



# User Exits

---

**Adabas Version 7.4.2**



This document applies to Adabas Version 7.4.2 and to all subsequent releases.

Specifications contained herein are subject to change and these changes will be reported in subsequent release notes or new editions.

© Copyright Software AG 1999-2003.  
All rights reserved.

The name Software AG and/or all Software AG product names are either trademarks or registered trademarks of Software AG. Other company and product names mentioned herein may be trademarks of their respective owners.

# Table of Contents

<b>About this Documentation</b>	1
About this Documentation	1
<b>Other Exits Supported by Adabas</b>	2
Other Exits Supported by Adabas	2
<b>User Exit 1 (General Processing)</b>	3
User Exit 1 (General Processing)	3
Input and Output Parameters	3
Command Queue (CQ) Layout	4
Command Queue Header DSECT	5
Command Queue Element (CQE) DSECT	7
<b>User Exit 2 (Dual Log Processing)</b>	10
User Exit 2 (Dual Log Processing)	10
User Exit 2 Calling Sequence	11
Input Parameters	11
Output Parameter	14
BS2000 Options for Invoking User Exit 2	14
Method 1	14
Method 2	14
<b>User Exit 3 (User-Defined Phonetization)</b>	15
User Exit 3 (User-Defined Phonetization)	15
Input Parameters	15
<b>User Exit 4 (User-Generated Log Data)</b>	16
User Exit 4 (User-Generated Log Data)	16
Command Log Format	16
PRILOG : Printing the Command Log	17
Print Program Input	18
Control Card Parameters	18
PRILOG6 Messages	19
Installing and Using PRILOG under VSE/ESA	19
Installing and Using PRILOG under OS/390 or z/OS	20
<b>User Exit 5 (Adabas Review Hub Event Handler)</b>	22
User Exit 5 (Adabas Review Hub Event Handler)	22
Input Parameters	22
Other Register Values at Entry	23
Output Parameters	23
<b>User Exit 6 (User Processing Before Data Compression)</b>	24
User Exit 6 (User Processing Before Data Compression)	24
Input Parameters	26
Output Parameters	27
<b>User Exit 8 (Operator Interface)</b>	28
User Exit 8 (Operator Interface)	28
Input Parameters	29
<b>User Exit 9 (ADAULD)</b>	30
User Exit 9 (ADAULD)	30
<b>User Exit 12 (Multiple Dataset Log Processing)</b>	32
User Exit 12 (Multiple Dataset Log Processing)	32
User Exit 12 Calling Sequence	33
User Exit Interface	34

Parameters . . . . .	34
DSECT of the EX12PARM Parameter Block . . . . .	34
Output Parameter . . . . .	35
Activating the Sample User Exit . . . . .	35
OS/390 or z/OS . . . . .	36
VSE/ESA . . . . .	36
BS2000 . . . . .	37
Method 1 . . . . .	38
Method 2 . . . . .	38
<b>Hyperdescriptor Exits 01 - 31</b> . . . . .	<b>40</b>
Hyperdescriptor Exits 01 - 31 . . . . .	40
Main Parameter Area . . . . .	40
INPUT Parameter Area (Pointed to by First Parameter Address) . . . . .	41
Header Element Fields . . . . .	41
Parent Element Fields . . . . .	41
OUTPUT Parameter Area . . . . .	42
Header Element . . . . .	43
Value Elements . . . . .	43
Null Value Option . . . . .	44
<b>Collation Descriptor Exits 01 - 08</b> . . . . .	<b>45</b>
Collation Descriptor Exits 01 - 08 . . . . .	45
Collation Descriptor Exit Interface . . . . .	45
Initialization Parameters . . . . .	46
Encoding/Decoding Parameters . . . . .	46

# About this Documentation

*User Exits and Hyperexits* refers to the user exits activated by the ADARUN parameters UEXn, HEXnn, and CDXnn (see the *Adabas Operations* documentation for descriptions of the ADARUN parameters).

The user exits documented in this document are as follows:

User Exit	ADARUN	Use
User Exit 1	UEX1	Command processing (Adabas nucleus)
User Exit 2	UEX2	Dual log processing
User Exit 3	UEX3	User-defined phonetization
User Exit 4	UEX4	User-generated log data
User Exit 5	UEX5	Adabas Review hub event handler
User Exit 6	UEX6	Data compression (ADACMP)
User Exit 8	UEX8	Operator interface
User Exit 9	UEX9	Data unload (ADAULD)
User Exit 12	UEX12	Multiple log processing

Hyperexit	ADARUN	Use
1 . . 31	HEX01 . . HEX31	User-supplied algorithm to create hyperindex values . . .

Collating Exit	ADARUN	Use
1 . . 8	CDX01 . . CDX08	User-supplied algorithm to encode and decode values for the corresponding collation descriptors ..

# Other Exits Supported by Adabas

Other user exits supported by Adabas include the following:

Entry Name	Use
ADACDCUX	Allows you to obtain control at strategic points during ADACDC utility processing. See the <i>Adabas Utilities</i> documentation.
ADACSHUX	Allows you to obtain control at strategic points during Adabas Caching Facility processing. See the <i>Adabas Caching Facility</i> documentation.
ADALERTX	Allows you to examine heartbeat records and/or change the heartbeat thresholds and interval. See <i>Adabas Online System Demo Version</i> in the DBA Tasks documentation.
ADASMXIT	Allows you to supply parameters to a PIN routine or examine a condition when it is encountered before the PIN routine is invoked so that recovery actions other than those provided by Adabas can be implemented. See <i>Adabas Online System Demo Version</i> in the DBA Tasks documentation..
DSFEX1	Automatically submits the necessary job to prevent overflow of the DLOG area. See the <i>Adabas Delta Save Facility</i> documentation.
UEXITA	Linked with Adalink: receives control after a command is processed by a target, the router, or Adalink itself. See the <i>Adabas Installation</i> documentation .
UEXITB	Linked with Adalink: receives control before a command is passed to a target with the router 04 call. See the <i>Adabas Installation</i> documentation .
UEXRAI	Allows you to change automatically generated ADARAI RECOVER JCL before it is written to DDJCLOUT. See the <i>Adabas Utilities</i> documentation.

# User Exit 1 (General Processing)

This user exit is given control by Adabas immediately after a command is received by the Adabas nucleus. The command itself has yet to be processed except for the determination of the type of command (simple access, complex access, update).

One of the most common applications of this user exit is to insert a security password and/or a cipher code into the Adabas control block.

The call to the user exit is made using a standard BALR 14,15 assembler instruction. Register 1 contains the address of a parameter list. All registers must be saved when control is received and restored immediately prior to returning control to Adabas.

## Notes:

1. User exit 1 must return the same AMODE value to the calling program that was active when user exit 1 was called.
2. The file number specified in the Adabas control block cannot be changed. If it is necessary to change a file number with the user exit, change the field CQEFNR in the command queue element (CQE), offset X'A6' (see the figure in *Input and Output Parameters* and the example in *Command Queue Header DSECT*).
3. The command code field in the Adabas control block cannot be changed; a response code is returned if you attempt to do so.
4. The length of an Adabas buffer in the Adabas control block cannot be changed.

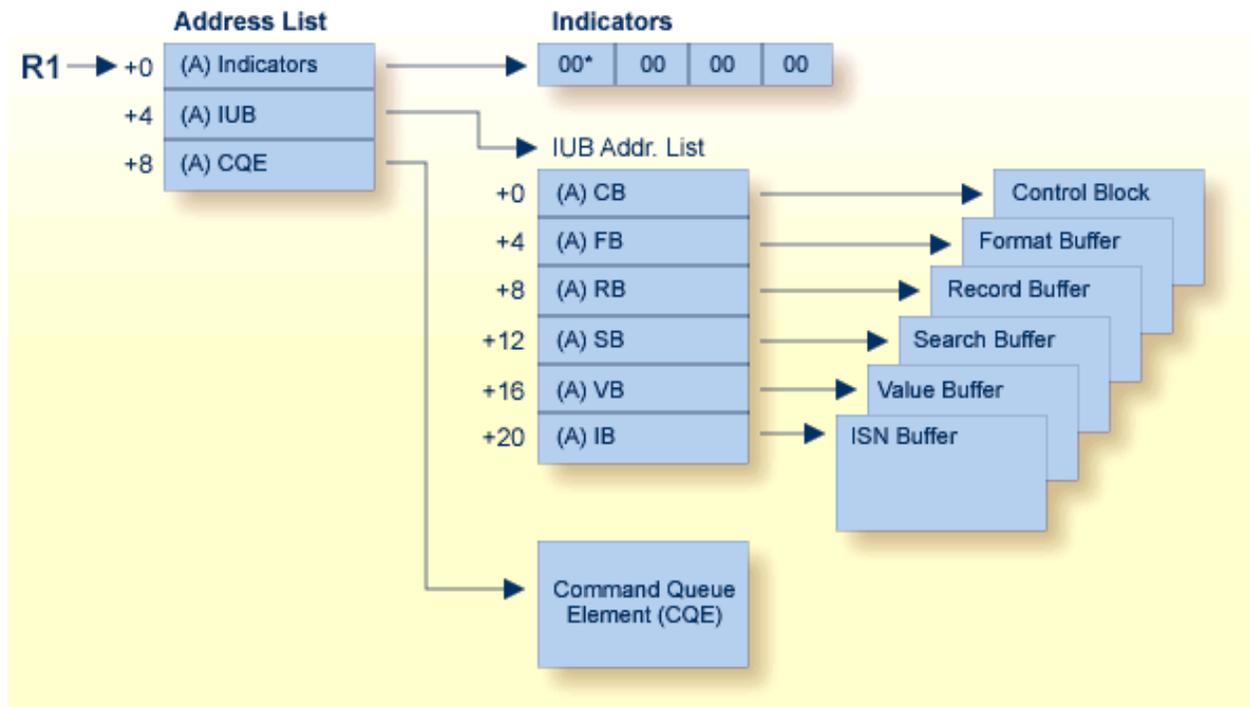
If a buffer is not supplied by the caller, the associated address field in the IUB address list is set to zero. Depending on the command type, if a buffer is not needed the corresponding address is disregarded. Do not change important buffer addresses.

This chapter covers the following topics:

- Input and Output Parameters
- Command Queue (CQ) Layout
- Command Queue Header DSECT
- Command Queue Element (CQE) DSECT

---

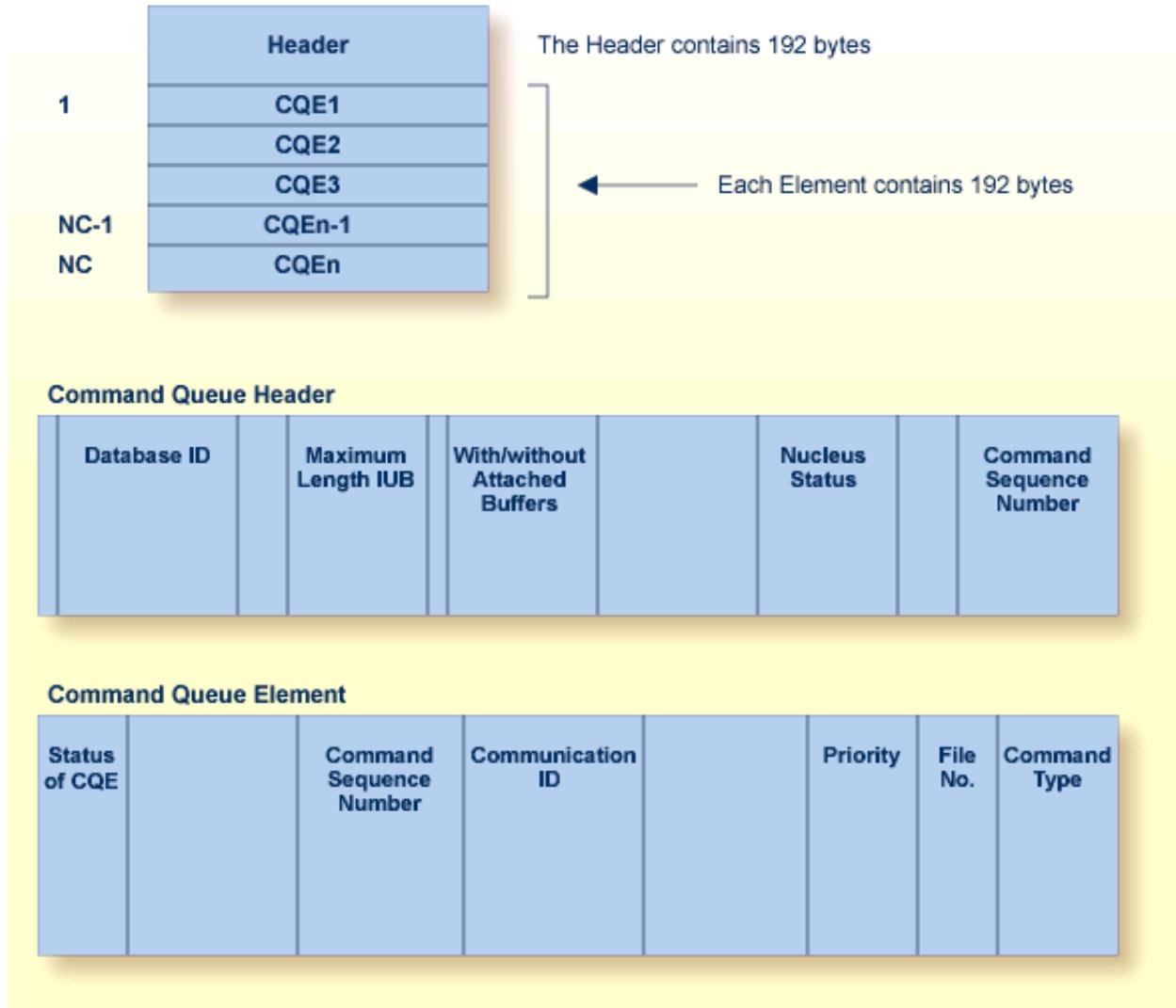
## Input and Output Parameters



### General Processing User Exit (1) Parameters

\* *Indicator:* Before calling user exit 1, the fullword indicator area are set to zero. As output from the user exit 1 call, 00 in the first Indicator byte means that the command can be executed; a non-zero value in the first byte means that the command should not be executed, and returns response code 22.

