Error Handling and Message Buffering

This chapter covers the following topics:

- Overview
- Range of Operations
- Recovery or Plug-In (PIN) Routines
- PIN Routine User Exit

Overview

The error handling and message buffering facility helps implement 24X7 operations by analyzing and recovering from certain types of errors automatically with little or no DBA intervention. It also generates additional information so that the error can be diagnosed by the user and by Software AG.

The ADARUN SMGT parameter is set to activate the facility; if message buffering is to be used, the ADARUN MSGBUF parameter is used to size the buffer. The wrap-around message buffer collects Adabas messages for later review by Adabas Online System in case online access to the console or to DDPRINT messages becomes unavailable. The buffer aids problem analysis and performance tuning.

The error handling functions of the facility are implemented as operands of the operator command SMGT and can be invoked from the operator console or from Adabas Online System. See the Adabas Operations documentation for detailed information.

The current implementation of the facility makes it possible for the nucleus to protect itself or provide additional error recovery information for

- a parameter error 31 : autorestart error;
- a parameter error 73 : checkpoint file is full during initialization;
- non-response codes by capturing pertinent areas of storage to aid in the diagnosis; and
- program interruptions by providing additional dump areas.

Range of Operations

User Exit Failures

User exits and hyperexits that are essential to the operation of the Adabas nucleus can be marked as critical (the default) or not using one of the operands of the SMGT operator command:

- If a user exit is defined as *critical*, is is not affected by the error handling and message buffering facility: an abnormal termination in it causes the Adabas nucleus to terminate abnormally as well.

- If the user exit is defined as *notcritical* and an abnormal termination occurs in it, the facility maintains an active Adabas nucleus, optionally refrains from invoking that exit, takes a dump of the nucleus at the point when the exit failed, and issues messages to the system log to inform the DBA of the problem. The DBA can then examine the diagnostic information, use it to fix the problem, then load and reactivate the corrected exit using operands of the SMGT operator command.

Note:
If an Adabas exit attaches a subtask, the subtask is not protected by the error handling and message buffering facility.

Recovery or Plug-In (PIN) Routines

The extensions (plug-in routines or "PINs") are designed to analyze and, in some cases, determine the cause of an abend while allowing the nucleus to continue processing. The PIN determines whether it is safe to allow the nucleus to continue processing and prints appropriate messages to notify the DBA when this is the case.

The PIN routine user exit ADASMXIT can be used to obtain additional information about response codes and abends. The user exit allows you to specify particular response codes or response code/subcode combinations to be monitored. Once you have modified the user exit, you can reload it and make your changes effective without bringing the database down.

Each plug-in (PIN) service routine handles a predefined condition when encountered, allowing the Adabas nucleus to

- remain active when it otherwise would terminate abnormally; or
- print extended error diagnostics as an aid to error recovery.

Based on its execution, a PIN module can either transfer control to the Adabas nucleus so that it can resume normal processing-usually with a response code-or it can return control to the error handling and message buffering facility, allowing the Adabas nucleus to terminate abnormally.

While the PIN is executing, most Adabas functionality is available to the PIN as the registers at the time of the abnormal event are available. The PIN decides whether the nucleus should remain active.

A PIN can also be used to format an intelligent dump in a number of circumstances to help debug a particular response or abend code.

If the PIN determines that the nucleus is to remain active, the PIN sets a response code.

- PIN Processing
- Default PIN Module ADAMXY
- Additional PIN Modules Provided

PIN Processing

In the event of an abnormal termination or nonzero response code, the error handling and message buffering facility looks for a PIN routine first for the specific condition and location detected, then for the specific condition, and finally for the location (any condition). If an appropriate PIN is found, it is invoked; otherwise, the nucleus is terminated.

If a PIN routine is invoked and it

- handles the condition, processing then continues with no further intervention from the error handling facility.
- cannot handle the condition, the PIN returns control to the error handling facility and the nucleus is terminated.

For a response code error, the error handling facility first determines whether the response code is one that it monitors.

- If the answer is no, the PIN returns control to the nucleus and the response code is returned to the user normally.
- Otherwise, the appropriate PIN is invoked to print additional information about the response code that to help resolve the problem. The PIN then returns control to the nucleus and the response code is returned to the user normally.

