IBM IMS Recovery Solution Pack for z/OS
Version 1  Release 1

IMS Database Recovery Facility:
Extended Functions User's Guide
This edition replaces SC19-2904-01.

This edition applies to Version 1 Release 1 of IBM IMS Recovery Solution Pack for z/OS: IMS Database Recovery Facility: Extended Functions (program number 5655-V86) and subsequent releases and modifications, unless otherwise indicated in new editions.

© Copyright IBM Corporation 2007, 2011.
US Government Users Restricted Rights – Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.
Contents

About this information ........................................ v
Service updates and support information ................. v
Highlighting conventions ........................................ v
How to look up message explanations ....................... vi
  Searching an information center ......................... vi
  Using a Web search ........................................ vi
  Using LookAt ................................................ vi
How to send your comments .................................. vi

Part 1. Extended Functions overview .......................... 1

Chapter 1. Extended Functions overview ..................... 3
What does Extended Functions do? ........................ 3
Backup and recovery solutions ............................. 3
Hardware and software prerequisites ..................... 4
Extended Functions features and benefits ................ 5
Extended Functions components ........................... 6
Extended Functions architecture and process flow ....... 7
Extended Functions terminology ............................ 9
Extended Functions publications and updates .......... 11
Accessibility features ....................................... 11
Summary of changes ....................................... 12

Chapter 2. Scenarios - Using Extended Functions .......... 13
HCHECK scenarios ........................................... 13
RPCR scenarios ............................................. 14
RPID scenarios ............................................. 15
VERIFY scenarios .......................................... 16
RCU scenarios .............................................. 16

Part 2. Configuring Extended Functions ..................... 19

Chapter 3. Extended Functions configuration .............. 21
Creating a shared Extended Functions configuration member ........................................ 21
  Rules for Extended Functions PROCLIB members .... 21
Configuration parameters .................................. 22
Supporting functions ........................................ 24
  DBLIST processing ........................................ 24
  RECONCOPY processing .................................. 27
  RECONID locate processing .............................. 29
TIME and RANGE parameter processing ................. 30
  TIME parameter ......................................... 30
  RANGE parameter ....................................... 31
Extended Functions JCL reference ....................... 32
  Extended Functions sample summary report ........... 34

Part 3. Using the Extended Functions ....................... 39

Chapter 4. IMS Command (IMSCMD) function ............... 41
IMSCMD overview .......................................... 41
IMSCMD input ............................................. 41
IMSCMD usage examples .................................. 42
IMSCMD output .......................................... 42

Chapter 5. Health Checker (HCHECK) function ............. 45
HCHECK overview .......................................... 45
HCHECK input ............................................. 45
HCHECK usage examples .................................. 51
HCHECK output .......................................... 52

Chapter 6. Recovery Point Creation (RPCR) function ...... 55
RPCR overview ............................................. 55
RPCR input ................................................. 56
RPCR usage examples ..................................... 58
RPCR output ............................................. 58

Chapter 7. Recovery Point Identification (RPID) function ..... 65
RPID overview ............................................. 65
RPID input ................................................. 66
RPID usage examples ..................................... 67
RPID output ............................................. 67

Chapter 8. Verify Recovery Assets (VERIFY) function ...... 73
VERIFY overview .......................................... 73
VERIFY input ............................................. 73
VERIFY usage example .................................... 75
VERIFY output .......................................... 75

Chapter 9. RECON cleanup (RCU) function ................... 79
RCU overview ............................................. 79
Preparing for site recovery ................................ 80
Clean up time ............................................. 81
RECON records that are updated ......................... 81
Special considerations for using RCU ................... 83
RCU input ................................................. 84
RCU usage example ..................................... 84
RCU output ............................................. 85
RCU return and reason codes ............................ 89

Part 4. IMS Database Recovery Facility job generation .... 91
About this information

IBM IMS Recovery Solution Pack for z/OS: IMS Database Recovery Facility: Extended Functions (also referred to as IMS Database Recovery Facility: Extended Functions or Extended Functions) is a tool that you can use to help ensure that your database environment is recoverable before you have to do a recovery.

This user’s guide provides instructions for using IMS Database Recovery Facility: Extended Functions.

To use the procedures in this user’s guide, you must have already installed IMS Database Recovery Facility: Extended Functions by completing the SMP/E installation process that is documented in the Program Directory for IMS Recovery Solution Pack for z/OS, XXXX-XXXX-XX, which is included with the product. You must also perform the post-installation steps as outlined in the IMS Recovery Solution Pack: Overview and Installation User’s Guide, XXXX-XXXX-XX.

This user’s guide is designed to help database administrators, system programmers, application programmers, and system operators perform the following tasks:

• Plan for the installation of Extended Functions
• Operate Extended Functions
• Configure your Extended Functions environment
• Diagnose and recover from Extended Functions problems

Before using this book, you should understand basic IMS™ concepts.

Specific changes since the previous edition of this guide are indicated by a vertical bar (|) to the left of a change. Editorial changes that have no technical significance are not noted.

Always check the DB2® and IMS Tools Product page for the most current version of this publication:

http://www.ibm.com/software/data/db2imstools/imstools-library.html

Service updates and support information

To find service updates and support information, including software fix packs, PTFs, Frequently Asked Question (FAQs), technical notes, troubleshooting information, and downloads, refer to the following Web page:


Highlighting conventions

This book uses the following highlighting conventions:

• Boldface type indicates commands or user interface controls such as names of fields, folder, icons, or menu choices.
• Monospace type indicates examples of text that you enter exactly as shown.
• *Italic type* indicates variables that you should replace with a value. It is also used to indicate book titles and to emphasize significant words.

## How to look up message explanations

You can use any of the following methods to search for messages and codes:

### Searching an information center

In the search box that is located in the top left toolbar of any Eclipse help system, such as the [IBM® Information Management Software for z/OS® Solutions Information Center](http://www.ibm.com/eserver/zseries/zos/bkserv/lookat/), enter the number of the message that you want to locate. For example, you can enter DFS1065A in the search field.

Use the following tips to help you improve your message searches:

- You can search for information on codes by entering the code; for example, enter -327.
- Enter the complete or partial message number. You can use wild cards (*) or (?) in the message number to broaden your search; for example, DFS20??I.

The information center contains the latest message information for all of the information management products that are included in the information center.

### Using a Web search

You can use any of the popular search engines that are available on the Web to search for message explanations. When you type the specific message number or code into the search engine, you will be presented with links to the message information in IBM information centers.

### Using LookAt

LookAt is an online facility that you can use to look up explanations for most of the IBM messages you encounter, as well as for some system abends and codes. Using LookAt to find information is faster than a conventional search because in most cases LookAt goes directly to the message explanation.

You can use LookAt from the following locations to find IBM message explanations for z/OS elements and features, z/VM®, VSE/ESA, and Clusters for AIX® and Linux:

- Your z/OS TSO/E host system. You can install code on your z/OS or z/OSe systems to access IBM message explanations, using LookAt from a TSO/E command line (for example, TSO/E prompt, ISPF, or z/OS UNIX System Services running OMVS).
- Your Microsoft Windows workstation. You can install code to access IBM message explanations on the z/OS Collection (SK3T-4271) using LookAt from a Microsoft Windows command prompt (also known as the DOS command line).
- Your wireless handheld device. You can use the LookAt Mobile Edition with a handheld device that has wireless access and an Internet browser (for example, Internet Explorer for Pocket PCs, Blazer, or Eudora for Palm OS, or Opera for Linux handheld devices). Link to the LookAt Mobile Edition from the LookAt Web site.
You can obtain code to install LookAt on your host system or Microsoft Windows workstation from a disk on your z/OS Collection (SK3T-4271) or from the LookAt Web site (click Download, and select the platform, release, collection, and location that suit your needs). More information is available in the LOOKAT.ME files available during the download process.

---

### How to send your comments

Your feedback is important in helping to provide the most accurate and high-quality information. If you have any comments about this book or any other IMS Database Recovery Facility: Extended Functions documentation, use either of the following options:

- Use the online reader comment form, which is located at:
  

- Send your comments by e-mail to comments@us.ibm.com. Be sure to include the name of the book, the part number of the book, the version of IMS Database Recovery Facility: Extended Functions, and, if applicable, the specific location of the text you are commenting on (for example, a page number or table number).
Part 1. Extended Functions overview

IMS Database Recovery Facility: Extended Functions inspects information that is stored in the RECON data sets, system catalogs, and in the repositories of IBM IMS Tools Knowledge Base. Extended Functions detects problems that can affect database recovery and will ensure that your IMS database environment is recoverable.

Topics:
• Extended Functions Overview
• Scenarios - Using Extended Functions
Chapter 1. Extended Functions overview

Extended Functions helps you to perform database recoveries.

Extended Functions helps you ensure that your IMS database environment is recoverable, before you ever have to do recoveries.

Topics:
- “What does Extended Functions do?”
- “Backup and recovery solutions”
- “Hardware and software prerequisites” on page 4
- “Extended Functions features and benefits” on page 5
- “Extended Functions components” on page 6
- “Extended Functions architecture and process flow” on page 7
- “Extended Functions terminology” on page 9
- “Extended Functions publications and updates” on page 11
- “Accessibility features” on page 11

What does Extended Functions do?

IMS Database Recovery Facility: Extended Functions is useful at both the local and disaster recovery site for identifying problems that affect your database recovery. At the disaster recovery site, this tool also assists you in ensuring that all of the required resources are present and helps you prepare the RECON data sets to perform recoveries.

Extended Functions inspects information that is stored in the RECON data sets, system catalogs, and in the repositories of IBM IMS Tools Base for z/OS: IMS Knowledge Base.

Extended Functions detects problems that can affect database recovery. This tool provides you the ability to create clean recovery points across multiple IMS systems and to perform synchronized log switches.

Extended Functions also offers features to determine valid database recovery points, makes sure that all of the resources that are necessary to perform a recovery are present, and provides you the ability to manage the RECON data sets to ensure that the data sets match the disaster recovery environment.

Backup and recovery solutions

IMS Database Recovery Facility: Extended Functions is just one of several IMS tools that provide enhancements to the process of managing backup and recovery operations for your databases in the event of a system outage or application failure.

For example, IBM IMS Recovery Solution Pack for z/OS: IMS Database Recovery Facility (5655-V86) is a product that works with IMS Database Recovery Facility: Extended Functions to simplify your database recovery process by eliminating the need to run a separate recovery job for each database that requires recovery.
In addition, IMS Database Recovery Facility provides the ability to interface automatically with the following IMS tools and utilities during the recovery process:

- IMS Tools Online System Interface
- IMS Index Builder (the IMS DFSPREC0 utility)
- IMS High Performance Image Copy (5655-N45) tool
- DEDB Pointer Checker (from IBM IMS High Performance Fast Path Utilities for z/OS (5655-W14), IMS Basic Fast Path Tools) utility

Other DB2 and IMS tools that can assist with database recovery include:

- IBM IMS Database Control Suite for z/OS V3 (5655-L08)
- IBM IMS DEDB Fast Recovery for z/OS (5655-E32)
- IBM IMS Recovery Solution Pack for z/OS: IMS High Performance Change Accumulation Utility V1 (5655-V86)
- IBM IMS High Performance Image Copy for z/OS V4 (5655-N45)
- IBM Application Recovery Tool for IMS and DB2 Databases V1 (5697-F56)

**Hardware and software prerequisites**

IMS Database Recovery Facility: Extended Functions is installed by using SMP/E and standard RECEIVE, APPLY, and ACCEPT processing.

Complete information about installation requirements, prerequisites, and procedures for IMS Recovery Solution Pack for z/OS is located in the Program Directory for IMS Recovery Solution Pack for z/OS, GI10-8824-00.

**Hardware prerequisites**

Extended Functions will run on any S/390® hardware configuration that supports the required version of IMS: IMS V9.1 (5655-J38), IMS V10.1 (5635-A01), and IMS V11.1 (5635-A02).

**Installation software prerequisites**

To install Extended Functions the following software is required:

- 5694-A01 z/OS, V1.7 or later
- 5655-G44 SMP/E for z/OS, V3.4 or later

**Operational software prerequisites**

These topics describe the software that is required to run IMS Database Recovery Facility: Extended Functions.

Extended Functions is designed to run with IMS V9.1 (5655-J38), IMS V10.1 (5635-A01), and IMS V11.1 (5635-A02), and will run on any S/390 software configuration that supports the required version of IMS.

Extended Functions requires that the IMS Tools Base for z/OS be installed and the following components be installed and configured:

- IMS Tools Knowledge Base
- IMS Tools Online System Interface
- IMS Tools Generic exits
Extended Functions features and benefits

By using IMS Database Recovery Facility: Extended Functions, you can save time and resources by detecting and correcting problems that can affect database recoveries before you need to recover your databases.

Extended Functions helps you perform database recoveries at the local and disaster recovery site by assisting you in many aspects of IMS database recovery tasks by providing the following features:

- IMS Command Interface
- Health Checker
- Recovery Point Creation
- Recovery Point Identification
- Verify Recovery Assets
- RECON Data Set Clean up
- IMS Database Recovery Facility JCL Generation

The tool also provides a process for creating a copy of your live RECON data sets. The copied data sets are accessed instead of your live RECON data sets, which eliminates I/O against your live RECON data sets.

IMS Command interface

The IMS command interface (IMSCMD) allows you to execute IMS type 1 AOI commands in a batch environment. IMSCMD uses the IMS Tools Online System Interface for entering commands and must first be installed in each IMS control region in which commands are issued. IMSCMD can be used in conjunction with other functions of the tool. The output of the commands is written to a report for viewing.

Health Checker

The Health Checker function (HCHECK) analyzes information in the RECON data sets and identifies problems which could impact IMS database recovery. HCHECK currently checks for over 20 different conditions which could impact IMS database recovery. Health Check uses information from the RECON data sets and the z/OS catalog when looking for potential problems.

Recovery Point Creation

The Recovery Point Creation function (RPCR) provides the ability to create recovery points for one or more databases by deallocating the database using /DBR or /DBD command. This process causes the allocation records in the RECON data sets to be updated, indicating a period of time when the database is not allocated by any IMS subsystem, thus creating a valid recovery point. When complete, RPCR also optionally restarts the databases to bring them back into their original state. RPCR also has the ability to issue a /SWI LOG command across connected IMS systems to create archived logs that are associated with the RPCR time.

Recovery Point Identification

The Recovery Point Identification function (RPID) analyzes records in the RECON data sets and locates valid recovery time spans for one or more databases. A
Recovery Time Span (RTS) is a period of time within which a database is not allocated and may be recovered. RPID analyzes database allocation records in the RECON data sets to identify RTSs to which one or more databases can be recovered using timestamp recovery (TSR).

RPID also analyzes database recovery records in the RECONs to identify Recovery Point timestamps to which individual databases can be recovered using TSR. These recovery points include times when a database was the object of a batch IC run, online IC run (in cases where the databases are not truly allocated), a complete CA run, a recovery run (full or partial) or an offline reorganization run; all of which are recorded in RECON recovery records. Run times of HALDB online reorganizations and IMS Online Reorganization Facility are also analyzed; for these RPID indicates whether the databases were truly allocated and thus whether the run time is a valid TSR recovery point.

**Verify Recovery Assets**

The Verify Recovery Assets function (VERIFY) examines the RECON data sets and identifies the recovery assets that are required to recover one or more databases. VERIFY ensures that the necessary assets exist and that they are valid before performing a recovery. Specifically, the VERIFY function performs one or all of the following functions:

- Lists the recovery assets that are required to recover the specified databases
- Allocates all recovery assets in order to ensure that they exist
- Opens all recovery assets to ensure that they are valid

**RECON Data Set Clean up**

The RECON Data Set Clean up function (RCU) is used to prepare a set of RECON data sets for your disaster recovery environment. RCU uses a copy of your RECON data sets and alters them to match the recovery environment at the disaster recovery site. These RECONs can then be used for a disaster recovery test or in the event of an actual disaster.

**IMS Database Recovery Facility JCL Generation**

The IMS Database Recovery Facility JCL generation function allows you to easily create and run IMS Database Recovery Facility JCL for performing IMS database recoveries and creating incremental image copies. This function parses the information from the Recovery Point Identification report and presents it in an easy to use ISPF interface.

**Extended Functions components**

IMS Database Recovery Facility: Extended Functions consists of several components that work together to provide the product functions.

The main components of Extended Functions include:

**BPE architecture**

The Extended Functions product is built on the Base Primitive Environment (BPE) architecture. BPE provides most of the underlying address space services such as dispatcher services, tracing, diagnostics, and storage services. The other components use these services when performing their tasks.
The main address space
The main address space controls job step initialization and setup, drives the requested functions, and performs address space clean up and termination. This component parses all input and prepares the environment for processing the specified functions. The main address space drives each function by performing the following steps:
- Parse all function control cards and initialize the function environment
- Invoke the function and allow it to process
- Check for successful or abnormal completion
- Clean up the function environment
- Terminate the function

Functions
Each function is processed by a separate set of execution routines. These functions are driven by the main address space and process the requested action, as specified by the control card input. Each function generates its own report and error messages.