### Command Queue (CQ) Layout



User Exit 1 Command Queue

Command Queue Header DSECT

```

000000    CQH      DSECT          COMMAND QUEUE HEADER
000000    CQHAI DTE DS      A          ADDR(IDTE) (SET BY 0-CALL)
          * THE FOLLOWING 2 LABELS MUST REMAIN CONTIGUOUS IN THE SAME ORDER
000004    CQHPI D   DS      H          PHYSICAL ID (SET BY SERVICE)
000006    CQHFL AG DS      X          FLAGS
          CQHFTRAN EQU X'80'        TRANSLATOR (SET BY SERVICE)
          CQHFCOMM EQU X'40'        COMMUNICATOR (SET BY SERVICE)
          CQHFND BT EQU X'20'        NOT DATABASE TARGET (SET BY
          *                          SERVICE)
          CQHFISO EQU X'10'        ISOLATED ID (SET BY SERVICE)
          CQHFABR EQU X'08'        ATTACHED BUFFERS REQUIRED (SET
          *                          BY SERVICE)
          CQHFORCE EQU X'04'        FORCE NEW IDTE (SET BY MPM)
          CQHFANCH EQU X'02'        ANCHOR SERVICE (SET BY SERVICE)
000007    CQHFLA G1 DS      X          FLAGS
          CQH1NMC EQU X'80'        NO MORE COMMANDS (SET BY
          *                          SERVICE)
    
```

```

CQH1UPG EQU X'40'          USER PROBABLY GONE (SET BY
*                          8/12/20-CALLS)
* 8/12-CALL PARMS (THE FOLLOWING 5 LABELS MUST REMAIN CONTIGUOUS IN
* THE SAME ORDER)
000008 CQHPARMS DS 0A      PARMS (SET BY MPM)
000008 CQHPARM1 DS 0A     PARM 1
000008 CQHAIUB DS A      ADDR(CURRENT IUB)
00000C CQHPARM2 DS 0A     PARM 2
00000C CQHACQE DS A      ADDR(CURRENT CQE)
* MPM PARMS
000010 CQHNCQES DS F      NUM(CQES) (SET BY MPM)
000014 CQHIUBL DS F      MAX LEN(IUB) (SET BY MPM)
* SERVICE POST PARMS (THE FOLLOWING 4 LABELS MUST REMAIN CONTIGUOUS
* IN THE SAME ORDER)
000018 CQHAECB DS A      ADDR(SERVICE ECB) (SET BY
* SERVICE)
00001C CQHTSN DS 0F      ADDR(SERVICE TSN) (BS2000) (SET
* BY 0-CALL)
00001C CQHTID DS 0F      SERVICE TID (DOS) (SET BY 0-CALL)
00001C CQHAASCB DS A     ADDR(SERVICE ASCB) (MVS) (SET BY
* 0-CALL)
* ATTACHED BUFFERS
000020 CQHAABST DS A     ADDR(ATTACHED BUFFER SEGMENT
* TABLE (SET BY MPM)
000024 CQHLABST DS F     LEN(ATTACHED BUFFER SEGMENT
* TABLE (SET BY MPM)
000028 CQHAABA DS A     ADDR(ATTACHED BUFFER AREA) (SET
* BY MPM)
00002C CQHLABP DS F     LEN(ATTACHED BUFFER PREFIX) (SET
* BY SERVICE)
000030 CQHABHWM DS F     ATTACHED BUFFER SEGMENT TABLE
* HIGH WATER MARK
CQHABSTS EQU 4          ATTACHED BUFFER SEGMENT TABLE
* SHIFT
CQHABAS EQU 8           ATTACHED BUFFER AREA SHIFT
CQHLABS EQU 256         LEN(ATTACHED BUFFER SEGMENT)
CQHABSEG EQU X'EE'     ATTACHED BUFFER SEGMENT TABLE
* SEGMENT ALLOCATED BYTE(MUST
* BE <0)
CQHABBAR EQU X'77'     ATTACHED BUFFER SEGMENT TABLE
* BARRIER (MUST BE >=0)
000034 CQHAECQ DS A     ADDR(END OF CQ) (SET BY MPM)
* SYSTEM DEPENDENT OVERLAY AREA
000038 CQHOVLAY DS XL105
* DOS
0000A1          ORG CQHOVLAY
000038 CQHACOMR DS A     ADDR(SERVICE COMREG) (SET BY
* 0-CALL)
* VS1
00003C          ORG CQHOVLAY
000038 CQHAJTCB DS A     ADDR(SERVICE JOB STEP TCB) (SET
* BY 0-CALL)
* MVS
00003C          ORG CQHOVLAY
* THE FOLLOWING 3 LABELS MUST REMAIN CONTIGUOUS IN THE SAME ORDER
000038 CQHLXL DS 0F      LX LIST
000038 CQHLXLN DS F'1'   NUM(LXS)
00003C CQHLXLLX DS F     LX VALUE
000040 CQHET DS XL44     ENTRY TABLE (LENGTH MUST BE AT
* LEAST ETDLEN+2*ETDELEN)
00006C CQHTSECB DS F     TRANSWAP ECB
000070 CQHSAVE2 DS F     R2 SAVE AREA

```

```

* THE FOLLOWING 2 LABELS MUST REMAIN CONTIGUOUS IN THE SAME ORDER
000074 CQHSAVEC DS F RC SAVE AREA
000078 CQHSAVED DS F RD SAVE AREA
00007C CQHLXRES DS 2F LXRES PARM LIST
000084 CQHETCON DS 3F ETCON PARM LIST
* THE FOLLOWING 2 LABELS MUST REMAIN CONTIGUOUS IN THE SAME ORDER
000090 CQHCPUID DS CL8 CPU ID (SET BY 0-CALL)
000098 CQHVMID DS CL8 VM ID (SET BY 0-CALL)
0000A0 CQHKEY DS X SERVICE PSW-KEY (SET BY 0-CALL)
*
0000A1 ORG
0000A1 DS XL19 RESERVED
*
0000B4 CQHANCHR DS F ANCHOR (CQHFANCH SET) (SET BY
* SERVICE)
*
* THE FOLLOWING 2 WORDS MUST BE THE LAST 2 WORDS IN THE CQH AND
* REMAIN CONTIGUOUS IN THE SAME ORDER
0000B8 CQHWI DS F SERVICE WAIT INDICATOR
0000BC CQHSEQNR DS F COMMAND SEQUENCE NUMBER (SET BY
* 4-CALL)
CQHLEN EQU *-CQH CQH LENGTH (MUST BE GE CQELEN)
CQHCQE EQU * FIRST CQE
    
```

## Command Queue Element (CQE) DSECT

and Queue Element (CQE) DSECT

```

000000 CQE DSECT
000000 CQECNTL DS X TS CONTROL BYTE
000001 CQEUBF DS X UB FLAGS (UBFLAG)
*
000002 CQEBIN DS X BUFFERS IN (UBBIN)
000003 CQEBOUO DS X BUFFERS OUT (UBBOUO)
000004 CQEID DS H ID (UBID) (UBFPID SET IN
* CQEUBF IF PHYSICAL)
*
000006 CQEFLAG DS X G E N E R A L P U R P O S E F L A G S
CQEFBUF EQU X'80' USER BUFFERS IN SERVICE
* PARTITION, REGION, ADDR
* SPACE (MUTUALLY EXCLUSIVE
* WITH CQEFAB)
CQEFETW EQU X'40' ET COMMAND WAITING FOR 12-CALL
CQEFW16 EQU X'20' WAITING FOR 16-CALL (MUTUALLY
* EXCLUSIVE WITH CQEF16R)
CQEF16R EQU X'10' 16-CALL REQUIRED (MUTUALLY
* EXCLUSIVE WITH CQEFW16)
CQEFAB EQU X'08' ATTACHED BUFFER (MUTUALLY
* EXCLUSIVE WITH CQEFBUF)
CQEFABR EQU X'04' ATTACHED BUFFER REQUIRED
* (MUTUALLY EXCLUSIVE WITH
* CQEFBUF)
CQECML EQU X'02' CML LOCK HELD (MVS)
CQEHELD EQU CQEFLAG
*
000007 CQESFLAG DS X S E L E C T I O N F L A G S
CQSFPROC EQU X'80' IN PROCESS
CQSFRDYS EQU X'40' READY TO BE SELECTED
CQSFSUQD EQU X'20' SEARCH FOR UQE DONE
CQSFUQEF EQU X'10' UQE FOUND
CQSFA9 EQU X'08' NOT SELECTABLE DURING
    
```

```

*                               BSS=X'80' STATUS
*
CQS FETSY EQU X'04'           NOT SELECTABLE DURING ET-SYNC
CQS FWSPC EQU X'02'           WAITING FOR SPACE
CQS FWISN EQU X'01'           WAITING FOR ISN IN HQ
*
*                               P O S T
000008    CQEAECB DS A          ADDR(USER ECB) (UBECB/UBAECB)
00000C    CQETID DS 0F         USER TID (VSE)
00000C    CQETSN DS 0F         USER TSN (BS2000)
00000C    CQEAASCB DS A        ADDR(USER ASCB) (MVS)
*
*                               U S E R   I N T E R F A C E
000010    CQEAAB DS A          ADDR(ATTACHED BUFFER) (CQEFAB
*                               SET)
000014    CQEA FABS DS A        ADDR(FIRST ATTACHED BUFFER
*                               SEGMENT TABLE BYTE) (CQEFAB
*                               SET)
000018    CQENABS DS F         NUM(ATTACHED BUFFER SEGMENT
*                               TABLE BYTES) (CQEFAB SET)
00001C    CQEAUI DS A          ADDR(USER INFO) (UBAUIFO IF
*                               UBFINUB SET)
000020    CQEJNAME DS CL8      USER JOB NAME (UBJNAME IF
*                               UBFINUB SET)
000028    CQECKSUM DS D        CHECKSUM (TOD-CLOCK)
000030    CQESEQNR DS F        COMMAND SEQUENCE NUMBER
*                               (CQHSEQNR)
000034    CQE LSB DS F         LEN(REQUIRED IUB) (UBL SB)
000038    CQEAUPL DS A         ADDR(USER PARAMETER LIST)
*                               (UBAUPL)
00003C    CQEAUB DS A          ADDR(UB)
*                               W O R K A R E A   FOR SVC AND PC
*                               ROUTINES (LENGTH MUST BE AT
*                               LEAST MAX(UBLEN,64))
000040    CQEWORK DS XL64
000080    ORG CQEWORK
000040    CQESAVE DS 16F        SAVE AREA
000080    ORG CQEWORK
000040    CQEUB DS XL64        UB (MVS)
000080    ORG CQEWORK
000040    CQEAPL DS 5A         APL WITHOUT ADDR(ACB) (MVS)
000054    ORG
*
*                               U S E R   I D   (COMMUNICATION)
000080    CQEUSID DS 0XL28     USER ID (UBAUID IF UBFINUB SET)
000080    CQECPUID DS D        CPU ID (IDCPUID) (THE USER WILL
*                               BE IDENTIFIED EXTERNALLY BY THE
*                               CONTENTS OF CQEJNAME IF THE
*                               LOW-ORDER BIT OF CQECPUID IS 0,
*                               AND BY THE CONTENTS OF CQEUID
*                               IF THE BIT IS 1)
000088    CQEV MID DS CL8      VM ID (IDVMID)
000090    CQEOSID DS F         OPSYS ID
000094    CQEUID DS XL8        ID (UBUID/ACBADD2)
*
00009C    CQERBIBL DS 0F       ACTUAL RB/IB LENS
00009C    CQERBL DS H          ACTUAL RB LEN
00009E    CQEIBL DS H          ACTUAL IB LEN
0000A0    CQEPASID DS H        CALLER'S PASID (MVS)
0000A2    CQE16CTL DS X        16-CALL TS CONTROL BYTE (CQEW16
*                               SET)
0000A3    CQEKEY DS X         CALLER'S PSW-KEY

