as the queueing indicator, originating and terminating partition

Definition

The field refinements for the NSI ISUP parameter are shown in Table 24-24.

Table 24-24
NSI PARM fields

Field	Description	Values	Default
TPART	Terminating Partition	0 to 15	NA
OPART	Originating Partition	0 to 999	NA
QUEUE	Queueing Indicator	NONE,NOT_APPLY,APPLY NONE—no queueing indicator NOT_APPLY—no previous queueing APPLY—previous queueing	NA

Example

An example of the NSI ISUP parameter is:

```
NSI 5 5 NONE
```

NSI ISUP parameter restriction and limitations

There are no identified restrictions for this parameter.

OI Parameter

The Operator Information parameter (OI) is an optional parameter in the IAM, ANM and FAR messages. It provides operator information to the bridging UCS DMS-250 switch in an RLT (release link trunk) call.

Definition

The field refinements for the OI ISUP parameter are shown in Table 24-25.

Table 24-25
OI PARM fields

Field	Description	Values	Default
NUM	Operator Number	0 to 16383	NA
RE_TYPE	Reorigination Type	NORMAL_REORIG, BOOMER_REORG, BLOCK_REORIG, NO_REORIG	NA
ENTRYCD	Entry Code	UNKNOWN,OPER_INIT, STN_PD_OPR_ASST,STN_COL, STN_SPL_CLG,PER_PD,PER_COL, PER_SPL_CLG,STA_SPL_CLD, PER_SPL_CLD	NA
TROUBLE	Trouble Indicator	NO_TRBL	NA
TAC_IND	Time and Charge Request Indicator	NOT_REQD,REQD	NA
ACTRESP	Action Response	NONE	NA
FEATCODE	Feature Code	NONE	NA
TERMRTE	Terminating Route Code	NONE	NA
BILLING_IND	Billing Indicator	OFCPARM, FIRST_ANM, LAST_ANM	NA
SPAREDIGS	Supplementary Digits	Vector of up to 24 digits	NA

—end—

Example

An example of the OI ISUP parameter is:

```
OI 5 NORMAL_REORIG OPER_INIT NO_TRBL REQD NONE NONE NONE
LAST_ANM 234234
```

OI ISUP parameter restriction and limitations

There are no identified restrictions for this parameter.

OCN Parameter

The field refinements for the Original Called Number (OCN) ISUP parameter are shown in Table 24-26.

Table 24-26
OCN PARM fields

Field	Description	Values	Default
NOA	Nature of Address	UNKNOWN, SUBR, VPN, NATL, INTL, TREAT_OP, SUBR_OP, NATL_OP, INTL_OP, NO_NUM_OP, NO_NUM_CT, 950_CT, TEST, PANI, INTL_INB_OP, INTL_OP_WZ1	NA
NUMPLAN	Numbering Plan	UNKNOWN, ISDN, RSVD0, RSVD1, RSVD2, PRIV, RSVD4, RSVD5	NA
PRESREST	Address Presentation Restriction	ALLOW, RESTRICT, SPARE1, SPARE2	NA
ADDRESS	Address	Vector of up to 24 of {0 to 9}	NA

Example

An example of the OCN ISUP parameter is:

```
OCN NATL UNKNOWN ALLOW 2143211234
```

OCN ISUP parameter restriction and limitations

There are no identified restrictions for this parameter.

OSI Parameter

The Operator Service Indicator parameter (OSI) is an optional parameter in the IAM message. This parameter indicates that a call requires the switch to perform a charge rating lookup operation based on information (such as the ANICLAS and PANICLAS fields) in the originating trunk group.

Definition

The field refinements for the OSI ISUP parameter are shown in Table 24-27.

Table 24-27
OSI PARM fields

Field	Description	Values	Default
OSITYPE	Indicator Type	CHG_RATE	NA
RATING	Rating	NONE, AUTO	NA

Example

An example of the OSI ISUP parameter is:

```
OSI CHG_RATE AUTO
```

OSI ISUP parameter restriction and limitations

There are no identified restrictions for this parameter.

OLI Parameter

The Originating Line Information parameter (OLI) is an optional parameter in the IAM and FAR messages. This parameter indicates any special information about the type of calling party address.

Definition

The field refinement for the OLI ISUP parameter is shown in Table 24-28.

Table 24-28
OLI PARM field

Field	Description	Values	Default
OLI	Originating Line Information	0 to 99	NA

Example

An example of the OLI ISUP parameter is:

```
OLI 12
```

OLI ISUP parameter restriction and limitations

There are no identified restrictions for this parameter.

SLI Parameter

The Supplementary Line Information parameter (SLI) is an optional parameter in the IAM message. In the IAM for a third-party interaction or services platform callback call, this parameter performs either of the following functions:

- identifies the call as a callback

  or

- indicates that the receiving switch must include a call reference parameter in the ACM or ANM.

Definition

The field refinement for the SLI ISUP parameter is shown in Table 24-29.

Table 24-29
SLI PARM fields

Field	Description	Values	Default
SLI	Supplementary Line Information	Vector of up to 10 of {BGL_REST_ORG, BGL_REST_TRM, BGL_ATTEND_LN, NON_UNIQUE_LN, CALL_WAITING, AUTO_CALL_BK, MULTI_CONF, MALC_CALL, TOLL_FREE, RLT, RLT_CALLBK} BGL_REST_ORG—business group line restriction orig BGL_REST_TRM—business group line restriction term BGL_ATTEND_LN—business group line attendant line NON_UNIQUE_LN—non unique line CALL_WAITING—call waiting AUTO_CALL_BK—auto call back or NRAG MULTI_CONF—multi–way conference MALC_CALL—malicious call indicator TOLL_FREE—toll free RLT—release line trunk RLT_CALLBK—release line trunk callback	NA

Example

An example of the SLI ISUP parameter is:

```
SLI (RLT) $
```

SLI ISUP parameter restrictions and limitations

There are no identified restrictions for this parameter.

TNS Parameter

The Transit Network Selector parameter (TNS) is an optional parameter in the IAM and identifies the carrier identification code (CIC) for the call. The TNS is only expected on international calls, and is delivered for international calls according to feature functionality.

Definition

The field refinements for the TNS ISUP parameter are shown in Table 24-30.

Table 24-30
TNS PARM fields

Field	Description	Values	Default
NETID_PLAN	Network ID Plan	UNKNOWN, 3_DIGIT_CIC, 4_DIGIT_CIC, PUB_DATA, PUB_LAND 3_DIGIT_CIC—3-digit CIC 4_DIGIT_CIC—4-digit CIC PUB_DATA—public data network PUB_LAND—public land mobile network	NA
NETID_TYPE	Network ID Type	CCITT,NATL CCITT—CCITT standardized NATL—national network	NA
CIC	Three Carrier ID Code Digits	Table of 4 of {N,1,2,3,4,5,6,7,8,9,0,B,C,D,E,F}	NA
CODE	Circuit Code	UNSPEC, INTL_NO_OP, INTL_OP UNSPEC—unspecified INTL_NO_OP—international, no operator requested INTL_OP—international, operator requested	NA
		—end—	

Example

An example of the TNS ISUP parameter is:

```
TNS 3_DIGIT_CIC CCITT 2342 INTL_OP
```

TNS ISUP parameter restriction and limitations

There are no identified restrictions for this parameter.

Table FLEXSIMT example

An example of a complete FLEXSIMT table entry is:

```
KEY MSGTYPE     PARMS
------------------------------------------------------------
  1     IAM
(NOC NONE NONE NONE)
(FCI NATL NONE IW EE ATW PREF ISDN) (CPC UNKNOWN)
(USI SPEECH CCITT INFO CIRCUIT N N N N N )
(CPA SUBR ISDN 2145551234)
(CLG NATL ISDN ALLOW 2135554542) (OLI NORM_ANI) $
```

System requirements

The FLEXSIMT table allocates a total of 3312 bytes of store for a maximum of 10 table entries. Store is allocated at system initialization time.

Datafill order

The FLEXSIMT table is not dependent on any other tables for provisioning.

Dump and restore

This table is not supported for dump and restore operations.

Requirements and limitations

The NOC, FCI, CPC, USI, and CPA parameters must be provisioned for the IAM message type. An attempt to provision an IAM message without one of these parameters results in the following error message:

```
Tuple must contain the NOC, FCI, CPC, USI, and CPA
parameters.
```

Appendix B
FlexDial conversion tool

This appendix describes the FlexDial conversion tool, FLEXCONV. Examples of using the tool are also provided.

Overview

FLEXCONV converts non-AXXESS trunk agencies to AXXESS trunk agencies used by the FlexDial framework feature. The tool performs the conversions in steps to minimize service interruption.

It is recommended that this conversion be performed during low traffic periods and the “move percentage of trunk members at a time” option be used.

This tool implements the following functions:

- moves trunk members from one trunk group CLLI to another trunk group CLLI (for example, from a non-AXXESS trunk group type to an AXXESS trunk group type, or from an AXXESS trunk group type to a non-AXXESS group type)
- changes the route references from the old trunk group CLLI to the new CLLI
- deletes the old trunk group and renames the new trunk group CLLI with the name of the old group

FLEXCONV moves trunks from one trunk agency to another by groups. Since this process takes time, you can query the status of the process or halt the process at any time.

Before activating FLEXCONV, all minimum provisioning must be in place. This minimum is equivalent to the provisioning required before datafilling table TRKMEM:

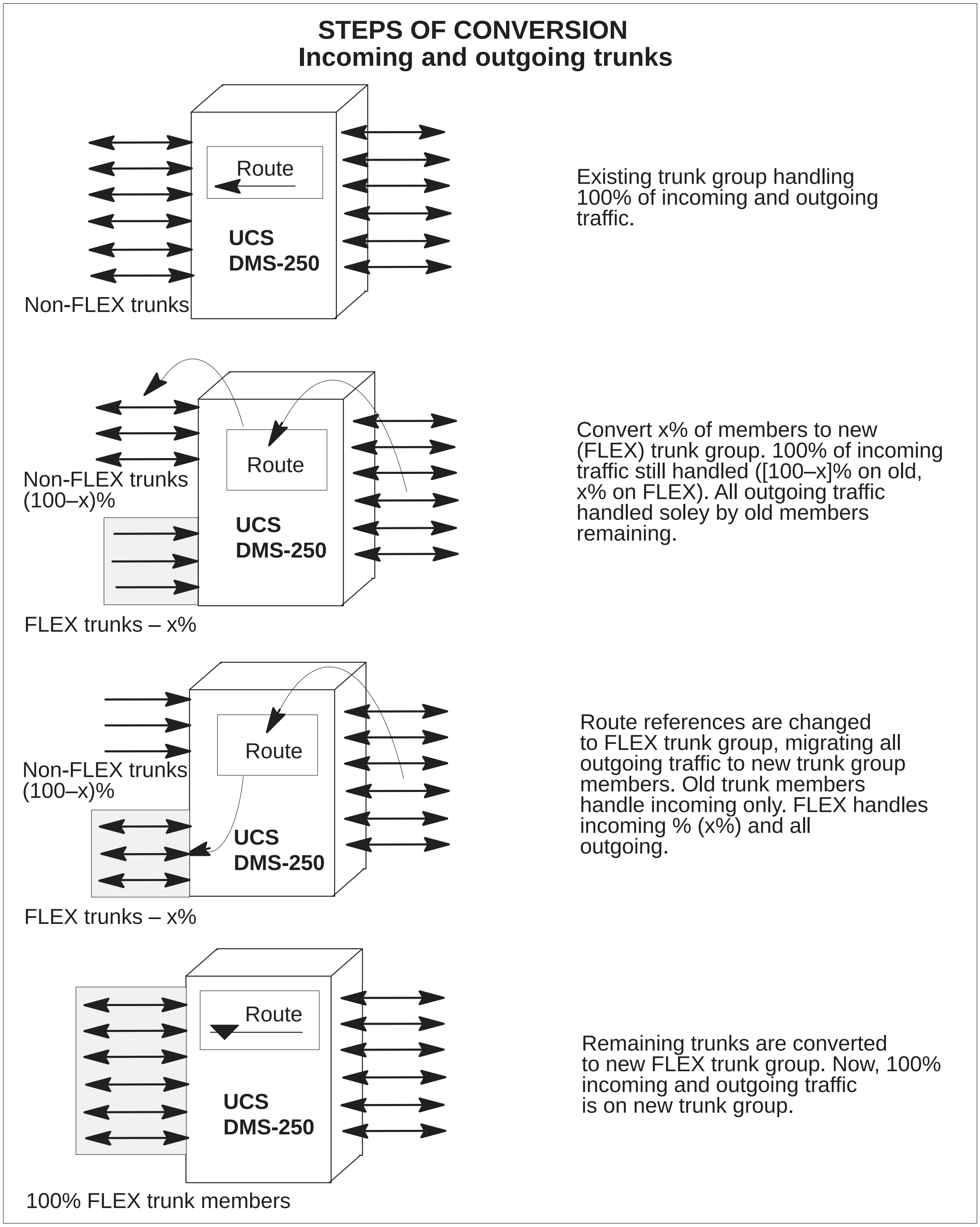
- Datafill tables CLLI, CLLICDR, and TRKGRP.
- Datafill tables TRKGRP1 and TRKSGRP for non-AXXESS trunks.