Report Writer
The report writer provides an interface for writing reports to the IMS Tools Base for z/OS: IMS Tools Knowledge Base repository, to report data sets, or to SYSOUT. Other components make requests of the report writer to generate report output and based on JCL and control card specifications, the report writer writes data to all necessary locations.

ISPF Interface
The Extended Functions ISPF interface allows you to view the data from the Recovery Point Identification (RPID) report and generate IMS Database Recovery Facility JCL for performing database recoveries and creating incremental image copies.

Extended Functions architecture and process flow
IMS Database Recovery Facility: Extended Functions operates as a batch job in a single address space. The job can be set up to run in a job scheduler and perform required functions on a regular basis. A single Extended Functions job can consist of multiple job steps. Each job step can process a separate function or a single job step can process multiple functions.

Within a single job step, the order in which the functions are processed is controlled by the tool. The functions are processed consecutively in a predefined order. No parallel processing of functions is performed. The order in which functions are processed is:
- RECON cleanup
- Recovery Point Create
- Recovery Point Identification
- Verify
- Health Checker
- IMS Command

If you specify a function multiple times in the same job step, all occurrences of that function are processed consecutively before the next function is attempted.

If you want to control the order in which the functions are processed, you must use multiple job steps. For example, within a single job, STEP1 might issue IMS
commands that display and stop certain IMS resources. STEP2 might then execute the RRPCR process to create a clean recovery point for one or more databases. STEP3 might then issue IMS commands to restart those resources.

Extended Functions uses the IMS Tools Base for z/OS: IMS Tools Knowledge Base or repository for the following purposes:

- Output from reports that are generated by Extended Functions can be stored in an output repository. IMS Tools Base for z/OS: IMS Tools Knowledge Base has an interface to store and retrieve output reports. You can view these reports at a later time, using the IMS Tools Base for z/OS: IMS Tools Knowledge Base ISPF interface.
- You can also process your RPID reports with the RPID ISPF interface.
- IMS Tools Base for z/OS: IMS Tools Knowledge Base is used to store RECONID records that describe the IMS environments against which Extended Functions can process.

You control Extended Functions processing by specifying input parameters. You can specify the following 2 types of input parameters:

**Configuration parameters**
These parameters specify overall job configuration settings. These parameters are not directly related to any one feature but they control address space setup and processing.

**Function control cards**
The function control cards drive the processing that is performed by Extended Functions for this run. These control cards specify the functions that are performed and the information that controls their processing.

The following diagram illustrates the general process flow for a Extended Functions environment that includes input sources, the BPE address space where the functions process, and the other components with which the Extended Functions interacts.
**Extended Functions terminology**

IMS Database Recovery Facility: Extended Functions includes several unique terms that you should understand before you begin to use the tool.

**Change Accumulation (CA) data set**
Data set created by a database Change Accumulation utility run that contains all of the database updates between the last Image Copy and the time the CA utility completed execution. There are two basic kinds of CA data sets; complete CA data sets (based on available system log data sets (SLDS)) and incomplete CA data sets that are created at the time that CA utility was generated and when the required SLDS are not yet available.

**IMS Database Recovery Facility**
IMS database tool that can recover databases using Full Recovery, Time Stamp Recovery, or Point In Time Recovery.

**Disaster recovery site (DR)**
Location at which data sets necessary to recover IMS databases are located in the event of a disastrous IMS outage such as natural calamities, fires, power failures, or other disasters.

**Full recovery**
Recovery of a database using the most current information available: image copy (IC), CA, Logs.

**Recovery health check (HCHECK)**
This feature consists of many different options to check and verify the recoverability of your environment. The majority of these options
interrogate and verify information in the RECON data sets, and compare
the information in the RECONs with information that is outside of the
RECONs (like information that is in the system catalog and in other
places). You can specify each piece of information that you want checked.
And you can exclude the pieces of information that you do not want
checked.

IMS command (IMSCMD)
This feature allow you to issue IMS commands in batch. You can view the
output of the commands that is returned in a report file.

Point-in-time recovery (PITR)
Timestamp input to which a database can be restored. The database was
typically allocated and being accessed at the selected recovery timestamp.

Recovery point create (RPCR)
This feature allows you to create a recovery point for one or more
databases by issuing synchronized /DBRECOVERY or /DBDUMP
commands for the databases, waiting for the databases to be deallocated
on all of your IMS systems, and then optionally restarting them. You can
also optionally perform a synchronized log switch on all active IMS
systems in the group with RPCR.

RECON cleanup (RCU)
This feature allows you to prepare a set of RECON data sets for your
disaster recovery environment by modifying the contents of the RECONs
to match their DR site.

Recovery point
Timestamp that can be used as input to database recovery utilities such as
IMS Database Recovery Facility or the standard IMS Recovery utility
DFSURDB0, as a time to which a database can be restored.

Recovery point identification (RPID)
This feature interrogates information in the RECON data sets to determine
common recovery points for one or more databases.

Recovery time range (RTR)
RPID input parameter containing two times within which the search for
recovery time spans and recovery points is bounded.

Recovery time span (RTS)
Span of time within which recovery points can be selected and to which a
database or group of databases can be recovered.

Timestamp recovery (TSR)
Timestamp input to either IMS Database Recovery Facility or another
standard database recovery program such as DFSURDB0. The database is
typically offline at this timestamp. In the RPID documentation, TSR refers
to non-PITR timestamps, though PITR is actually a special case of TSR.

Verify recovery assets (VERIFY)
This feature interrogates the RECON data sets and first generates a list of
the recovery assets that are needed in order to recover one or more
databases. When this list is generated, this feature provides options to
perform the following tasks
• Print the list
• Allocate each asset to ensure that it exists
• Open each recovery asset that exists to ensure that it is valid
Extended Functions publications and updates

This topic explains where to find DB2 and IMS Tools information on the Web, and explains how to receive information updates automatically.

Extended Functions information on the Web

The IMS Tools Library Web page provides current product documentation that you can view, print, and download. To locate publications with the most up-to-date information, refer to the following Web page:

http://www.ibm.com/software/data/db2imstools/library.html

You can also access documentation for many DB2 and IMS Tools for z/OS from the Information Management Software for z/OS Solutions Information Center:

http://publib.boulder.ibm.com/infocenter/imzic

IBM Redbooks® publications that cover DB2 and IMS Tools are available from the following Web page:

http://www.ibm.com/software/data/db2imstools/support.html

The Data Management Tools Solutions Web site shows how IBM solutions can help IT organizations maximize their investment in DB2 and IMS databases while staying ahead of today's top data management challenges:


Receiving updates automatically

To automatically receive a weekly email that notifies you when new DCF documents are released, when existing product documentation is updated, and when new product documentation is available, you can register with the IBM My Support service. You can customize the service so that you receive information about only those IBM products that you specify.

To register with the My Support service:

1. Go to http://www.ibm.com/support/mysupport
2. Enter your IBM ID and password, or create one by clicking register now.
3. When the My Support page is displayed, click add products to select those products that you want to receive information updates about. The DB2 and IMS Tools category is located under Software → Data and Information Management → Database Tools & Utilities.
4. Click Subscribe to email to specify the types of updates that you would like to receive.
5. Click Update to save your profile.

Accessibility features

Accessibility features help a user who has a physical disability, such as restricted mobility or limited vision, to use a software product successfully.

The major accessibility features in IMS Database Recovery Facility: Extended Functions enable users to:
• Use assistive technologies such as screen readers and screen magnifier software. Consult the assistive technology documentation for specific information when using it to access z/OS interfaces.
• Customize display attributes such as color, contrast, and font size.
• Operate specific or equivalent features by using only the keyboard. Refer to the following publications for information about accessing ISPF interfaces:
  – z/OS ISPF User’s Guide, Volume 1
  – z/OS TSO/E Primer
  – z/OS TSO/E User’s Guide

These guides describe how to use ISPF, including the use of keyboard shortcuts or function keys (PF keys), include the default settings for the PF keys, and explain how to modify their functions.

Summary of changes

This topic summarizes the technical changes for this edition.

New and changed information is indicated by a vertical bar (|) to the left of a change. Editorial changes that have no technical significance are not noted.

### SC19-2904-02
- Added a new topic that documents the return and reason codes issued by the RECON Cleanup (RCU) function on the IRO0102E message. To view this revised topic, see “RCU return and reason codes” on page 89.

### SC19-2904-01
- Removed the 'D' abbreviation from the list of accepted HCHECK(DBNOTCA) abbreviations. To view this revised topic, see “HCHECK input” on page 45.
Chapter 2. Scenarios - Using Extended Functions

The functions of IBM IMS Recovery Solution Pack for z/OS: IMS Database Recovery Facility: Extended Functions can be used in several ways before you need to perform a recovery, to help you to ensure that your database environment is recoverable.

The scenarios are presented by function.

Topics:
- “HCHECK scenarios”
- “RPCR scenarios” on page 14
- “RPID scenarios” on page 15
- “VERIFY scenarios” on page 16
- “RCU scenarios” on page 16

HCHECK scenarios

To interrogate DBRC to determine if a database is recoverable, use the HCHECK function to help you make this determination without manually listing all of the necessary DBRC components.

Some of these scenarios might be applicable to your environment.

Scenario 1: Determine the statuses of all image copies related to a specific group of databases
1. Run the HCHECK function with the ICALL option to perform all of the following tasks.
   a. Check whether any image copies are needed, recommended, or whether no image copies exist for a specific DBDS in the database list.
   b. Generate a report with the image copy status of each DBDS.

Scenario 2: Determine if the databases in your list are available
1. Run the HCHECK function with the BACKOUT option to determine if any of the databases in the list need to be backed out. A report will be generated with the status of each database in the list that requires a backout.
2. Run the HCHECK function with the PROAUTH option to determine if any of the databases in the list have prohibit authorization turned on and therefore, cannot be accessed. A report will be generated with the authorization status of each database in the list.
3. Run the HCHECK function with the HALDBNOTINIT option to determine if any of the HALDB databases in the list are not initialized and therefore, cannot be accessed. A report will be generated with the initialization status of each HALDB partition in the list.
4. Run the HCHECK function with the NOADS option to determine if any of the Fast Path DEDBs in the list are undefined or unavailable. A report will be generated with the availability status of each Fast Path area in the list.
5. Run the HCHECK function with the ICNEED option to determine if any of the databases in the list require an image copy and are therefore unavailable until an image copy is taken. A report will be generated with the ICNEEDED status of each database data set in the list.

RPCR scenarios

The RPCR function can assist you in creating a valid recovery point.

When attempting to create a valid recovery point during which one or more databases are not being updated, perform the following tasks:

- Manually /DBR or /DBD a single database or group of databases
- Manually start the databases after all of the databases were stopped to create a recovery point during which the databases were not allocated to an online subsystem.

However, this manual process is error prone because, not only would you have to manually verify that each IMS command is done correctly in all IMS subsystems, you would also manually have to verify that the databases were not started or allocated before all of the databases in the list are deallocated (both batch and online). This manual processing could be inconvenient and this processing can cause the databases to remain offline for some amount of time.

The IMS Database Recovery Facility: Extended Functions RPCR function can help you create a valid recovery point easily, without manually invoking the IMS commands and simultaneously verify that the DBDSs were not reallocated by any other subsystem or batch job during the deallocation process. RPCR will also pause any BMPs that have the databases allocated to allow the deallocation of all of the databases in the list to occur simultaneously. Here are some scenarios that might be applicable to your environment:

Scenario 1: Determine if a large group of allocated databases can be deallocated

In this scenario, you deallocate a large group of databases that are allocated and updated by several online subsystems to create a valid recovery point.

1. Run the RPCR function with PARTIAL(N) to perform all of the following tasks:
   a. Quickly pause any BMPs that have reached a checkpoint or prevent any BMPs that have not yet started during the process.
   b. Successfully deallocate all of the databases in the list for all of the IMS subsystems that have the databases allocated.
   c. Verify that no other jobs or subsystems have accessed the databases during the time it took for the RPCR deallocation to complete.
   d. Immediately restart the databases in their appropriate online systems with little or no impact to the online systems.
   e. Generate a report with the timestamp of the last database that was deallocated by RPCR. This timestamp can be used as a common recovery point for all of the databases in the list. If any of the databases in the list could not be deallocated, the entire process fails. The report indicates which databases caused the process to fail.
Scenario 2: Determine if a small group of unrelated databases can be deallocated

In this scenario, you deallocate a small group of databases that are unrelated to each other.

1. Run the RPCR function with the PARTIAL(Y) option to quickly deallocate the databases in the list for all IMS subsystems that have the databases allocated. A report will be generated with the timestamp that the last database was successfully deallocated by RPCR. If any of the databases in the list could not be deallocated, the report will indicate which databases failed and which databases were successful.

RPID scenarios

To check whether a single database or group of databases can be recovered to a common recovery point can be a complicated and timely process.

The IMS Database Recovery Facility: Extended Functions RPID function can help you to determine whether databases can be recovered to a common recovery point easily without manually listing and comparing all of the DBRC history for each database in the list.

These scenarios might be applicable to your environment:

Scenario 1: Determine a common recovery point for several databases within an application

In this scenario, you determine a common recovery point for several databases within an application for a specific time period.

1. Run the RPID function with a RANGE to perform all of the following tasks:
   a. Determine when all of the databases in the list were deallocated at a common point in time.
   b. Generate a report with the individual database recovery points and a list of common recovery points for all of the databases in the list. Any common timestamp within the recovery times that are listed in the report can be used as a valid recovery point for all databases.

Scenario 2: Determine a recovery point for a single database for a specific time period

In this scenario, you determine if all data set assets that are needed to recover a database to a specific timestamp (TSR) can be allocated or opened successfully.

1. Run the RPID function with a RANGE to perform all of the following tasks:
   a. Determine when the database in the list was deallocated.
   b. Generate a report with the individual database recovery points; common recovery points will be identical because just one database was specified. Any timestamp within the recovery times that are listed in the report can be used as a valid recovery point for that database.
**VERIFY scenarios**

You have many methods to choose from to verify assets before a recovery of one or more databases. However, the VERIFY function can help you verify assets before a recovery with one step.

The following scenarios might be applicable to your environment.

**Scenario 1: Determine if all assets necessary for a current time DB recovery can be allocated or opened**

In this scenario, you determine if all of the data set assets that are needed to recover a database to the current time can be allocated or opened successfully.

1. Run the VERIFY function with TYPE(OPEN) to perform all of the following tasks:
   a. List, allocate, and open all of the assets that are needed to recover a database to the current time.
   b. Generate a report with the status of each asset that is necessary for the recovery.

**Scenario 2: Determine if all necessary assets for a TSR can be allocated or opened**

In this scenario, you determine if all data set assets that are needed to recover a database to a specific timestamp (TSR) can be allocated or opened successfully.

1. Run the VERIFY function with PARTIAL(Y) to perform all of the following tasks:
   a. Quickly deallocate the databases in the list for all IMS subsystems that have the databases allocated using the RPCR function.
   b. Generate a VERIFY report with the timestamp that the last database was successfully deallocated by RPCR. If any of the databases in the list could not be deallocated, the RPCR report will indicate which databases failed and which databases were successful.

**Scenario 3: Determine if all necessary assets for a PITR recovery can be allocated or opened**

In this scenario, you determine if all of the data set assets that are needed to recover a database to a specific point in time (PITR) can be allocated or opened successfully.

1. Run the VERIFY function with TYPE(OPEN), RCVTIME(timestamp), and RCVTYPE(PITR) to perform all of the following tasks:
   a. List, allocate, and open all of the assets that are needed to recover a database to a specific point in time (PITR).
   b. Generate a report with the status of each asset that must be used in the PITR.

**RCU scenarios**

IMS Database Recovery Facility: Extended Functions can help you determine a valid cleanup time with several functions that are available.

The following scenarios might be applicable to your environment.
Scenario 1: Databases can remain offline during image copies

Perform the steps in the following list to perform an RCU when your databases can remain offline during image copies.

1. Determine the time period that you are going to schedule the Disaster Recovery (DR) backup for. Schedule the DR backup at a time with the least amount of system activity.
2. Run the RPCR function to issue the /DBR command on your databases. Specify STARTDB(N). The RPCR function process occurs quickly.
3. Create batch image copies for all of the databases that are to be recovered at your DR site.
4. Run the IMSCMD function to issue the /STA DB command on your databases.
5. Run the RCU function with a cleanup time (a timestamp from the last batch image copy).

Scenario 2: Databases cannot remain offline during image copies

Perform the steps in the following list to perform an RCU when your databases cannot remain offline during image copies.

1. Determine the time period that you are going to schedule the DR backup for. Schedule the DR backup at a time with the least amount of activity.
2. Run the RPCR function to issue the /DBR and /STA commands on your databases. The RPCR process occurs quickly and should have minimal impact on your online system.
3. Create concurrent image copies for all of the databases that are to be recovered at your DR site.
4. Run the RPCR function again to issue the /DBR and /STA commands on your databases. This will return a single timestamp that is the time that the last database was /DBR’d. This timestamp that can be used as your cleanup time in the next step.
5. Run the RCU function with a cleanup time that is the timestamp from last RPCR.