```

```

0000A4  CQEPRTY  DS    X          PRIORITY OF USER (UBPRTY IF
*                               UBFINUB SET)
0000A5  CQECMDT  DS    X          COMMAND TYPE (UBCMDT)
*                               EQU    X'20'    RESERVED
*                               EQU    X'10'    RESERVED
CQETET  EQU    X'08'    ET COMMAND
CQETUC  EQU    X'04'    UPDATE COMMAND
CQETCC  EQU    X'02'    COMPLEX COMMAND
CQETSC  EQU    X'01'    SIMPLE COMMAND
0000A6  CQEFNR   DS    H          FILNR (ACBFNR)
*
* TARGET AND SYSTEM DEPENDENT OVERLAY AREA, WHICH CAN BE USED AS
* FOLLOWS:
* - BY THE ROUTER 04-CALL PRIOR TO SETTING 'CQSFRDYS' IN
*   'CQESFLAG';
* - BY THE TARGET FROM THE TIME 'CQSFRDYS' IS SET UNTIL EITHER A
*   ROUTER 12-CALL FOR WHICH NO 16-CALL IS REQUIRED ('CQEF16R' IN
*   'CQEFLAG' IS NOT SET) OR UNTIL THE ROUTER 16-CALL OR COMMAND
*   TIMEOUT OCCURS ('CQE16CTL' CONTAINS X'FF' INSTEAD OF X'00');
* - BY THE ROUTER 16-CALL FROM THE TIME IS ACQUIRES THE CQE
*   (SUCCESSFULLY SETS 'CQE16CTL' TO X'FF') UNTIL IT FREES THE
*   CQE (SETS 'CQECNTL' TO X'00').
0000A8  CQEOVLAY DS    XL20
*
0000BC  ORG     CQEOVLAY          M V S
* THE FOLLOWING 2 LABELS MUST REMAIN CONTIGUOUS IN THE SAME ORDER
0000A8  CQESAVED DS    A          RD SAVE AREA
0000AC  CQESAVEE DS    A          RE SAVE AREA
* THE FOLLOWING 3 LABELS MUST REMAIN CONTIGUOUS IN THE SAME ORDER
0000B0  CQESAVE3 DS    0F        R3 SAVE AREA
*
0000B0  CQEARB   DS    A          ADDR(RB)
0000B4  CQEAIIB  DS    A          ADDR(IB)
0000B8  ORG     CQEOVLAY          N U C L E U S
0000A8  CQEAUQE  DS    A          ADDR(UQE)
0000AC  CQEACA  DS    A          ADDR(WORK-AREA) FOR CONTINUATION
0000B0  CQERQST DS    F          REQUESTED RESOURCE (SPACE/FNR/
*                               ISN)
0000B4  CQETHROB DS    F          TIME OF LAST THROWBACK
0000B8  CQECTHR  DS    0F        CMD COUNT OF LAST THROWBACK
0000B8  CQEAIUB  DS    0A        ADDR(IUB)
0000B8  CQET12C DS    F          TIME OF MPM 12-CALL
0000BC  ORG
0000BC  CQECMD  DS    XL2      COMMAND
*
0000BE  CQEX     DS    H          CQE INDEX
CQELEN  EQU    *-CQE      CQE LENGTH
*
*
AQUQE   EQU    CQEAUQE-CQE
    
```

## User Exit 2 (Dual Log Processing)

This user exit is given control by the Adabas nucleus during a switch from one dual log to the alternate dual log for the purpose of copying the log before it is reused by Adabas. This switch occurs only if dual data protection logging and/or dual command logging is in effect for the session.

The user exit routine must invoke a procedure whereby the appropriate function of the ADARES utility (CLCOPY or PLCOPY) is executed.

User exit 2 is invoked

- during Adabas nucleus startup if a PLOG/CLOG has to be copied;
- whenever a dual command or dual protection log switch occurs between two log datasets;
- during Adabas nucleus shutdown.

The user exit is provided with information about the status of the dual log datasets.

The user exit can decide which action is to be taken:

- Ignore the call;
- Submit a job to copy the log dataset just filled up (ADARES utility);
- Wait for completion of the copy job just submitted.

An example of user exit 2 is supplied with the Adabas installation procedure. Refer to the *Adabas Installation* documentation for more information.

The call to the user exit is made using a standard BALR 14,15 Assembler instruction. All registers must be saved when control is received and restored immediately prior to returning control to Adabas. Register 15 contains an action code as described in *Output Parameter*.

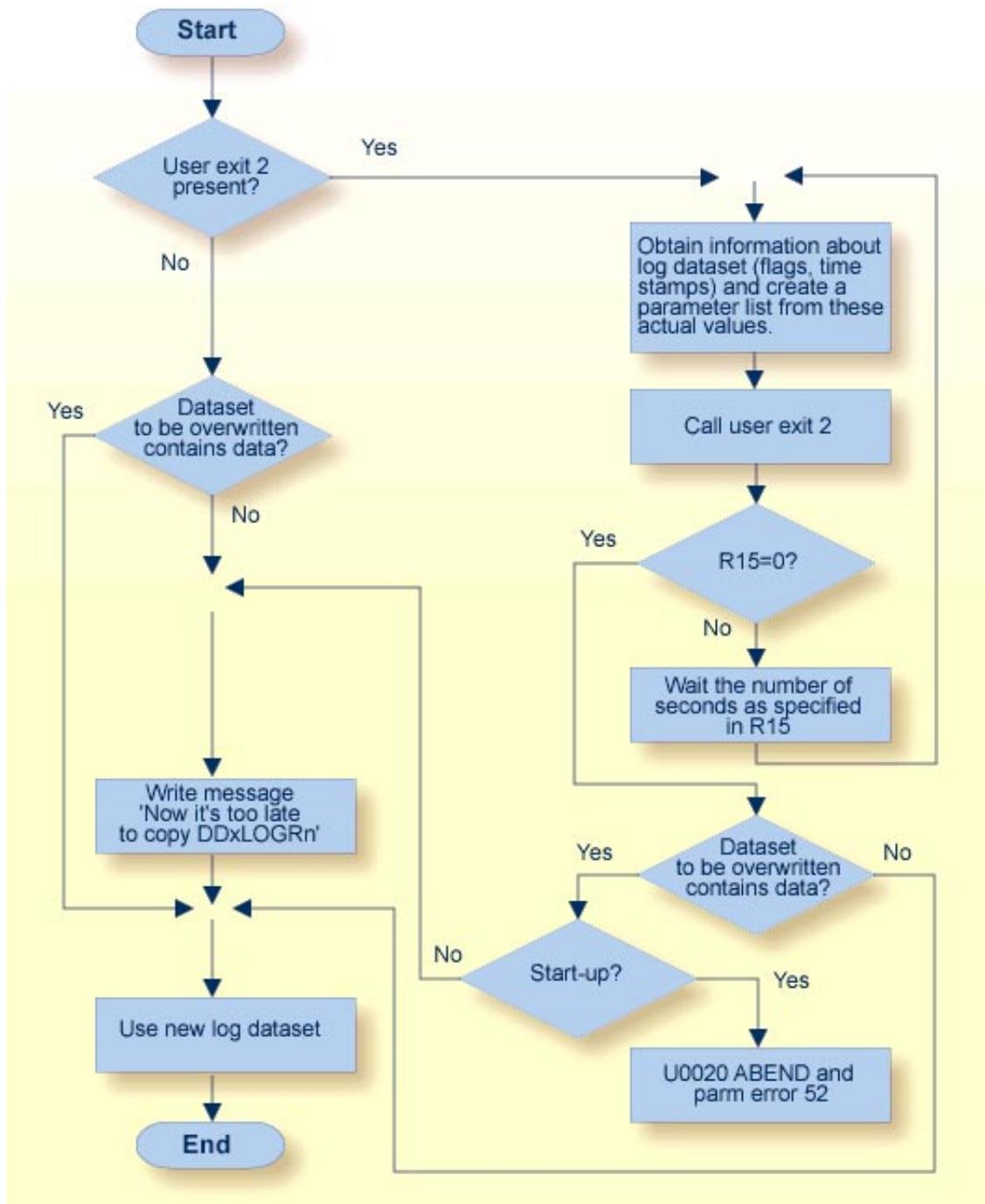
**Note:**

User exit 2 must return the same AMODE value to the calling program that was active when user exit 2 was called.

This chapter covers the following topics:

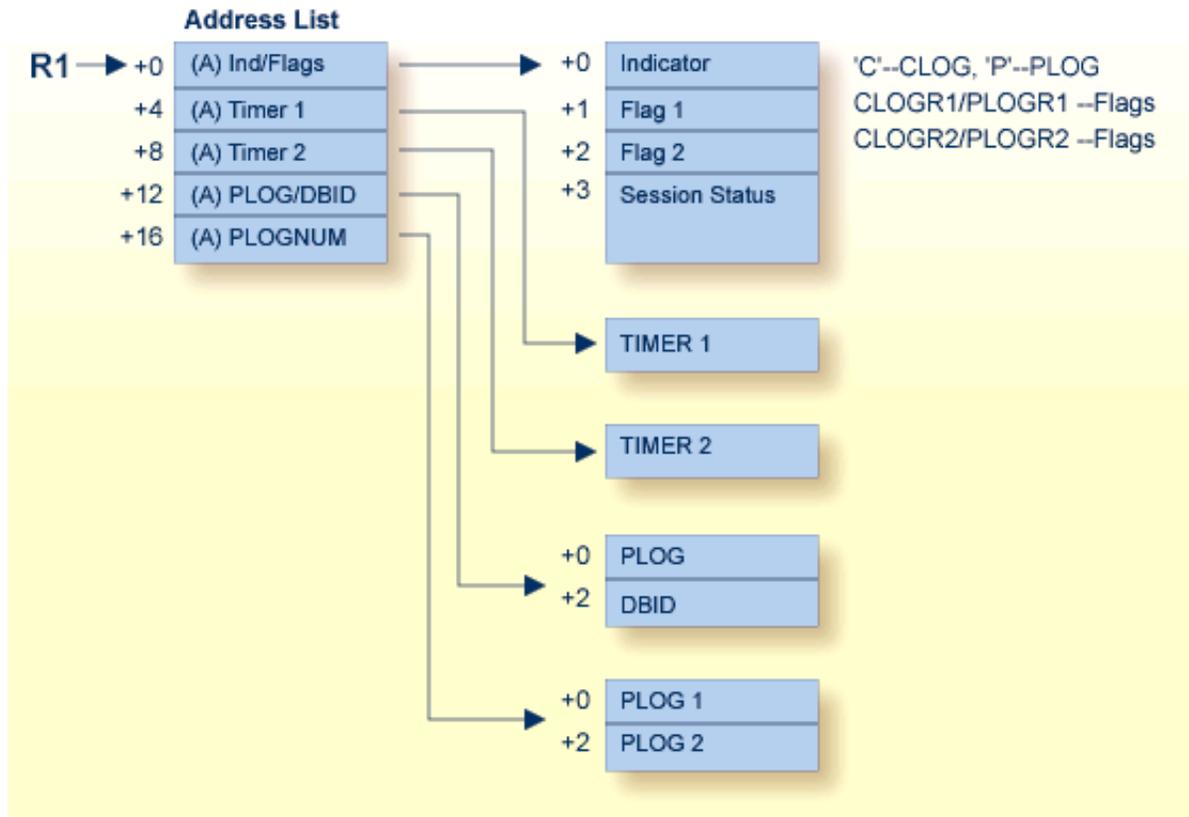
- User Exit 2 Calling Sequence
  - Input Parameters
  - Output Parameter
  - BS2000 Options for Invoking User Exit 2
-

# User Exit 2 Calling Sequence



Dual Log Processing Flow

## Input Parameters



### Dual Log Processing User Exit (2) Parameters

The input parameters for the address list are as follows:

Parameter	A fullword address of . . .
0 (R1)	the C/PLOG indicators and flag 1/2.
4 (R1)	the four-byte timer 1 field.
8 (R1)	the four-byte timer 2 field.
12 (R1)	the current session's PLOG number, followed by the database ID.
16 (R1)	a four-byte area where the first two bytes contain the number of PLOG1, and the second two bytes hold the number of PLOG2.

Other input parameters are explained in the following table:

Parameter	Usage
Flag 1	Status flags for DD/PLOGR1 and DD/CLOGR1; and

Parameter	Usage
Flag 2	<p>Status flags for DD/PLOGR2 and DD/CLOGR2:</p> <p>B'1... ....' : Dataset being written by nucleus</p> <p>B'.1.. ....' : Dataset has been completed by nucleus</p> <p>B'.11. ....' : Being copied by ADARES</p> <p>B'0000 0000' : Dataset is empty (or copied) and reusable for the nucleus.</p> <p>All other flag 1/2 field values are reserved. For DD/CLOGR1/2 only: X'08' for CLOGLAYOUT=5. Flag 1/2 bit settings can be combined (X'40' and X'20' as X'60', for example).</p> <p>If OPENOUT is specified, these flags are set after OPEN is issued for the output dataset; otherwise, the flags are set before the OPEN is issued.</p>
Session Status	<p>Contains information about the status of the nucleus when the exit was called:</p> <p>X'S'        Called during nucleus session start-up.</p> <p>X'T'        Called while terminating the nucleus session.</p> <p>X'W'        Called following a dual protection log switch.</p>
TIMERn	Time-stamp (highest four bytes of a STCK instruction) for the time the first block of the log dataset has been written. TIMER1 for DD/PLOGR1 and DD/CLOGR1, and TIMER2 for DD/PLOGR2 and DD/CLOGR2
PLOG	Current session protection log number (two bytes). This value is set for PLOG only; the field contains X'00' for CLOG.
DBID	Database ID (two bytes).
PLOG1/2	Two two-byte PLOG numbers found on PLOG 1 and PLOG 2. If the previous nucleus session ended abnormally, these four bytes contain that session's PLOGNUM value, which can be used in the initial user exit 2 call to copy that session's PLOG. During any subsequent session, these bytes contain the current PLOGNUM value. If the preceding session ends abnormally, these four bytes contain the ended session's PLOG numbers during the nucleus start phase. This PLOG information is needed during the start phase to assign the correct PLOG numbers to the PLOG areas to be copied. During subsequent exit calls, the current PLOG values are in these fields.