- Datafill tables TRKFEAT, TRKSIG, and FLEXDIAL for AXXESS trunks. Also datafill table ISUPDEST for SS7 trunks.

The goal of FLEXCONV is to move trunk members of an old trunk group type to a new trunk group without disrupting traffic on the trunks or causing downtime on the trunk group. Figure 25-1 shows conceptually how the conversion is done.

Figure 25-2 shows the actual steps of the conversion. The text following this figure describes the steps in detail.

Figure 25-1
FLEXCONV conversion steps overview



STEPS OF CONVERSION
Incoming and outgoing trunks

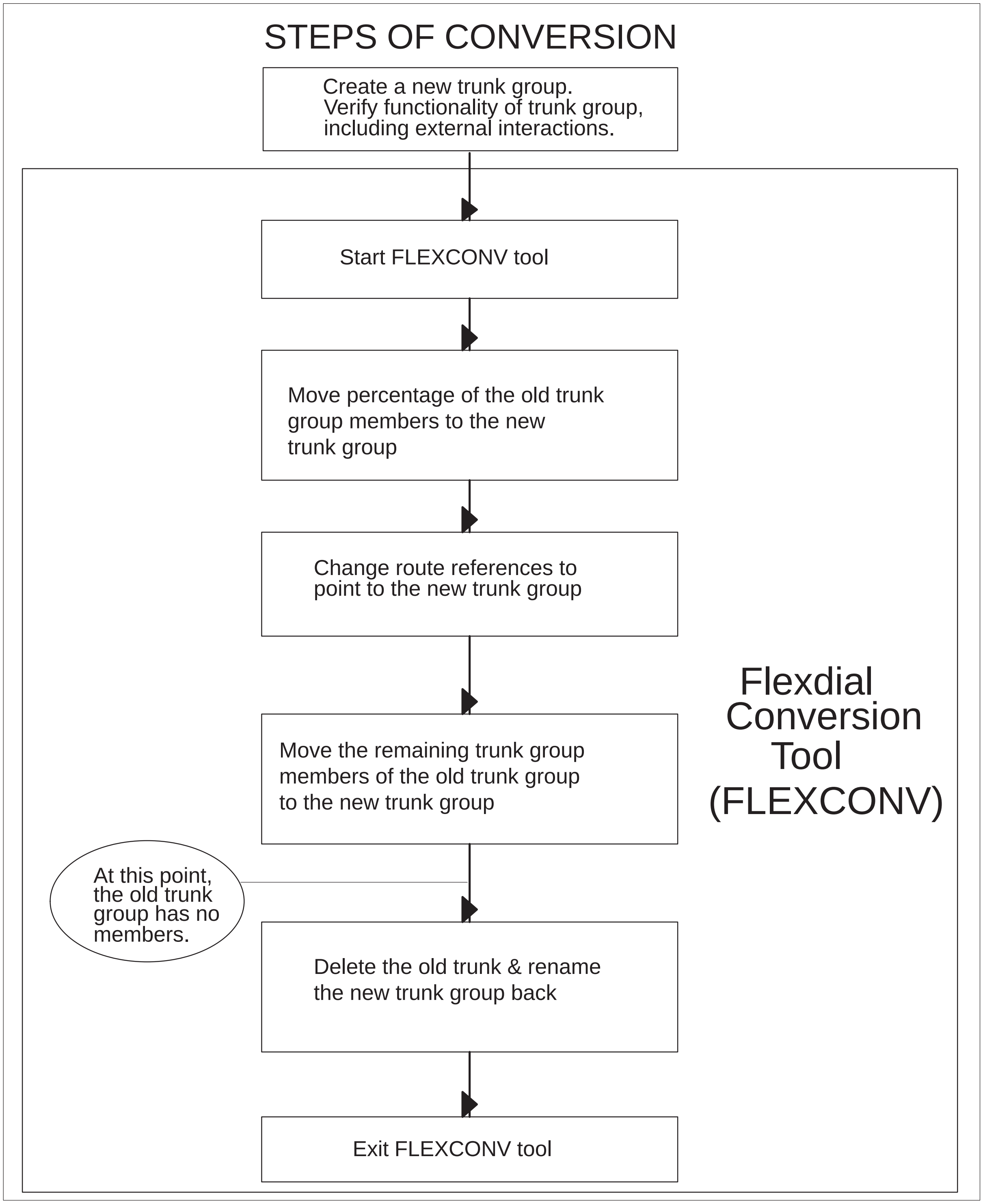
Existing trunk group handling 100% of incoming and outgoing traffic.

Convert x% of members to new (FLEX) trunk group. 100% of incoming traffic still handled ([100–x]% on old, x% on FLEX). All outgoing traffic handled soley by old members remaining.

Route references are changed to FLEX trunk group, migrating all outgoing traffic to new trunk group members. Old trunk members handle incoming only. FLEX handles incoming % (x%) and all outgoing.

Remaining trunks are converted to new FLEX trunk group. Now, 100% incoming and outgoing traffic is on new trunk group.

Figure 25-2
FLEXCONV conversion steps


STEPS OF CONVERSION
Create a new trunk group. Verify functionality of trunk group, including external interactions.
Start FLEXCONV tool
Move percentage of the old trunk group members to the new trunk group
Change route references to point to the new trunk group
Move the remaining trunk group members of the old trunk group to the new trunk group
At this point, the old trunk group has no members.
Delete the old trunk & rename the new trunk group back
Exit FLEXCONV tool
Flexdial Conversion Tool (FLEXCONV)

FlexDial conversion steps

Refer to Figure 25-2 for a picture of these steps.

Step 1: Create a new trunk group

In the first step, define a new trunk group type AXXESS in the FlexDial tables. For example, for AXXESS trunks, datafill tables TRKSIG, TRKFEAT, FLEXDIAL, and TRKGRP.

Note: This feature does not support these definitions and all the required tables have to be datafilled manually. The conversion tool does not check the consistency between old and new definitions; you must perform this verification before the tool is activated.

After the new trunk group has been defined and all the necessary consistency checks are performed, verify the functionality of the new trunk group. Verify all external interactions for the trunk group, and coordinate the trunk member percentages.

Step 2: Move a percentage of the trunk members

In the second step, move the trunk members from one trunk group to another trunk group, as defined in the previous step. During this process, these trunks are out of service. In order not to disrupt services, move the trunk members in at least two portions: for example, move one portion before updating route references, and the second after updating route references. You determine the percentage of trunk members to move in each phase.

Note: During the process, the trunk members are out of service, moved to the new trunk group, and are returned to their original state. Trunks are taken care of one-by-one in such a way, that only one trunk remains out of service at any moment. As soon as a trunk member is returned to service, it is immediately available for originations.

At the end of this step, only the remaining portion of the old trunk members are available for terminations. All incoming members are still being used.

Step 3: Change route references

In the third step, change the route references from the old trunk group to the new trunk group. In this step, change every reference to the old trunk group in the reference tables to the new trunk group. The FLEXCONV tool provides changes to the following reference tables:

- STDPRTCT subtable STDPRT
- HNPACONT subtable HRTEREF
- FNPACONT subtable FRTEREF

- TMTCNTL subtable TREAT
- OFRT
- OFRT2
- OFRT3
- OFRT4
- CCTR
- CTRTE
- TERMRTE
- TANDMRTE
- OPERRTE
- POSITION

At the end of this step, the trunk members of the new trunk group are available for terminations as well as originations. At this point the trunk members from the remaining portion of the old trunk group are only used for call originations.

Step 4: Move the rest of the trunk members

In the fourth step, move the remaining trunk members of the old trunk group to the new trunk group.

Step 5: Delete the old trunk group and rename the new group to the old name

The fifth step takes place after all trunk members have been moved to the new group, and no members remain in the old trunk group. At this point, delete the old trunk group. Change the name of the new trunk group to the old trunk group name (if desired).

Halting the movement process

Since the process of moving the trunk members from one trunk group to another can be timely, you can halt the process at any time. The halt request does not move back the trunk members that have been already moved to the new trunk group. It only discontinues further movement of trunk members from the old trunk group to the new one. All the trunks are returned to their original state.

Querying the movement process

You can use the FLEXCONV tool to query the status of processed requests and report on how many trunk members have already been moved.

FLEXCONV MAPCI level

The commands available on the MAPCI level for FLEXCONV are:

- CHNGROUT—finds all route references to the old trunk group and changes them to the new trunk group name.
- DELREN—deletes the old trunk group name, and renames the new trunk group name back to the old name.
- MVMEMBER—invokes an additional MAPCI level. This level controls movement of trunk members from the one trunk group to another trunk group.

  The commands available in the MAPCI MVMEMBER level are:

  — MOVE—starts the process of moving trunk members from the old trunk group to the new trunk group.

  — QUERYMV—queries the status of the movement process. If the specified group is being processed, this command displays the number of trunk members that have been already processed.

  — HALTMV—stops the process, if it is not completed.

MAPCI command syntax

See “FLEXCONV tool example” on page 25-10.

Data verification for FLEXCONV commands

For each of the following commands, data verification is performed before execution.

Data verification for the MOVE command

The following verifications are performed before executing the the MOVE command:

- If the old trunk group name or the new trunk group name does not exist, the movement does not take place. The following error message is displayed:

  ```
  clli is not valid
  ```

- If the old trunk group has no members, there are no members to move. No action is taken. The following error message is displayed:

  ```
  group has no members to move
  ```

- If the entries in the table CLLICDR for the old and new trunk groups contain different EXTNUM values, you can decide to modify the EXTNUM for the new trunk group CLLI to have the same value as the old trunk group CLLI. For this purpose, a message is displayed, and you are prompted to determine if the modification should be performed.

- If the percentage of members to be moved is 100%, this can cause a disruption of traffic on the trunk group. A warning message is displayed, and you will be asked whether or not to continue.
- If you attempt to move members from a PTS trunk group to a SS7 trunk group, the following warning message is displayed:

  ```
  use the TRKCONV to convert the PTS to SS7
  ```

  For SS7 to SS7 conversions, the tool only takes care of modifying table C7TRKMEM.