Scenario 3: 24 x 7 environment—databases cannot be quickly /DBR’d and /STA’d

Perform the following steps to perform an RCU when your databases cannot be quickly DB recovered (by using the /DBR command) and started (by using the /STA DB command).

1. Determine the time period that you are going to schedule the DR backup for. Schedule the DR backup at a time with the least amount of activity.
2. Create concurrent image copies for all of the databases that are to be recovered at your DR site.
3. Run the RCU function with a cleanup time that is the timestamp from last concurrent image copy stop time.
Part 2. Configuring Extended Functions

You must complete IMS Database Recovery Facility: Extended Functions installation and configure the tool to your environment. Before continuing with this section, make sure that you have performed the SMP/E install as outlined in the Program Directory for IMS Recovery Solution Pack for z/OS, XXXX-XXXX-XX and then performed the post-installation steps as outlined in the IMS Recovery Solution Pack: Overview and Installation User's Guide, XXXX-XXXX-XX.

Topics:
• Configuring IMS Database Recovery Facility: Extended Functions
Chapter 3. Extended Functions configuration

To use IBM IMS Recovery Solution Pack for z/OS: IMS Database Recovery Facility: Extended Functions you must set up JCL, specify configuration parameters, specify supporting functions, and configure several other options.

Topics:
- “Configuration parameters” on page 22
- “Supporting functions” on page 24
- “RECONID locate processing” on page 29
- “TIME and RANGE parameter processing” on page 30
- “Extended Functions JCL reference” on page 32

Creating a shared Extended Functions configuration member

There are certain IMS Database Recovery Facility: Extended Functions configuration parameters which must be set to the same value across all Extended Functions jobs. You can set these parameters either by specifying them individually within the job or, you can define them in a common configuration member where they can be shared.

The one parameter that will be the most helpful to place in this member is the ITKBSRVR() parameter. This parameter identifies the IMS Tools Base for z/OS: IMS Tools Knowledge Base server name (XCF group name) that is associated with your IMS Tools Base for z/OS: IMS Tools Knowledge Base environment. If all Extended Functions jobs will share the same IMS Tools Base for z/OS: IMS Tools Knowledge Base environment, creating this shared PROCLIB member will ensure that all Extended Functions jobs specify the correct name. SAMPLIB member IROCFG01 contains default settings that can be used by all Extended Functions jobs. Tailor member IROCFG01 to specify the correct IMS Tools Base for z/OS: IMS Tools Knowledge Base server name. Specify the Extended Functions configuration PROCLIB member on the EXEC statement by using the IMSRECFG= parameter in the PARM= string.

Rules for Extended Functions PROCLIB members

IMS Database Recovery Facility: Extended Functions can use PROCLIB members in order to share BPE configuration and Extended Functions configuration options among several jobs. This data set should be a standard IMS-type PROCLIB data set. Adhere to these rules when you set up these PROCLIB members.

- The rightmost 8 columns of each line are ignored and can be used for sequence numbers or any other notation that you might need. For example, if your LRECL is 80, columns 73 to 80 are ignored.
- You can add blanks between words for readability, but you cannot add them within words.
- You can code multiple statements on one line.
- Use commas or spaces to delimit keywords.
- Use an asterisk (*) or a pound sign (#) in the first column of a line to identify the line as a comment.
- Comments can also be included between pairs of /* and */ characters.
Configuration parameters

Configuration parameters are used to control the overall execution of IMS Database Recovery Facility: Extended Functions. These parameters are not directly related to an individual function but control the overall Extended Functions environment.

You can specify configuration parameters in one of two places:

- Using the JCL EXEC statement in the PARM= parm specification, you can specify IMSRECFG=member. If you specify this member it is read from the //PROCLIB DD concatenation and can contain your configuration settings.
- In the //IMSREIN DD input along with the function control cards. If you specify a configuration parameter in both the //IMSREIN DD and IMSRECFG=, the value that is specified in the //IMSREIN DD overrides the value that is specified in IMSRECFG=.

Tip: Configuration parameters are processed before any function control statements.

Specify the Extended Functions configuration parameters according to their use and syntax as they are described here:

Tip: The spaces that are shown in the configuration parameter examples are optional.

**ITKBSRVR parameter**

Use this parameter to identify the IMS Tools Base for z/OS: IMS Tools Knowledge Base repository server name that is associated with the IMS Tools Base for z/OS: IMS Tools Knowledge Base environment that is to be used for this run of Extended Functions. It is the 8-character XCF group name that is associated with the IMS Tools Base for z/OS: IMS Tools Knowledge Base server. This parameter is required and has no default value.

Here is the syntax for ITKBSRVR:

```
ITKBSRVR( ITKB-XCF-group )
```

**RECONID parameter**

This parameter identifies the RECONID member that is obtained from the repository in order to retrieve environmental information. This is an 8-character name that is assigned to the RECONID member as defined in the repository. This parameter is optional but, if you do not supply it, the RECON1 data set name must be available either through the JCL (allocated to the RECON1 DD statement), or by the DFSMDA dynamic allocation member in the //STEPLIB or //IMSDALIB.

Here is the syntax for RECONID:

```
RECONID( reconid-member-name )
```

**REPORT parameter**

Use this parameter to specify your preferences for the reports that are created by Extended Functions.
Here is the syntax for REPORT:

```
REPORT( ITKBOUT( YES | Y | NO | N )
       PAGEBR( YES | Y | NO | N )
       TIMEFMT( LOCAL | DATA | UTC )
       DATEFMT( YYYYDDD | MMDDYYYY | DDMYYYY | YYYYMMDD )
    )
```

The REPORT parameters are described here:

**ITKBOUT( YES | Y | NO | N )**

Use this parameter to specify whether the reports that are generated by Extended Functions are placed into the IMS Tools Base for z/OS: IMS Tools Knowledge Base output repository. The default is ITKBOUT(YES).

**PAGEBR( YES | Y | NO | N )**

Use this parameter to specify whether the reports that are generated by Extended Functions include page breaks and headers that make the reports easier to interpret when they are printed. If your reports will be viewed online only, PAGEBR( NO ) is recommended. The default is PAGEBR( NO ).

**TIMEFMT( LOCAL | UTC | DATA )**

Use this parameter to specify whether timestamps that are displayed on your reports are in local time ( LOCAL ), global time ( UTC ), or the time that is associated with the data ( DATA ). For TIMEFMT( DATA ), if the processed data (such as, log records) are from a location other than the current location, the timestamps that are displayed are from the location where the data was created. The default is TIMEFMT( LOCAL ).

**DATEFMT( YYYYDDD | MMDDYYYY | DDMYYYY | YYYYMMDD )**

Use this parameter to specify the format of the date that is displayed on all of your report headings. The YYYY variable represents the value for the year. The variable MM represents the value for the month. The variable DD or DDD represents the value for the day. The default is DATEFMT(MMDDYYYY).

**TEMPDSN parameter**

Use this parameter to control the allocation of temporary data sets that might be needed by internal services. If you do not specify this parameter, the system defaults are used to allocate all necessary temporary data sets.

Here is the syntax for TEMPDSN:

```
TEMPDSN( TUNIT( unit-name )
       TVOLSER( volser )
       TSTORCLASS( SMS-storage-class )
       TMGMTCLASS( SMS-management-class )
       TDATACLASS( SMS-data-class )
    )
```

The TEMPDSN parameters are described here:

**TUNIT( unit-name )**

Use this parameter to specify the DASD unit that is used to allocate temporary data sets.

**TVOLSER( volser )**

Use this parameter to specify the DASD volume serial that is used to allocate temporary data sets.

**TSTORCLASS( SMS-storage-class )**
Supporting functions

The supporting functions, DBLIST and RECONCOPY, provide information and services that are used by other IMS Database Recovery Facility: Extended Functions functions when those Extended Functions are processed. The supporting functions do not function stand-alone, but operate in conjunction with the other functions.

DBLIST processing

Most of the functions that can be processed by IMS Database Recovery Facility: Extended Functions require that you specify the databases for which that function is to be performed. Specify the database list with the DBLIST parameter.

Before Extended Functions invokes any specified function, the database list is pre-processed to validate and expand the parameters into a simple list of databases, database data sets, areas, and partitions that will be processed by the functions.

Database list processing takes the input that is specified by the DBLIST subparameters (as explained in the subsequent topic about target database list processing), and accesses the RECON data sets to resolve the list into individual elements. All of the DBLIST subparameters (such as, DB(), AREA(), and other subparameters) allow wildcard characters, such as, * and %, in the names, following standard IMS rules. Each subparameter is processed, as described in the next subtopic and an entry is created in the list. This process ensures that there are no duplicate entries in the list.

Use the DBLIST parameter to identify the databases that Extended Functions is to process. The DBLIST parameter is used only in conjunction with other functions, it does not operate alone.

The Extended Functions that require the use of the DBLIST parameter are as follows:

- HCHECK (Recovery health checker)
- RPCR (Recovery point create)
- RPID (Recovery point identification)
- VERIFY (Verify recovery assets)

You can specify the DBLIST parameter only once per job step. All of the functions that are specified in the job step will process the same set of databases, as specified by DBLIST().

Extended Functions accesses the RECON data sets, validates the DBLIST, expands the DBLIST as needed, and passes the individual databases to the functions. You can specify the databases using the following subparameters:

- A single or multiple database (DB)
- A single or multiple database and database data set (DBDS)
- A single or multiple fast path DEDB and AREA pair (AREA)
A single or multiple HALDB master and partition pair (PART)
A single or multiple RECON DBDS group (DBDSGRP)
A single or multiple RECON recovery group (RECOVGRP)
A single or multiple RECON CA group (CAGRP)

All of the values that can be specified, as shown in the next example allow the standard use of wildcard characters (for example, * and %).

Here is the syntax for DBLIST:

```
DBLIST(
  DB( dbname, dbname, ... )
  DBDS( ( dbname,dbdsname),(dbname,dbdsname),... )
  AREA( ( dbname,areaname),(dbname,areaname),... )
  PART( ( dbname,partname),(dbname,partname),... )
  DBDSGRP( recon-dbds-group,... )
  CAGRP( recon-ca-group,... )
  RECOVGRP( recon-recovery-group,... )
)
```

The DBLIST parameters are described here:

**DB(dbname,dbname,...)**
Use this parameter to specify one or more full function, fast path, or HALDB master databases. You cannot specify HALDB partition names. You can specify multiple names separated by commas or blanks.

**Example of DBLIST(DB()):**
```
DBLIST(DB((
  PARTSDB,PAYROLL,ACCT%)
))
```

**DBDS((dbname,dbdsname),(dbname,dbdsname),...)**
Use this subparameter to specify one or more pairs of database names (DBD) followed by the database data set name (DD name). You can specify only full function and fast path databases. You can specify one or more pairs of databases but, you must specify them in pairs. You must specify the pairs within single parenthesis, separated by commas or blanks. You cannot specify HALDB master and partition names in this field.

**Example of DBLIST(DBDS()):**
```
DBLIST(DBDS((
  DBHDOJ01,DJOHDG1O),(
  ACCTA,*
)))
```

**AREA( dbname,areaname),(dbname,areaname),...**
Use this subparameter to specify one or more pairs of fast path database names (DEDB) followed by the area name. You can specify one or more pairs but, you must specify them in pairs. You must specify the pairs within single parenthesis, separated by commas or blanks.

**Example of DBLIST(AREA()):**
```
DBLIST(AREA((
  DEDBD01,DD01AR0)
))
```

**PART( dbname,partname),(dbname,partname),...**
Use this subparameter to specify one or more pairs of HALDB master database names followed by the partition name. You can specify one or more pairs but, you must specify them in pairs. You must specify the pairs within single parenthesis, separated by commas or blanks.

**Example of DBLIST(PART()):**
```
DBLIST(PART((
  MASTER4,PART41),(MASTER4,PART43),( MASTER7,*)))
```
DBDSGRP(\textit{recon-dbds-group,...})

Use this subparameter to specify one or more RECON DBDS groups. You can specify multiple RECON DBDS group names separated by commas or blanks.

Example of DBLIST(DBDSGRP(\textit{}))

\texttt{D\textbf{B}LIST(D\textbf{B}\textbf{D}\textbf{S}\textbf{G}\textbf{R}\textbf{P}(\textit{DBDSGRP1, DBDSGRP2}))}

CAGRP(\textit{recon-ca-group,...})

Use this subparameter to specify one or more RECON change accumulation groups. You can specify multiple RECON change accumulation group names separated by commas or blanks.

Example of DBLIST(CAGRP(\textit{}))

\texttt{D\textbf{B}LIST(C\textbf{A}\textbf{G}\textbf{R}\textbf{P}(\textit{CAGRP01, CAGRP02}))}

RECOVGRP(\textit{recon-recovery-group,...})

Use this subparameter to specify one or more RECON recovery groups. You can specify multiple RECON recovery group names separated by commas or blanks.

Example of DBLIST(RECOVGRP(\textit{}))

\texttt{D\textbf{B}LIST(RE\textbf{C}\textbf{O}\textbf{V}\textbf{G}\textbf{R}\textbf{P}(\textit{RECOVG01, RECOVG02}))}

Target database list processing

Extended Functions processes the database list by using the subparameters as described in the following sections.

\textbf{DB() subparameter processing}

The \textit{DB()} subparameter allows for a series of one or more full-function (FF), fast path (FP), or HALDB master database names. This list of database names is processed using the DBRC API and any database that matches a name in the list (allowing for wildcards) is processed. For any database names in the list that are defined in the RECONs, an entry is added to the list.

\textbf{DBDS() subparameter processing}

The \textit{DBDS()} subparameter allows for a series of one, or more pairs of full-function (FF) or fast path (FP) database names followed by the associated database data set (for example, DD name or AREA). Each entry must be paired but each piece can specify wildcard characters. The list of databases is processed using the DBRC API and the database entries, along with all related database data sets, are returned. A list entry is created for each FFDB+DBDS or FPDB+AREA entry that matches the input list. If a database that matches the input is located but is not a full function or fast path database, the entry is not added to the list.

\textbf{AREA() subparameter processing}

This subparameter allows for a series of one or more pairs of fast path database names followed by the associated area name. Each entry must be paired but each piece can specify wildcard characters. The list of databases is processed using the DBRC API and the database entries, along with all related areas are returned. A list entry is created for each of the DEDB AREA+DDname entries that match the input list. If a database that matches the input is located, but is not a fast path database, the entry is not added to the list.
PART() subparameter processing

This subparameter allows for a series of one or more pairs of HALDB master database names, followed by the associated partition name. Each entry must be paired but each piece can specify wildcard characters. The list of databases is processed using the DBRC API and the database entries, along with all of the related areas are returned. A list entry is created for each of the Partition+DDname entries that match the input list. If a database that matches the input is located, but it is not a HALDB database, the entry is not added to the list.

DBDSGRP() subparameter processing

This subparameter allows for a series of one or more RECON DBDS group names. This list is processed using the DBRC API and all of the databases that are associated with the specified groups are returned. A list entry is created for each DB+DD (or AREA, or PART) entry in the RECON group.

CAGRP() subparameter processing

This subparameter allows for a series of one or more RECON Change Accumulation group names. This list is processed using the DBRC API and all of the databases that are associated with the specified groups are returned. A list entry is created for each DB+DD (or AREA or PART) entry in the RECON group.

RECOVGRP() subparameter processing

This subparameter allows for a series of one or more RECON Recovery group names. This list is processed using the DBRC API and all of the databases that are associated with the specified groups are returned. A list entry is created for each DB+DD (or AREA or PART) entry in the RECON group.

RECONCOPY processing

The RECON copy (RECONCOPY) supporting function creates a copy of the RECON data sets that are used by IMS Database Recovery Facility: Extended Functions function processing instead of the live RECON data sets.

The RECONCOPY supporting function is used only in conjunction with other functions, it does not operate alone. This process uses the DBRC BACKUP:RECON RECON1 command, so that whichever copy of the RECON data set is COPY1 is used. This process then makes a second and third copy and then allocates these three data sets to the RECON1, RECON2, and RECON3 DD statements.

Attention: If you do not specify the RECONCOPY parameter, the Extended Functions will access the live RECON data sets.

Not all of the Extended Functions can operate on RECON copies. Those functions that cannot use RECON copies are:

- Recovery Point Create (RPCR)
- IMS Commands (IMSCMD)

The RECON cleanup (RCU) function requires the use of RECONCOPY. The RCU function will not process without RECONCOPY.

The functions that use can use RECON copies are:

- Recovery point identification (RPID)
• RECON cleanup (RCU)
• Recovery health checker (HCHECK)
• Verify recovery assets (VERIFY)

Here is the syntax for RECONCOPY:

```
RECONCOPY( DSNPREF( dsn-prefix )
  REUSE( YES | NO | Y | N )
  DELETE( YES | NO | Y | N )
  COPY1VOL( volser-list )
  COPY2VOL( volser-list )
  COPY3VOL( volser-list )
  SPACE( space-parameters )
  STORCLASS( SMS-storage-class )
  MGMTCLASS( SMS-management-class )
  DATACLASS( SMS-data-class )
)
```

The RECONCOPY subparameters are described in the following list. The only required subparameter is DSNPREF. If you do not specify the allocation type subparameters (for example, SPACE and VOLSER), their values will be obtained from the existing RECON1 data set that is used as a model.