## Output Parameter

Parameter	Usage
R15 = 0	Nucleus continues processing.
R15 > 0	R15 is treated as the number of seconds to wait before calling user exit 2 again. During this time, the nucleus is in a "hard" wait. No commands are processed during the wait.

## BS2000 Options for Invoking User Exit 2

When using user exit 2 with BS2000 systems, the name of the ADARES CLCOPY or PLCOPY job is no longer "hard-coded" in the user exit as in previous releases. Therefore, there are now two ways of specifying the /ENTER job:

### Method 1

Create a job variable containing the complete "enter job" command. For example:

```
/DCLJV   ENTER.ADARES.PLCOPY, LINK=*DDJBPLC
/SETJV   *DDJBxLC, C'ENTER JOB. ENTER.ADARES.PLCOPY, ST=IMM'
```

Note that all operands of the /ENTER command may be used. The presence of the /DCLJV statement overrides the other possibilities. If necessary, the content of the job variables can be changed during a nucleus session.

Use the link names "\*DDJBPLC" for assigning the ADARES PLCOPY job, and "\*DDJBCLC" for assigning the ADARES CLCOPY job.

### Method 2

Omit the JV specification. The user exit 2 will then issue the following command:

```
'ENTER   RES.E.xLCO'
```

#### Note:

For BS2000 systems, user exit 2 is delivered as a source element only.

## User Exit 3 (User-Defined Phonetization)

This user exit may be used to perform user-defined phonetization. It is given control by the ADACMP utility or the Adabas nucleus whenever phonetic processing is required.

The user exit must develop a three-byte phonetic key using the value supplied. The address of the resulting phonetic key must be placed at 8(R1) before control is returned.

### Note:

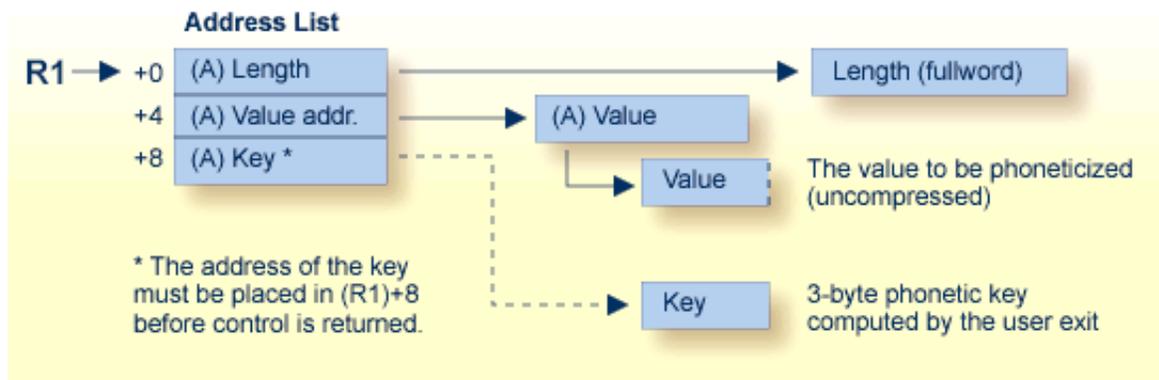
User exit 3 must return the same AMODE value to the calling program that was active when user exit 3 was called.

This chapter covers the following topics:

- Input Parameters

## Input Parameters

Register 1 contains the address of the following parameter list:



### User-Defined Phonetization User Exit (3) Parameters

Parameter	A fullword address of . . .
0(R1)	the four-byte length for the value to be phonetized.
4(R1)	the address of the value to be phonetized.
8(R1)	a three-byte location to contain the phonetic key. This address is set to zero before the user exit and must be set to the actual address during the user exit.

The call to the user exit is made using a standard BALR 14,15 assembler instruction. All registers must be saved when control is received and restored immediately prior to returning control to Adabas. The content of R15 is ignored.

# User Exit 4 (User-Generated Log Data)

User exit 4 is called immediately before an Adabas command log record is to be written. It may be used to generate any required user log data (SMF records) special statistics, or to suppress writing a log record.

This chapter covers the following topics:

- Command Log Format
  - PRILOG : Printing the Command Log
- 

## Command Log Format

Adabas supports two different command log formats. The ADARUN CLOGLAYOUT parameter determines which format is used:

- CLOGLAYOUT=4 (the default) specifies the older format. This format has been in use since Adabas version 4.

**Note:**

Some Adabas releases using this older format are no longer supported by Software AG. Future releases of Adabas will no longer support this older format.

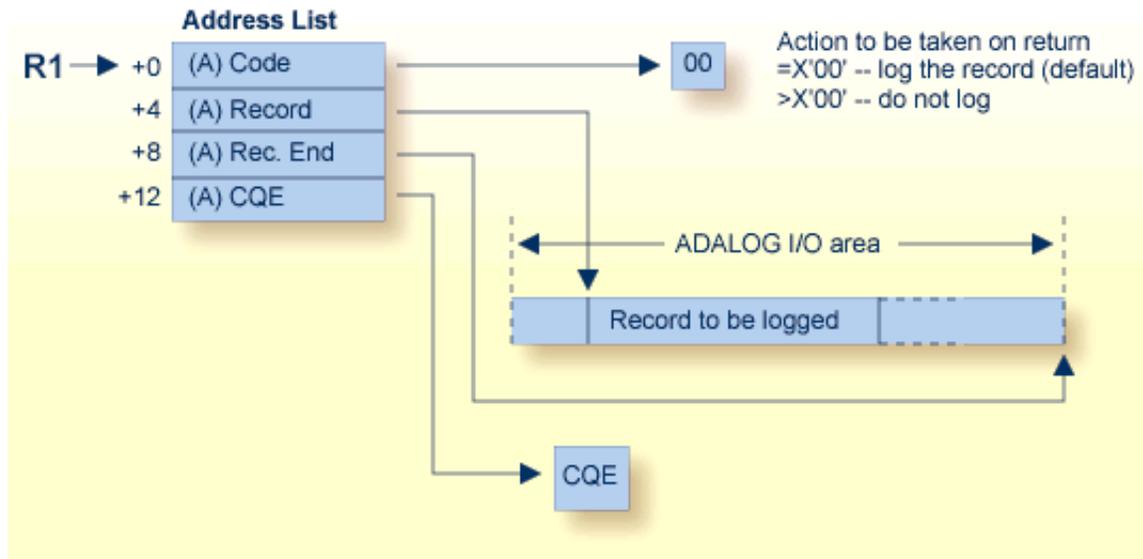
- CLOGLAYOUT=5 specifies the new format, which is supported only in Adabas versions 5.2 and above.

Both formats are displayed in *Command Block Formats*.

Ensure that your user exit and command log evaluation programs recognize the format in use before switching to it.

**Note:**

User exit 4 must return the same AMODE value to the calling program that was active when user exit 4 was called.



### User-Generated Log Data User Exit (4) Parameters

Parameter	Address of . . .
0(R1)	a byte containing a logging action code. This byte contains 00 (to log the record) upon each call. If changed to a nonzero value, this record will not be written to DDLOG.
4(R1)	the record to be logged. This address is zero if the exit is called at the end of the nucleus session.
8(R1)	the end of the Adabas I/O area. This address is zero if the exit is called at the end of the nucleus session.
12(R1)	the command queue element (CQE). This address is zero if the exit is called at the end of the nucleus session.

The record to be logged may be modified by the user exit. The record's address in 4(R1) may also be modified. The logging action code must always be specified before returning to the Adabas nucleus.



**Warning:**

**When modifying the record, do not exceed the end address of the ADALOG I/O area contained in 8(R1).**

## PRILOG : Printing the Command Log

Adabas provides the PRILOG print program to read and report the contents of Adabas command logs in either version 4 or version 5 command log layout format.

PRILOG reads a sequential Adabas command log that has been produced directly by the Adabas nucleus (DD/LOG file) or by the ADARES CLCOPY utility when the Adabas nucleus uses dual or multiple comand logging.

PRILOG is supplied in both source and object form.

**Note:**

Although PRILOG can be customized, Software AG does not support the program when it has been modified by the user.

In source form, three modules (PRILOG, CCSTCK, and PRILOGD) are supplied for the OS/390 or z/OS, VSE/ESA, and BS2000 operating systems. These modules replace all PRILOG versions supplied with earlier versions of Adabas.

Two of the PRILOG modules are system-independent components and one is specific to a particular operating system:

- PRILOG interprets control statements; generates report lines from CLOG records.
- CCSTCK converts internal timestamp information on CLOG records into a more useful form before printing, making it compliant with Year 2000 standards. CCSTCK is provided independently so it can also be used by other programs.
- PRILOGD retrieves an input control card image and a CLOG record and prints a line. PRILOGD is system-dependent. It contains a number of parameters that are described in the source.

## Print Program Input

As input, the PRILOG program requires CLOG records and control cards.

Control cards must begin with the program name "PRILOG6" in columns 1 through 7; at least one space must follow the program name before parameters are entered.

Parameters can be entered up to column 71. No continuation or parameter splitting is permitted. Additional parameters can be entered on a separate PRILOG6 card.

A comment line begins with an asterisk (\*). Comments may also be added to the right side of the parameter string as long as the comment is separated from the parameter value by at least one space.

## Control Card Parameters

Two parameters can be entered on the control cards:

**CLOGLAYOUT**={ 4 | 5 }

CLOGLAYOUT indicates the format of the CLOG records being used as input to the PRILOG program. Valid values are 4 and 5; the default value is 5.

**FIELDS**= { ( item, ... ) | ( LIST ) }

FIELDS indicates the item (or items) from the CLOG records that are to be printed where "item" is one of the following:

Item	Description
FB	format buffer
IB	ISN buffer
IOL	I/O list
LIST	Adabas control block field list
RB	record buffer
SB	search buffer
UXB	user exit B buffer
VB	value buffer

The default value is LIST. Multiple "items" can be listed in any order.

If a data item listed in the FIELDS parameter is not being captured during the ADALOG session and is therefore not present in the CLOG record, the request to print that data item is ignored.

## PRILOG6 Messages

PRILOG6 messages are documented in the *Adabas Messages and Codes* manual as "PL6nnna" (independent) and "PL6ann" (system-dependent) messages.

## Installing and Using PRILOG under VSE/ESA

The following components comprise the PRILOG print program for VSE/ESA:

Member	Description
PRILOG.A	Independent PRILOG assembly language source module
PRILOGD.A	VSE/ESA-dependent PRILOG assembly language source module
CCSTCK.A	Independent Adabas STCK conversion assembly language source module
PRILOG.OBJ	Object deck for PRILOG
PRILOGD.OBJ	Object deck for PRILOGD
CCSTCK.OBJ	Object module for CCSTCK
PRILOG.PHASE	Executable phase for PRILOG
ASMPLOG.X	Sample VSE/ESA JCS to assemble, catalog, and link the PRILOG, CCSTCK, and PRILOGD components into the PRILOG.PHASE
LNKPLOG.X	Sample VSE/ESA JCS to link the PRILOG.PHASE from the PRILOG.OBJ, CCSTCK.OBJ, and PRILOGD.OBJ modules
RUNPLOG.X	Sample VSE/ESA JCS to execute the PRILOG utility

The PRILOG.PHASE may be executed from the library without additional preparation. However, if it becomes necessary to reassemble and relink the PRILOG.PHASE, the following points must be observed:

- The PRILOG.A, CCSTCK.A, and PRILOGD.A modules must be assembled using the IBM high-level Assembler (ASMA90).
- When linking PRILOG.PHASE, set the phase AMODE and RMODE attributes to 24-bit (AMODE(24) RMODE(24)).
- The provided sample JCS members ASMPLOG.X and LNKPLOG.X must be modified to suit your site's requirements. Check the LDEST, DEST, volume, and extent information as well as the library and sublibrary information provided in the JCS members.

### Executing PRILOG under VSE/ESA

Execute the PRILOG print program by running the PRILOG.PHASE in either a static or dynamic partition.

The PRILOG program uses three files:

File	Description
SYSIPT	PRILOG control card data; may be read from any valid device that can be assigned to SYSIPT
SYSLST	PRILOG report file; may be written to any device that can be assigned to SYSLST
SYS001 (DDCLOG)	Sequential command log input file; this file may reside on any IBM-supported disk device, or it may be read from tape

#### Notes:

1. The PRILOG control cards must be available from SYSIPT. If on disk or tape, they must be 80-byte records with FIXED record format.
2. The PRILOG report file must be written to SYSLST. If assigned to disk or tape, the output records are 121 bytes in length with FIXED record format. ASA control characters are used for printer control and are in the first byte of each print record.
3. The sequential command log file must be assigned to SYS001. The file may be on disk or tape and may not have a block size greater than 32,760 bytes. The record format is VARBLK.