- If you attempt to move members from trunk group SS7 to trunk group PTS, the following message is displayed:

  ```
  cannot convert SS7 to PTS
  ```

  Note: SS7 to PTS can be converted in two stages: first convert PTS to PTS using the FlexDial conversion tool, and then use the TRKCONV tool to convert it to SS7. Alternatively, the TRKCONV tool could be used first followed by the FLEXCONV tool.

- If the old trunk group uses a trunk subgroup (SGRP) that is not datafilled for the new trunk group. Then, the following error message is displayed:

  ```
  Table <table name> must be filled for the <new trunk CLLI>
  at SGRP <trunk subgroup number>.
  ```

Data verification for the DELREN command

The following verifications are performed before executing the DELREN command:

- If the old trunk group name or the new trunk group name does not exist, the “delete and rename” action does not take place. The following error message is displayed:

  ```
  clli is not valid
  ```

- If the old trunk group still has trunk members, the trunk group can not be deleted. The “delete and rename” action does not take place. The following message is displayed:

  ```
  group has members
  ```

Data verification for the CHNGROUT command

The following verifications are performed before executing the CHNGROUT command:

- If the old trunk group name or the new trunk group name does not exist, the route reference change does not take place. The following message is displayed:

  ```
  clli is not valid
  ```

- The new trunk group must have at least one trunk member in an idle state. If there are no trunk members in the idle state, the traffic on this trunk group can be disrupted. In this case, route reference changes do not take place. The following error message is displayed:

  ```
  At least one trunk should be in idle state
  ```

Data verification for the QUERYMV command

The following verifications are performed before executing the QUERYMV command:

- If the old trunk group name does not exist, the Query movement in process does not take place. The following error message is displayed:

  ```
  clli is not valid
  ```

- If there is no active process, the following message is displayed:

  ```
  The group <clli> is not being processed
  ```

Data verification for the HALTMV command

The following verifications are performed before executing the HALTMV command:

- If the old trunk group name does not exist, the Halt request is not performed. The following error message is displayed:

  ```
  clli is not valid
  ```

- If the old trunk group is not being processed, the following error message is displayed:

  ```
  group is not being processed
  ```

FLEXCONV tool step-by-step example

Follow the following steps to use the FLEXCONV tool:

1 First, define a new trunk group. This trunk group replaces an existing trunk group. As a minimal requirement, datafill the following tables: CLLI, CLLICDR and TRKGRP. For non-AXXESS trunks, also datafill tables TRKGRP1 and TRKSGRP. For AXXESS agencies, datafill tables TKRFEAT, TRKSIG and FLEXDIAL. For SS7 trunks, datafill table ISUPDEST.

2 Issue the the following sequence of commands to access the FLEXCONV tool:

```
> MAPCI; MTC; TRKS; TTP; LEVEL FLEXCONV; MVMEMBER
```

3 At this point, you can start moving trunk members from the old trunk group to the new trunk group. It is recommended that you move a portion of the members first; for example, move 50 percent of all members in the group. Use the following command to do this:

```
> MOVE <old trunk group name> <new trunk group name> 50
```

Here, 50 is the percentage of members to be moved. You can choose the percentage you want.

Note: The parameter “percentage” relates to the members that are still in the trunk group when the command is executed. For example, if a trunk group had 100 members and a user said to move 10%, then 10 members would be moved, leaving 90 members in the group. A subsequent move of 10% would move 9 members, leaving 81 in the group.

The movement has started. You must wait before proceeding to the next step. In the meantime, you can exit this MAPCI level. If no failure occurs, a log report FLCV302 is generated signifying a successful completion.

4 During the movement, you can use the QUERYMV command to **c**heck the status of the request. For example:

```
> QUERYMV <old trunk group name>
```

If the request is being processed, the command displays a number of trunk members that have been already moved to the new group.

5 Once the movement is completed and the log report FLCV302 is generated, you can return to the FLEXCONV level. In order to change the route references of the old trunk group to the new trunk group, use the following command:

```
> CHNGROUT <old trunk group name> <new trunk group name>
```

When the command is completed, a message is displayed with results of this step.

6 You should repeat steps 2–4, but this time you can move 100% of trunk members (instead of 50%). This moves all the trunk members of the old trunk group to the new trunk group. The old trunk group now has no trunk members, and you can execute the DELREN command.

7 Once all the trunk members have been moved to the new trunk group, you can delete the old trunk group, and you can rename the new trunk group to the old trunk group name. For this purpose, you can use the DELREN command:

```
> DELREN <old trunk group name> <new trunk group name>
```

This operation deletes all occurrences of the old trunk group name and renames all occurrences of the new trunk group name in all affected tables to the old name. Upon a successful completion of the DELREN command, a log report FLCV302 is generated. If the deletion of the old trunk group cannot be done, the process is stopped and an error message is displayed.

Failure conditions

While executing any of commands, the following failures can occur:

- If the state of a trunk cannot be changed during the movement process, the conversion of that trunk is aborted and the conversion process proceeds to the next trunk. A log report FLCV300 is generated.
- If the table TRKMEM or C7TRKMEM cannot be successfully updated during the movement process, a log report FLCV301 is generated and the process is aborted.
- If a table cannot be updated during the change of route references, the change process is aborted. An appropriate message is displayed and a log report FLCV301 is generated.
- If the “move members” process is interrupted by either a COLD or WARM restart, the process is renewed automatically after the restart is completed. A RELOAD restart aborts the process.
- Any restart (RELOAD, COLD or WARM) aborts the DELREN or CHNGROUT commands.

Restrictions and limitations

The following restrictions and limitations apply to the FLEXCONV tool:

- The conversion tool supports PTS to PTS and SS7 to SS7 conversions only.
- The conversion tool does not support PTS to PTS conversions, when pulse types differ (for example, the old trunk group is set up for MF and the new trunk is set up for DTMF).
- You must verify that the new trunk group signaling corresponds to the old trunk group signaling.
- The conversion tool does not support PRI trunk conversions.
- The following functionalities are not covered by this feature:
  - — subscriber provisioning conversion
  - — trunk group provisioning conversion
  - — creation of new trunk group
  - — full trunk group migration
  - — provisioning consistency check between old and new trunk groups.
- The tool can process only one “movement” request at a time. Another request, whether from a different map or the same one, will result in the following message: “Error: Another conversion request is being processed.”
- When DELREN or CHNGROUT commands are performed, the MAP terminal is locked until the request is completed.

Appendix C
FlexDial provisioning examples

This appendix provides FlexDial provisioning examples.

Agent provisioning example

To show the improved organization of the data model for FlexDial framework, Figures 26-1 and 26-2 provide agent provisioning examples without using FlexDial and with using FlexDial, respectively.

Figure 26-1
Non-AXXESS agent provisioning example

```
TRKGRP

EAN670TWMFWK
    GRPTYP: EANT
    TRAFSNO: 0
    PADGRP: NPDGP
    NCCLS: NCIT
    COS: 6
    DIR: 2W
    PRTNM: EAN
    SELSEQ: MIDL
    ODSCFLTR: 16
    ORIGFLTR: 7
    TDSCFLTR: 16
    ANSWFLTR: 16
    IDPRTRAN: EAPT
    SNDRPSIG: 5
    SNDRPDIL: 5
    SNPA: 214
    ISUPIDX: NILIDX
    TRAFCLS: NIL
    CONNGNPA: 703
    OPART: 111
    RECALLDT: MANUAL
    TIMEBIAS: –2
    ZEROMPOS: RTE1
    ZONE: 0
    ADIN: 1
    BCNAME: 3_1KHZ
    TSUSR: 160
    OPTION: ALTTRTMT
    OPTION: TMANIDLV
                CGNONLY
    OPTION: OHQ
    OPTION: OHQTERM
```

```
TRKGRP1

EAN670TWMFWK
    GRPTYP: EANT
    ACPROMPT: N
    AUTHFRST: Y
    AUTHDIAL: 7
    DIALTONE: C
    MLTSTAGE: Y
    CN: 0
    OSSSIG: NONE
    ANICLAS: NOLOOK
    PINADDRP: N
    VAUTHFLD: NOAUTHS
    SPARE1: Y
    SPARE3: 0
    SNXX: 555
    SPARE2: Y
    SPARE4: 0
    LCDDUR: 0
```

```
TRKSGRP

EAN670TWMFWK 0
    CARDCODE: DS1SIG
    SGRPVAR: STD
    DIR: 2W
    IPULSTYP: DT
    ISTARTSG: WK
    OVLP: N
    PSPDSEIZ: 5
    PARTDIAL: 5
    OPULSTYP: MF
    OSTARTSG: WK
    IDGTIME: 6
    NUMSTOPS: 0
    GLAREYD: N
    CCONT: NO
    RNGBCK: NO
    ESUPR: N
    SAT: N
    REMBSY: Y
    DIALMODE: M
    TRKGRDTM: 70
    ECSTAT: UNEQ
```

Figure 26-2
AXXESS agent provisioning example

```
TRKGRP

AXX670TWMFWK

    GRPTYP: AXXESS

    TRAFSNO: 0

    PADGRP: NPDGP

    NCCLS: NCIT

    SELSEQ: MIDL

    SIGIDX: MF_WK_IDX

    FEATIDX: AXX670_IDX

    DPIDX: I_OLI_ANI_CV
           ITC_AU_SD_AD

    OGRPTYPE: EANT
```

```
TRKFEAT

AXX670

    ORIGOPTS: (ALTTRTMT) (OHQ) (REORIGAL $) (SNPA 214)
              (SNXX 555) (TIMEBIAS –2) (TRKCOS 6) (MSGCTR
              670)$

    TERMOPTS: (NOANSDUR 10 TRMT RODR) (OHQTERM) (SNPA
              703) (TRKCOS 6)$

MSGCTR

670

    ADDRESS: (ADDR N Y PRTNM EAN) (OLI N Y PRTNM EAPT)
             (ADDR OPER NORMAL OFRT 1 EAN OFRT 2)$


TRKSIG

MF_WK_IDX

    SIGTYPE: DS1

    IPULSTYP: MF

    ISTARTSG: WK

    PSEIZTMR: 5

    PDILTMR: 5

    MINRTMR: 2

    FDIGMASK: KP

    LDIGMASK: ST STP ST2P ST3P

    DIALMODE: M

    OSTARTSG: WK

    OPULSTYP: MF

    OIDGTMR: 6

    TRKGRDTM: 70

    OPTIONS: (ANSWFLTR 16) (DELIVER CGNONLY)
             (ACKWINK) (MLTSTAGE) (ODSCFLTR 16)
             (ORIGFLTR 7) (REMBSY) (TDSCFLTR 16)$
```

Subscriber number provisioning example

Figure 26-3 provides a subscriber number provisioning example without using FlexDial and using FlexDial, respectively.