Once the PIN has processed the response code, it returns control to the error handling facility so that normal response code processing can continue.

Default PIN Module ADAMXY

The ADAMXY module is the default PIN module comprising the PIN routines that are distributed with Adabas and automatically installed during initialization.

Note:
It is possible to disable the default PIN ADAMXY using the SMGT,DELPIN or SMGT,DEACTPIN operator commands.

The following table describes the interrupts that are handled by the PIN routines in ADAMXY. For each interrupt, extended dump formating is provided to aid in error analysis:

Code	Exception Type	The processor . . .
01	Operation	is about to execute an instruction that has an invalid operation code.
02	Privileged Operation	attempts to execute a supervisory instruction while in problem state.
03	Execute	interrupts a program deliberately to aid problem diagnosis.
04	Protection (also Segment and Page)	attempts to alter system or hardware storage; access fetch protected system or hardware storage; or access or modify storage that is not allocated. Requires record/file-level locking with user notification in job log. Note that code 16 (segment exception) and code 17 (page exception) are also presented to the error handling facility as code 04.
05	Addressing	encounters a reference to an invalid read address.
06	Specification	attempts to either set or branch to an old address or an instruction that required a field to be aligned but did not have an aligned argument.
07	Data	encounters a corrupted data record, probably a field that should be packed decimal is not.
08	Fixed Point	a high-order carry occurs; or high-order significant bits are lost in a fixed-point add, subtract, shift, or sign-control operation.
09, 11, 15	Divide	encounters a zero divisor in a division instruction; probably a corrupted record. Code 9 is for binary; code 11 is for packed decimal; and code 15 is for floating point arithmetic.

The message

```
*****DEFAULT PIN OUTPUT************
```

is generated whenever the default PIN ADAMXY is invoked. This is followed by all output concerning the program interrupt processing of ADAMXY. The message

```
*****END OF DEFAULT PIN OUTPUT*****
```

-is generated whenever ADAMXY is completed.

Additional PIN Modules Provided

Some of the PIN modules discussed in this section are delivered with selectable units of Adabas. They are established automatically when the relevant server component initializes at nucleus startup:

PINAFP
PINATM
PINAVI
PINCOR
PINSAF

The remaining PIN modules discussed in this section are included with Adabas but are not part of ADAMXY and are not automatically installed at Adabas initialization:

PINAUTOR
PINOPRSP
PINRSP
PINUES

PINRSP and PINUES are installed using the SMGT,ADDPIN=module-name command when the nucleus is active. Because PINAUTOR and PINOPRSP are invoked during system initialization when operator commands are not available, they are activated by renaming a particular module in the Adabas load library:

- renaming NOOPRSP to PINOPRSP activates that PIN;
- renaming NOAUTOR to PINAUTOR activates that PIN.

See the Adabas Installation documentation for more information about installing PIN modules.

PINAFP

PINAFP is delivered with the selectable unit Adabas Fastpath. It is established automatically when the Adabas Fastpath server component initializes at nucleus startup (ADARUN FASTPATH=YES).

In the event of a program interrupt (see the table in the section *Default PIN Module ADAMXY*) in the Adabas Fastpath server component, control is passed to PINAFP, which formats and prints the main memory areas used by the component. These diagnostics are written to the DDPRINT dataset with the title

```
ADABAS FASTPATH - memory-area-name : SNAP BY PINAFP
```

PINAFP then returns control to the error handling and message buffering facility so that Adabas can terminate abnormally.

If necessary, PINAFP can be activated and deactivated. However, after PINAFP is reactivated, it will not be reestablished until the next nucleus session.

PINATM

PINATM is delivered with the selectable unit Adabas Transaction Manager (ATM). It is established automatically when the ATM job initializes (ADARUN DTP=TM).

In the event of a program interrupt (see the table in the section *Default PIN Module ADAMXY*) in the ATM logic, control may be passed to PINATM, which formats and prints the main memory areas used by ATM. These diagnostics are written to the DDPRINT dataset with the title

```
ADABAS TRANSACTION MANAGER - memory-area-name : SNAP BY PINATM
```

PINATM then returns control to the error handling and message buffering facility so that Adabas can terminate abnormally.