The provided RUNPLOG.X JCS member may be modified according to your installation's requirements to execute the PRILOG print program. Modify the DEST, extent, DLBL, sublibrary, and volume information before submitting the job.

### Installing and Using PRILOG under OS/390 or z/OS

The following components comprise the PRILOG print program for OS/390 or z/OS:

<b>Member</b>	<b>Library</b>	<b>Description</b>
PRILOG	ADAvrs.MVSSRCE	Independent PRILOG assembly language source module
PRILOGD	ADAvrs.MVSSRCE	OS/390- or z/OS-dependent PRILOG assembly language source module
CCSTCK	ADAvrs.MVSSRCE	Independent Adabas STCK conversion assembly language source module
ASMPLOG	ADAvrs.MVSJOBS	Sample JCL to assemble and link the PRILOG, CCSTCK, and PRILOGD components into the PRILOG load module.
JPRILOG	ADAvrs.MVSJOBS	Sample JCL to execute the PRILOG utility

The PRILOG program uses three files:

<b>Files</b>	<b>Description</b>
DDCARD	used for input parameter data; may be any sequential 80-byte record file supported by QSAM.
DDPRINT	used for the output command log report; may be assigned to SYSOUT or to any 121-byte record dataset with record format of FBA.
DDCLOGIN	used for the input sequential command log file; must be a sequential file produced by the ADARES CLCOPY utility, or the direct DDLOG sequential file produced by an Adabas nucleus when single command logging is used.

# User Exit 5 (Adabas Review Hub Event Handler)

User exit 5 is called by the Adabas nucleus when an *event* occurs with the Adabas Review hub.

An event is defined as

- a connection made with the Adabas Review hub during Adabas session open;
- a connection broken with the Adabas Review hub during Adabas session close; or
- a non-zero return code received from the send operation for a command log record.

The exit is invoked with AMODE=31 and should return control in the same state.

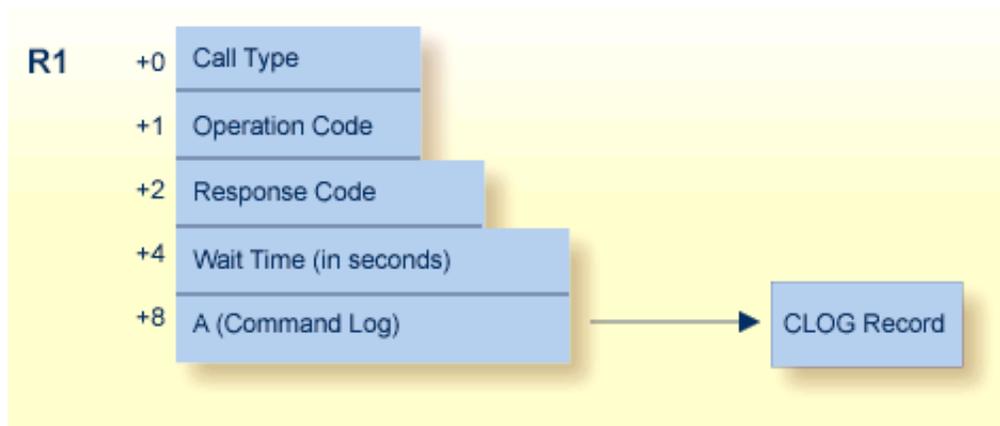
The exit is required to process logging errors. It determines how the failure is handled. The record that was not logged and the response code received from the Adabas Review hub logging request are provided to assist in making the determination.

This chapter covers the following topics:

- Input Parameters
  - Output Parameters
- 

## Input Parameters

On entry, the register 1 points to the following parameter list:



Parameter	Usage
0(R1)	Exit call indication. The value of this byte can be:  O      connection with Adabas Review hub opened;  C      connection with Adabas Review hub closed; or  L      sending logging error to Adabas Review hub.
1(R1)	Action to handle a logging error (ignored for open and close). The exit must provide one of the following values for this field in the parameter list for a logging error:  W      wait a specified time and then retry;  R      retry logging operation immediately; or  I      ignore the logging failure and continue without consequence.
2(R1)	Response code for logging errors. This response code is the same as the Adabas response code found in the <i>Adabas Messages and Codes</i> .
4(R1)	Fullword where the exit must provide a wait time (in seconds) for the logging failures that are to be retried after waiting.
8(R1)	Address of the command log record that the Adabas nucleus was attempting to send to the Adabas Review hub.

### Other Register Values at Entry

R13	save area of calling Adabas nucleus routine
R14	return address in Adabas nucleus
R15	entry point address for exit

### Output Parameters

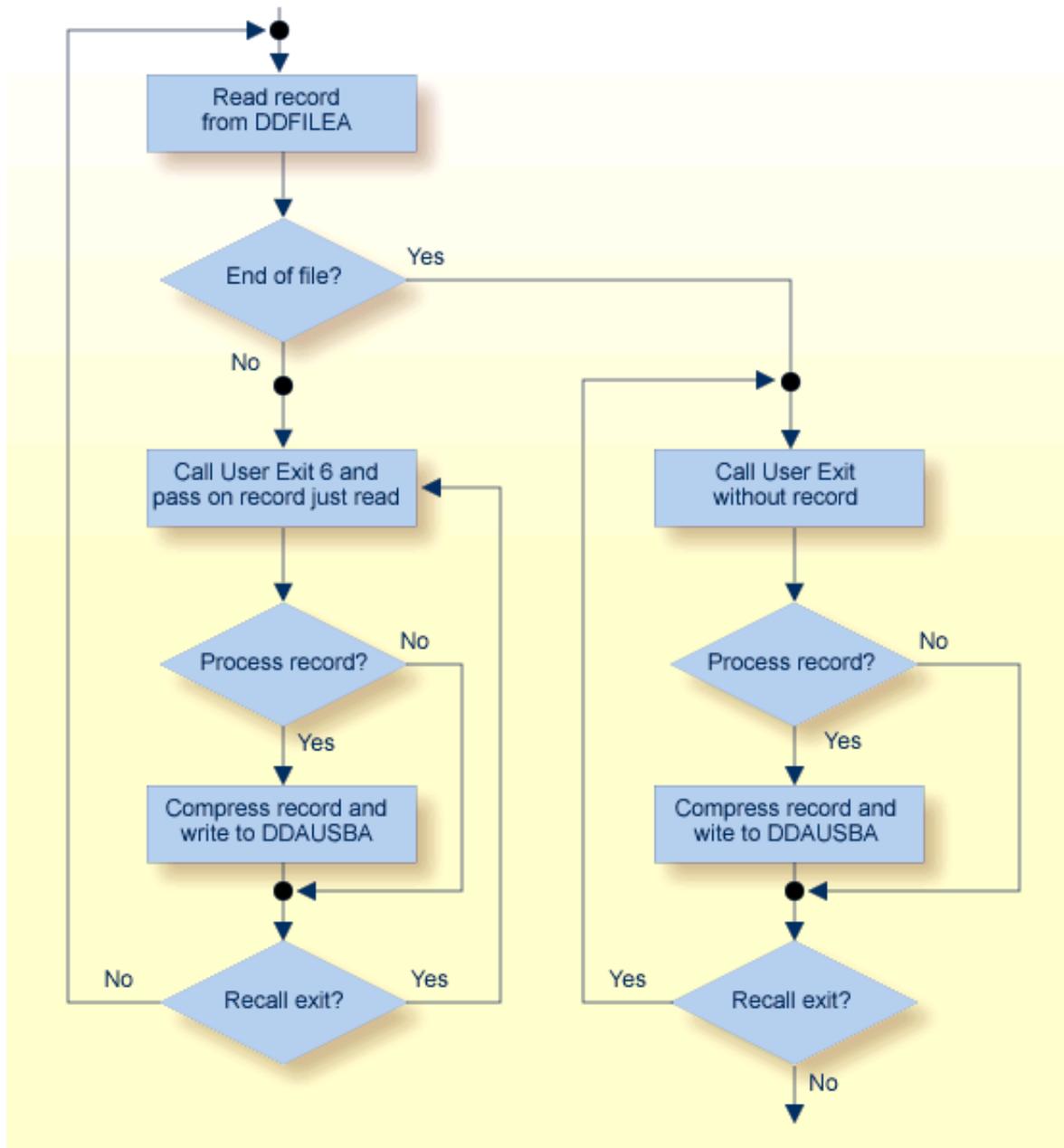
- For logging errors, the exit is required to set a value in the ‘operation’ field. If the wait value (W) is chosen, the exit is also required to provide a non-zero time value.
- Register 15 should be set to zero. All other registers should be returned intact.

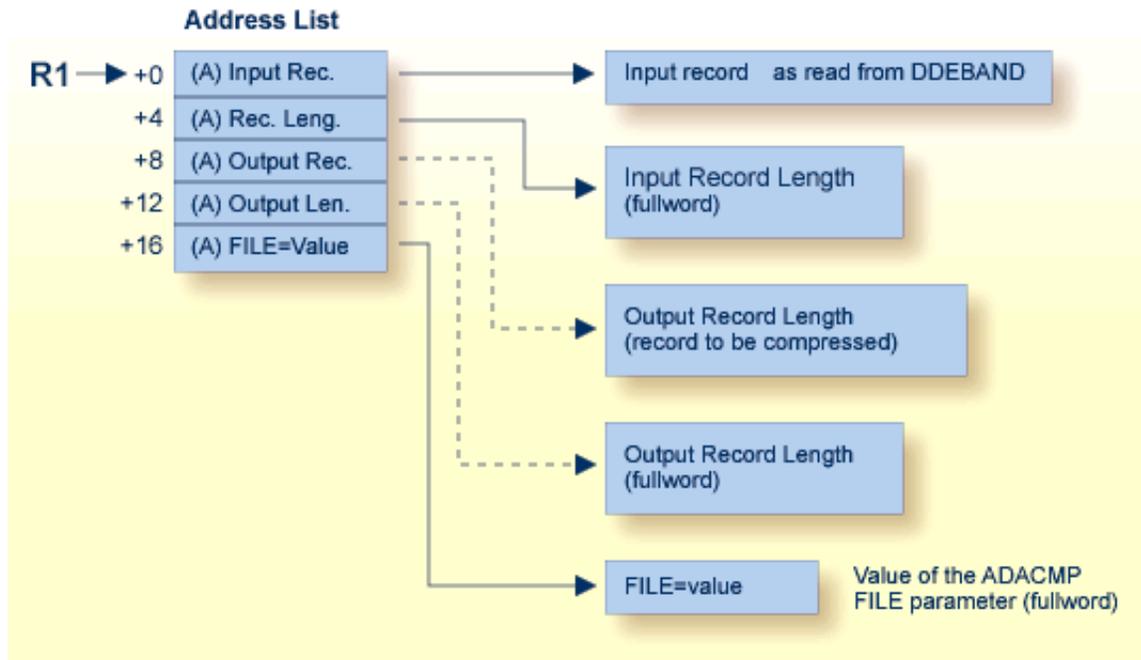
# User Exit 6 (User Processing Before Data Compression)

This user exit can be used to perform user processing on a record before it is processed by the ADACMP COMPRESS utility. The user program can be written in Assembler language or COBOL. The ADACMP COMPRESS utility job must specify the ADARUN UEX6=programparameter, where "program" is the name of the user program.

So that user exit 6 can append records to the input, it is called by the ADACMP COMPRESS utility function immediately after

- a record has been read from DD/EBAND; or
- an end-of-file condition has occurred on DD/EBAND.





## ADACMP User Processing User Exit (6) Parameters

### Note:

User exit 6 must return the same AMODE value to the calling program that was active when user exit 6 was called.

This chapter covers the following topics:

- Input Parameters
- Output Parameters

---

## Input Parameters

Parameter	Usage
0 (R1)	Address of an input record. The length field preceding the variable record is skipped. The address is of a fullword containing -1 (X'FFFF FFFF') if the user exit is called after ADACMP detects end-of-file in DD/EBAND.
4 (R1)	Address of the field containing the input record length. For fixed records, this is a logical record length. For variable records, this is the length of the actual data only (excluding the length field itself). The address points to a fullword containing minus 1 (X'FFFFFFFF') if the user exit is called after ADACMP detects end-of-file in DD/EBAND.
8 (R1)	Contains binary zeros on entry to the user exit (see <i>Output Parameters</i> ).
12 (R1)	Contains binary zeros on entry to the user exit (see <i>Output Parameters</i> ).
16 (R1)	Address of the FILE parameter value specified by the ADACMP COMPRESS utility job. The address is in the rightmost/low-order two bytes. The location and content of this fullword must remain unchanged during the time of the user exit. If ADACMP COMPRESS did not specify the FILE parameter, the fullword is X'0000'.

## Output Parameters

Parameter	Usage
8 (R1)	Address of the user exit output record. This record will be used as input to the ADACMP compression algorithm. The address of this record must be placed into 8 (R1) each time the user exit is called. If this field contains binary zeros on return, ADACMP will ignore the input record and will continue processing.
12 (R1)	Address of a 4-byte field containing the length of the returned record. The address of this field must be placed into 12 (R1) each time the exit is called. If this field contains binary zeros on return, ADACMP will ignore the record and will continue processing. Though the length field pointed to by 12 (R1) has a length of 4 bytes, only the low-order/rightmost halfword is used (bytes 3 and 4). If byte 2 contains a X'01' on return, the exit is recalled before the next record is read from DDEBAND. This enables the user to return more than one record to ADACMP for each record read from DD/EBAND.