Figure 26-3
Subscriber number provisioning example

Without FlexDial

```
ANISCUSP

2145555511
     STATUS: AL
     ACCTLEN: 6
     ACCTVAL : N
     BCNAME: 3_1KHz
     OPART: 511
     TERMPART: 31
     SATRES: N
     PINLEN: 0
     PININDEX: 0
     PINDIGS: $
     COSINDEX: 11
     ANIDELV : ALWAYS
     OPTIONS: ACCTIDX 10011
```

With FlexDial

```
FLEXTYPE

ANI
     OPTIONS: (BILLFLD ANISP) (CALLING) (FLEXLOG)$

FLEXVAL

ANI 1 2145555511
     FEATIDX: 10011

FLEXFEAT

10011
     OPTIONS: (MLTCOSID 11) (DELIVER ALWAYS) (DPIDX
                 APPEND ACCT_CTONE) (MSGCTR 10011)
                 (REORGACT ADDR_IDX 110011 15)
                 (REORGTYP ONKEY STR S 20 1) (TRANSTS
                 PARTITION 31 511)$

MSGCTR

10011
     ADDRESS: (SUBR ACCT MINMAX 6 6) (SUBR ACCT
                 INDEXES (10011)$)$
```

FlexDial provisioning example

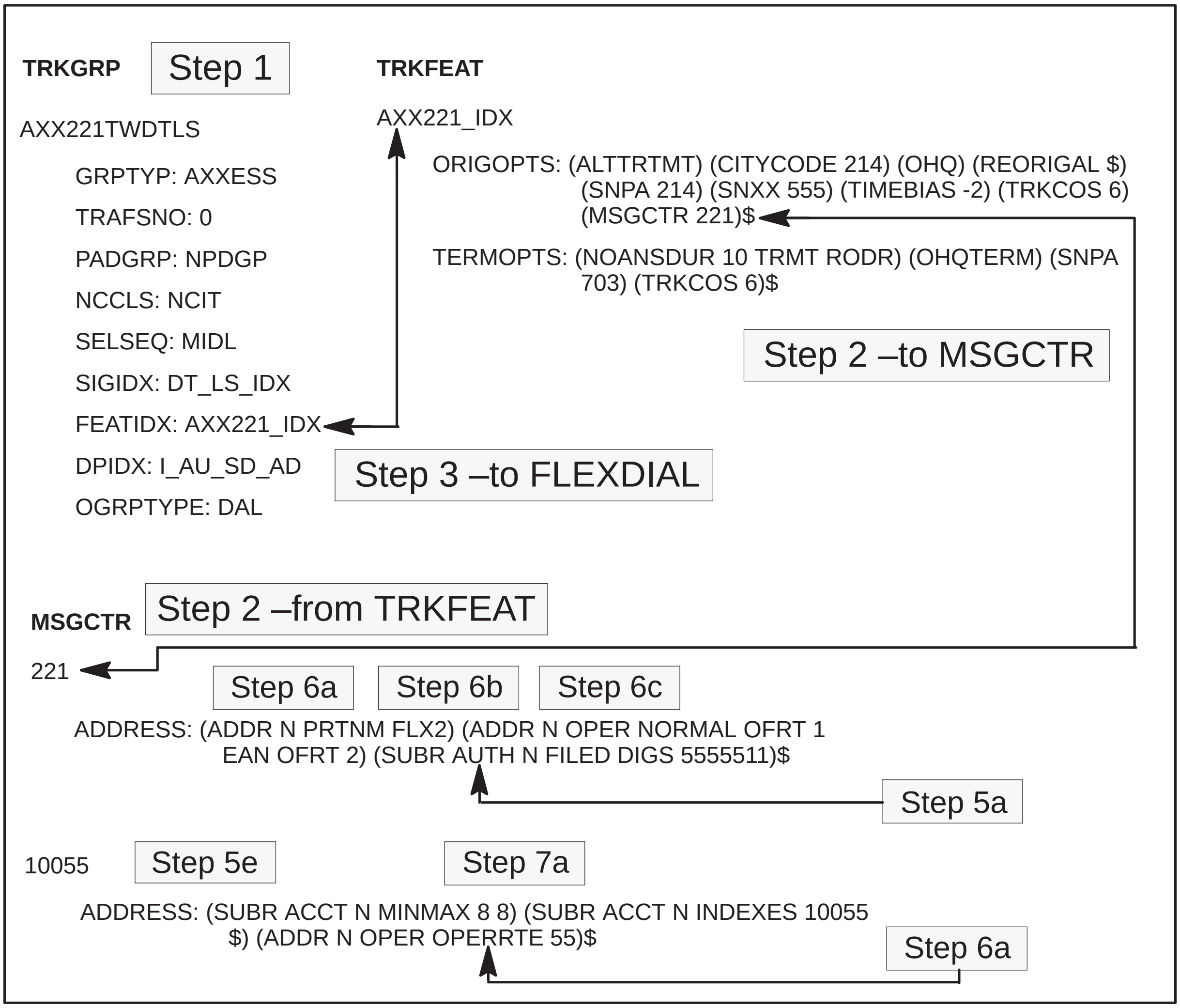
This section provides a detailed FlexDial provisioning example, including step-by-step instructions.

Basic UCS DMS-250 switch authorization code service

Figures 26-4, 26-5, and 26-6 contains the datafill for a basic authcode service example that includes the following:

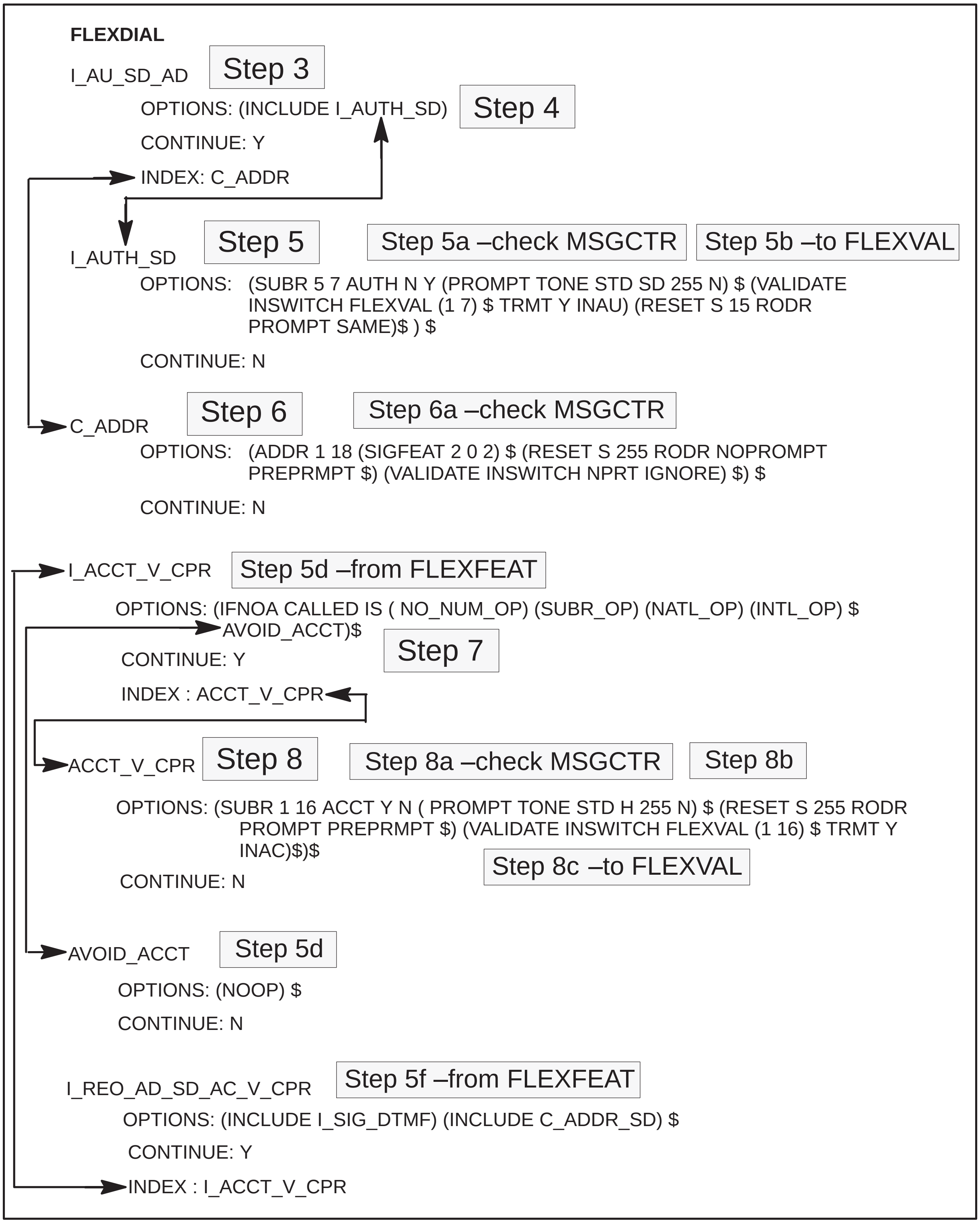
- detailed agent, dialplan, and subscriber provisioning
- filed authcode for agent
- account code collection associated with authcode screened

Figure 26-4
Basic authcode example—agent provisioning



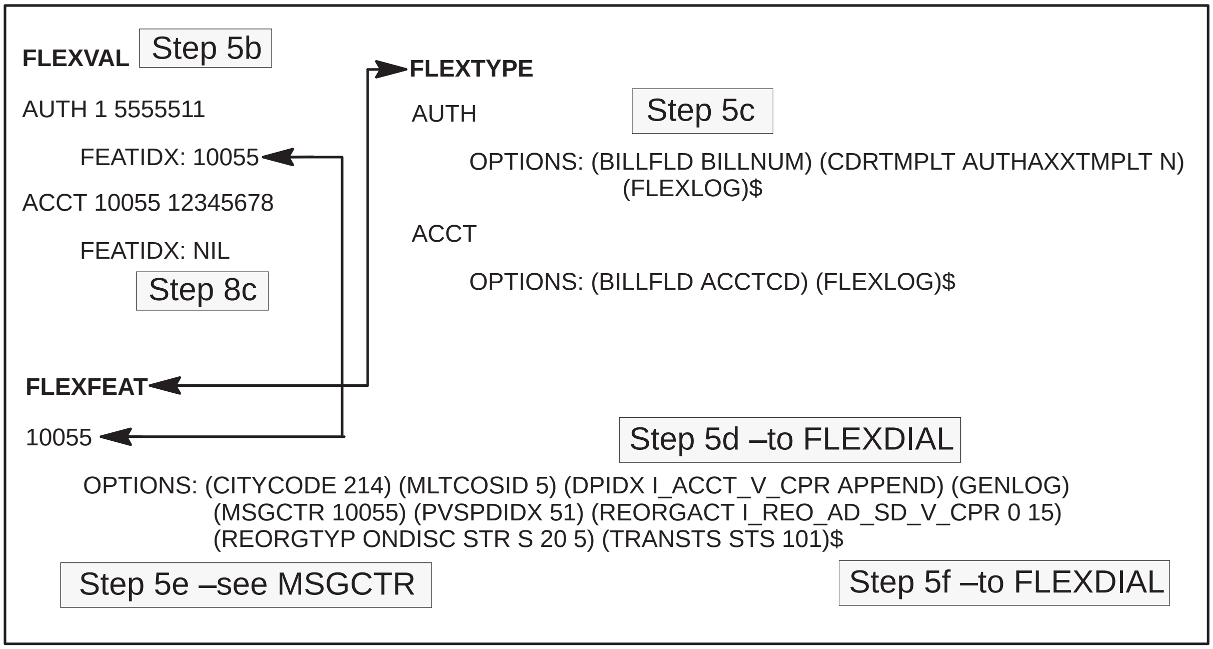
```
TRKGRP

AXX221TWDTLS

     GRPTYP: AXXESS
     TRAFSNO: 0
     PADGRP: NPDGP
     NCCLS: NCIT
     SELSEQ: MIDL
     SIGIDX: DT_LS_IDX
     FEATIDX: AXX221_IDX
     DPIDX: I_AU_SD_AD
     OGRPTYPE: DAL

TRKFEAT

AXX221_IDX

     ORIGOPTS: (ALTTRTMT) (CITYCODE 214) (OHQ) (REORIGAL $)
               (SNPA 214) (SNXX 555) (TIMEBIAS -2) (TRKCOS 6)
               (MSGCTR 221)$

     TERMOPTS: (NOANSDUR 10 TRMT RODR) (OHQTERM) (SNPA
               703) (TRKCOS 6)$

MSGCTR

221
     ADDRESS: (ADDR N PRTNM FLX2) (ADDR N OPER NORMAL OFRT 1
              EAN OFRT 2) (SUBR AUTH N FILED DIGS 5555511)$

10055
     ADDRESS: (SUBR ACCT N MINMAX 8 8) (SUBR ACCT N INDEXES 10055
              $) (ADDR N OPER OPERRTE 55)$
```