If necessary, PINATM can be activated and deactivated. However, after PINATM is reactivated, it will not be reestablished until the next ATM session.

PINAUTOR

If NOAUTOR has been renamed to PINAUTOR in the Adabas load library and a parameter error 31 occurs during autorestart, PINAUTOR acquires control. PINAUTOR attempts to identify the file that is in error and exclude it from autorestart if possible.

Before excluding a file from autorestart processing, PINAUTOR checks that

- the file is not the security or checkpoint file; and
- the response code is not 9, 65, 72, 88, 97, 99, 148, or 151 as these are not valid for the exclusion process.

Additionally, PINAUTOR checks whether certain files or particular response codes for a particular database are designated as ineligible for exclusion. For example, it may be senseless to start a database without the particular file on which it depends. You can customize ADASMXIT to include the files and the response codes that cannot be excluded. You can also specify the maximum number of autorestarts with exclusions that can be attempted.

The AREXCLUDE procedure is automatically invoked to exclude a file. See the Adabas Operations documentation, ADARUN parameter AREXCLUDE for more information. Note that excluded files may become inconsistent and need to be restored from backup using ADASAV RESTORE.

If a file is excluded from autorestart, an SM-PINAUTOR2 message is generated followed by an ADAN50 message, both indicating the file number of the excluded file.

Whenever PINAUTOR is invoked, the message PINAUTOR OUTPUT is generated followed by messages pertaining to the specific PINAUTOR situation as described in the Adabas Messages and Codes documentation. To indicate that PINAUTOR processing is completed, the message END PINAUTOR OUTPUT is generated.

PINAVI

PINAVI is delivered with the selectable unit Adabas Vista. It is established automatically when the Adabas Vista server component initializes at nucleus startup (ADARUN VISTA=YES).

In the event of a program interrupt (see the table in the section *Default PIN Module ADAMXY*) in the Adabas Vista server component, control is passed to PINAVI, which formats and prints the main memory areas used by the component. These diagnostics are written to the DDPRINT dataset with the title

```
ADABAS VISTA - memory-area-name : SNAP BY PINAVI
```

PINAVI then disables the program in which the interrupt occurred and returns control to Adabas so that Adabas can continue. Disabling the program does not disrupt the Adabas service; however, access to Adabas Vista files may be restricted. In this case, a nonzero response code returned to the user identifies the restrictions.

If necessary, PINAVI can be activated and deactivated. However, after PINAVI is reactivated, it will not be reestablished until the next nucleus session.

PINCOR

PINCOR is delivered with System Coordinator for Adabas Options. It is established automatically when the System Coordinator server component (ADAPOP) initializes at nucleus startup.

If a program interrupt occurs in the System Coordinator server component, control is passed to PINCOR, which formats and prints the main memory areas used by the component.

These diagnostics are written to the DDPRINT dataset with the title

```
COMMON RUNTIME - memory-area-name : SNAP BY SMGT
```

PINCOR then returns control to the error handling and message buffering facility so that Adabas can terminate abnormally.

PINOPRSP

⚠ **Warning:**
PINOPRSP makes it possible to initialize a database and operate it without writing checkpoints. If database recovery procedures become necessary, the missing checkpoint information can result in critical errors. To prevent this, you must reorder the checkpoint file immediately after PINOPRSP is invoked and be able to account for all changes in the status of the database between initialization and the reordering of the checkpoint file.

If NOOPRSP has been renamed to PINOPRSP in the Adabas load library and a parameter error 73 occurs during system initialization indicating a checkpoint overflow condition, PINOPRSP is invoked.

The message "PINOPRSP OUTPUT" is generated indicating that PINOPRSP has been invoked. PINOPRSP then generates a message warning the DBA that Adabas will activate even though the checkpoint file is full:

```
response code  INTERCEPTED BY PINOPRSP BECAUSE THE CHECKPOINT FILE IS
FULL. THE ADABAS NUCLEUS WILL ACTIVATE BUT THE CHECKPOINT FILE NEEDS
TO BE REORDERED AS SOON AS POSSIBLE.
```

The "response code" is either 75 or 77 in this case. No checkpoint is written but the nucleus activates. Corrective action needs to be taken as soon as possible. The message "END PINOPRSP OUTPUT" is then generated to indicate that PINOPRSP processing is completed.