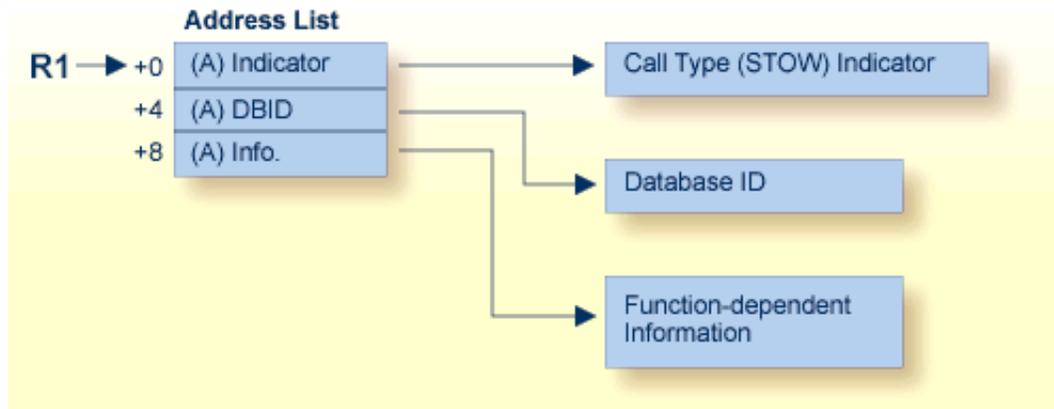
# User Exit 8 (Operator Interface)

This user exit receives control from the Adabas nucleus whenever the nucleus starts or stops, or whenever the nucleus or an Adabas utility receives a message from or sends a message to the operator. User exit 8 can be used to provide specific instructions to the operator

- when the nucleus starts and (normally) stops operation;
- as added information when Adabas sends console messages to the operator;
- to confirm commands entered by the operator.

User exit 8 is invoked

- (MODE=MULTI only) after Adabas startup, as soon as the nucleus is able to answer calls from user programs. At this point, the nucleus is now active.
- immediately after the Adabas nucleus or utility issues a console operator message. The user exit call is in addition to the standard message processing; the message itself cannot be changed during the user exit.
- after the Adabas nucleus or utility receives an operator command. The exit is called before the command is actually processed, and can reject or replace the command. The command cannot be modified in its original area.
- before a normal Adabas nucleus stop. At this point, the nucleus is no longer active; any more nucleus calls result in response code 148. This exit is not called if the nucleus ends abnormally.



## Operator Interface User Exit (8) Parameters

### Note:

User exit 8 must return the same AMODE value to the calling program that was active when user exit 8 was called.

This chapter covers the following topics:

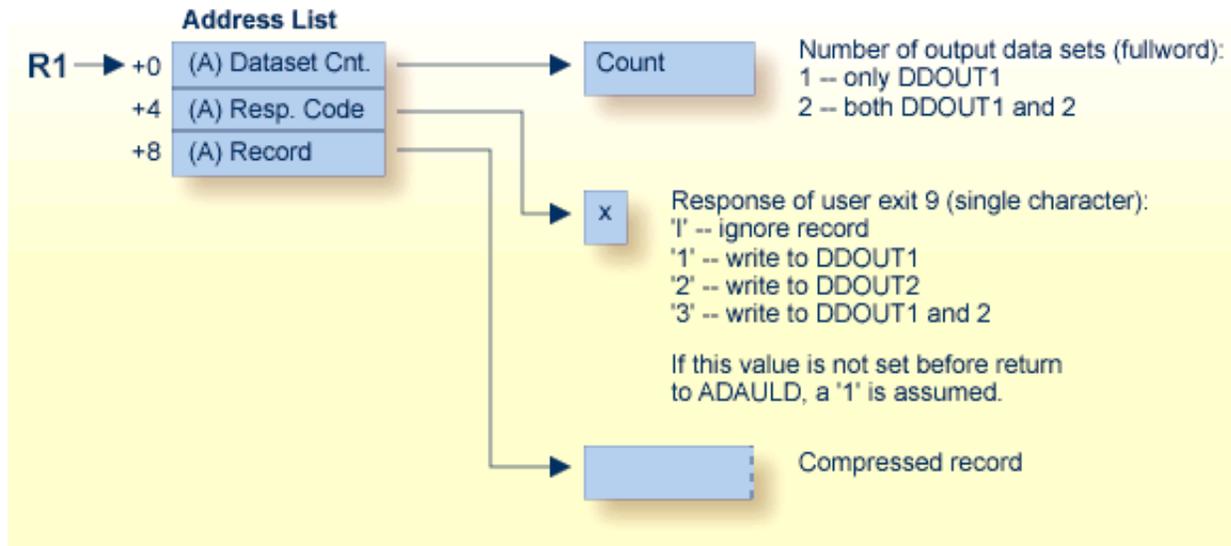
- Input Parameters

## Input Parameters

Parameter	Usage
0 (R1)	<p>Address of the byte containing the call type ("STOW") indicator:</p> <p>S      called at nucleus start</p> <p>T      called at normal nucleus termination</p> <p>O      called with an operator message to the nucleus/utility</p> <p>W      called with a nucleus/utility message to the operator</p>
4 (R1)	Address of the fullword containing the database ID.
8 (R1)	<p>Address of variable-length message-related information for "O" and "W" type calls. The information at this address has the following format: Call format:</p> <p>O      The one-byte message length, followed by the alphanumeric message. The length excludes the length byte itself. If the message is to be changed, location 8(R1) must point to the new message on return. This message is structured as described above. If the message is to be suppressed, location 8(R1) must point to a two-byte field containing X'0140'.</p> <p>W      8(R1) points to the message, which has the following structure:</p> <pre>DC X'message-length+4' DC X'0' DC X'message-text'</pre>

# User Exit 9 (ADAULD)

User exit 9 is called by ADAULD whenever a compressed record is ready to be written. The user exit decides whether a record is written to DD/OUT1, DD/OUT2, both, or neither.



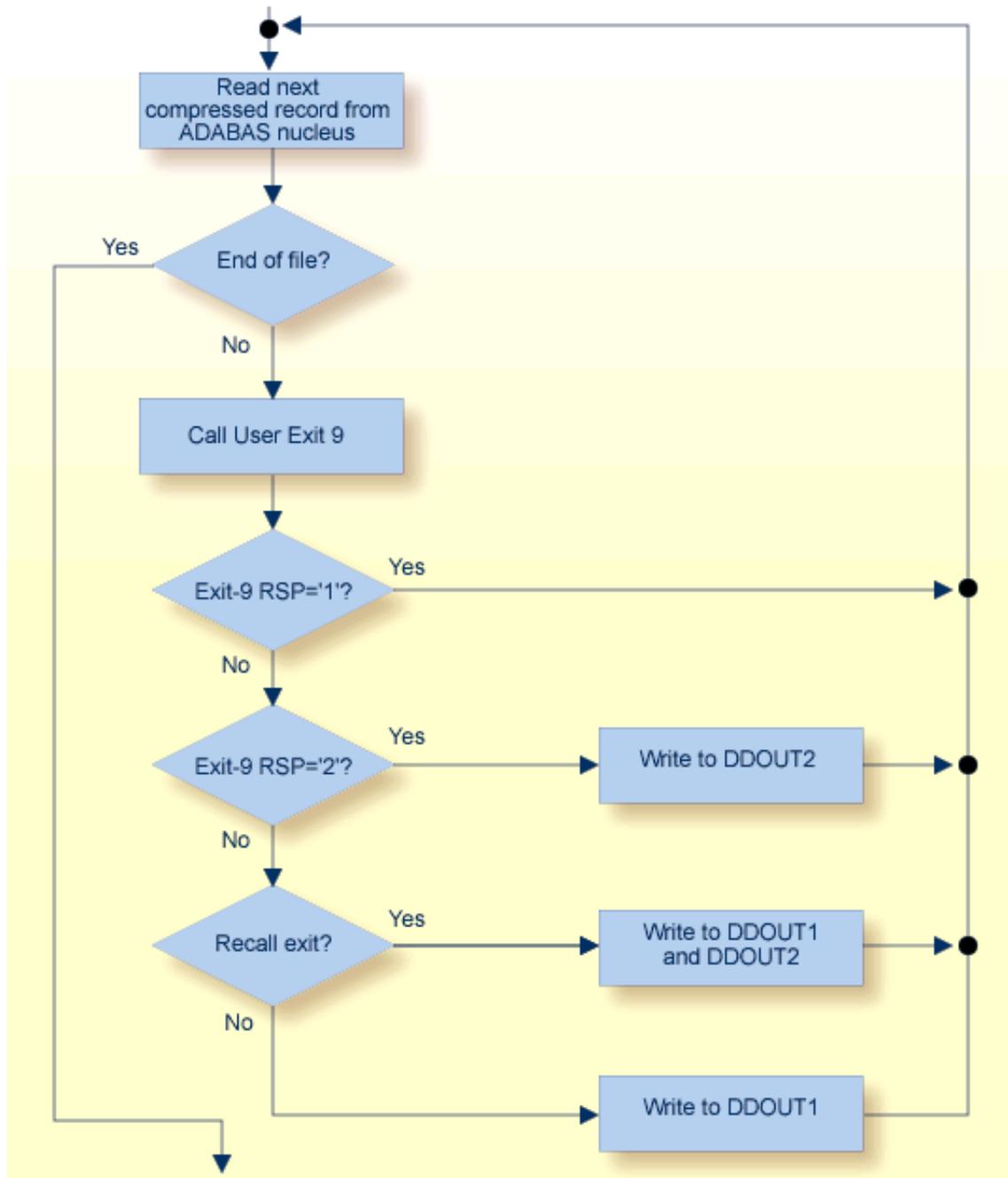
## ADAULD User Exit (9) Parameters

### Notes:

1. User exit 9 must return the same AMODE value to the calling program that was active when user exit 9 was called.
2. DDOUT1 & 2 must have the same block size, or an ADAULD error occurs.

The compressed record pointed to by the third address has the following structure:

```
AL2 (L1) total length (inclusive)
AL2 (L2) record length (inclusive)
AL4 (ISN)
XL (L2 - 8) '...compressed fields...'
XL (L1 - L2 - 2) '...DVT entries...'
```



### User Exit 9 Output Control Flow

The call to the user exit is made via a standard BALR 14,15 assembler instruction. All registers must be saved when control is received and restored immediately prior to returning control to ADAULD. The content of R15 is ignored.

# User Exit 12 (Multiple Dataset Log Processing)

**Note:**

UEX2 and UEX12 are mutually exclusive for an Adabas nucleus session: only one can be specified.

This user exit is given control by the Adabas nucleus during a switch from one multiple log dataset to another for the purpose of copying the log dataset before it is reused by Adabas. This switch occurs only if multiple dataset data protection logging and/or multiple dataset command logging is in effect for the session.

The user exit routine is designed to invoke a procedure that will execute the appropriate function (CLCOPY or PLCOPY) of the ADARES utility.

User exit 12 is invoked

- during Adabas nucleus startup if a multiple PLOG/CLOG dataset has to be copied;
- whenever a switch to another log dataset occurs;
- during Adabas nucleus shutdown.

The user exit is provided with information about the type of log (PLOG or CLOG) and the status of the multiple log datasets.

The user exit can decide which action is to be taken:

- Ignore the call and allow Adabas to proceed;
- Submit a job to copy and mark as empty the log dataset just filled (ADARES utility);
- Direct Adabas to wait for a specified interval, then call the user exit again with updated PLOG/CLOG dataset status information. During the wait interval, no commands that may produce log records for the log type being processed are allowed to proceed.

An example of user exit 12 is supplied with the Adabas installation procedure. Refer to the *Adabas Installation* documentation for more information.

The call to the user exit is made using a standard BASSM R14,R15 Assembler instruction. All registers must be saved when control is received and restored immediately prior to returning control to Adabas. Register 15 contains an action code as described on page 19.