**DSNPREF(dsn-prefix)**

Use this subparameter to specify the pattern to use to create the prefix for the RECON copy data sets. The prefix length is limited to 38 characters. The RECON copy data set names will consist of this prefix with .C1 C2, or .C3 as the suffix. This subparameter is required and there is no default.

The prefix allows for several special character strings that cause Extended Functions to substitute values into the prefix. These special character strings are as follows:

- **&date**  Substitute a date stamp in the format Dyyyyddd within the name.
- **&time**  Substitute a timestamp in the format Thhmmss within the name.
- **&jobnm** Substitute the job name within the name.
- **&stepnm** Substitute the jobstep name within the name.

**REUSE( Y | YES | N | NO )**

Use this subparameter to indicate whether to reuse an existing RECON copy data set, if one is found, with the same name as the one that is being created. A specification of YES or Y causes the process to delete and redefine the existing data sets. A specification of NO or N causes the process to terminate when one or more of the data sets already exist. The default for REUSE is REUSE(NO).

**DELETE( Y | YES | N | NO )**

Use this subparameter to indicate whether the RECON copy data sets are deleted at job step termination. The default is DELETE(NO).

**COPY1VOL( volser-list ), COPY2VOL( volser-list ), COPY3VOL( volser-list )**

Use these subparameters to specify a list of one-to-five volser-list on which to allocate the specified RECON copy data set. Use the COPY1VOL subparameter to specify the list of volser-list for RECON COPY1, Use COPY2VOL to specify the list for RECON COPY2 and use COPY3VOL to specify the list for RECON COPY3.

**SPACE( space-subparameters )**

Use this subparameter to specify the space characteristics for the RECON
copy data sets. The values that you specify for this subparameter must adhere to the SPACE() subparameter that are allowed by IDCAMS as shown below.

- SPACE(CYLINDERS(\textit{primary, secondary}))
- SPACE(TRACKS(\textit{primary, secondary}))
- SPACE(RECORDS(\textit{primary, secondary}))

The abbreviations that are allowed are: CYL, TRK, and REC.

\textbf{STORCLASS( SMS-storage-class )}

\textbf{MGMTCLASS( SMS-management-class )}

\textbf{DATACLASS( SMS-data-class )}

Use the three SMS subparameters to specify the various SMS classes that are used to allocate COPY1, COPY2, and COPY3 of the RECON data sets.

\section*{RECONID locate processing}

Before Extended Functions can begin processing, it must locate the RECONID member and be able to retrieve it from the repository.

The RECONID member contains information that IMS Database Recovery Facility: Extended Functions needs in order to process. This information includes:

1. The eight-byte external name associated with the RECONID member
2. The RECON1, RECON2, and RECON3 data set names
3. The TOI XCF group name that is used to communicate with IMS Tools Online System Interface in the IMS control regions
4. The IMSPLEX name as defined to the RECON data sets, if present
5. The RACF® security class to be used for securing IMS commands that are issued through Extended Functions

The ways that the RECONID member can be specified, and the order that is used to locate the RECONID member are outlined in the next list. When the RECONID member is located, the search terminates (in other words, subsequent steps are performed only until the RECONID member is located).

1. If you specify the RECONID() parameter in either the IMSRECFG= PROCLIB member or the IMSREIN DD, this RECONID() parameter value is used to look up the RECONID member in the repository. If this RECONID member is not located, the job step terminates.
2. If the RECON1 DD is allocated in the JCL, the data set name that is associated with the DD is retrieved. A lookup is performed for that data set name to locate the RECONID member. If no RECONID member with this RECON1 data set name is located, the job step terminates.
3. If the IMSDALIB DD is allocated in the JCL, a BLDL request is performed for member RECON1. If found, the RECON1 member is loaded and the RECON1 data set name is retrieved. A lookup is performed for that data set name to locate the RECONID member. If no RECONID member with this RECON1 data set name is located, the job step terminates.
4. A load is issued for member RECON1 which will attempt to load the member from JOBLIB, STEPLIB, or the link-pack area. If loaded successfully, the RECON1 data set name is retrieved. A lookup is performed for that data set name to locate the RECONID member. If no RECONID member with this RECON1 data set name is located, the job step terminates.
If, after attempting these steps, Extended Functions is unable to locate the
RECONID member, the job terminates with error messages and an ABEND.

TIME and RANGE parameter processing

Many of the IMS Database Recovery Facility: Extended Functions functions allow
and in some cases, require you to specify a timestamp or a range of timestamps for
processing.

There are two parameters that are used by many of the Extended Functions to
specify these timestamps:
- TIME() allows you to specify a single timestamp.
- RANGE() allows you to specify a begin and end timestamp, along with other
  parameters for computing the begin and/or end timestamps. These timestamps
  form the range, or boundaries that the function will use in performing its
  processing.

TIME parameter

The TIME parameter is used by IMS Database Recovery Facility: Extended
Functions functions that require only a single timestamp.

You can specify the format of the timestamp in either compressed or punctuated
format. The timestamp must be enclosed in single quotes. An example of the
TIME() parameter is:

TIME('06/25-08.24.45.7 -8:00')

Timestamp format

You can specify timestamps for the TIME() parameter in any format that is
recognizable to DBRC with the exception that symbolic offset values, such as PST
or UTC, are not allowed.

Additionally, timestamps must always be enclosed within single quotation marks.
For detailed information about the formats that are recognizable to DBRC, see the
topics about DBRC Time Stamps in the DBRC Guide and Reference.

The format of the timestamp subparameter is either compressed or punctuated. For
all timestamp formats that are shown in the next examples, the use of brackets []
indicates that the value that is contained in the brackets is optional.

Compressed timestamps follow this format:

' [yy]yydddhhmmss [offset]' 

Punctuated timestamps follow this format:

' [yy]yy|ddd|hh|mm|ss|t [offset]' 

The meaning of each portion of the timestamp is described in this list:
- [yy]yy is the year specification (0000 – 9999). You can abbreviate the year to yy,
specifying only the last two digits of the year. 2007 and 07 are equivalent values.
- ddd is the day specification (001 – 365).
- hh is the hour specification (0-23).
• **mm** is the minute specification (0-59).
• **ss** is the second specification (0-59).
• **t** is the tenth of a second specification (0-9).
• **offset** is optional. You can use **offset** to specify the offset for computing local time. It is a numeric offset in the form **h[h:mm]** or **h[h:mm]** that, when added to UTC, gives local time. **h[h]** is a numeric value from 0 to 14. For the compressed format if you specify **mm**, then also specify **hh:mm** is a value from the set {00, 15, 30, 45}.

• In the punctuated timestamp, the delimiter character (|) can be any non-numeric character delimiter including blank except the single quotation mark (').

Here are 2 examples of compressed timestamps:

'06250824457'

'06250824457-0800'

Here are 4 examples of punctuated time stamps:

'06.252/08:24:45.7'

'06.252 08:24:45.7 -8'

'06/252-08.24.45.7 -8:00'

'2006 252 16.24.45.7 +0'

**Note:** If a timestamp does not contain an offset, it is assumed to be local time and the offset of the local MVS™ is used to calculate the UTC timestamps that are used internally.

### RANGE parameter

The RANGE parameter is used by Extended Functions that require both a beginning and ending timestamp (for example, a range).

**RANGE(timestamp, direction, amount, scale)**

There are three formats that you can specify for this parameter that are described in these topics.

**timestamp**

Use this subparameter to specify a timestamp in either compressed or punctuated format, as already described.

**direction**

Use this subparameter to specify the direction to be applied to the timestamp that you specified in computing the begin and/or end timestamps. Acceptable values are: +-(plus,minus), +, and -.

**amount**

Use this subparameter to specify a number from 1-to-366 which indicates the amount to be added to and/or subtracted from the specified timestamp in computing the begin and/or end timestamps.

**scale**

Use this subparameter to specify the scale that is associated with the
amount that you specified. Acceptable values are SECS (seconds), MINS (minutes), HRS (hours), and DAYS (days).

The allowed values for the amount subparameter depends on the scale that is specified.
- SECS nnnn must be from 1-59
- MINS nnnn must be from 1-59
- HRS nnnn must be from 1-23
- DAYS nnnn must be from 1-366

The beginning and ending timestamps are computed as follows:
- If direction is specified as +- (plus, minus), the beginning timestamp is computed by subtracting the amount of time that is specified (by amount and scale) from the timestamp value. The ending timestamp is computed by adding the amount of time specified to the timestamp value.
- If direction is specified as + (plus), the timestamp that is specified is the beginning timestamp. The ending timestamp is computed by adding the amount of time that is specified to the timestamp value.
- If direction is specified as - (minus), the beginning timestamp is computed by subtracting the amount of time that is specified (by amount and scale) from the timestamp value. The timestamp that is specified is used as the ending timestamp.

Extended Functions JCL reference

IMS Database Recovery Facility: Extended Functions requires JCL statements to run. A complete set of the sample JCL jobstreams that you need to run Extended Functions is delivered in SAMPLIB member FRXIVPR1.

Sample JCL jobstreams in the SAMPLIB are documented in the member with descriptive and instructive comments.

The following sections describe the EXEC, input DD, and output DD statements:

The EXEC statement

The following EXEC statement runs Extended Functions:

```plaintext
//STEP1 EXEC PGM=IROMAIN,
   PARM=('TYPE=IRE,IMSRECFG=IRECFGXX,BPECFG=BPECFG')
```

The IROMAIN program is run for all of the Extended Functions. The following definitions describe the PARM= keywords:

**TYPE=**
Use this keyword to indicate the type of address space that is used. The only value for this keyword is IRE. This keyword is optional and its default is IRE.

**IMSRECFG=**
Use this keyword to specify the name of the Extended Functions configuration member that is located in the //PROCLIB DD statement. If you specify it, the member that you indicate is read and parsed by Extended Functions. You can use this member to specify configuration parameters only. No function control cards are allowed in this member.
This keyword is optional and has no default however, if IMSRECFG= is omitted, you must provide the ITKBSRVR keyword in the //IMSREIN DD input.

BPECFG=
Use this keyword to specify the name of the BPE configuration member that is located in the //PROCLIB DD statement. If you specify it, the member that you indicate is read and parsed by BPE. This keyword is optional and has no default.

Input DD statements

The following DD statements are allowed or required as indicated for specifying Extended Functions run input.

STEPLIB / JOBLIB
Use this DD to specify the load libraries that are necessary to access the Extended Functions code. This specification must also include the load libraries for IMS Tools Online System Interface, the IMS Tools Base for z/OS: IMS Tools Knowledge Base, and the IMS load library. These concatenated libraries must be APF-authorized.

IMSDALIB
Use this optional DD to point to the dynamic allocation load library that contains the dynamic allocation member for your RECON1 data set. This data set is used by Extended Functions to locate the RECON1 dynamic allocation member in order to retrieve the RECON1 data set name when the RECON1 data set name is not specified by any other method. The complete steps for locating the RECON1 data set name are specified in “RECONID locate processing” on page 29.

PROCLIB
Use this DD to specify the JCL procedure library (PROCLIB) that contains the members that you specified with the IMSRECFG= or BPECFG= parameters on the MVS parameter string. If you specified either of these parameters, this DD is required. Otherwise, this DD is optional.

RECON1 / RECON2 / RECON3
You can use these DD statements to specify the names of the RECON1, RECON2 and RECON3 data sets. These DD statements are optional but, you must specify the RECON1 data set name in some manner. The complete steps for locating the RECON1 data set name are described in “RECONID locate processing” on page 29.

IMSEREIN
Use this DD to specify configuration parameters and function control cards that drive Extended Functions processing. This DD is required. The parameters that can be specified in IMSREIN are described in “Supporting functions” on page 24 and also the other product functions.

Output DD statements

The following output DD statements are allowed or required as indicated for specifying the location of outputs that are produced by running Extended Functions.

IREDIAG
Use this DD statement for debugging. This DD is currently used by the
RECON copy function as the output data set for IDCAMS and DSPURXRT. The characteristics of this data set are: RECFM=FBA,LRECL=133. This DD is optional.

**SYSPRINT**
This DD statement is written to by some internal functions that are used by Extended Functions. The characteristics of this data set are: RECFM=FBA,LRECL=133. This DD is optional but if you do not specify it, this DD will be dynamically allocated to SYSOUT=*.

**RPTSUMM**
This DD is used by Extended Functions to put out information to a general purpose DD. Several summary reports are written to this DD that contain operational and execution information. The characteristics of this data set are: RECFM=FBA,LRECL=133. This DD is optional but if you do not specify it, this DD will be dynamically allocated to SYSOUT=*.

Each Extended Functions has its own report DD that must be specified when that function is requested. If the function-specific DD is not present then the report DD that is associated with that function is dynamically allocated to SYSOUT=*.

The characteristics of these data sets are: RECFM=FBA,LRECL=133. The report DD names for each function are listed in the next table.

<table>
<thead>
<tr>
<th>Report DD Name</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>RPTRPCR</td>
<td>Recovery Point Create (RPCR)</td>
</tr>
<tr>
<td>RPTRPID</td>
<td>Recovery Point Identify (RPID)</td>
</tr>
<tr>
<td>RPTVERIFY</td>
<td>Verify (VERIFY)</td>
</tr>
<tr>
<td>RPTRCU</td>
<td>RECON Cleanup Utility (RCU)</td>
</tr>
<tr>
<td>RPTHCHECK</td>
<td>Health Checker (HCHECK)</td>
</tr>
<tr>
<td>RPTIMSCM</td>
<td>IMS Command (IMSCMD)</td>
</tr>
</tbody>
</table>

**Extended Functions sample summary report**
When any IMS Database Recovery Facility: Extended Functions function runs, it creates a summary report that is written to the RPTSUMM output DD. This report contains overall system execution information and reflects the configuration and function control cards and other processing options.

If the RPTSUMM DD is not specified in the JCL, it will be dynamically allocated to SYSOUT=*.

Report output can be written to SYSOUT or to a data set. The characteristics of the output are RECFM=FBA,LRECL=133. Report output can also be written to the IMS Tools Base for z/OS: IMS Tools Knowledge Base by specifying the REPORT(ITKBOUT(YES)) keyword and parameter.

**Note:** REPORT(ITKBOUT(YES)) is the default.

The next figure shows a sample SUMMARY report. The format of this sample report has been modified to fit the publication formatting.
The major sections of the summary report are described in this list:

**MVSP - MVS PARM= STRING**

This report section shows the value of the PARM= string that was specified
on the EXEC statement for this job step. If the PARM= string was specified, it shows the BPE and Extended Functions configuration members that were used.

**CFG - IMSRECFG= CONFIGURATION MEMBER**
This report section shows the contents of the Extended Functions configuration member, if specified. This member is read by Extended Functions and any parameters that are specified will be in effect for this run. This member is specified using the IMSRECFG= keyword on the EXEC statement PARM= string.

**REIN - IMSREIN INPUT MEMBER**
This report section shows the contents of the IMSREIN DD. This data is read by Extended Functions and any parameters that are specified will be in effect for this run.

**PARM - ACTIVE CONFIGURATION PARMS**
This report section shows the settings for the configuration parameters that were used for this job run.

**RCID - ACTIVE RECONID RECORD**
This report section shows the RECONID member that was used for this job run.

**RCGR - DBLIST() RECON GROUP EXPANSIONS**
This report section shows any DBLIST( CAGRP(), DBDSGRP() or RECOVGRP() ) input and their expansions. The report headings and descriptions follow.

**INPUT:**
This field indicates the specific input that was specified.

**GROUP:**
This field shows the database name (DB=) and DD name (DD=) for each member of the group.

**MEMBER:**
This field shows the database name (DB=) and DD name (DD=) for each member of the group.

**DBLS - LIST OF DATABASES PROCESSED**
This report section shows the databases that were processed in this run. These are the report headings and their descriptions:

**DATABASE**
This column specifies the database name.

**AREA/PART**
This column specifies the area name for fast path databases, or the partition name for HALDB databases. This column is blank for full function databases.

**DD**
This column specifies the DD name that is associated with this area, partition, or full function database.

**TYPE**
This column specifies the type of database. Possible values are FF for full function databases, FP for fast path areas, and HALD for HALDB partitions.

**DSN**
This column specifies the data set name that is associated with this area, partition, or full function database.
PART-TYPE
For HALDB partitions, this column specifies the type of data set. Possible values are INDEX for the index portion of the HALDB, ILE for the index list element (or ILDS) portion of the HALDB, or DATA for the data portion of the HALDB.

OLR-PARTNER
For HALDB partitions that are OLR capable, this column specifies the OLR partner DD name that is associated with the partition.

ERLS - DBLIST() ERRORS
This report section shows the databases that were specified in the DBLIST, or expanded based on the DBLIST but were found to be in error. The report headings and descriptions are:

DATABASE
This column shows the database name.

DD/AREA/PART
This column shows the DD name, area name for fast path databases, or the partition name for HALDB databases.

ERROR
This column indicates the specific error that was found. The possible values are:
• DATABASE HAS NO DEFINED DD/AREA/PART
• ENTRY NOT DEFINED TO RECON
• HALDB PART SPECIFIED WHERE MASTER REQUIRED
Part 3. Using the Extended Functions

Each IMS Database Recovery Facility: Extended Functions perform a specific operation. You need to understand each function to fully operate Extended Functions.