PINRSP

The SMGT,ADDPIN=PINRSP operator command activates PINRSP, which provides extended information to aid in diagnosing a response code.

Note:
Only response codes set by Adabas can be logged. A response code such as 22 (invalid command code), which is set by the Adabas SVC before it reaches Adabas, is not logged.

If PINRSP is installed without modifying the Adabas PIN routine user exit ADASMXIT, all response codes are logged. You can customize ADASMXIT to

- include specific response codes or response code/subcode combinations;
- indicate the number of times a particular response code can be monitored.

When a nonzero response code is encountered, PINRSP acquires control. The message "PINRSP OUTPUT" is generated to indicate that PINRSP has control. Depending on the response code encountered, different areas are logged.

With respect to areas logged, five categories of response codes are described:

1. Basic response code logging includes
   - the active thread
   - the FCB if possible
   - the areas that have been GETMAINed and are currently in use
2. Index-related response codes such as 177
   - basic response code logging
   - the index structure from the thread
   - active CQEs
   - buffer pool headers
3. Response codes such as 40 where additional IUB areas may be pertinent
   - basic repsonse code logging
   - IUBs
   - active CQEs
4. Response codes such as 255 where additional attached buffer information may be necessary
   - basic reponse code logging
   - active CQEs
   - attached buffer information
5. Response codes such as 72 where the user queue may be helpful
   - basic response code logging
   - active CQEs
   - user queue

Example:

Rather than obtain a CLOG when you need additional information to diagnose a particular response code, you can modify ADASMXIT to capture the response code, reassemble it, and load it while the nucleus is up. The information is then logged the next time the response code is encountered.

Once you have the information, you can modify ADASMXIT to remove the response code and reload it so that information is no longer captured. Alternatively, you can set ADASMXIT initially to log the information only ‘n’ number of times.

You can also use PINRSP in conjunction with ADASMXIT to suppress the ADAN77 message that is generated for response codes 201, 202, or 203. This may be useful in situations where a new application receives enough security errors to fill the SYSLOG. Although Software AG does not recommend this action, you may temporarily modify ADASMXIT to suppress N77 messages and activate PINRSP with response codes 201, 202 and 203 indicated in ADASMXIT.

If message suppression is activated, the ADAN77 message "Message suppression in effect" is generated and the PINRSP output providing format information related to the response code is suppressed.

Once PINRSP has completed processing, the message END PINRSP OUTPUT is generated.

PINSAF

PINSAF is delivered with the selectable unit Adabas SAF Security (ADASAF). It is invoked automatically when the ADASAF initializes at nucleus startup.

In the event of a program interrupt (see the table in the section *Default PIN Module ADAMXY*) in ADASAF, control is passed to PINSAF, which formats and prints the main memory areas used by ADASAF. These diagnostics are written to a dataset with the title

```
ADABAS SAF INTERFACE - control-block-name : SNAP BY SMGT
```

PINSAF then returns to the error handling facility so that Adabas can terminate abnormally.

Note:
For security reasons, PINSAF does not allow Adabas to continue after an abend in ADASAF.

Like other PIN routines, PINSAF can be activated and deactivated. However, after PINSAF is reactivated, ADASAF itself must be restarted before PINSAF will function correctly. Refer to the Adabas SAF Security documentation for more information.

PINUES

PINUES handles Adabas response codes in the context of the universal encoding support (UES) system. PINUES captures input/output errors when trying to

- load an encoding object that does not exist; or
- convert invalid data.

Note:

When used with other PIN routines that handle the same error conditions, the PIN loaded last is called to handle the error. For example:

```
F NUC227,SMGT,ADDPIN=PINRSP
F NUC227,SMGT,ADDPIN=PINUES
```

In this example, PINUES is loaded after PINRSP and therefore handles the error conditions it can handle (response codes 17, 48, and 55). All other response codes are processed by PINRSP.

The message PINUES OUTPUT is generated to show that PINUES has acquired control. The message END PINUES OUTPUT is generated when PINUES processing is completed.