**Note:**

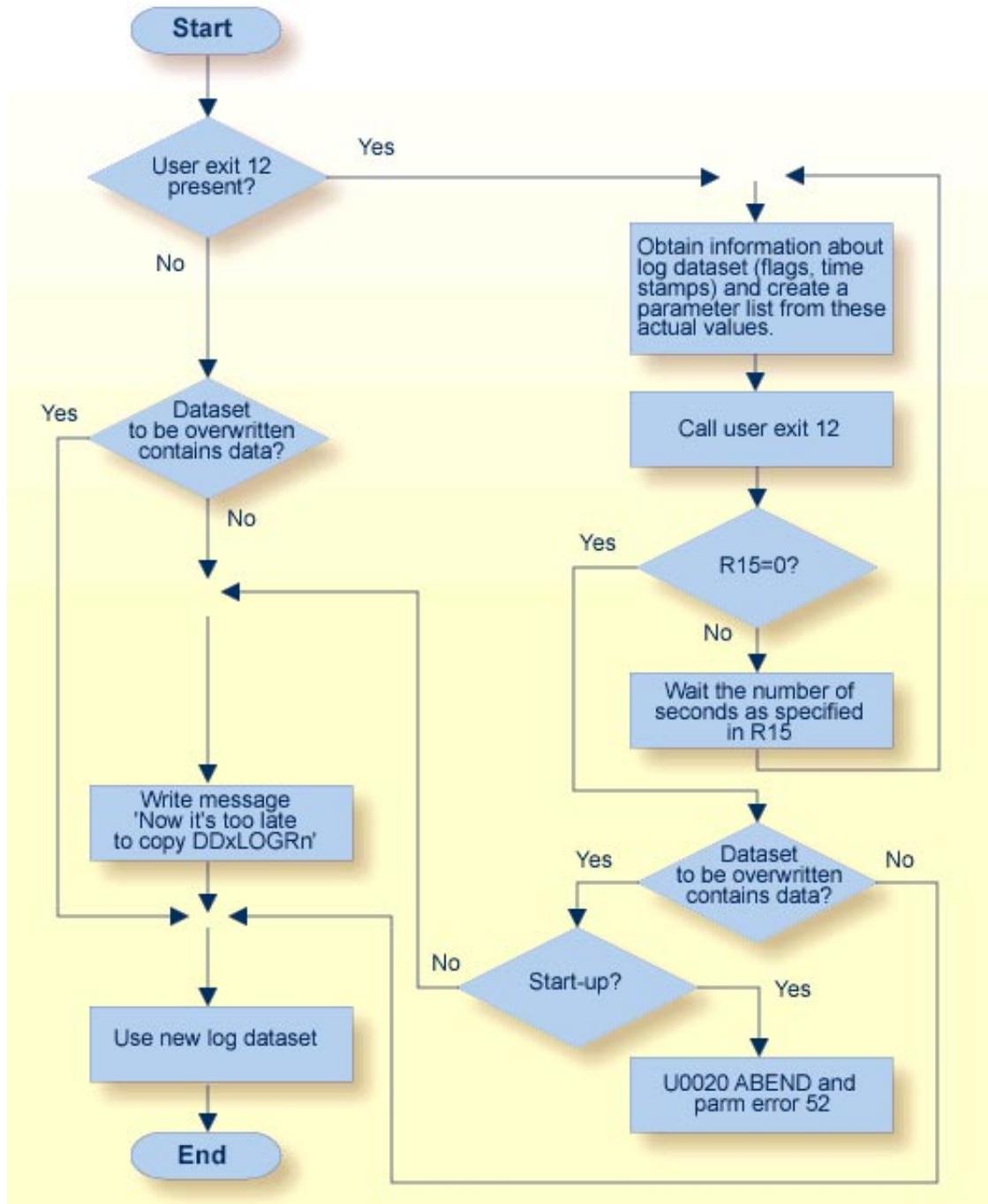
User exit 12 must return the same AMODE value to the calling program that was active when user exit 12 was called. The recommended Assembler instruction to return is BSM 0,R14.

This chapter covers the following topics:

- User Exit 12 Calling Sequence

- User Exit Interface
- Output Parameter
- Activating the Sample User Exit

## User Exit 12 Calling Sequence



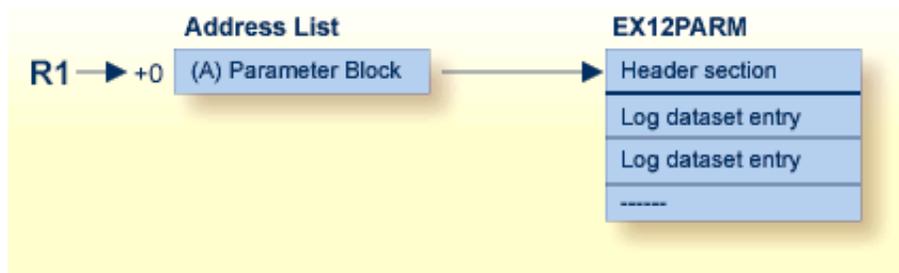
## Multiple Log Processing Flow

## User Exit Interface

Optionally, the user exit may initialize its operation. It may store any value in field EX12USER of the EX12PARAM parameter block to keep track of its resources. This field is considered as "owned" by the user exit and is supplied again for all subsequent executions of the exit. It is set to zero when the exit is first called and is not modified by Adabas thereafter.

The user exit is called again during termination to do any necessary finishing or 'cleanup' work.

## Parameters



## User Exit 12 Parameters

### DSECT of the EX12PARAM Parameter Block

```

Macro
EX12PARAM
*****
.* Name           Exl2Parm                      *
.* Component      ADABAS User Exit              *
.* Function       Parameter list for User Exit 12 *
.*               (replacement for User exit 2 for use when there are *
.*               more than 2 PLOGs or CLOGs)    *
.* Parameters     None                          *
.* Restrictions   None                          *
.* Notes         None                            *
*****

.*
EX12PARAM DSECT ,           User Exit 12 Parameter List
*
EX12HDR  DS      0F          Common header section
*
EX12USER DS      F          Reserved for the user. This field +
                             is initialized to zero before the +
                             exit is called the first time, and +
                             will not be altered by ADABAS after +
                             that. It can be used to maintain +
                             information across invocations.

*
EX12LOGT DS      X          Log type
EX12PLOG EQU     C'P'      PLOG
EX12CLOG EQU     C'C'      CLOG
*

```

```

EX12TYPE DS      X          Call type
EX12TBEG EQU     C'S'      Nucleus start
EX12TSW EQU     C'W'      Log switch
EX12TEND EQU     C'T'      Nucleus termination
                DS      XL2      Reserved
*
EX12NLOG DS      F          Number of logs
EX12DBID DS      F          Database ID
EX12NUCI DS      F          Nucleus ID
*EX12PLGN DS     F          Current session PLOG number      +
                        (zero for CLOGs)
EX12NCMP DS      F          Log just completed
EX12STAT DS      X          Flags of next log in sequence
EX12WNUC EQU     X'80'     Being written by the nucleus
EX12FULL EQU     X'40'     Completed by the nucleus
EX12RES EQU     X'20'     Being copied by ADARES
EX12CL5 EQU     X'08'     CLOGLAYOUT=5
EX12UNUS EQU     X'00'     Unused and/or copied
                DS      XL3      Reserved
                DS      4F      Reserved
*
EX12HDRL EQU     *-EX12HDR  Length of header section
*
EX12LOG DS       0F          Start of individual log dataset      +
                        entries. This section is repeated      +
                        for the number of logs specified in      +
                        field EX12NLOG
*
EX12LTIM DS      XL8       Time stamp of write to log dataset
EX12LNUM DS      F          Number of log dataset
EX12LFLG DS      X          Flags (mapped as in EX12STAT)
                DS      XL3      Reserved
                DS      4F      Reserved
*
EX12LOGL EQU     *-EX12LOG  Length of a log dataset entry
*
*
                MEnd
    
```

## Output Parameter

Parameter	Usage
R15 = 0	Nucleus continues processing.
R15 > 0	R15 is treated as the number of seconds to wait before calling user exit 12 again with updated status for all log datasets. During this time, no commands that may create log entries are processed.

## Activating the Sample User Exit

The sample user exit is written in Assembler language. It performs the following functions:

- Issues a message identifying the reason and the type of log for which it was called.

- Issues a message with the status and timestamp of all log datasets that are not empty.
- If any log dataset is full and at least one log dataset has a status that is different from the last time the exit was called, the exit reads 80-byte records from an input file and writes them to an output file. It replaces all occurrences of the character "?" with either "P" or "C", depending on whether the exit was invoked to process a PLOG or CLOG event, respectively. This allows the input file to accommodate an event for either log type. Normally, the input file contains job control statements and the output file is directed to a job execution queue.
- If at least one log dataset is not full, the exit returns to the caller with R15 zero, which allows Adabas to proceed.
- If all log datasets are full, the exit returns to the caller with R15 nonzero, which directs Adabas to wait for the number of seconds in R15, then call the exit again with an updated status of all log datasets. The default delay time is 30 seconds.

## OS/390 or z/OS

The sample user exit UX12SAMP is delivered on OS/390 and z/OS as source and as a load module that can be used without change or reassembly.

The source and load forms of the user exit are delivered in the Adabas source and load libraries, respectively. The job to assemble the user exit UX12ASML is located in the Adabas jobs library. The jobs library also contains a sample job UX12CJOB to be customized and submitted by the user exit that invokes the ADARES utility PLCOPY or CLCOPY function.

Activate the sample user exit as follows:

1. In addition to ADARUN NxLOG={2-8}, specify ADARUN UEX12=UX12SAMP for the Adabas nucleus.
2. Supply the job control model that the user exit is supposed to submit under the DDNAME COPYJOB.

Provide the following DD statement:

```
//INTRDR2 DD SYSOUT=(*,INTRDR)
```

### Note:

The same DD statement is used by the sample user exit 2 or 12 for submitting PLCOPY or CLCOPY jobs.

## VSE/ESA

The sample user exit UX12SAMP is delivered on VSE/ESA as source and as a phase that can be used without change or reassembly.

The source A.book and phase are delivered in the Adabas VSE sublibrary. The job to assemble the user exit UX12ASML.X is also provided in the Adabas sublibrary. The sample job UX12CJOB.X is to be customized and submitted by the user exit that invokes the ADARES utility PLCOPY or CLCOPY function.

Activate the sample user exit as follows:

1. In addition to ADARUN NxLOG={2-8}, specify ADARUN UEX12=UX12SAMP for the Adabas nucleus.
2. Supply the job control model that the user exit is supposed to submit under the DLBL COPYJOB.

## BS2000

The sample user exit UX12SAMP is delivered on BS2000 as source and as an object module that can be used without change or reassembly.

The source form of the user exit is delivered in the Adabas source library. The job to assemble the user exit, UX12ASML, is located in the Adabas jobs library. The jobs library also contains a sample job UX12CJOB to be submitted by UX12SAMP that invokes the ADARES utility PLCOPY or CLCOPY function.

Activate the sample user exit as follows:

1. In addition to ADARUN NxLOG={2-8}, specify ADARUN UEX12=UX12SAMP for the Adabas nucleus.
2. If the machine can process job variables, place the following statement in the Adabas nucleus job:

```
/SET-JV-LINK LINK-NAME=*DDJBDSF , -
/          JV-NAME=#JV.DSF
/MODIFY-JV  JV-ID=JV-NAME(JV-NAME=#JV.DSF) , -
/          VALUE='ENTER-JOB FROM-FILE=*LIB-ELEM( , -
/LIB=library,ELEM=dsfjob) , -
/JOB-CLASS=JCDSF'
```

where

*library* is the library; and  
*dsfjob* is the member where the customized job from step 2 is stored.

If the machine cannot process job variables, Adabas attempts to submit the job

```
(ADABAS.SRC,UX12CJBP,J)
```

for the PLCOPY job and

```
(ADABAS.SRC,UX12CJBC,J)
```

for the CLCOPY job.

An example job that allows the database ID to be supplied as a parameter is supplied in the Adabas source library ADAvrs.SRC.

If this is not satisfactory, you can modify the supplied source UX12SAMP.

When using user exit 12 with BS2000 systems, the name of the ADARES CLCOPY or PLCOPY job is no longer "hard-coded" in the user exit as in previous releases. Therefore, there are now two ways of specifying the /ENTER job:

## Method 1

Create a job variable containing the complete "enter job" command. For example:

```
/DCLJV   ENTER.ADARES.PLCOPY, LINK=*DDJBPLC
/SETJV  *DDJBx12,C'ENTER-PROC(srclib,ux12cjbx),(DB=dbid),J-N=X12COPY'
```

where

*srclib* is the library where the UX12CJBx(J) is stored (x=P for the PLCOPY job and x=C for the CLCOPY job)

*ux12cjbx* is the J element name of the respective procedure job.

*dbid* is the database ID of this nucleus.

Note that all operands of the /ENTER command may be used. The presence of the /DCLJV statement overrides the other possibilities. If necessary, the content of the job variables can be changed during a nucleus session.

Use the link names "\*DDJBPLC" for assigning the ADARES PLCOPY job, and "\*DDJBCLC" for assigning the ADARES CLCOPY job.