Topics:
- IMS Command (IMSCMD) function
- Health Checker (HCHECK) function
- Recovery Point Creation (RPCR) function
- Recovery Point Identification (RPID) function
- Verify Recovery Assets (VERIFY) function
- RECON Clean up (RCU) function
Chapter 4. IMS Command (IMSCMD) function

The IMS command function (IMSCMD) provides a batch interface for IMS Type-1 AOI commands in IBM IMS Recovery Solution Pack for z/OS: IMS Database Recovery Facility: Extended Functions.

Topics:
- “IMSCMD overview”
- “IMSCMD input”
- “IMSCMD usage examples” on page 42
- “IMSCMD output” on page 42

IMSCMD overview

The IMSCMD function issues the commands that you specify to one or more active IMS subsystems. IMS Type-1 AOI commands all begin with a slash (/) (for example, /DBR).

IMSCMD function uses the IMS Tools Online System Interface to issue IMS commands. For this function to operate, IMS Tools Online System Interface must be installed and active on the IMS systems to which this command is issued. The available target IMS systems are determined by the IMS Tools Online System Interface XCF group in the RECONID member that is used by the job. You can issue the IMSCMD function to all of the IMS systems that are connected to the IMS Tools Online System Interface XCF group, or to individual IMS systems, by using the SYSTEMS subparameter on the IMSCMD function specification.

Use of the RECONCOPY supporting function has no effect on the IMSCMD function. Any commands that you issue run under the control of the IMS control region and therefore, those commands access the RECON data sets that are allocated to that IMS subsystem.

Similarly, use of the DBLIST supporting function has no effect on the IMSCMD function. Any commands that you issue must be specified completely and no database list processing is involved.

IMSCMD input

Run the IMS Command function by running IMSCMD.

IMSCMD function syntax

```
IMSCMD ( COMMAND ( command )
    SYSTEMS ( systems-list )
)
```

Parameters

COMMAND

Use this required parameter to specify a single IMS Type-1 AOI command
to be issued. Specify the IMS commands as you would normally; starting with a slash (/) and following the same syntax as the command follows when it is issued manually.

**SYSTEMS**

Use this optional parameter to specify one or more active IMS systems to which the command is issued. If you omit this parameter, the command is issued to all of the active IMS systems in the IMS Tools Online System Interface XCF group. If a system that is specified in the list is not active, the fact that the system was inactive is noted in the output, but the command is still issued for any active systems in the Tools Online System Interface XCF group.

When you use the GLOBAL parameter on IMS commands, the SYSTEMS parameter is not allowed. If you specify both the GLOBAL and SYSTEMS parameters, an error message is displayed and the IMS command is not performed. The GLOBAL parameter requires internal resource lock manager (IRLM) in order to run the commands on all of the sharing online subsystems.

---

**IMSCMD usage examples**

You can use the IMS Command function (IMSCMD) to alter or display the status of IMS resources across all of the IMS systems in an IMSplex or data sharing environment.

**Example 1: Specifying multiple parameters**

You can specify multiple IMSCMD parameters within the same job step. For example, to display the status of a series of databases, programs and transactions across the IMSplex, you would specify the following input:

```
IMSCMD(COMMAND(/DIS DB FFDB01 FFDB02))
IMSCMD(COMMAND(/DIS PGM FFPGM01 FFPGM02))
IMSCMD(COMMAND(/DIS TRAN FFTRAN01 FFTRAN02))
```

**Example 2: Displaying the same set of IMS resources**

If you wanted to display the same set of IMS resources for just the IMS1 system, you would specify the following input:

```
IMSCMD(COMMAND(/DIS DB FFDB01 FFDB02) SYSTEMS(IMS1))
IMSCMD(COMMAND(/DIS PGM FFPGM01 FFPGM02) SYSTEMS(IMS1))
IMSCMD(COMMAND(/DIS TRAN FFTRAN01 FFTRAN02) SYSTEMS(IMS1))
```

---

**IMSCMD output**

The IMS Command function (IMSCMD) output is written to the RPTIMSCM DD.

If you do not specify the RPTIMSCM DD in the JCL, it is dynamically allocated to SYSOUT=* . Report output can be written to SYSOUT or to a data set. The characteristics of the output are RECFM=FBA, LRECL=133. You can also have the report output written to the IMS Tools Base for z/OS: IMS Tools Knowledge Base if you specify the REPORT(ITKBOUT(YES)) keyword and parameter.

**Note:** REPORT(ITKBOUT(YES)) is the default.
The following figure shows a sample IMSCMD report output.

IMS Recovery Solution Pack V1.1.0 (5655-V86)
IMS DRF Extended Functions IMS COMMAND REPORT Run Time: 11/06/2007 08:57

********************************************************************************
-> Invocation Parameters
********************************************************************************

Command: /DIS DB DI21PART
Systems: ALL

********************************************************************************
-> IMS Command Output
********************************************************************************

--> IMSID: Q91A COMMAND: /DIS DB DI21PART
D71 DATABASE TYPE TOTAL UNUSED TOTAL UNUSED ACC CONDITIONS
D01 DI21PART UP STOPPED, NOTOPEN, NOTINIT
X99 *07310/085729*

********************************************************************************
-> IMS Command Output
********************************************************************************

--> IMSID: Q91B COMMAND: /DIS DB DI21PART
D71 DATABASE TYPE TOTAL UNUSED TOTAL UNUSED ACC CONDITIONS
D01 DI21PART UP STOPPED, NOTOPEN, NOTINIT
X99 *07310/085729*

Figure 3. Sample IMSCMD report output

The major report sections are described under their headings in the following list:

**Invocation Parameters**
This section of the report shows the IMS command that the IMSCMD function issued and the systems to which the IMS command was issued. This sample report shows that the /DIS DB DI21PART command was issued to all of the active IMS systems.

**IMS Command Output**
This section of the report is repeated for each IMS system to which the IMS command was issued. This section also contains all of the command output. This sample report shows that the command was issued to IMSIDs, Q91A and Q91B because they were all of the active IMS systems in the Tools Online System Interface XCF group.
Chapter 5. Health Checker (HCHECK) function

The Health Checker function (HCHECK) provides an interface to the RECONs that identifies a number of issues that can impact IMS database recovery in IBM IMS Recovery Solution Pack for z/OS: IMS Database Recovery Facility: Extended Functions.

Topics:
- “HCHECK overview”
- “HCHECK input”
- “HCHECK usage examples” on page 51
- “HCHECK output” on page 52

HCHECK overview

The HCHECK function interrogates information in the RECON data sets that is associated with the IMS database subsystem, looking for potential problems with the recoverability of one or more databases. HCHECK uses information from the RECON data sets and the MVS catalog when it looks for potential problems.

The HCHECK function requires that you specify the target databases using the DBLIST supporting function.

The HCHECK function can use the RECON copy data sets that are created by the RECONCOPY supporting function. It is recommended that you use RECONCOPY in conjunction with HCHECK to prevent I/O to the live RECON data sets.

HCHECK input

Run the Health Checker function by running HCHECK.

HCHECK function syntax

```
HCHECK( hcheck-option
  HOURS(nnn)
  DAYS(nnn)
  RANGE(range-specification)
  NUMBER(nnn)
  )
```

Parameters

`hcheck-option`

Use this positional parameter to specify the HCHECK option to be processed. The list of valid options is described in the next table. This parameter must be the first parameter that you specify on the HCHECK keyword.

Use the following HCHECK parameters to specify the time frame range, or number that HCHECK will use for various options. Not all HCHECK options allow these parameters. Also, these parameters are mutually exclusive. The valid list of HCHECK options and the parameters that are allowed is shown in the next table.
HOURS(nn)

Use the HOURS parameter to specify the number of hours to be used to limit the HCHECK option. HOURS allows an integer between 1 and 999. There is no default for the HOURS parameter.

DAYS(nn)

Use the DAYS parameter to specify the number of days to be used to limit the HCHECK option. DAYS allows an integer between 1 and 999. There is no default for the DAYS parameter.

NUMBER(nn)

Use the NUMBER parameter to specify the number to be used to limit the HCHECK option. NUMBER allows an integer between 1 and 999. There is no default for the NUMBER parameter.

RANGE(range-specification)

Use the RANGE parameter is used to specify the beginning and ending time frame (or range) to be used to limit the HCHECK option. There is no default for the RANGE parameter.

For more detailed information about the format of the range specification, see “TIME and RANGE parameter processing” on page 30.

The following table shows all of the valid HCHECK options, their abbreviations, acceptable parameters, and a brief description of each option. A more detailed description of each option appears after this table.

<table>
<thead>
<tr>
<th>Option</th>
<th>Abbreviations</th>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ICALL</td>
<td>IA, ICA</td>
<td>N/A</td>
<td>Produces the ICNONE, ICNEED, and ICREC reports.</td>
</tr>
<tr>
<td>ICNONE</td>
<td>IO, ICNO</td>
<td>N/A</td>
<td>DBs for which no ICs exist.</td>
</tr>
<tr>
<td>ICNEED</td>
<td>IE, ICNE</td>
<td>N/A</td>
<td>DBs for which ICs are needed.</td>
</tr>
<tr>
<td>ICREC</td>
<td>IR, ICR</td>
<td>N/A</td>
<td>DBs for which ICs are recommended.</td>
</tr>
<tr>
<td>ICFREQ</td>
<td>IF, ICF</td>
<td>HOURS()</td>
<td>DBs for which no ICs exist within the specified time. Processes batch and online ICs.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DAYS()</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>RANGE()</td>
<td></td>
</tr>
<tr>
<td>MINBAT</td>
<td>MB, MINB</td>
<td>HOURS()</td>
<td>DBs for which no minimum number of batch ICs exist within the specified time. Processes batch ICs only.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DAYS()</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>RANGE()</td>
<td></td>
</tr>
<tr>
<td>MINONL</td>
<td>MO, MINO</td>
<td>HOURS()</td>
<td>DBs for which no minimum number of online ICs exist within the specified time. Processes online ICs only.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DAYS()</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>RANGE()</td>
<td></td>
</tr>
<tr>
<td>EEQE</td>
<td>EQ, E</td>
<td>N/A</td>
<td>DBs that have error queue elements (EEQEs).</td>
</tr>
<tr>
<td>DBNOTCA</td>
<td>DC</td>
<td>N/A</td>
<td>DBs that are not in a CA group.</td>
</tr>
<tr>
<td>HALDBDISABLEDPART</td>
<td>HD, HALDBD</td>
<td>N/A</td>
<td>HALDBs that have disabled partitions.</td>
</tr>
<tr>
<td>HALDBNOTINIT</td>
<td>HN, HALDBN</td>
<td>N/A</td>
<td>HALDBs that are not initialized.</td>
</tr>
<tr>
<td>NOADS</td>
<td>NA</td>
<td>N/A</td>
<td>Produces the NOADSDNEFINE and NOADSAVAIL reports.</td>
</tr>
<tr>
<td>NOADSDNEFINE</td>
<td>ND, NOADSD</td>
<td>N/A</td>
<td>DEDBs that have no area data sets defined.</td>
</tr>
<tr>
<td>NOADSAVAIL</td>
<td>NV, NOADSA</td>
<td>N/A</td>
<td>DEDBs that have no area data sets available.</td>
</tr>
<tr>
<td>BACKOUT</td>
<td>BO, B</td>
<td>N/A</td>
<td>DBs that show backout needed.</td>
</tr>
<tr>
<td>MINCARECS</td>
<td>MC, MINC</td>
<td>NUMBER()</td>
<td>DBs that do not have the minimum number of CA records available.</td>
</tr>
</tbody>
</table>
### Table 2. The valid HCHECK options, abbreviations, parameters, and descriptions (continued)

<table>
<thead>
<tr>
<th>Option</th>
<th>Abbreviations</th>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SECLOG</td>
<td>SL, SECL</td>
<td>N/A - global</td>
<td>No valid corresponding SECLOG entry for a PRILOG entry.</td>
</tr>
<tr>
<td>SECSLD</td>
<td>SS, SECS</td>
<td>N/A - global</td>
<td>No valid corresponding SECSLDS entry for a PRISLDS entry.</td>
</tr>
<tr>
<td>CARUN</td>
<td>CR, CAR, CARU</td>
<td>HOURS()</td>
<td>CA group does not have a valid CARUN within specified time.</td>
</tr>
<tr>
<td>PROAUTH</td>
<td>PA, PR</td>
<td>N/A</td>
<td>DBs that indicate that prohibit authorization is on.</td>
</tr>
<tr>
<td>PI SEC</td>
<td>PS, PI</td>
<td>HOURS()</td>
<td>Primary IC with no corresponding valid secondary IC within that specified time.</td>
</tr>
<tr>
<td>VERIFY</td>
<td>VF, VER, VFY</td>
<td>N/A</td>
<td>Ensures that all recovery assets are cataloged.</td>
</tr>
</tbody>
</table>

You can specify just one of the HCHECK options within each HCHECK command. However, you can specify the HCHECK command any number of times within a single JCL input stream. Also, you can specify the HCHECK option using the fully-qualified option, or any of the acceptable abbreviations that are displayed in the last table. For those options that require additional parameters, if you do not specify the required additional parameters, the HCHECK command is marked in error and is not processed.

The following sections describe the HCHECK options and their relationships in more detail.

**Image copy needed or recommended**

Syntax for the ICALL, ICNONE, ICNEED, and ICREC options:

```
HCHECK ( ICALL | ICNONE | ICNEED | ICREC )
```

**ICALL**
Use this option to specify that the ICNONE, ICNEED, and ICREC options are to be processed.

**ICNONE**
Use this option to check for database data sets for which no image copy has been taken.

**ICNEED**
Use this option to check for database data sets for which an image copy is needed.

**ICREC**
Use this option to check for database data sets for which an image copy is recommended.
No Batch or Online Image Copy within Specified Time Frame

HCHECK ( ICFREQ <time-specification> | MINBAT <time-specification> | MINONL <time-specification> )

ICFREQ
Use this option to specify that the MINBAT and MINONL options are processed.

MINBAT
Use this option to specify that a check is performed for valid batch image copies within the specified time frame.

MINONL
Use this option to specify to check for valid online image copies within the specified time frame.

Databases that have extended error queue elements (EEQEs)

Syntax for the EEQE option:

    HCHECK ( EEQE )

EEQE Use this option to check for databases that have EEQEs, indicating that recovery might be needed.

Databases not in a change accumulation group

Syntax for the DBNOTCA option:

    HCHECK ( DBNOTCA )

DBNOTCA Use this option to check for databases that are not included in any change accumulation group. Databases that are not in change accumulation groups might adversely affect database recovery.

HALDB not initialized

Syntax for the HALDBNOTINIT option:

    HCHECK ( HALDBNOTINIT )

HALDBNOTINIT Use this option to check for High Availability Large Databases (HALDBs) that have not been initialized. These databases are currently unusable.

No area data set defined, no area data set available

Syntax for the NOADS, NOADSDEFINE, and NOADSAVAIL options:

    HCHECK ( NOADS | NOADSDEFINE | NOADSAVAIL )

NOADS Use this option to specify that the NOADSDEFINE and NOADSAVAIL
options are processed. Use the NOADSDEFINE option to check for fast path databases that have no area data sets defined.

**NOADSDEFINE**
Use this option to check for fast path databases that have no area data sets defined.

**NOADSAVAIL**
Use this option to check for fast path databases that have no currently available area data sets.

**Backout needed**

Syntax for the BACKOUT option:

```
HCHECK ( BACKOUT )
```

**BACKOUT**
Use this option to check for databases marked as backout needed.

**Minimum change accumulation records not available**

Syntax for the MINCARECS option:

```
HCHECK ( MINCARECS NUMBER ( number ) )
```

**MINCARECS**
Use this option to check for databases that do not have the specified number of change accumulation data sets available.

**NUMBER(number)**
Use this option keyword to specify the minimum acceptable available change accumulation data sets.

**No valid corresponding SECLOG entry for the PRILOG record**

Syntax for the SECLOG option:

```
HCHECK ( SECLOG )
```

**SECLOG**
Use this option to check for databases that have a valid PRILOG entry, but no corresponding SECLOG record.

**No valid corresponding SECSLDS entry for PRISLDS record**

Syntax for the SECSLD option:

```
HCHECK ( SECSLD )
```

**SECSLD**
Use this option to check for databases that have a valid PRISLDS entry, but no corresponding SECSLDS record.
Change accumulation (CA) group does not have a valid CA run within the specified time range

Syntax for the CARUN option:

HCHECK ( CARUN <time-specification> )

**CARUN**

Use this option to check for databases that do not have a valid change accumulation within the specified time frame.

*time-specification*

Use this required option keyword to specify the time frame to check. Specify the *time-specification* with either the HOURS(), DAYS(), or RANGE() keyword. You must specify one of the HOURS(), DAYS(), or RANGE() keywords or the HCHECK CARUN function will be rejected. For a description of HOURS(), DAYS(), or RANGE(), see "TIME and RANGE parameter processing" on page 30.