Figure 26-5
Basic authcode example—dialplan provisioning



```
FLEXDIAL
I_AU_SD_AD
        OPTIONS: (INCLUDE I_AUTH_SD)
        CONTINUE: Y
        INDEX: C_ADDR

I_AUTH_SD
        OPTIONS:   (SUBR 5 7 AUTH N Y (PROMPT TONE STD SD 255 N) $ (VALIDATE
                   INSWITCH FLEXVAL (1 7) $ TRMT Y INAU) (RESET S 15 RODR
                   PROMPT SAME)$ ) $
        CONTINUE: N

C_ADDR
        OPTIONS:   (ADDR 1 18 (SIGFEAT 2 0 2) $ (RESET S 255 RODR NOPROMPT
                   PREPRMPT $) (VALIDATE INSWITCH NPRT IGNORE) $) $
        CONTINUE: N

I_ACCT_V_CPR
      OPTIONS: (IFNOA CALLED IS ( NO_NUM_OP) (SUBR_OP) (NATL_OP) (INTL_OP) $
                 AVOID_ACCT)$
        CONTINUE: Y
        INDEX : ACCT_V_CPR

ACCT_V_CPR
      OPTIONS: (SUBR 1 16 ACCT Y N ( PROMPT TONE STD H 255 N) $ (RESET S 255 RODR
                 PROMPT PREPRMPT $) (VALIDATE INSWITCH FLEXVAL (1 16) $ TRMT Y
                 INAC)$)$
        CONTINUE: N

AVOID_ACCT
       OPTIONS: (NOOP) $
       CONTINUE: N

I_REO_AD_SD_AC_V_CPR
        OPTIONS: (INCLUDE I_SIG_DTMF) (INCLUDE C_ADDR_SD) $
         CONTINUE: Y
         INDEX : I_ACCT_V_CPR
```

Figure 26-6
Basic authcode example—subscriber provisioning



```
FLEXVAL
AUTH 1 5555511
     FEATIDX: 10055
ACCT 10055 12345678
     FEATIDX: NIL

FLEXTYPE
AUTH
     OPTIONS: (BILLFLD BILLNUM) (CDRTMPLT AUTHAXXTMPLT N)
              (FLEXLOG)$
ACCT
     OPTIONS: (BILLFLD ACCTCD) (FLEXLOG)$

FLEXFEAT
10055
     OPTIONS: (CITYCODE 214) (MLTCOSID 5) (DPIDX I_ACCT_V_CPR APPEND) (GENLOG)
              (MSGCTR 10055) (PVSPDIDX 51) (REORGACT I_REO_AD_SD_V_CPR 0 15)
              (REORGTYP ONDISC STR S 20 5) (TRANSTS STS 101)$
```

Basic authcode example steps

The following steps correspond to the steps in Figures 26-4, 26-5, and 26-6:

1 Origination occurs on a member of the AXX221TWDTLS trunk group.

2 Features and characteristics of the AXXESS agent are set for the call, including the processing of the TRKFEAT MSGCTR option, which posts the ADDR PRTNM, ADDR OPER, and SUBR AUTH FILED messages at the per-call message center.

3 The initial interaction with the originating agent is defined by the I_AU_SD_AD FLEXDIAL index. This identifies the collectable list to be processed as (INCLUDE I_AUTH_SD) (ADDR).

4 The INCLUDE collectable identifies that the I_AUTH_SD FLEXDIAL index is processed. The collectable list is modified and the new list is:

(INCLUDE) (SUBR AUTH) (ADDR)

5 The SUBR AUTH collectable is executed with the following highlights:

a. Initial processing of the collectable consumes the SUBR AUTH FILED message from the message center. The FILED message identifies the required digits for the digit collectable, so no digits are collected.

b. Validation of the SUBR AUTH digits occurs through table FLEXVAL. The FLEXFEAT index of 10055 identifies the features and characteristics of the subscriber number.

c. As specified by FLEXTYPE, the identified SUBR AUTH digits are captured in the BILLNUM field of the CDR.

d. Through the DPIDX option, the collectable list identified by the I_ACCT_V_CPR FLEXDIAL table index is appended to the currently processing collectable list, appending a SUBR ACCT collectable. The list becomes: (INCLUDE) (SUBR AUTH) (ADDR) (SUBR ACCT).

e. Through the MSGCTR option, the messages identified by index 10055 are posted at the message center. This includes messages SUBR ACCT MINMAX, SUBR ACCT INDEXES, and the ADDR OPER message.

   – Using SUBR ACCT MINMAX, you redefine the number of digits the SUBR ACCT collectable will collect. This way you don’t have to provision multiple versions of the SUBR ACCT collectable, each with different MIN/MAX values.

   – With the INDEXES message, the index to use for validation is specified. This way you don’t have to provision multiple versions of SUBR ACCT with different VALIDATE options.

   – The ADDR OPER message identifies information for operator call types. This is the same sort of information you could get from OPCHOICE, but using FlexDial you’re not limited to 255 indexes.

f. Other features and characteristics are processed, and the STS is also identified.

6 After the SUBR AUTH collectable completes, the ADDR collectable is executed.

a. Initial processing of the collectable consumes all messages from the message center for ADDR, including the ADDR PRTNM and ADDR OPER messages. The last ADDR OPER message posted (ADDR N OPER OPERRTE 55) is used for operator calls.

b. Up to six digits are collected and used to index STDPRTCT. MIN/MAX values from STDPRTCT are used to update the MIN/MAX values for ADDR. Remaining digits are then collected.

c. Validation occurs using the FLX2 pretranslator name.

7 After ADDR, the IFNOA collectable is executed. If the call is not an operator call, then branching of collectable processing does not occur.

8 After IFNOA, the SUBR ACCT collectable is executed (for the non-operator call).

   a. Initial processing of the collectable consumes all messages from the message center for SUBR ACCT, including the MINMAX and INDEXES messages.

   b. A continuous high tone prompt is provided for ACCT collection.

   c. Collection and validation of the SUBR ACCT digits occur, with information from the message overriding provisioned information.

9 After SUBR ACCT, collectable processing is complete, and the call proceeds with translations and routing.

Appendix D FlexDial interactions

This appendix describes the following interactions with UCS DMS-250 features:

- CAIN and FlexDial interactions
- TCAP and FlexDial interactions

CAIN and FlexDial interactions

This section is a high-level overview of the interactions between CAIN and the FlexDial framework. You should understand and be proficient in normal CAIN (non-FlexDial) call processing, as well understanding the provisioning of the FlexDial Framework.

Note: For more information on Carrier AIN, refer to the *UCS DMS-250 NetworkBuilder Application Guide*. Please refer to *UCS DMS-250 CAIN/FlexDial Interactions* for more detailed information on CAIN and FlexDial interactions.

Overview

CAIN 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.

The FlexDial Framework also changes the way the switch handles a call by creating an open software system provisioned entirely through datafilled options and features. You provision the interface between your network, the originating switch, and the end subscriber.

Together, CAIN and FlexDial offer a more flexible approach to both in-switch and SCP processing of a UCS DMS-250 call.

What is Carrier AIN?

Carrier AIN (CAIN) is Nortel Networks’ optional advanced intelligent network (AIN) offering for the UCS DMS-250 market and is part of the NetworkBuilder software platform. Based on Bellcore’s AIN 0.2 specifications, CAIN was developed specifically for interexchange carriers (IXC).

General AIN

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.

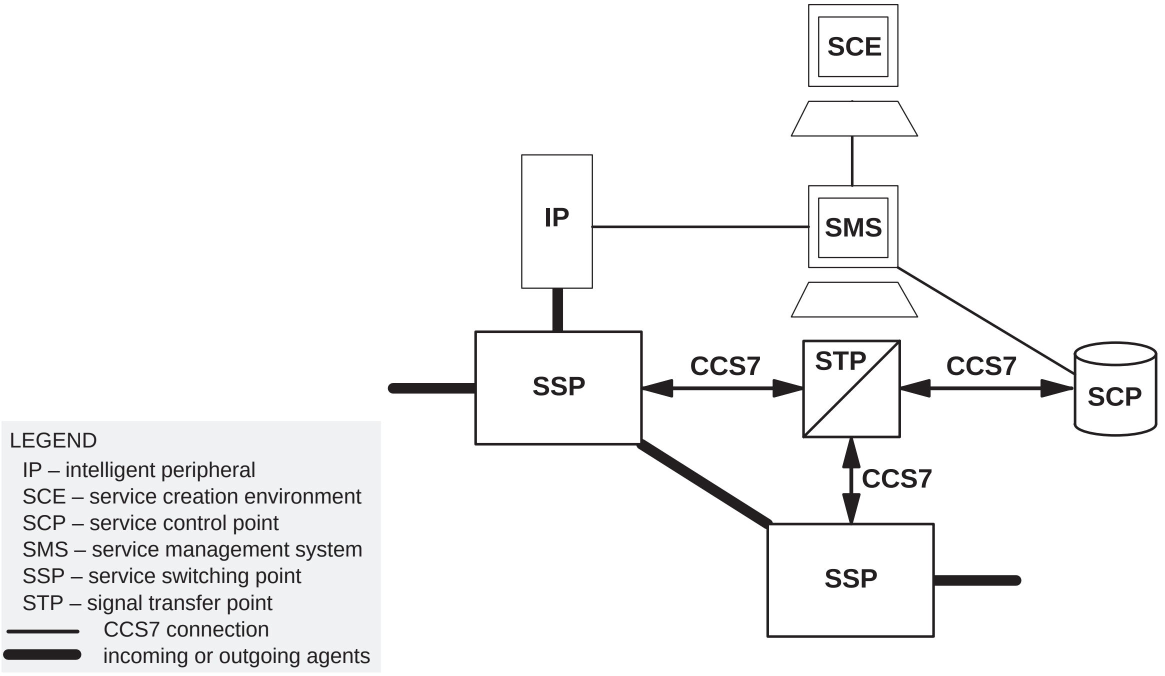
Note 1: CAIN requires the use of a UCS DMS-250 switch as the SSP.

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

Figure 27-1 shows the hardware components available for designing an AIN system. Descriptions of each component follow Figure 27-1.

For more information on the hardware components, refer to the *UCS DMS-250 NetworkBuilder Application Guide*.

Figure 27-1
Example of an AIN network model


SCE
IP
SMS
SSP
CCS7
STP
CCS7
SCP
CCS7
SSP
LEGEND
IP – intelligent peripheral
SCE – service creation environment
SCP – service control point
SMS – service management system
SSP – service switching point
STP – signal transfer point
CCS7 connection
incoming or outgoing agents

Note: CAIN requires that a UCS DMS-250 switch be used as the SSP.

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.

The UCS08 software release also provides support for EDPs; refer to the *UCS DMS-250 NetworkBuilder Application Guide* for more information.

Note: Interaction between CAIN software and the FlexDial Framework does not support EDPs.

Subscription

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

Addresses, authcodes, ANIs, originating agents, SUBR collectables, SUBRPARM collectables, CALLTYPE collectables, or the office can subscribe to a CAIN group provisioned in table CAINGRP. The CAIN group, in turn, enables one or more triggers. Enabling a trigger provides an index into the appropriate trigger table where trigger criteria (required to query the SCP) is defined.