Error Conditions Handled

- Response codes 17 and 48 may occur on an OP command if the ECS objects are not available that are needed to determine the data conversion between user and Adabas file. In this case, PINUES calls ADAMXF with the options IUB and UQE to obtain diagnostic output.
- Response code 55 may occur if ECS returns a response indicating that the conversion or moving of text failed. In this case, the conversion parameters and buffers are snapped to obtain diagnostic output.

Output Produced

Whenever PINUES writes diagnostic information, the following lines are printed on the console:

```
  ******** P I N U E S  OUTPUT ********'

  ADANX1 dbid COMMAND cmd COMMAND ID hex-cid FNR file-number
         RESPONSE adabas-response-code SUBCODE adabas-subcode FLD
  field-name'
         TID hex-internal-user-id UID open-userid JOB job-name'
  ****** END P I N U E S  OUTPUT ******'
```

Session Snapshots

```
18:17:37 ADAN19 00227 BUFFERFLUSH IS  A S Y N C H R O N O U S
18:17:37 ADAN01 00227 A D A B A S  V7.1.0  IS ACTIVE
18:17:37 ADAN01 00227 MODE = MULTI
18:17:37 ADAN01 00227 RUNNING WITHOUT RECOVERY-LOG
18:18:04 ADAI29 OPER CMD: SMGT,ADDPIN=PINUES
18:18:04 ADANTG 00227 PIN MODULE PINUES   LOADED
18:18:04 ADANO2 00227 SMGT COMMAND PROCESSED
18:18:04 ADAN41 00227 1999-01-00 18:18:03 FUNCTION COMPLETED

18:36:33 ADAN7A 00227 ECS ERROR     -2 IN FUNCTION GETHANDL
18:37:21
18:37:21 ******** P I N U E S  OUTPUT ********
18:37:21 ADANX1 00227 COMMAND OP COMMAND ID 00000000 FNR 00014
18:37:21              RESPONSE 017 SUBCODE 023
18:37:21              TID 00000013 UID BLAUTOPF JOB TXG.....
18:37:21 ADAH51 00227 DUMP FORMAT CALLED
```

The following output is produced by the ADAMXF module:

```
18:37:22 ADAH52 00227 DUMP FORMAT COMPLETED
18:37:22 ****** END P I N U E S  OUTPUT ******
18:37:22
18:49:45 ADAN7A 00227 ECS ERROR     54 IN FUNCTION CVFTXTX
18:49:45
18:49:45 ******** P I N U E S  OUTPUT ********
18:49:45 ADANX1 00227 COMMAND A1 COMMAND ID 00000000 FNR 00014
18:49:45              RESPONSE 055 SUBCODE 004
18:49:45              TID 00000017 UID ANDECHS. JOB TXG.....

   ECS CONVERSION PARAMETERS

   0C190BE0+0000  00106570 00000080 00000200 00000000 *................*
   0C190BF0+0010  00000004 0C10ACB4 00000004 0010A6E0 *..............w.*
   0C190C00+0020  00004000 0C190BEC 0000020C 00000000 *................*
   0C190C10+0030  001065AD 0C190BE4 00000000 00000000 *.......U........*

   ECSE FROM ENCODING 3026 TO ENCODING 3035

   00106570+0000  02000002 00000BD2 00000BDB 00000004 *.......K........*
   00106580+0010  00000000 001062C8 00106350 001064E8 *.......H.......Y*
   00106590+0020  0000BD20 0000BDB0 0000000C 00000000 *................*
   001065A0+0030  001065AD 0004362C 40000000 00FEFE00 *................*
   001065B0+0040  00000340 40000000 04404000 00000000 *................*
   001065C0+0050  00000000 00000000 00000000 00000000 *................*
   001065D0+0060  00000000 00000000 00000000 00000000 *................*
   001065E0+0070  00000000 00000000 00000000 00000000 *................*
   001065F0+0080  00000000 00000000 02000001 00000BDB *................*

   ECONV INPUT AREA

   0C10ACB4+0000  4141F1F2 40404040 40404040 40404040 *..12............*

   ECONV OUTPUT AREA

   0010A6E0+0000  414150C2 50C350C4 50C550C6 50C750C8 *...B.C.D.E.F.G.H*
   0010A6F0+0010  50C150C2 50C350C4 50C550C6 50C750C8 *.A.B.C.D.E.F.G.H*
       LINES 0010A700 TO 0010A7C0 SAME AS ABOVE
   0010A7D0+00F0  50C150C2 50C350C4 50C550C6 00000000 *.A.B.C.D.E.F....*
   0010A7E0+0100  00000000 00000000 00000000 00000000 *................*
       LINES 0010A7F0 TO 0010E6D0 SAME AS ABOVE

18:49:45 ****** END P I N U E S  OUTPUT ******
```