Database is in prohibit authorization state

Syntax for the PROAUTH option:

HCHECK ( PROAUTH )

**PROAUTH**

Use the PROAUTH option to check for databases that are in prohibit authorization state.

Primary image copy with no corresponding valid secondary image copy within the specified time range

Syntax for the PISEC option:

HCHECK ( PISEC <time-specification> )

**PISEC**

Use this option to check for databases that have a valid primary image copy, but no valid secondary image copy within the specified time frame.

*time-specification*

Use this required option keyword to specify the time frame to check. Specify the *time-specification* with either the HOURS(), DAYS(), or RANGE() keyword. You must specify one of the HOURS(), DAYS(), or RANGE() keywords or the HCHECK CARUN function will be rejected. For a description of HOURS(), DAYS(), or RANGE(), see "TIME and RANGE parameter processing" on page 30.

Disabled HALDB partition

Syntax for the HALDBDISABLEDPART option:

HCHECK ( HALDBDISABLEDPART )

**HALDBDISABLEDPART**

Use this option to check for High Availability Large Databases (HALDBs) that have disabled partitions.
Verify that recovery assets are cataloged

Syntax for the VERIFY option:

\[ \text{HCHECK()} \text{ VERIFY} \]

**VERIFY**

Use this option to verify that the recovery assets that are associated with the selected databases are catalogued. All of these types of recovery assets that are registered with DBRC are verified:

- Database data set names
- Image copy data set names
- Log data set names
- Change accumulation data set names

**HCHECK usage examples**

You can set up the Health Check function (HCHECK) in several different ways.

You can set up HCHECK to run on a scheduled basis to periodically check for problems that can affect database recovery. You can also use HCHECK at the Disaster Recovery (DR) site to verify that your DR site is recoverable. You must specify the options to be checked individually, each with its own HCHECK keyword. You can although, specify multiple HCHECK keywords within the same job step.

**Example 1: Identify problems related to image copies**

To identify problems that are related to image copies, specify the following control cards:

To report on any databases that do not have a batch or online image copy within the last 14 days, issue the first line in the next example.

To report on any databases for which one of the following conditions are true, issue the second line that is shown in the next example:

- No image copy has been taken
- An image copy is needed
- An image copy is recommended

\[ \text{HCHECK}(\text{ICFREQ DAYS(14)}) \]
\[ \text{HCHECK}(\text{ICALL}) \]

**Example 2: Identify problems with partitions or unavailable areas**

To identify problems that are related to disabled partitions, partitions that are not initialized, or unavailable areas, specify these control cards:

\[ \text{HCHECK}(\text{HALDBDISABLEDPART}) \]
\[ \text{HCHECK}(\text{HALDBNINIT}) \]
\[ \text{HCHECK}(\text{NOADS}) \]
Example 3: Identify databases with potential backout or authorization problems

To identify databases with potential backout or authorization problems, use this input:

```
HCHECK(BACKOUT)
HCHECK(PROAUTH)
```

HCHECK output

The HCHECK function report is written to the RPTHCHEK output DD.

If you do not specify the RPTHCHEK DD in the JCL, it will be dynamically allocated to SYSOUT=*.

You can also have the output report written to SYSOUT, or to a data set. The characteristics of the output are RECFM=FBA, LRECL=133. Or, you can have the output report written to the IMS Tools Base for z/OS: IMS Tools Knowledge Base by specifying the REPORT(ITKBOUT=YES) keyword and parameter.

Note: REPORT(ITKBOUT=YES) is the default.

The following figure shows a sample HCHECK report:

```
 IMS Recovery Solution Pack V1.1.0 (5655-V86)
 IMS DATABASE FACILITY EXTENDED FUNCTIONS HEALTH CHECKER REPORT
 Run Time: 11/05/2007 12:30

> Invocation Parameters
> Database List
> Health Checker Report
```

The major report sections are described here with their headings:

**Invocatin Parameters**
This section of the report shows the options that were used by HCHECK
processing. In this example you can see that the CARUN and ICNEED options were specified. Any parameters that you specified for the options are also displayed. In the example of CARUN, a range was specified.

**Database List**
This section of the report shows the databases that were processed in this run. The report headings and descriptions are listed here:

- **DATABASE**
  This column lists the names of the databases that were processed.

- **AREA/PART**
  This column lists the area names for fast path databases or the partition names for HALDB databases. This column is blank for full-function databases.

- **DD**
  This column shows the DD name that is associated with this area, partition, or full-function database.

- **TYPE**
  This column lists the type of databases that were processed. Valid values are FF for full-function databases, FP for fast path areas, and HALD for HALDB partitions.

- **DSN**
  This column lists the data set names that are associated with this area, partition, or full function database.

- **PART-TYPE**
  For HALDB partitions, this column specifies the type of data set. Valid values are INDEX for the index portion of the HALDB, ILE for the index list element (or ILDS) portion of the HALDB, or DATA for the data portion of the HALDB.

- **OLR-PARTNER**
  For HALDB partitions that are IMS Online Reorganization (OLR) capable, this column specifies the OLR partner DD name that is associated with the partition.

**Health Checker Report**
This section includes a line for any condition that HCHECK identifies as a problem. The report headings and their descriptions are:

- **DBD**
  This column specifies the database name for which the condition was identified. If the database is a HALDB type, this column indicates the partition name.

- **DDN/AREA**
  This column specifies the DD name (for full-function or HALDB databases), or the fast path area name for which the condition was identified.

- **TYPE**
  This column indicates the type of database that was processed. Valid values are FF (for full-function database), FP (for Fast Path area), and HALD (for HALDB partition).

- **FUNCTION**
  This column indicates the HCHECK option that is associated with the condition.

- **HEALTH CHECKER MESSAGE(S)**
  This column indicates the condition that was identified by HCHECK. Possible values and their meanings are listed in the next table. An asterisk in the message column indicates that the condition is associated with a specified range.
### Table 3. Health checker messages and their issuing functions

<table>
<thead>
<tr>
<th>Message</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>No Image Copy has been taken</td>
<td>ICNONE</td>
</tr>
<tr>
<td>Image Copy is Needed</td>
<td>ICNEED</td>
</tr>
<tr>
<td>Image Copy is Recommended</td>
<td>ICREC</td>
</tr>
<tr>
<td>No Batch Image Copy taken *</td>
<td>MINBAT</td>
</tr>
<tr>
<td>No Online Image Copy taken *</td>
<td>MINONL</td>
</tr>
<tr>
<td>DB has EEQE, so recovery may be needed</td>
<td>EEQE</td>
</tr>
<tr>
<td>DB is not in a Change Accumulation group</td>
<td>DBNOTCA</td>
</tr>
<tr>
<td>HALDB is not initialized</td>
<td>HALDBNOTINIT</td>
</tr>
<tr>
<td>No Area Data Set is defined</td>
<td>NOADSDEFINE</td>
</tr>
<tr>
<td>No Area Data Set is available</td>
<td>NOADSAVAIL</td>
</tr>
<tr>
<td>One or more Area Data Sets are unavailable</td>
<td>NOADSAVAIL</td>
</tr>
<tr>
<td>Backout is needed</td>
<td>BACKOUT</td>
</tr>
<tr>
<td>Minimum CA records not available *</td>
<td>MINCARECS</td>
</tr>
<tr>
<td>No corresponding SECLOG for PRILOG</td>
<td>SECLOG</td>
</tr>
<tr>
<td>No corresponding SECSLDS for PRISLDS</td>
<td>SECSLD</td>
</tr>
<tr>
<td>CA group has no valid CA run *</td>
<td>CARUN</td>
</tr>
<tr>
<td>HALDB has Prohibit Authorization status</td>
<td>PROAUTH</td>
</tr>
<tr>
<td>Primary IC has no valid secondary IC *</td>
<td>PISEC</td>
</tr>
<tr>
<td>HALDB partition is disabled</td>
<td>HALDBDISABLEDPART</td>
</tr>
<tr>
<td>DB Recovery Asset(s) not catalogued</td>
<td>VERIFY</td>
</tr>
<tr>
<td>LOG Recovery Asset(s) not catalogued</td>
<td>VERIFY</td>
</tr>
<tr>
<td>CA Recovery Asset(s) not catalogued</td>
<td>VERIFY</td>
</tr>
<tr>
<td>Recovery Asset(s) not catalogued</td>
<td>VERIFY</td>
</tr>
</tbody>
</table>
Chapter 6. Recovery Point Creation (RPCR) function

The Recovery Point Creation (RPCR) function allows you to create recovery points for one or more databases in IBM IMS Recovery Solution Pack for z/OS: IMS Database Recovery Facility: Extended Functions.

Topics:
- “RPCR overview”
- “RPCR input” on page 56
- “RPCR usage examples” on page 58
- “RPCR output” on page 58

RPCR overview

RPCR sets the allocation records in the RECON data sets to be updated, indicating a period of time where the database is not allocated by an IMS subsystem, thereby creating a valid recovery point.

The following items are performed to ensure that a valid database recovery point is created when RPCR runs. In this list, references to database include databases, partitions, and areas.

- Issues a CHANGE.DB NOAUTH command for each database in the list to set prohibit authorization.
- Issues an ENQ for each database to communicate with BMPs and CICS/ODBA applications.
- Issues a /DIS DB command to obtain the status of each database.
- Checks if any BMPs are currently holding any of the databases in the list. If a BMP is currently holding a database that is in the list, RPCR issues a WTOR and waits for $n$ number of seconds as specified on the WAITTIME() parameter. If you respond to the outstanding WTOR, the check is performed again and the BMP list is refreshed. This process continues until the wait time expires, until all of the BMPs have been paused or ended, or until you cancel the RPCR operation. If you specified PARTIAL(Y), RPCR processes those databases that are not allocated at the end of the WAITTIME. If you specified PARTIAL(N), RPCR fails if any databases are allocated at the end of the WAITTIME.
- Issues synchronized /DBR or /DBD commands for the databases across all systems on which the database is allocated.
- Dynamically allocates all database data sets for all databases in the list with DISP=OLD to prevent allocation.
- Checks for new allocation records in the RECON data sets to ensure that a database is not allocated after the /DBR or /DBD and before all other databases in the list that were deallocated by RPCR. This check is done to ensure that all of the databases that were deallocated by RPCR have a common recovery point.
- Optionally, issues a /SWI OLDS command on all IMS subsystems on which databases were deallocated.
- Dynamically deallocates all database data sets that were previously allocated as DISP=OLD.
- Issues a CHANGE.DB AUTH command to reset prohibit authorization.
- Optionally restarts the databases, if it was already started before RPCR.
• Issues a DEQ command for each previously issued ENQ to allow BMPs and CICS/ODBA applications to continue.

RPCR uses the IMS Tools Online System Interface to issue all IMS commands. The XCF group name that is used for IMS Tools Online System Interface must be unique within each RECONID and the IMS online subsystems that must share the same RECONs to avoid conflicts with IMS Tools Online System Interface and RPCR.

The RPCR function requires that you specify the target databases using the DBLIST supporting function.

RPCR cannot be used in combination with RECONCOPY. If RECONCOPY is specified with RPCR, the job step terminates.

When RPCR attempts to issue a /DBR or /DBD command for a database, an active BMP or a CICS/ODBA thread can prevent successful deallocation. To prevent that, install the BMP interface and the CICS/ODBA interface. Installing the Extended Functions BMP interface and the CICS/ODBA interface will pause these applications at checkpoint time to allow RPCR to complete. Installing the IMS RE BMP interface and the CICS/ODBA interface is highly recommended.

If the BMP interface is installed, any active BMPs that reference databases that are specified in the DBLIST at the time RPCR is invoked, will be paused at the next checkpoint and continued after RPCR processing completes. If a BMP cannot be paused because it has not reached a checkpoint or the BMP Interface was not installed or used by the active BMP, RPCR displays a list of BMPs so that you can respond accordingly. If a BMP list is displayed, all full-function and HALDB BMPs are displayed first, followed by any FastPath BMPs.

**Related concepts**

“RPID ISPF interface initialization errors” on page 97

---

### RPCR input

Run the Recovery Point Create (RPCR) function by running RPCR.

#### RPCR function syntax

```plaintext
RPCR( CMDTYPE(DBR | DBD)
    SYNCLOG(YES | Y | NO | N)
    STARTDB(YES | Y | NO | N)
    PARTIAL(YES | Y | NO | N)
    WAITTIME(0 – 1800)
)
```

**Parameters**

The RPCR parameters are described here:

**CMDTYPE** (DBR | DBD)

Use the CMDTYPE parameter to specify whether RPCR will issue a synchronized /DBR or a synchronized /DBD command to deallocate the databases. The ability to issue synchronized /DBR and /DBD commands is provided through IMS Tools Online System Interface.
If you specify CMDTYPE(DBR), a synchronized /DBR is issued for all databases in the DBLIST.

If you specify CMDTYPE(DBD), a synchronized /DBD is issued for all full-function databases in the DBLIST. Because there is no /DBD command for fast path databases, a synchronized /DBR will be issued for all fast path databases. The default is CMDTYPE(DBR).

**SYNCLOG** (YES | Y | NO | N)
Use the SYNCLOG parameter to specify whether RPCR will issue a /SWI OLDS command on all connected IMS systems when the RPCR is completed. Performing a synchronized log switch will assist in creating a coordinated set of archived logs associated with RPCR.

The default is SYNCLOG(YES).

**STARTDB** (YES | Y | NO | N)
Use the STARTDB parameter to specify whether RPCR should start the databases when RPCR completes.

If you specify STARTDB(Y), the databases are put back into their original state before RPCR. For example, if a database was not allocated to the online system prior to RPCR, the STARTDB(Y) parameter will be ignored for that database.

If you specify STARTDB(NO), the databases are left in their /DBR or /DBD state.

The default is STARTDB(YES).

**PARTIAL** (YES | Y | NO | N)
Use the PARTIAL parameter to specify whether RPCR will attempt to create a recovery point for some of the specified databases, even though RPCR was unable to create a recovery point for other databases.

If you specify PARTIAL(YES), RPCR will attempt to create a recovery point for as many databases in the DBLIST as possible. If one or more of the databases cannot be deallocated, RPCR is performed for those databases that can be deallocated.

If you specify PARTIAL(NO), RPCR will return an error indicating that it is unable to create a recovery point for at least one database.

The default is PARTIAL(NO).

**WAITTIME** (YES | Y | NO | N)
The WAITTIME parameter specifies the maximum amount of time, in seconds, that RPCR should wait when attempting to deallocate the databases in the DBLIST. If RPCR is unable to deallocate any database in the DBLIST before the specified time has expired, it will return an error message and RPCR for that database will fail. A value of zero indicates that there is no limit to the amount of time RPCR will wait for the database to become deallocated.

**Recommendation:** Install the BMP Pausing interface that is provided with IMS RE so that RPCR can pause any BMPs that have access to the databases in the DBLIST and prevent any new BMPs from starting during the RPCR process. The BMP Pausing interface will temporarily halt processing of a BMP after the next checkpoint to allow the /DBR or /DBD command to complete. Also consider installing the BMP Pausing interface as an SMP/E usermod so that all BMP JCL does not need to be modified.

The default is WAITTIME(300), which is 300 seconds (or 5 minutes).
RPCR usage examples

The Recovery Point Creation function (RPCR) uses several different interfaces to perform its functions.

The interfaces that RPCR uses require specific installation procedures. The interfaces are:
- IMS Tools Online System Interface
- BMP Pausing interface
- ODBA/CICS interface

Example 1: /DBR, perform a /SWI OLDS, and start 3 databases

When the following example code is run, RPCR attempts to:
- /DBR three databases within two minutes (120 seconds) only if all of the databases can be deallocated
- Perform a /SWI OLDS to switch the online log data sets
- Start the databases after RPCR only if they have been deallocated by RPCR

```
DBLIST ( DB (dbd1 dbd2 dbd3) )
RPCR ( CMDTYPE(DBR)
       SYNCLOG(YES)
       STARTDB(YES)
       PARTIAL(NO)
       WAITTIME(120)
)
```

Example 2: /DBD the databases that can be deallocated, then leave databases in the same state

When the following example code is run, RPCR attempts to:
- /DBD any of the two databases within five minutes (300 seconds)
- Leave the affected databases in the same state as the state that was performed

```
DBLIST ( DB (dbd1 dbd2 dbd3) )
RPCR ( CMDTYPE(DBD)
       SYNCLOG(NO)
       STARTDB(NO)
       PARTIAL(YES)
       WAITTIME(300)
)
```

RPCR output

The RPCR function report is written to the RPRPCR output DD.

If you do not specify the RPRPCR DD in the JCL, it is dynamically allocated to SYSOUT=*.
Report output can be written to SYSOUT or to a data set. The characteristics of the output are RECFM=FBA,LRECL=133.
You can also write the report output to the IMS Tools Base for z/OS: IMS Tools Knowledge Base if you specify the REPORT(ITKBOUT=YES)) keyword and parameter.