## Method 2

Omit the JV specification. The user exit 12 will then issue the following command:

```
'ENTER-PROC RES.EPdbid.xLCO,(DB=dbid),J-N=xL12<dbid>'
```

Use the following code to extract the job element in your database job deck prior to starting the database:

```
/DELETE-FILE RES.EPdbid.CLCO
/SET-JOB-STEP
/DELETE-FILE RES.EPdbid.PLCO
/SET-JOB-STEP
/ASS-SYSDTA *SYSCMD
/STA-LMS
//O srclib
//EXT-E (,ux12cjbc,J),RES.EPdbid.CLCO
//EXT-E (,ux12cjbpc,J),RES.EPdbid.PLCO
//END
/ASS-SYSDTA *P
```

where

*dbid* is the database ID of the P/CLCOPY database (this could be &DBID. in a procedure)

*srclib* is the source library where the template copy procedures are held

*ux12cjbc* is a procedure to perform the exit 12 ADARES CLCOPY job

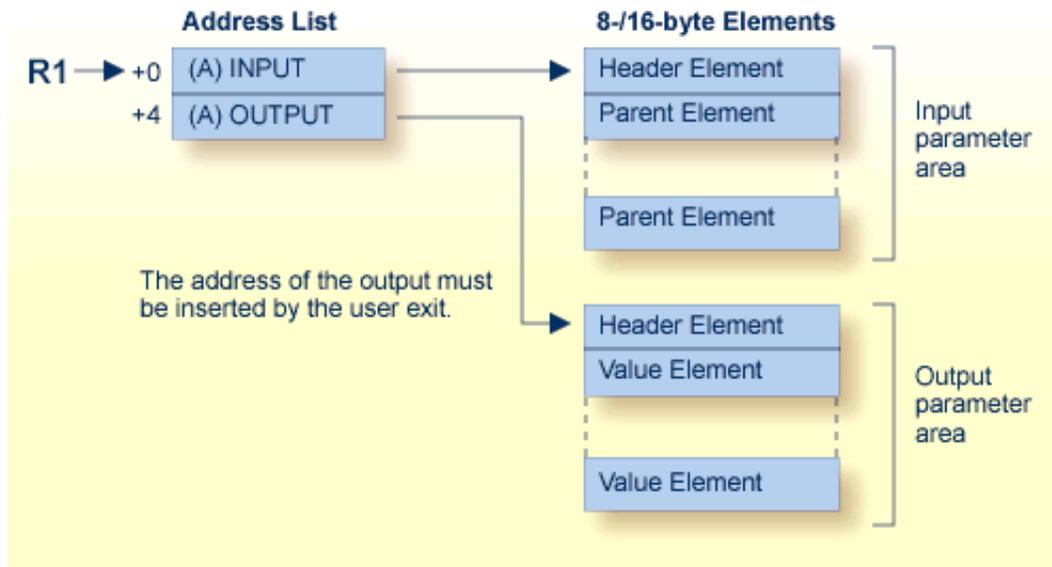
*ux12cjbpc* is a procedure to perform the exit 12 ADARES PLCOPY job

**Note:**

For BS2000 systems, user exit 12 is delivered both as a source and as an object element.

# Hyperdescriptor Exits 01 - 31

The hyperdescriptor exits (hyperexits) 1 through 31 (HEX01...HEX31) are required to define the algorithm for user-supplied descriptor values (see the *Adabas Utilities* documentation ). A hyperexit is called by ADACMP or the Adabas nucleus whenever a hyperdescriptor value is to be generated. ADACMP always uses the hyperdescriptor exit specified in its own ADARUN statement. When ADAINV specifies a hyperdescriptor exit, the exit used is the one specified in the Adabas nucleus' ADARUN statement.



## Hyperdescriptor Exit Parameters

### Note:

Hyperexits must return the same AMODE value to the calling program that was active when the hyperexit was called.

This chapter covers the following topics:

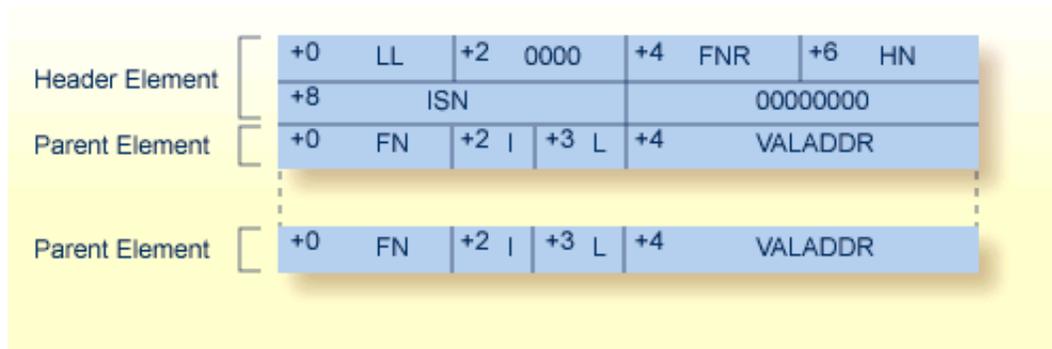
- Main Parameter Area
- INPUT Parameter Area (Pointed to by First Parameter Address)
- OUTPUT Parameter Area
- Null Value Option

---

## Main Parameter Area

Parameter	Address of the beginning of the . . .
0 (R1)	input parameter area.
4 (R1)	output parameter area. This address must be inserted by the user-written program. If no values are to be returned, the address value must be set to zero.

## INPUT Parameter Area (Pointed to by First Parameter Address)



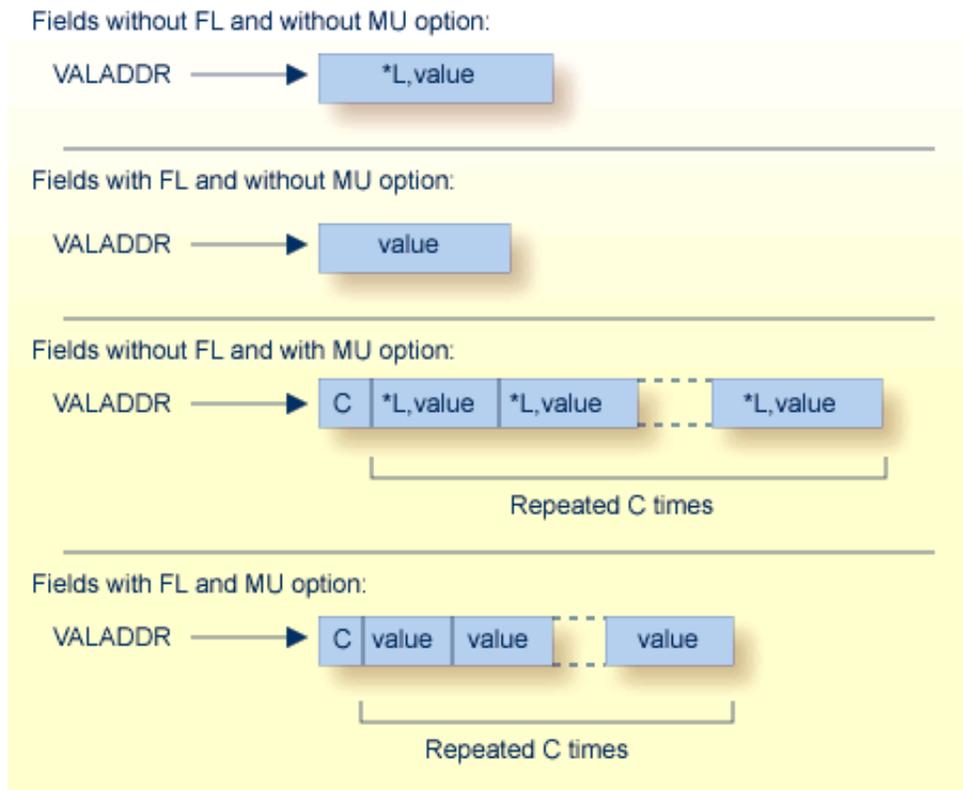
### Header Element Fields

LL	total length of the input parameter area, including this length field
FNR	file number
HN	name of the hyperdescriptor
ISN	ISN assigned to the record

### Parent Element Fields

FN	name of the parent field
I	periodic group index of the parent field. If the parent field is not part of a PE group, this byte contains a zero.
L	length of the value pointed to by VALADDR if the parent field is defined with the FI option.
VALADDR	address of the value of the parent field. The format of the value depends on the options of the fields. If the parent field is defined with the NU (null value suppression) option and the value for this field is suppressed, no input parameter element is created.

The following examples show formats for the value pointed to by VALADDR for parent fields with combinations of the FI (fixed storage) and MU (multiple-value) options:



where

C is a one-byte value representing the MU count.

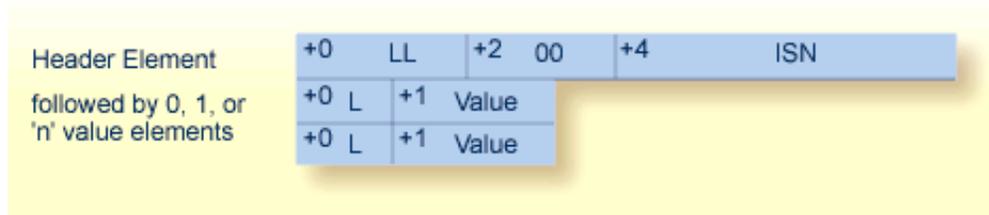
\*L is a hexadecimal value length, including this one- or two-byte length value. For lengths from 1 through 127, only a single byte is required. For lengths ranging 128 to 255, two bytes are needed: the first byte is set to X'80', and the second byte is set to the actual length value (see the following example table):

Length	Byte 1	Byte 2
L=127:	x'7F'	(x'80')
L=128:	x'80'	x'80'
L=255:	x'80'	x'FF'

## OUTPUT Parameter Area

This area must be allocated and filled within the hyperdescriptor user exit. The address of this area must be placed into the 2nd position of the main parameter area.

This area consists of a 8-byte header followed by the generated hyperdescriptor values in compressed format.



## Header Element

LL	total length of the output parameter area, including this length field.
00	reserved space. This must be set to zeros.
ISN	the ISN to be assigned to the descriptor values. If the original ISN is to be changed, the new ISN must be inserted here. If these four bytes contain zero on return to the Adabas nucleus, the original ISN is used. This is a four-byte binary value.

### Note:

If the hyperexit returns an ISN in the ISN field of the header element, the file must be defined with USERISN=YES to prevent ISN reassignment when the file is later reloaded.

## Value Elements

L	length of the following value, including this length byte. The maximum length depends on the format in use for the hyperdescriptor.
Value	the descriptor value to be inserted into the index. The value must follow the rules which are in effect for the format assigned to this hyperdescriptor. If the hyperdescriptor is defined with the PE option, one byte containing the one-byte PE index must immediately follow the value and be included in length L. The nucleus checks values of packed or numeric format for validity. Valid signs for packed fields are A,C,E,F (positive) and B,D (negative). The nucleus changes all signs to F or D.

Examples:

L	Value	
04	R E D	
06	B L U E02	where X'02' is a PE index
03	123F	packed 123
04	123F01	packed 123 in PE group with index 1

## Null Value Option

The NU (null value) option is possible for the hyperdescriptor and/or parent fields. The possible combinations are as follows:

- The hyperdescriptor is not NU:
  - The parentfield is not NU and the value is null, the hyperexit is called and the null value is passed.
  - The parentfield is NU and the value is null, the hyperexit is called and no input parameter element is created for this parent field.
  - All parentfields are NU and all values are null, the hyperexit is called and no input parameter element is created for any parent field.
- The hyperdescriptor is NU:
  - The parentfield is not NU and the value is null, the hyperexit is called and the null value is passed.
  - The parentfield is NU and its value is null, the hyperexit is called and no input parameter element is created for this parent field.
  - All parentfields are NU and all values are null, the hyperexit is not called.

# Collation Descriptor Exits 01 - 08

The collation descriptor exits 1 through 8 (CDX01 through CDX08) are used for encoding and decoding values for the corresponding collation descriptors.

A collation descriptor may be defined for a field with alphanumeric or wide format. Its values are stored in the index, not in the record itself. The number of the collation descriptor exit used to derive the values is associated with the collation descriptor.

A sample collation descriptor exit CDXE2A is provided in the Adabas source dataset. It converts EBCDIC to ASCII for the encoding function and the reverse (ASCII to EBCDIC) for the decoding function.

The Collation Exit implements three function entry points which are called on the following events:

## **INITIALIZE function**

- nucleus session start
- utility initialization when collation exits have been defined (ADARUN parameters)

## **ENCODE function**

- update/insert/delete of the parent's value (Nucleus)
- Search specifying the collation descriptor with the search value (Nucleus)
- compression of a record (ADACMP)

## **DECODE function**

- Read Index (L9) by Collation DE, only if the exit supports the DECODE function (Nucleus)

This chapter covers the following topics:

- Collation Descriptor Exit Interface
- 

## **Collation Descriptor Exit Interface**

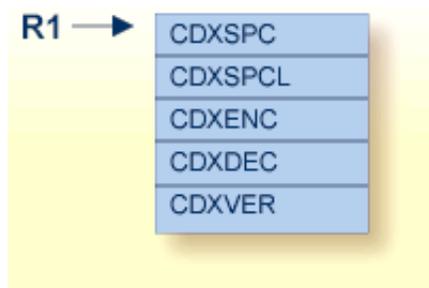
The collation descriptor exit interface is defined in the CDXPARM DSECT in the Adabas source dataset. The interface has three functions:

- initialization
- encoding
- decoding (optional)

## Initialization Parameters

R1 points to a list of addresses that point to five storage areas of the caller. The collation descriptor exit must set the five areas as follows:

CDXSPC	default space character; a maximum of 4 bytes
CDXSPCL	fullword containing the size of the space character
CDXENC	address of encoding function
CDXDEC	address of decoding function If the returned address is zero, decoding is not supported. The collation descriptor cannot then be used for L9 processing.
CDXVER	address of zero-byte delimited version string



## Encoding/Decoding Parameters

R1 points to a list of five fullword-sized parameters:

CDXIA	address of the input string
CDXIL	length of the input string
CDXOA	address of the output string
CDXOL	size of the output area
CDXARL	address of the length of the returned output string

The collation descriptor exit stores

- the output string in the area at the address specified by CDXOA; and
- the input string length in the fullword at the address specified by CDXARL.