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

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*
- ignore the criteria and continue processing
- continue processing as directed by datafill
- block the call and apply AINF treatment

CAIN call processing stores up to six subscription groups for use throughout the call. The six groups are determined through:

- SCP-returned CAIN group
- Address subscription
- Authorization code, SUBR collectable, SUBRPARM collectable, or CALLTYPE collectable subscription
- ANI, SUBR collectable, or SUBRPARM collectable subscription
- Agent subscription
- Office subscription

Supported PICs

The UCS08 CAIN software release supports the PICs, TDPs, and triggers shown in Table 27-1.

Table 27-1
Supported PICs, TDPs, EDPs, and triggers

PIC	TDP/EDP	Trigger
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	*Offhook_Delay* trigger
		Shared_Interoffice_Trunk trigger
		PRI_B-Channel trigger
PIC 4: **Analyze_Information**	***Info_Analyzed*** TDP	*Specific_Feature_Code* trigger
		Customized_Dialing_Plan trigger
		Specific_Digit_String trigger
		Office_Code trigger
PIC 5: **Select_Route**	***Network_Busy*** TDP	*Network_Busy* trigger
	Network_Busy EDP	*Network_Busy* 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 1: Since the FlexDial Framework does not support PRI, CAIN/FlexDial interaction does not support *PRI_B-Channel*.

—continued—

Table 27-1
Supported PICs, TDPs, EDPs, and triggers (continued)

PIC	TDP/EDP	Trigger
PIC 8: **O_Alerting**	***O_Answer*** EDP	*O_Answer* event
	O_No_Answer TDP	*O_No_Answer* trigger
	O_No_Answer EDP	*O_No_Answer* event
	O_Mid_Call TDP	*O_IEC_Reorigination* trigger
PIC 9: **O_Active**	***O_Disconnect*** EDP	*O_Disconnect* event
	O_Mid_Call TDP	*O_IEC_Reorigination* trigger
	O_Mid_Call EDP	*Timeout* event
PIC 10: **O_Suspended**	***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 1: Since the FlexDial Framework does not support PRI, CAIN/FlexDial interaction does not support *PRI_B-Channel*.		
—end—		

SCP interaction

CAIN 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.

Why CAIN/FlexDial?

CAIN/FlexDial interaction enhances the existing CAIN call processing functions by

- supporting calls originating on an AXXESS agent
- allowing subscription on any subscriber digits (such as authorization codes, personal identification numbers, account codes, or calling card numbers)

- allowing subscription based on the dial plan (using subscription by call type)

Changes required by CAIN/FlexDial interaction

Interactions with the FlexDial Framework changes CAIN call processing in the following ways:

- allows originations from AXXESS agents
- provides new subscription methods for FlexDial collectables
- recognition of digits collected by the FlexDial Framework

The following changes were made to the FlexDial Framework in order to allow interaction with CAIN call processing:

- support for CAIN-capable AXXESS agents in table TRKFEAT
- allows subscription to CAIN groups in tables FLEXFEAT and TRKFEAT through the CAINGRP option
- adds the CAINFLG option to table FLEXTYPE. The CAIN flag maps the subscriber number to a digit type that CAIN can understand.

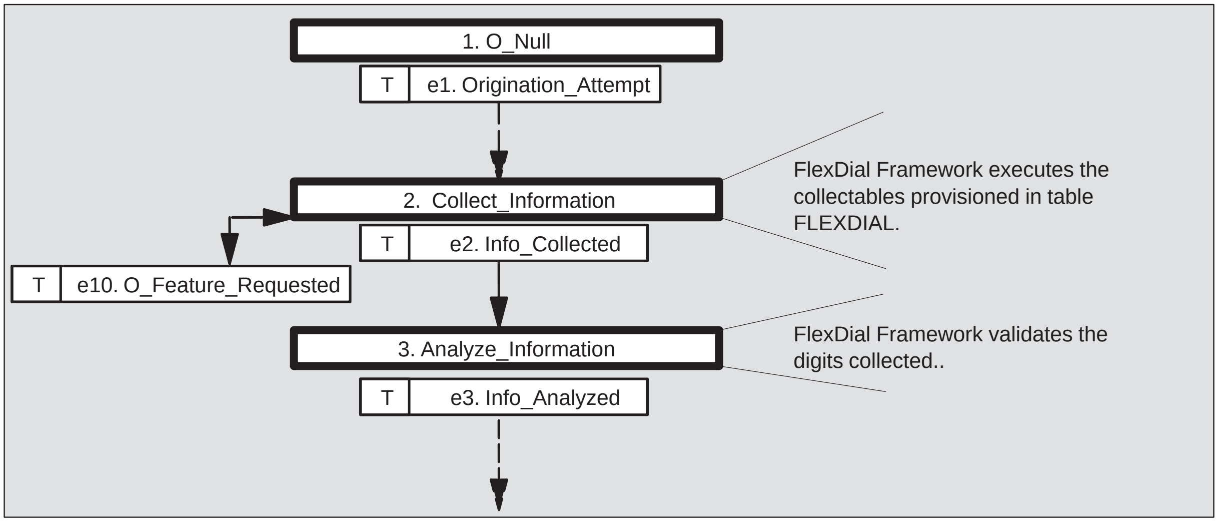
CAIN interacts with the FlexDial Framework in two ways:

- supporting originating AXXESS agents (which requires use of the datafillable dial plans in the FlexDial Framework)
- digit collection provided through the datafillable dial plans

Digit collection and digit validation

The FlexDial Framework provides the digit collection and digit validation for originating AXXESS agents. The majority of the interaction between CAIN and FlexDial occurs during **Collect_Information** and **Analyze_Information** (Figure 27-2).

Figure 27-2
CAIN call model



TCAP and FlexDial interactions

FlexDial supports call processing of data received in a Transaction Capabilities Part (TCAP) Version 2 response message. This section discusses how this functionality is implemented in tables FEATBYTE and TCAPANNC, for both AXXESS and non-AXXESS agents.

Note: For more information on TCAP, refer to the *UCS DMS-250 Transaction Capabilities Application Part (TCAP) Application Guide*. Please refer to *UCS DMS-250 Data Schema Reference Manual* for more detailed information on tables FEATBYTE and TCAPANNC.

FEATBYTE table

This section describes the Feature Byte (FEATBYTE) table.

Purpose

The FEATBYTE table is only used with the FlexDial framework. The FEATBYTE table identifies certain call types and interaction requirements directly related to information contained in a TCAP response message.

When a Feature Bytes parameter is received in the TCAP response message, the parameter data is used to index the FEATBYTE table to identify data for call processing or to specify additional interactions with the originating agent that need to be processed. The Feature Byte parameter contains eight feature bytes.

The FEATBYTE table is indexed by a numeric index, the feature byte number, and individual feature byte values used to identify the required processing options for that feature byte.

Key

The key for indexing the FEATBYTE table consists of a three-part key:

- number index with range of 0 to 8,191

  This number includes a unique numeric index that allows a particular feature byte and feature byte value to be provisioned multiple times with different data.

- feature byte number with range of 1 to 8

  This index identifies which feature byte value to use as the third part of the key.

- feature byte value with range of 0 to 255

  This index is the actual value of the identified feature byte received in the TCAP response message.

This three-part key is used to index the FEATBYTE table. Because the initial lookup in the table consists of two unknowns (the unique index and which feature byte to use first), values for the numeric index and feature byte number are initially identified by the office parameters, FEATBYTE_FIRST_INDEX and FEATBYTE_FIRST_BYTE. The FEATBYTE_FIRST_MASK office parameter masks the feature byte value in the initial lookup.

The FEATBYTE table is limited to 1,048,576 tuples, although the actual capability storage of the table is limited by system memory.

Table 27-2 provides the FEATBYTE table key.

Table 27-2
FEATBYTE table key

Key field	Description	Values
NUM_INDX	The first part of the table key consists of a unique numeric index value used to key the FEATBYTE table.	0 to 8191
FEATB_NUM	The second part of the table key consists of the feature byte number that needs to be screened next.	1 to 8
FEATB_VAL	The third part of the table key consists of the resulting value after the mask value has been applied to the feature byte value being used.	0 to 255

Fields

The FEATBYTE table does not contain a fixed set of fields, but consists of an options vector that is provisioned with the index, byte number, and feature byte values necessary to identify the required processing options for that feature byte. Each option in turn defines its own data refinement for datafill purposes. Therefore each option is listed independently.

The default field for the options identifies the value that is assumed for the feature options if this option is not explicitly provisioned. Normally, when a tuple consists of an options vector, the default is to not have any options provisioned. Due to the nature of feature specifications, some feature options require a default value if the option is not provisioned in the list.

The FEATBYTE table tuples contain one field as outlined in Table 27-3.

Table 27-3
FLEXMOD table fields

Field	Description	Values	Default
OPTIONS	This field identifies an options vector that indicates the call type or dial plan interaction required for the call.	DPIDX, FEATB, PROCESS, NIL	NA

DPIDX option

The dial plan index option identifies an index into the FLEXDIAL table. This index contains provisioned collectables that are processed for the call.

Definition

Table 27-4 contains the field refinements for the DPIDX option.

Table 27-4
Table FEATBYTE DPIDX option fields

Fields	Description	Values	Default
FLEXDIAL_INDX	Identifies FLEXDIAL table index that has been previously provisioned within the FLEXDIAL table.	Valid range of FLEXDIAL table indices	NA
ACTION	Identifies the action that is to occur with the addition of the collectable identified by the FLEXDIAL index to the setup information.	INSERT, APPEND, REPLACE, EXEC INSERT adds the new list into the current processing list before the next collectable. APPEND adds the new list to the end of the current processing list. REPLACE modifies the remainder of the unprocessed collectables with those identified by the FLEXDIAL index. EXEC executes the list identified by the FLEXDIAL table index as a sublist.	NA

Example

Examples of the DPIDX option provisioning are:

```
DPIDX ACCT_COL_IDX APPEND
```

In this example, the collectable list identified by the ACCT_COL_IDX FLEXDIAL table index is appended to the currently processing list of collectables.

DPIDX option restrictions and limitations

The FLEXDIAL table index stored in the INDEX field must be previously provisioned before it can be referenced by this option.

FEATB option

The FEATB option indicates that the identified feature byte must be screened next. This option provides a means of concatenating the processing of feature bytes.

Definition

Table 27-5 contains the field refinements for the FEATB option.

Table 27-5
Table FEATBYTE FEATB option fields

Fields	Description	Values	Default
NUM_INDX	This field identifies the unique numeric index value used to key the FEATBYTE table.	0 to 8191	NA
FEATB_NUM	This field identifies the feature byte number that needs to be screened next.	1 to 8	NA
FEATB_MASK	This field identifies the mask value that is applied to the feature byte value being used. The resulting value after the masking operation is right shifted to the least signification mask bit set.	0 to 255	NA

Example

Examples of the FEATB option provisioning are:

```
FEATB 5 3 255
```

In this example, the next feature byte processed is feature byte 3, using a unique numeric index value of 5 and a mask of 255 (#FF).

FEATB option restrictions and limitations

There are no identified table control restrictions for the FEATB option.

PROCESS option

The PROCESS option identifies that the feature byte value is to be used as a specific call processing data value.

Definition

Table 27-6 contains the field refinements for the PROCESS option.