REPORT(ITKBOUT=YES) is the default.
Report

The following figure shows a sample RPCR report:

IMS Recovery Solution Pack V1.1.0 (5655-V86)
IMS DATABASE FACILITY EXTENDED FUNCTIONS RECOVERY POINT CREATE REPORT Run Time: 11/08/2007 11:56

********************************************************************************
-> Invocation Parameters
********************************************************************************
CMDTYPE: DBR  SYNCLOG: NO
STARTDB: YES  PARTIAL: YES
WAITTIME: 00000010

********************************************************************************
-> Database List
********************************************************************************

<table>
<thead>
<tr>
<th>DATABASE</th>
<th>AREA/PART</th>
<th>DD</th>
<th>TYPE</th>
<th>DSN</th>
<th>PART-TYPE</th>
<th>DLR-PARTNER</th>
</tr>
</thead>
<tbody>
<tr>
<td>F1O1P1</td>
<td>F1O1P11</td>
<td>FF</td>
<td>IMS</td>
<td>IMSTOOL.IMSRE.QA.Q91A.F1O1P1.F1O1P11</td>
<td></td>
<td></td>
</tr>
<tr>
<td>F1O1X1</td>
<td>F1O1X11</td>
<td>FF</td>
<td>IMS</td>
<td>IMSTOOL.IMSRE.QA.Q91A.F1O1X1.F1O1X11</td>
<td></td>
<td></td>
</tr>
<tr>
<td>F2O1P1</td>
<td>F2O1P11</td>
<td>FF</td>
<td>IMS</td>
<td>IMSTOOL.IMSRE.QA.Q91A.F2O1P1.F2O1P11</td>
<td></td>
<td></td>
</tr>
<tr>
<td>F2O2P1</td>
<td>F2O2P11</td>
<td>FF</td>
<td>IMS</td>
<td>IMSTOOL.IMSRE.QA.Q91A.F2O2P1.F2O2P11</td>
<td></td>
<td></td>
</tr>
<tr>
<td>F2O2P1</td>
<td>F2O2P12</td>
<td>FF</td>
<td>IMS</td>
<td>IMSTOOL.IMSRE.QA.Q91A.F2O2P1.F2O2P12</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

********************************************************************************
-> Recovery Point Create Report
********************************************************************************

<table>
<thead>
<tr>
<th>DATABASE</th>
<th>AREA/PART TYPE</th>
<th>RPCR TIME (UTC)</th>
<th>RPCR TIME (LOCAL)</th>
<th>INFORMATIONAL MESSAGES</th>
</tr>
</thead>
<tbody>
<tr>
<td>F1O1X1</td>
<td>IMS</td>
<td>2007.312 19:56:18.2 +00:00 2007.312 11:56:18.2 -08:00</td>
<td>SUCCESS</td>
<td></td>
</tr>
<tr>
<td>F2O1P1</td>
<td>IMS</td>
<td>2007.312 19:56:18.2 +00:00 2007.312 11:56:18.2 -08:00</td>
<td>SUCCESS</td>
<td></td>
</tr>
</tbody>
</table>

Figure 5. Sample RPCR report

The RPCR report contains the following sections:

Invocation Parameters

This section of the report shows the options that were used for this run of RPCR process.

CMDTYPE

This field indicates whether a DBR or DBD command was requested.

STARTDB

This field indicates whether the databases will be restarted after RPCR is complete.

WAITTIME

This field indicates the value specified for the WAITTIME() parameter.

SYNCLOG

This field indicates whether a synchronized log switch will be performed after RPCR is complete.

PARTIAL

This field indicates whether a partial RPCR is allowed.
Database List
This section of the report shows the databases that were processed in this run.

DATABASE
This column specifies the database name.

AREA/PART
This column specifies the area name for fast path databases or the partition name for HALDB databases. This column is blank for full function databases.

DD
This column specifies the DD name that is associated with this area, partition, or full function database.

TYPE
This column specifies the type of database that was processed. Valid values are FF for full function databases, FP for fast path areas, and HALD for HALDB partitions.

DSN
This column specifies the data set name that is associated with this area, partition, or full function database.

PART-TYPE
For HALDB partitions, this column specifies the type of data set. Valid values are INDEX for the index portion of the HALDB, ILE for the index list element (or ILDS) portion of the HALDB, or DATA for the data portion of the HALDB.

OLR-PARTNER
For HALDB partitions that are OLR-capable, this column specifies the OLR partner DD name that is associated with the partition.

Recovery Point Create Report
This section of the report includes a line for each database, area, and partition that was processed and indicates the final status.

DATABASE
This column specifies the name of the databases that were processed.

AREA/PART
This column specifies the fast path area name or HALDB partition name that is associated with the database. If the database is full function, this column is blank.

TYPE
This column indicates the type of database that was processed. Valid values are IMS for full function database, FP for fast path area, and HALD for HALDB partition.

RPCR TIME (UTC)
This column indicates the RPCR time in coordinated universal time (UTC) format. If RPCR failed, this column will be blank.

RPCR TIME (LOCAL)
This column indicates the RPCR time in local format. If RPCR failed, this column will be blank.

INFORMATIONAL MESSAGES
This column indicates the final disposition for the database. The valid messages are shown in the following table.
### Table 4. RPCR report informational messages and their descriptions

<table>
<thead>
<tr>
<th>Message text</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>INVALID TOKEN FOR FUNCTION</td>
<td>Invalid function token passed to RPCR. Contact IBM Software Support.</td>
</tr>
<tr>
<td>TOKEN NOT FOR SAME TCB</td>
<td>Invalid TCB token passed to RPCR. Contact IBM Software Support.</td>
</tr>
<tr>
<td>UNABLE TO JOIN XCF GROUP</td>
<td>An attempt to join the XCF group for IMS Tools Online System Interface has failed. Ensure that you have specified the correct XCF group in the RECONID record and the IMS subsystems were initialized correctly with IMS Tools Online System Interface.</td>
</tr>
<tr>
<td>TARGET SYSTEM NOT AVAILABLE</td>
<td>An attempt to issue an IMS command for one or more target IMS systems has failed. Ensure that the IMS system is still active.</td>
</tr>
<tr>
<td>UNABLE TO OBTAIN RECON INFO</td>
<td>RPCR was unable to access the RECON data sets to obtain RECON information. Review the job log for additional messages that indicate the specific error.</td>
</tr>
<tr>
<td>UNABLE TO GET ONLINE DB STATUS</td>
<td>RPCR was unable to access the RECON data sets to obtain database information. Review the job log for additional messages that indicate the specific error.</td>
</tr>
<tr>
<td>UNABLE TO SWITCH LOG</td>
<td>RPCR was unable to complete the /SWI OLDS command processing. Review the job log and the IMS messages for additional messages that indicate the specific error.</td>
</tr>
<tr>
<td>UNABLE TO DBR A DB</td>
<td>RPCR was unable to complete the /DBR command processing for one or more databases. Review the job log and the IMS messages for additional messages that indicate the specific error.</td>
</tr>
<tr>
<td>UNABLE TO DBD A DB</td>
<td>RPCR was unable to complete the /DBD command processing for one or more databases. Review the job log and the IMS messages for additional messages that indicate the specific error.</td>
</tr>
<tr>
<td>UNABLE TO START DB</td>
<td>RPCR was unable to start one or more databases. Review the job log and the IMS messages for additional messages that indicate the specific error.</td>
</tr>
<tr>
<td>OPTIONS CONFLICT WITH DB TYPE</td>
<td>The RPCR options that were specified conflict with database type. Contact IBM Software Support.</td>
</tr>
<tr>
<td>GETMAIN REQUEST FAILED</td>
<td>Unable to acquire storage. Ensure that the region size for the Extended Functions job is adequate and that storage is available.</td>
</tr>
<tr>
<td>NAME TOKEN SERVICES FAILED</td>
<td>An error was encountered using MVS name token services. Contact IBM Software Support.</td>
</tr>
<tr>
<td>UNABLE TO LOAD PROGRAM</td>
<td>RPCR was unable to load a required program. Ensure that Extended Functions has been correctly installed and that all required load modules are included in the JOBLIB or STEPLIB.</td>
</tr>
<tr>
<td>DYNAMIC ALLOCATION FAILED</td>
<td>Dynamic allocation failed for one or more database data sets. Review the job log for details on the specific data set. This normally indicates that a job or user outside of IMS has the data set in use.</td>
</tr>
<tr>
<td>UNABLE TO LOCATE SUBTASK SVCS</td>
<td>Unable to locate subtask services. Contact IBM Software Support.</td>
</tr>
</tbody>
</table>
Table 4. RPCR report informational messages and their descriptions (continued)

<table>
<thead>
<tr>
<th>Message text</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SVCS NOT AVAILABLE FOR SUBTASK</td>
<td>Subtask services are unavailable. Contact IBM Software Support.</td>
</tr>
<tr>
<td>ERROR STARTING SERVICE SUBTASK</td>
<td>Unable to start the subtask services. Contact IBM Software Support.</td>
</tr>
<tr>
<td>IMS ENVIRONMENTAL ERROR</td>
<td>RPCR encountered an error within the IMS environment. contact IBM Software Support.</td>
</tr>
<tr>
<td>NOT APF AUTHORIZED</td>
<td>The RPCR program is not running in an APF authorized environment. Ensure that the load libraries in the JOBLIB or STEPLIB are APF authorized.</td>
</tr>
<tr>
<td>MISSING IMS DD</td>
<td>The specific RPCR options that were specified require the IMS DD to be specified. Add the appropriate IMS DD concatenation.</td>
</tr>
<tr>
<td>INCORRECT DSNAME FOR RECON</td>
<td>The data set name that is associated with the RECON that is being accessed does not match the data set name that was passed to RPCR. Ensure that the same RECON data sets are accessed by the Extended Functions job (JCL / RECONID) and the online IMS subsystem.</td>
</tr>
<tr>
<td>DB IN USE FOR TIMEOUT DURATION</td>
<td>One or more databases were in use for the timeout duration, and RPCR failed. If you specified PARTIAL(Y), some databases might have succeeded. If you specified PARTIAL(N), RPCR failed.</td>
</tr>
<tr>
<td>DB NOT PROCESSED, PARTIAL(N)</td>
<td>One or more databases were in use for the timeout duration and PARTIAL(N) was specified. All other databases in the list were skipped.</td>
</tr>
<tr>
<td>DB IS NOT REGISTERED</td>
<td>One or more databases that were passed to RPCR were not registered to DBRC. Ensure that all databases are registered.</td>
</tr>
<tr>
<td>DB TYPE IS UNSUPPORTED</td>
<td>The database type is either unsupported by RPCR or cannot be determined.</td>
</tr>
<tr>
<td>UNABLE TO LOCATE DBD</td>
<td>RPCR could not locate the DBD in the IMS DD concatenation. Ensure that the correct blocks are located in the IMS DD concatenation.</td>
</tr>
<tr>
<td>UNABLE TO LOCATE PART INFO</td>
<td>RPCR could not locate information for a HALDB partition. Ensure that the correct blocks are located in the IMS DD concatenation.</td>
</tr>
<tr>
<td>DUPLICATE DBD/PART IN DBLIST</td>
<td>A duplicate database or partition was passed to RPCR in the DBLIST. Correct the DBLIST.</td>
</tr>
<tr>
<td>ALLOCATED AFTER DBR</td>
<td>RPCR has determined that one or more of the databases processed were allocated after they were processed by the /DBR command and before all other databases in the list were processed by the /DBR command. The RPCR process must fail because it cannot ensure that a common recovery point was created for all of the databases.</td>
</tr>
<tr>
<td>CMDTYPE(DBD) NOT VALID FOR DB</td>
<td>Internal error, contact IBM Software Support.</td>
</tr>
<tr>
<td>DB TYPE NOT PROVIDED</td>
<td>Internal error, contact IBM Software Support.</td>
</tr>
<tr>
<td>DB DSLIST NOT PROVIDED</td>
<td>Internal error, contact IBM Software Support.</td>
</tr>
</tbody>
</table>
Table 4. RPCR report informational messages and their descriptions (continued)

<table>
<thead>
<tr>
<th>Message text</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DB IS CURRENTLY BEING USED</td>
<td>One or more databases were in use for the timeout duration, and RPCR failed.</td>
</tr>
</tbody>
</table>
Chapter 7. Recovery Point Identification (RPID) function

The Recovery Point Identification function (RPID) will help you identify common recovery time spans for databases or help you identify a valid recovery point for a single database in IBM IMS Recovery Solution Pack for z/OS: IMS Database Recovery Facility: Extended Functions.

Topics:
- “RPID overview”
- “RPID input” on page 66
- “RPID usage examples” on page 67
- “RPID output” on page 67

RPID overview

The RPID function can be used to analyze the RECON data sets and locate valid recovery time spans for one or more databases without the need to manually compare DBRC histories for each database.

A recovery time span (RTS) is a period of time during which a database is not allocated and, therefore, the database can be recovered. Although valid recovery times might exist during which the database is allocated, RPID does not currently analyze log records and therefore, does not report on these recovery times. The following sequence of events illustrates the concept of a simple RTS:

Event 1:
Database X was started with the /STA command at Time 1:
07.320 08:53:37.924206-0800
Allocated => 07.320 08:53:37.924206-0800 to 07.320 08:54:16.730540-0800

Event 2:
Database X was unallocated with the /DBR command at Time 2:
07.320 08:54:16.730541-0800

Event 3:
Database X was started with the /STA command at Time 3:
07.320 08:59:01.75235-0800

Time Span => 07.320 08:54:16.730541-0800 to 07.320 08:59:01.75235-0800

The sequence of these three events shows that database X was successfully started and allocated at Time 1. The database remained allocated until it was unallocated at Time 2. Database X was successfully started again at Time 3. This sequence shows that an RTS exists between Time 2 and Time 3. The database can be recovered successfully to any timestamp within this RTS.

RPID analyzes database allocation records in the RECON data sets to identify RTSs to which one or more databases can be recovered by using timestamp recovery (TSR). RPID also analyzes database recovery records in the RECONS to identify recovery point timestamps to which individual databases can be recovered by using TSR.
A recovery point is a timestamp that can be used as input to IMS Database Recovery Facility, or in some cases to other database recovery utilities such as, the standard IMS Recovery utility (DFSURDB0), as a time to which a database can be restored. These recovery points include times when a database was the object of any of the following utility runs, all of which are recorded in RECON recovery records:

- A batch image copy (IC) run
- An online IC run (in cases where the databases are not truly allocated)
- A complete change accumulation (CA) run
- A recovery run (Full or Partial), or an offline reorganization run

The run times of HALDB IMS Online Reorganizations (IMS OLRS) and IMS Online Reorganization Facility are also analyzed. For IMS OLR and IMS Online Reorganization Facility, RPID indicates whether the databases were truly allocated and therefore, whether the run time is a valid TSR point.

HALDB partitions that have been part of an IMS Online Reorganization have two sets of DBDSs, that are known as A-J and M-V. At any point in time, either set might be active and the other set might be inactive (an IMS OLR is not in progress), or either set might be a shadow (an IMS OLR is in progress). Both sets of DBDSs are recognized as one logical partition when RPID analyzes their associated RECON records.

When RPID lists the recovery time spans it also indicates specific recovery points (timestamps) that are available for recovery including:

- Times of complete change accumulation runs.
- Times of batch image copy runs, and times of online image copy runs when the databases are not allocated.
- Times of offline reorganizations and online reorganizations when the databases are not allocated. In the second case (OLR), RPID indicates whether there was a subsequent image copy during the recovery time range.
- Times of database recovery, either full or partial recoveries.

To manually create a recovery time span or recovery point for a TSR for all of the databases in a group, run the Recovery Point Create (RPCR) function that is described in Recovery Point Creation function (RPCR).

The RPID function requires that you specify the target databases by using the DBLIST supporting function.

The RPID function can use the RECON copy data sets that are created by the RECONCOPY supporting function.

**Attention:** Use RECONCOPY in conjunction with RPID to prevent I/O to the live RECON data sets.

**RPID input**

Run the Recovery Point Identification function by running RPID.

**RPID function syntax**

```
RPID( RANGE(range-specification) )
```
Parameters

The RPID parameters are described here:

RANGE(range-specification)

Use the required RANGE parameter to specify the beginning and ending time frame (or the time range) that is used by RPID to locate recovery time spans. There is no default for this keyword.

The format of the range specification can be found in the "TIME and RANGE parameter processing" on page 30.

RPID usage examples

Use the Recovery Point Identification function (RPID) to identify valid recovery time spans that occur between specified time spans for a single or group of databases.

You can use the RPID function to identify common recovery time spans for the following database types:

- A single database that is not associated with or related to any other database
- A group of databases that logically belong together (for example, a CAGRP, a RECOVGRP, or other DB groupings)

Example of identifying valid recovery times

To identify valid recovery time spans that occurred between 07.291 06:30:01.1 and 07.291 11:30:01.1 for one HDAM database and one HIDAM database with a secondary index and an external logical, you would specify these control cards:

```plaintext
DBLIST ( DB(HDAM1 HIDAMDB1 HIDAMIX PRIMESX EXTLOGDB) )
RPID ( RANGE ('07.291 06:30:01.1','07:291 11:30:01.1') )
```

In this example, RPID determines a separate recovery time span for each database that you specified in the DBLIST. This recovery time span is the timeframe in which a database was not allocated during the specified range. RPID also determines recovery time spans that are common to all of the databases in the DBLIST. This timeframe is a period during which all of the databases were not allocated at the same time during the specified range.