PIN Routine User Exit

The PIN routine user exit (entry name ADASMXIT) can be used to

- supply parameters to the various PINs. If the exit is not installed, the parameters are set to the default values.
- 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.

The ADASMXIT load module must be located so that it can be loaded by the nucleus, either in the load library concatenation or in a system call library such as the MVS system link list. If you are running either ADASMP or Adaplex+ on OS/390 MVS, the ADASMXIT module must be placed in an authorized load library.

The PIN routine user exit is written in Assembler.

User Exit Inputs

The exit is entered with the following registers set:

R13	Adabas PIN routine save area
R14	Return address / AMODE
R15	Entry point address
R0	Function code:
	1 - nucleus initialization
	2 - abend
	4 - response code
	5 - nucleus termination
R1	Parameter list
	+0 address of two user words
	+4 address of condition description block (CDB) for functions 2, 4

User Exit Outputs

There are no outputs. Return codes are ignored and all registers other than 15 must be returned unchanged.

Condition Description Block

For each program check, abnormal termination, or response code error, a control block called the condition description block (CDB) is generated that describes the event that occurred, where it occurred, and what the registers and machine state were when it occurred. The CDB is passed to the error handling and message buffering facility for use in determining whether a PIN routine is to be called or whether an Adabas user exit is to be terminated. A PIN routine uses the CDB to obtain information about the occurrence of the condition.

Modifying and Reloading the Exit

The PIN routine user exit may be modified, reassembled, and reloaded with the nucleus active. To load a newly reassembled exit, issue the console operator command

SMGT,XD=SX*nn*	to deactivate the PIN routine user exit
SMGT,XLOAD=SX*nn*	to load the modified version of the exit
SMGT,XA=SX*nn*	to activate the exit

See the Adabas Online System documentation for another way to accomplish this task.

Using the Exit with PINAUTOR

If PINAUTOR is enabled

- without the PIN routine user exit, the maximum number of files that can be excluded from autorestart is 10 (the default) and all files except the checkpoint and security (system) files are eligible for exclusion. All response codes are eligible for exclusion except those that Adabas disallows as a general rule.
- with the PIN routine user exit, you can modify its AUTOPARM to change the maximum number of files that can be excluded from autorestart and prevent specific files and/or response codes from ever being excluded.

AUTOPARM Example

```
AUTOPARM DS  0D
MAXARPIN DC  F’6’       Maximum of 6 files can be excluded from autorestart
BADRSPS  DC  XL1’48’    Response code 72 cannot be excluded from autorestart
        DC  XL29’00’    28 more entries are possible
NOTFILE DC  XL2’0041’   File number 65 cannot be excluded from autorestart
        DC  XL48’00’    23 more entries are possible
```

Using the Exit with PINRSP

If PINRSP is enabled

- without the PIN routine user exit, all response codes are monitored with no specific subcode checking. Each response code is monitored a maximum of ten times. The ADAN77 message is not suppressed.
- with the PIN routine user exit, you can modify the response code to indicate the specific response codes and subcodes that are to be monitored and the maximum number of times each response code is to be monitored. You can set ‘N77MSG’ to YES in conjunction with response code 201/202/203 monitoring to suppress message ADAN77.

Response Table Entry Example

Note:
There is one entry per response code up to 255.

```
XL5’000A000000’
.
.... subcodes (up to 3; specified in hexadecimal)
.
.... maximum number of times to invoke PINRSP for response code (default=10)
.
.... 00 don’t log, 01 log
```

Example:

The ninth entry in the table corresponds to response code 9.

The entry X’0105020311’ indicates that response code 9, subcodes 2, 3, and 16 are logged. Response code 9 is logged a maximum of 5 times.