Table 27-6
Table FEATBYTE PROCESS option fields

Fields	Description	Values	Default
PROC_DATA	This field identifies the data for which the value is used. Currently the only defined call processing specific data value contained in the feature bytes is the no answer duration timer value. NOANSTMR is derived as follows: Offset + (Featbyte_value * Multiplier) where Offset = FEATBYTE_NOANSDUR_OFFSET Multiplier= FEATBYTE_NOANSDUR_MULT	NOANSTMR The calculation of this value is based on two office parameters and the feature byte value (after the mask has been applied).	NA
PROC_MASK	This field identifies a mask that is applied to the feature byte value. The resulting value after the masking operation is right shifted to the least significant mask bit set.	0 to 255	NA

Example

Examples of the PROCESS option provisioning are:

```
PROCESS NOANSTMR 63
```

In this example,

PROCESS option restrictions and limitations

There are no identified table control restrictions for the FEATB option.

Table FEATBYTE Restrictions and Limitations

The following restrictions and limitations apply to table FEATBYTE:

- Call processing queries to the FEATBYTE table with a non-existing key generate a FLEX 401 log message.
- Duplicate options may not be provisioned for a table entry. Table control routines enforce this restriction.
- The FEATBYTE table is limited to 1,048,576 (2**20) tuples.

TCAPANNC table

The TCAP Announcement (TCAPANNC) table allows provisioning of a specific destination against a received TCAP standard or custom announcement parameter value. The call processing action directs the call to connect to an announcement, a tone, or an identified treatment.

The TCAPANNC table is fully defined in the *UCS DMS-250 Data Schema Reference Manual*. This section describes the FlexDial interactions with this table.

The TCAPANNC table is indexed by a two-part key. The first part is a numeric index, and the second part is an announcement index, which is used in turn as an index into several tables. Option TCAPANNC in table FLEXFEAT stores a range of values that match the numeric index range. The indices (0 to 2047) are allocated as follows:

- an index of 0 indicates a standard announcement for non-AXXESS agents
- an index of 1 indicates a custom announcement for non-AXXESS agents
- indices 2 to 2047 are reserved exclusively for AXXESS originating agents

TCAP and FlexDial call processing interactions

Utilization of provisioned data

The FEATBYTE table is accessed to collect call interactions during the processing of feature bytes. The TCAPANNC table is accessed during the processing of the Custom Announcement parameter.

Processing TCAP response message

Up to three DNIS and Billing Number parameters can be received in the TCAP Response message. This feature utilizes the first DNIS and Billing number parameters for all originating agents. For AXXESS agents only, all three possible DNIS and Billing Number parameters are utilized. DNIS numbers determine the final routing number, and the Billing numbers are written into the CDR.

Processing all routing numbers from TCAP response message

Up to three Routing parameter numbers can be received in the TCAP Response message. Only AXXESS agents can use each of these three Routing numbers.

Call Processing (AXXESS agent only)

This section describe call processing features developed only for AXXESS originating agents.

Use of the FEATBYTE table

Feature bytes are received and processed during the processing of the ADDR or ADDRPARM digit collectable. Eight feature bytes are received in the TCAP response to an SCP N00 query message. The feature bytes indicate that, for this N00 number, the following interaction with the originating agent or process variable must be used for this call.

When a Feature Bytes parameter is received in the TCAP response message, the parameter data is used to index the FEATBYTE table to identify data for call processing or specify additional interactions with the originating agent that need to be processed.

For calls originating on AXXESS agents, the Feature Bytes parameter is used in the connection control component operation (with ”connect” operation specifier) and consists of eight bytes of data that define how the call is to continue processing, as described below:

1. The feature bytes from a TCAP response message and the FEATBYTE office parameters are used by call processing to derive the initial set of key values.
2. The key values are used to index into the FEATBYTE table to obtain the options associated with that key.
3. The options returned from the FEATBYTE table define the interactions with the originating agent for that feature byte.

The FEATB option allows the examination of additional feature bytes within the FEATBYTE table. Thus, multiple feature bytes (and the data provisioned for them in the FEATBYTE table) can be concatenated together through the use of the FEATB option.

The DPIDX option provides an index into the FLEXDIAL table and an associated action to take. Call processing accesses the FLEXDIAL table with the given index. A filed MSGCTR message is then generated for the ADDR collectable with the tuple from the FLEXDIAL table and the associated action to take. As mentioned in the previous paragraph, through the use of the FEATB option, multiple DPIDX options can be utilized.

The PROCESS option allows a mechanism to utilize the feature byte value as a specific call processing data value. Currently the only defined use for this option is the ”No answer duration timer” (NOANSTMR) value. Thus, a feature byte value of 25 for this option would define that the ”No answer duration timer” be set utilizing the feature byte value of 25 (in the equation ”offset+[featbyte value * multiplier]”). Call processing builds a filed MSGCTR message for the CALLTYPE collectable with the given feature byte value.

Call processing access to the FEATBYTE table is activated by the agent type for that call. When the agent is an AXXESS agent, the FEATBYTE table is accessed to determine the interactions with the originating agent.

Note: For non-AXXESS agents, only the least significant bit of the eighth feature byte is checked for call processing related actions. This bit is identified as the N00 Passthru indicator. The N00 Passthru indicator determines if special routing should be used for the call. Refer to *Feature AD8348, V2 TCAP Enhancements*, for a detailed description of the N00 Passthru functionality.

Use of the TCAPANNC table

The TCAPANNC table allows custom announcements for a particular subscriber by utilizing the entire range of indices (and provisioned data) in the TCAPANNC table.

The TCAPANNC table is access via a two-part key, which consists of an announcement index and a numeric index. The announcement index is derived from the data value within the announcement parameter, and whether a referral number parameter is present. The valid range of the data value is limited to 0-127 for the announcement parameter. If a referral number is not present, this range (0-127) is used. However, if a referral number is present, 128 is added to the announcement parameter value before indexing the table (thus a range of 128 to 255 is used).

The derivation of the numeric index differs, based on which announcement parameter (Standard or Custom) is received within the TCAP response message.

- TCAP Standard announcement parameters

  The numeric index portion of the key into the TCAPANNC table is always 0 (zero).

- TCAP Custom announcement parameters

  The numeric index portion of the key is determined through processing of a FLEXFEAT table TCAPANNC option for a previously executed subscriber number or Call Type collectable. If a TCAPANNC option has not been processed., then a numeric index value of one (1) is used by default.

If a tuple is not found in the TCAPANNC table using the custom announcement numeric index, a numeric index of 0 (zero) is used by default.

Call processing access to custom announcements for a subscriber is determined by the originating agent of the call. If the originating agent is an AXXESS agent, then the new scheme of accessing the TCAPANNC table is utilized.

For non-AXXESS agents, the existing UCS DMS-250 switch capability is used. (The existing capability sets the numeric index portion of the key to 0 (zero) for the Standard Announcement parameter, and to 1 (one) for the Custom Announcement parameter.)

Identifying features associated with N00 digits

As part of the Feature Bytes parameter processing, a filed message is always posted to the MSGCTR for the SUBR collectable. This message contains the N00 digits dialed and a FLEXTYPE index (defined by the FLEXDIAL_N00_FLEXTYPE office parameter).

Note: For a detailed description of MSGCTR, see Chapter 4, ”Table MSGCTR.”

Once the filed message has been posted to the MSGCTR, the Feature Bytes parameter returned from the TCAP Response message are processed. The processing of the Feature Bytes (and the data provisioned in the FEATBYTE and FLEXDIAL tables) determine if the filed MSGCTR message for the SUBR collectable is ever utilized.

If the SUBR collectable (with the same FLEXTYPE as defined by FLEXDIAL_N00_FLEXTYPE) reads the filed MSGCTR message, the N00 digits and the features associated with it are processed. The features associated with the N00 digits are determined from the FLEXFEAT table.

Activation

Activation of identifying features associated with the N00 digits dialed involves the following items:

- The originating agent must be an AXXESS agent (for Feature Bytes parameter processing to be invoked).
- A SUBR collectable must process the filed MSGCTR message to access the features associated with the N00 dialed number.

N00 digits restrictions and limitations

The CLRFTRS (Clear Call Features) with MSGCTR option in FLEXDIAL, does not clear the filed MSGCTR message generated by this feature for the SUBR collectable.

The office parameter FLEXDIAL_N00_FLEXTYPE is defaulted to a value of N00NUM. A FLEXTYPE table tuple with a key of N00NUM is always provisioned.

Processing all routing number parameters

Up to three routing numbers are received and processed during the processing of the TCAP response to an SCP N00 query message. Routing numbers contain the ”real” called party number to use for call translations.

The first routing number received in the TCAP response message is processed as currently defined by the existing UCS DMS-250 switch capabilities. For each additional routing number parameter received, a filed MSGCTR message is generated and sent to the ADDR collectable. Thus, if the call returns to the collect information point, the next filed message for the ADDR collectable is processed and the additional routing number parameter is utilized.

To ensure that only one filed address is processed at a time by an ADDR collectable, the dominant flag is set in the last filed message for every ”set” of numbers filed in the MSGCTR. A set is defined as a routing number, a DNIS number, and a billing number (all of which are received in the TCAP response message). The dominant flag signals to the ADDR collectable that there are no more messages at the MSGCTR for that ADDR collectable.

Activation

The processing of up to three Routing Number parameters from the TCAP response message is based on the agent type. The originating agent must be an AXXESS agent to enable the processing of all three possible numbers.

For non-AXXESS agents, the existing UCS DMS-250 switch capability is used. (The existing capability processes only the first routing number from the TCAP Response message.)

Example

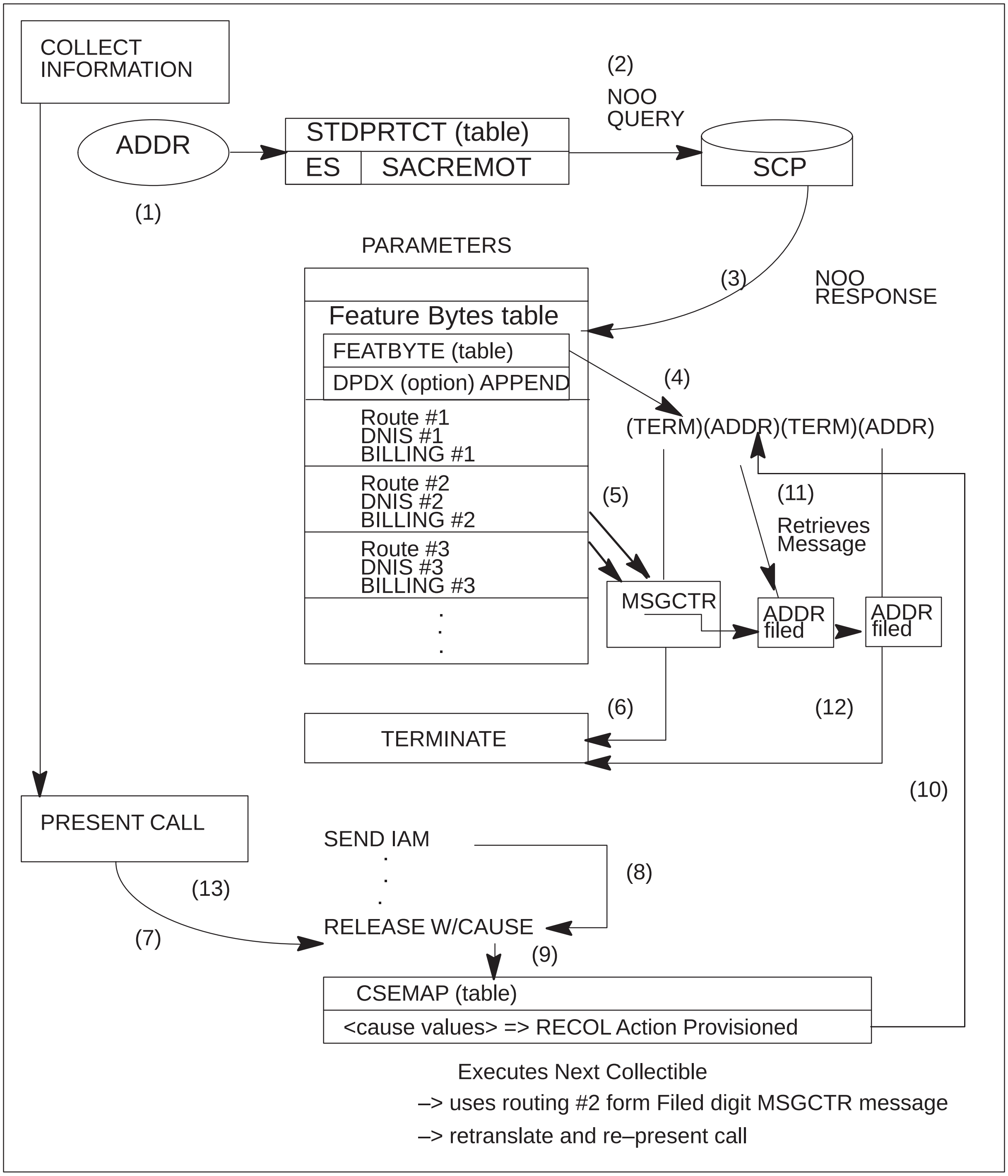
Figure 8-1 describes how multiple routing numbers are used by the TCAP response message in the AXXESS agent processing call scenario. The numbers in parentheses show each step taken in this process. Step 12 and step 13 correspond to step 6 and step 7, respectively, but the second routing number is used. The third routing number is processed in a similar way to the second routing number.

Restrictions and limitations

The following restrictions and limitations apply:

- Overflow routing numbers received in the TCAP response message are only utilized by subsequent ADDR or ADDRPARM digit collectables. (Overflow conditions exist when more than one routing number is received.)
- Call processing functionality depends on datafill in several tables. The tables include the FEATBYTE, FLEXDIAL, and CSEMAP tables.
- The CLRFTRS (Clear Call Features) with MSGCTR option in FLEXDIAL, does not clear the filed MSGCTR message generated by this feature for the ADDR collectable.

Figure 8–1
AXXESS agent processing call scenario



Call processing (all agents)

This section describes call processing features developed for all agents, including AXXESS originating agents.

Processing of DNIS parameters

DNIS numbers are received during the processing of the TCAP response to an SCP N00 query message. Up to three DNIS numbers can be received in the TCAP response message (one for each routing number received). DNIS numbers contain the destination party number. The DNIS number can be delivered according to the datafill in table RTEATTR.

The first DNIS number received in the TCAP response message is stored so that call processing can utilize the information when the call leaves the network. For AXXESS agents, each additional DNIS number parameter received results in a filed MSGCTR message generated and sent to the ADDR collectable (along with the associated routing number). Thus, if the call returns to the collect information point, the next filed message for the ADDR collectable is processed, and the additional DNIS number parameter is utilized.

To ensure that only one filed address is processed at a time by an ADDR collectable, the dominant flag is set in the last filed message for every ”set” of numbers filed in the MSGCTR. A set is defined as a routing number, and an optional DNIS and billing number (all of which are received in a TCAP response message). The dominant flag signals to the ADDR collectable that there are no more messages at the MSGCTR for that ADDR collectable.

Activation

This feature is activated only when the SOC option, ”N00R V2 TCAP DNIS Svs,” (option code N00R0003 is set to ON).

Processing of the first DNIS parameters is available for all agent types. Only AXXESS originating agents have the ability to process additional DNIS number parameters from the TCAP response message.

Restrictions and limitations

The following restrictions and limitations apply:

- Overflow DNIS numbers received in the TCAP response message are only utilized by subsequent ADDR or ADDRPARM digit collectables. (Overflow conditions exist when more than one DNIS number is received.)
- Call processing functionality depends on datafill in several tables. The tables include the FEATBYTE, FLEXDIAL, and CSEMAP tables.

- The CLRFTRS (Clear Call Features) with MSGCTR option in FLEXDIAL, does not clear the filed MSGCTR message generated by this feature for the ADDR collectable.

Processing of billing number parameters

Billing number digits are received and processed during the processing of the TCAP response to an SCP N00 query message. Up to three billing numbers can be received (one for each associated routing number received).

While all agents can process the first billing number parameter, only AXXESS originating agents can process each of the three possible billing numbers received in the TCAP response message.

Processing is performed as shown below:

- all originating agents

  The first Billing Number received in the TCAP response message is stored in the CDR. The office parameter, VER_2_BILL_FLEXTYPE, defines an index into the FLEXDIAL table. The BILLFLD option of the tuple identified by the VER_2_BILL_FLEXTYPE office parameter determines the field within the CDR where the billing number will be written.

- AXXESS originating agents only

  Each additional Billing Number parameter received results in a filed MSGCTR message generated and sent to the ADDR collectable (along with the associated routing number and DNIS number). Thus, if the call returns to the collect information point, the next filed message for the ADDR collectable is processed, and the additional Billing Number parameter is utilized.

  To ensure that only one filed address is processed at a time, by an ADDR collectable, the dominant flag is set in the last filed message for every ”set” of numbers filed in the MSGCTR. A set is defined as a routing number, and an optional DNIS and billing number (all of which are received in a TCAP response message). The dominant flag signals to the ADDR collectable that there are no more messages at the MSGCTR for that ADDR collectable.

Activation

Identifying the field in the CDR for writing the billing number digits, involves the provisioning of the VER_2_BILL_FLEXTYPE office parameter on the UCS DMS-250 switch.

Call processing utilizes additional options provisioned (not just the BILLFLD option) within the FLEXTYPE tuple identified by the VER_2_BILL_FLEXTYPE office parameter. The additional FLEXTYPE tuple options process are the following:

- billing field
- billing flags
- CallingParty
- CDR template
- Answer CDR (AXXESS trunks only)

The default value for the VER_2_BILL_FLEXTYPE office parameter is ANI.

Restrictions and limitations

The following restrictions and limitations apply:

- Overflow billing numbers received in the TCAP response message are only used by a subsequent ADDR or ADDRPARM digit collectable. (Overflow conditions exist when more than one billing number is received.)
- The CLRFTRS (Clear Call Features) with MSGCTR option in FLEXDIAL, does not clear the filed MSGCTR message generated by this feature for the ADDR collectable.
- The billing number is not utilized from the TCAP response message when the tuple identified by the VER_2_BILL_FLEXTYPE does not contain the BILLFLD option.
- Call processing functionality depends on datafill in several tables. The tables include the FEATBYTE, FLEXDIAL, and CSEMAP tables.

Office parameter provisioning

The office parameters for TCAP and FlexDial interactions are described in the Chapter 13, ”Office parameter provisioning.”

Logs

The FLEX 401 FEATBYTE access error log is described in section ”FlexDial logs and OMs” on page 27-23.

List of terms

ACCT	account code number
ADDR	address digits; generally 7 or 10 digits
ANM	Answer Message
AIN	advanced intelligent network
ANI	automatic number identification
AUTH	authorization code
AXXESS	trunk agent for FlexDial framework
CAIN	carrier AIN capability
CCB	call condense block
CCS7	Common Channel Signaling #7
CDR	call detail record

CLG
calling party address

CHG
charge number

CI
command interpreter

CIC
carrier identification code

CLLI
common language location identifier

DAL
direct access line trunk group type

DIRP
device independent recording package

DTC
digital trunk controller

DTMF
dual-tone multi-frequency

EANT
equal access network trunk (FGD)

FGA
Feature Group A trunk group type

FGB
Feature Group B trunk group type

FGC
Feature Group C trunk group type

FGD
Feature Group D trunk group type

FLEXCONV
FlexDial conversion tool to convert non-AXXESS trunks to AXXESS trunks

FLEXDIAL table

FlexDial Agent Interaction Definition table

FLEXFEAT table

FlexDial Subscriber Number and Call Type Features table

FLEXMOD table

FlexDial Digit Modification table

FLEXSIM

FlexDial simulator tool

FLEXSIMT table

FlexDial Simulator table

FLEXTYPE table

FlexDial Subscriber and Call Type Definitions table

FLEXVAL table

FlexDial Subscriber Number Validation table

FNAL

Feature Not Allowed treatment

GS

ground start

IAM

Initial Address Message

IMT

inter-machine trunk

IN

international

INTOA

international operator-assisted

IP

international partitioned

ISUP

integrated services user part

JIP

Jurisdiction Information Parameter

KP	signal designating beginning of string
LS	loop start
MB	make busy
MCCS	mechanized calling card service
MF	multi-frequency
MSGCTR table	FlexDial Message Center table
MTM	maintenance trunk module
N/A	nonapplicable
NOA	nature of address
NTRS	No Terminal Responding treatment
ONAL	off-network access line (FGA)
ONAT	off-network access trunk (FGB or FGC)
ONP	one night process
PANI	pseudo automatic number identification
PIN	personal identification number
PTS	per-trunk signaling

RLT

release link trunk

RMB

Remote Make Busy parameter

SOC

software optionality control

SNPA

serving numbering plan area

SNXX

serving numbering exchange

SSP

service switching point

ST

signal designating termination of string

STM

service trunk module

STP

service transfer point

STS

serving translation scheme

TCAP

Transaction Capabilities Application Part

TCN

travel card number

TRKCONV

tool to convert PTS trunks to SS7 trunks

TRKFEAT table

FlexDial Trunk Group Features table

TRKGRP table

Trunk Group table

TRKSGRP table

Trunk Subgroup table

TRKSIG table

FlexDial Trunk Group Signaling table

UA

universal access

UCS

Universal Carrier Services

UTR

universal tone receiver

VACT

Vacant Code treatment

XPM

extended multiprocessor system peripheral module

Ordering information

Use the following table for ordering Nortel Networks NTPs (Northern Telecom Publications) and Product Computing-Module Loads (PCLs):

Type of product	Source	Phone	Cost
Technical documents (paper or CD-ROM)	Nortel Networks Product Documentation	1-877-662-5669, Option 4 + 1	Yes
Individual NTPs (paper)	Merchandising Order Service	1-800-347-4850	Yes
Marketing documents	Sales and Marketing Information Center (SMIC)	1-800-4NORTEL (1-800-466-7835)	No
PCL software	Nortel Networks	Consult your Nortel sales representative	Yes

When ordering publications on CD

Please have the CD number and software version available, for example, **HLM-2621-001 02.02**.

When ordering individual paper documents

Please have the document number and name available, for example, **297-2621-001, UCS DMS-250 Master Index of Publications**.

When ordering software

Please have the eight-digit ordering code, for example, **UCSE0012**, as well as the ordering codes for the features you wish to purchase. Contact your Nortel Networks representative for assistance.

Digital Switching Systems

UCS DMS-250

FlexDial Framework Application Guide

Product Documentation–Dept 3423
Nortel Networks
P.O. Box 13010
RTP, NC 27709–3010
1-877-662-5669, Option 4 + 1

NORTEL NETWORKS CONFIDENTIAL: The information contained herein is the property of Nortel Networks and is strictly confidential. Except as expressly authorized in writing by Nortel Networks, the holder shall keep all information contained herein confidential, shall disclose the information only to its employees with a need to know, and shall protect the information, in whole or in part, from disclosure and dissemination to third parties with the same degree of care it uses to protect its own confidential information, but with no less than reasonable care. Except as expressly authorized in writing by Nortel Networks, the holder is granted no rights to use the information contained herein.

Information is subject to change without notice. Nortel Networks reserves the right to make changes in design or components as progress in engineering and manufacturing may warrant.

Publication number: 297-2621-390
Product release: UCS13
Document release: Preliminary 06.01
Date: March 2000
Printed in the United States of America

NORTEL NETWORKS™

How the world shares ideas.