RPID output

The output of the RPID function is written to the RPRPID DD.

If you do not specify the RPRPID DD in the JCL, it is dynamically allocated to SYSOUT=* You can write the RPID report output to SYSOUT or to a data set. The characteristics of the output are RECFM=FBA,LRECL=133. You can write the report to the IMS Tools Base for z/OS: IMS Tools Knowledge Base repository by specifying the REPORT(ITKBOUT(YES)) keyword and parameter.

The default is REPORT(ITKBOUT(YES)).

The RPID report consists of three report sections. The following figure shows the first two sections of the RPID report:
The RPID report contains the following sections:

**Invocation Parameters**
This section of the report shows the options that RPID used for its processing.

**Time Format**
This field indicates whether the timestamps that are displayed in the report are in Local or UTC (Coordinated Universal Time) format. Specify this parameter on the REPORT keyword by using the TIMEFMT() parameter.

Figure 6. Example of an Extended Functions Recovery Point ID report
**Time Range**
This field displays the formatted time range that you specified on the RPID RANGE() parameter.

**Database List**
This section of the report shows the databases that were processed in this RPID run.

**DATABASE**
This column specifies the name of the database that was processed.

**AREA/PART**
This column specifies the area name for fast path databases or the partition name for HALDB databases that were processed. This column is blank for full function databases.

**DD**
This column specifies the DD name that is associated with this area, partition, or full function database.

**TYPE**
This column specifies the type of database that was processed. Valid values are FF for full function databases, FP for fast path areas, and HALD for HALDB partitions.

**DSN**
This column specifies the data set name that is associated with this area, partition, or full function database.

**PART-TYPE**
For HALDB partitions, this column specifies the type of data set that was processed. Valid values are INDEX for the index portion of the HALDB, ILE for the index list element (or ILDS) portion of the HALDB, or DATA for the data portion of the HALDB.

**OLR-PARTNER**
For HALDB partitions that are IMS Online Reorganization (OLR) capable, this column specifies the OLR partner DD name that is associated with the partition.

**Recovery Time Spans and Points for Time Stamp Recovery**
The next section shows the following information:

- Recovery Time Spans that are common to all database entries
- Recovery Time Spans for Individual database entries
- Recovery Points (Utility End Times) for individual database entries

The following figure shows an example Recovery Time Spans and Points for Time Stamp Recovery:
The Recovery Time Spans and Points for Time Stamp Recovery report contains the following sections:

**Recovery Time Spans Common to All Entries in the DBLIST:**

This section of the report lists all of the recovery time spans (RTSs) that are

---

**Recovery Time Spans for Individual Entries in the DBLIST:**

DBD=POHIDKA  DDN=POHIDKAN DB_Type=HALDB-Part Partner_DD=POHIDKAB Master_DB=DBOHIDK5
05.283 14:15:40.100000+0000 to 07.139 10:54:27.712196+0000 Last_IMSID=n/a
07.139 11:10:19.946777+0000 to 07.364 14:15:40.100000+0000 Last_IMSID=IMS1

DBD=POHIDKA  DDN=POHIDKAM DB_Type=HALDB-Part Partner_DD=POHIDKAA Master_DB=DBOHIDK5
05.283 14:15:40.100000+0000 to 07.139 10:59:31.214366+0000 Last_IMSID=n/a
07.139 11:00:17.329938+0000 to 07.139 11:00:38.972179+0000 Last_IMSID=IMS1
07.139 11:10:19.562406+0000 to 07.364 14:15:40.100000+0000 Last_IMSID=IMS1

DBD=POHIDKA  DDN=POHIDKAB DB_Type=HALDB-Part Partner_DD=POHIDKAN Master_DB=DBOHIDK5
05.283 14:15:40.100000+0000 to 07.139 10:59:31.192224+0000 Last_IMSID=n/a
07.139 11:00:17.329938+0000 to 07.139 11:00:38.951749+0000 Last_IMSID=IMS1
07.139 11:10:19.478393+0000 to 07.364 14:15:40.100000+0000 Last_IMSID=IMS1

DBD=POHIDKA  DDN=POHIDKAA DB_Type=HALDB-Part Partner_DD=POHIDKAM Master_DB=DBOHIDK5
05.283 14:15:40.100000+0000 to 07.139 10:59:31.192224+0000 Last_IMSID=n/a
07.139 11:00:17.329938+0000 to 07.139 11:00:38.951749+0000 Last_IMSID=IMS1
07.139 11:10:19.478393+0000 to 07.364 14:15:40.100000+0000 Last_IMSID=IMS1

DBD=POHIDKB  DDN=POHIDKBN DB_Type=HALDB-Part Partner_DD=POHIDKBB Master_DB=DBOHIDK5
05.283 14:15:40.100000+0000 to 07.139 10:58:42.471480+0000 Last_IMSID=n/a
07.139 11:00:17.329938+0000 to 07.139 11:00:42.276666+0000 Last_IMSID=IMS1
07.139 11:10:19.690115+0000 to 07.364 14:15:40.100000+0000 Last_IMSID=IMS1

---

**Recovery Points (Utility End Times) for Individual Entries in the DBLIST:**

DBD=POHIDKA  DDN=POHIDKAN DB_Type=HALDB-Part Partner_DD=POHIDKAB Master_DB=DBOHIDK5
07.139 10:58:40.700316+0000 -- Batch-IC
07.139 11:11:19.946777+0000 -- Batch-IC
07.139 11:11:32.500000+0000 -- 07.139 11:11:32.500000+0000 Full-Recovery
07.139 11:12:26.800000+0000 -- 07.139 11:12:26.800000+0000 Full-Recovery
07.139 11:12:26.812300+0000 -- 07.130 11:12:26.800000+0000 Concurrent-IC

DBD=POHIDKB  DDN=POHIDKBB DB_Type=HALDB-Part Partner_DD=POHIDKBB Master_DB=DBOHIDK5
07.139 10:58:42.471480+0000 -- 00.000 00:00:00.000000+0000 Batch-IC
07.139 11:11:32.500000+0000 -- 07.139 11:11:32.500000+0000 Full-Recovery
07.139 11:12:26.800000+0000 -- 07.139 11:12:26.800000+0000 Full-Recovery
07.130 11:12:26.812300+0000 -- 07.130 11:12:26.800000+0000 Concurrent-IC

---

Figure 7. Example Recovery Time Spans and Points for Time Stamp Recovery report

The Recovery Time Spans and Points for Time Stamp Recovery report contains the following sections:

**Recovery Time Spans Common to All Entries in the DBLIST:**

This section of the report lists all of the recovery time spans (RTSs) that are
common for all of the databases in the DBLIST. The RTSs that are shown are based on allocation ranges bounded by the RANGE parameter input. If more than one common RTS is found, each one is listed. If no common RTS was found, no lines are printed under this heading. Each entry shows the beginning and ending timestamp that is associated with the common recovery time span.

When you review an RTS, any timestamp can be used on a Time Stamp Recovery job for recovering any and all of the databases on the DBLIST, including the beginning and the ending of the RTS. When you review this information, if it appears that the RTSs are more limited than what you expected, you can find the reasons for the limitation in the Utility End Times section of the report. For example, if any of the databases were reorganized and no image copy was taken after the reorganization, the end of the RTS will be the time immediately before the reorganization job started.

Recovery Time Spans for Individual Entries in the DBLIST:
This section of the report lists the RTSs for individual databases in the DBLIST, which are based on allocation ranges bounded by the RANGE parameter input. The last IMSID to allocate the database just before the beginning of the RTS is also listed.

If you recognize the Last_IMSID field as a DL/I batch job, the time to the left of the two dashes (--) is the log start time for the batch job, and the time to the right of the two dashes (--) is the log end time for the batch job.

If the database is a HALDB partition, the partner partition, and HALDB master database name are listed.

Any timestamp that is bounded by and includes the beginning and ending timestamps of the RTS is a valid recovery TSR time for the individual database. When RPID calculates the Common Recovery Time Spans, it assumes that if one of the HALDB partners is allocated the other HALDB partner is also allocated. And so, a TSR that is using a recovery point can recover both of the partner partitions.

Recovery Points (Utility End Times) for Individual Entries in the DBLIST:
This section of the report lists the subsequent recovery points (that is, the utility run and end times within the RANGE specification for individual entries in the DBLIST).

The following table lists the utility information that can be displayed:

<table>
<thead>
<tr>
<th>What is displayed in the report</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Batch-IC</td>
<td>Batch image copy</td>
</tr>
<tr>
<td>Concurrent-IC</td>
<td>Concurrent image copy</td>
</tr>
<tr>
<td>User-IC</td>
<td>User image copy</td>
</tr>
<tr>
<td>SMS-IC-DB-Exclusive</td>
<td>SMS image copy – database exclusive</td>
</tr>
<tr>
<td>SMS-IC-DB-Shared</td>
<td>SMS image copy – database shared</td>
</tr>
<tr>
<td>Full-Recovery</td>
<td>Full recovery</td>
</tr>
<tr>
<td>Partial-Recovery</td>
<td>Partial recovery</td>
</tr>
<tr>
<td>Offlin-REORG+SubseqIC</td>
<td>Offline reorganization and subsequent image copy</td>
</tr>
<tr>
<td>Offlin-REORG-SubseqIC</td>
<td>Offline reorganization and no subsequent image copy</td>
</tr>
</tbody>
</table>
### Table 5. RPID DB recovery points or utility end times for DBLIST entries (continued)

<table>
<thead>
<tr>
<th>What is displayed in the report</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Online-REORG+IC</td>
<td>Online reorganization and image copy</td>
</tr>
<tr>
<td>Online-REORG-IC</td>
<td>Online reorganization and no subsequent image copy</td>
</tr>
<tr>
<td>Full-Rcvr+SubseqIC</td>
<td>Full recovery and subsequent image copy</td>
</tr>
<tr>
<td>Full-Rcvr-SubseqIC</td>
<td>Full recovery and no subsequent image copy</td>
</tr>
<tr>
<td>Partial-Rcvr+SubseqIC</td>
<td>Partial recovery and subsequent image copy</td>
</tr>
<tr>
<td>Partial-Rcvr-SubseqIC</td>
<td>Partial recovery and no subsequent image copy</td>
</tr>
<tr>
<td>Complete-Change-Accum</td>
<td>Complete change accumulation</td>
</tr>
</tbody>
</table>

### RPID ISPF interface for viewing RPID reports and generating JCL statements

The RPID ISPF interface plays an important role for viewing RPID reports and generating JCL for a recovery. The RPID ISPF interface allows you to easily select database components and recovery times. After specifying database components and recovery times, you can use the RPID ISPF interface to generate IMS Database Recovery Facility control statements. For viewing RPID reports and generating control statements, you must have an understanding of the IMS Recover Point Identification ISPF interface.

**Related concepts**

[Chapter 10, “IMS Database Recovery Facility job generation database component and recovery time selection,” on page 93](#)
Chapter 8. Verify Recovery Assets (VERIFY) function

The VERIFY function ensures that the necessary data sets exist and that they are usable before performing a recovery with IBM IMS Recovery Solution Pack for z/OS: IMS Database Recovery Facility: Extended Functions.

Topics:
- “VERIFY overview”
- “VERIFY input”
- “VERIFY usage example” on page 75
- “VERIFY output” on page 75

VERIFY overview

The Verify Recovery Assets function (VERIFY) examines the RECON data sets and identifies the recovery assets that are required to recover one or more databases.

Specifically, the VERIFY function performs one or all of the following functions:
- Lists or prints the recovery assets that are required to recover the specified databases
- Allocates all recovery assets to ensure that they exist
- Opens all recovery assets to ensure that they are valid

The VERIFY function requires that you specify the target databases by using the DBLIST supporting function.

The VERIFY function can use the RECON copy data sets that are created by the RECONCOPY supporting function.

Attention: Use RECONCOPY in conjunction with VERIFY to prevent I/O to the live RECON data sets.

VERIFY input

Run the Verify Recovery Assets (VERIFY) function by running VERIFY.

VERIFY function syntax

```
VERIFY( TYPE(LIST | ALLOC | OPEN)
TIME( Time stamp)
RCVTYPE(TSR | PITR)
SOURCE(PRI | SEC)
)
```

Parameters

The VERIFY parameters are described in the following list:

```
TYPE(LIST | ALLOC | OPEN)
```

Use the TYPE parameter to specify the verification level that VERIFY will perform.
TYPE(LIST) provides a list of all log, image copy, and change accumulation data sets that are required for recovery. The list contains information that is extracted from DBRC about each data set.

TYPE(ALLOC) provides the same detail as TYPE(LIST), but TYPE(ALLOC) also allocates each data set with a deferred mount for tape data sets to verify the existence and physical accessibility of each data set.

TYPE(OPN) provides the same detail as TYPE(LIST) and TYPE(ALLOC) but TYPE(OPN) also opens each data set to verify its availability and usability.

The default is TYPE(LIST).

**TIME** *(timestamp)*

The TIME parameter provides a way to obtain information about recovery assets for a timestamp recovery, including recoveries to any prior point in time. TIME specifies the timestamp value that determines how much log and change accumulation data would be processed during recovery. Only log data sets that contain data from before the recovery timestamp will be reported. Only change accumulation data sets from before an allocation boundary, as defined by DBRC, will be reported. TIME() is an optional parameter. If TIME() is not specified, the VERIFY function will process recovery assets for performing a full recovery.

You must supply the timestamp value within an interval where the databases and fast path areas in the input list are unallocated if you specified RCVTYPE(TSR). The timestamp value can be any earlier point-in-time if RCVTYPE(PITR) is specified.

The format of the timestamp specification can be found in "TIME and RANGE parameter processing" on page 30.

**RCVTYPE** *(TSR | PITR)*

Use the RCVTYPE parameter to specify the type of recovery that is being requested.

RCVTYPE is an optional parameter.

TSR is the default parameter if TIME() is specified. TSR specifies that recovery assets verification for a timestamp recovery is to be performed. The timestamp must be such that no ALLOC records in the RECON span the timestamp that is supplied with TIME(). All RECON ALLOC records that have an ALLOC time that earlier than the recovery time must have one of the following specifications:

**DEALLOC time:**

The DEALLOC time must be earlier than the recovery time.

**STOP time:**

The associated PRILOG record must have a STOP time that is earlier than the recovery time. The associated PRILOG record is the one with a START time equal to the START time of the PRILOG START time specified in the ALLOC record.

Use the PITR parameter to specify that recovery assets verification for a timestamp recovery to any prior point in time (Point-In-Time Recovery) is to be performed. At the timestamp that is specified, database data sets and areas can be in any allocation state as recorded in the RECON data sets. Log data sets that span allocation intervals for the database data sets and FastPath areas in the input list, up to and including the specified recovery time, are reported. If PITR is specified, the timestamp that is supplied can
be any value before the current time, but after the run time of a valid image copy that is registered with DBRC.

**SOURCE**(*PRI | SEC*)

Use the SOURCE parameter to specify the source of image copies and log data sets for the recovery process.

If SOURCE(*PRI*) is specified, primary image copy and log data sets are used as the source for the verify process except for any that are marked as invalid in the RECON data sets. The status, as recorded in the RECON data sets, of the primary image copy and log data sets is reported.

If SOURCE(*SEC*) is specified, secondary image copy and log data sets are verified as the source for the recovery process. The status, as recorded in the RECON data sets, of the secondary image copy and log data sets is reported. The default is SOURCE(*PRI*).

**VERIFY usage example**

Use the Verify Recovery Assets function (VERIFY) to verify the availability of recovery assets.

You can set up VERIFY to run on a scheduled basis to periodically check for problems that might affect database recovery. VERIFY can also be used at the disaster recovery (DR) site to ensure that your DR site is recoverable. All recovery assets that are recorded in DBRC must be usable.

If you are attempting to recover a set of databases to a specific time, and you want to identify any problems related to image copies, change accumulation data sets, and log data sets that will be used as input to that recovery, specify the following control cards to report on all of the assets for all of the databases in the DBLIST.

**Example**

The following example uses primary log records and primary image copy data sets for verification. It allocates or opens all of the assets to verify that the data sets are usable. It reports on only those assets that are recorded before the timestamp.

```bash
DBLIST( DB (DB1 DB2 DB3) )
VERIFY( TYPE(OPEN)
   TIME(*timestamp*)
   RCVTYPE(TSR)
   SOURCE(PRI)
)
```

**VERIFY output**

The output of the VERIFY function is written to the RPTVERFY DD.

If you do not specify the RPTVERFY DD in the JCL, it is dynamically allocated to SYSPUT=*. Report output can be written to SYSPUT or to a data set. The characteristics of the output are RECFCM=FBA, LRECL=133. Report output can also be written to the IMS Tools Base for z/OS: IMS Tools Knowledge Base by specifying the REPORT(ITKBOU(YES)) keyword and parameter.

The default is REPORT(ITKBOU(YES)).
The Verify Asset report consists of three main report sections. The following figure shows the sections of the Verify Asset report.

---

**Invocation Parameters**

- **PROCESS**: VERIFY(OPEN)  
  - **TYPE**: TSR  
  - **SOURCE**: PRI  
  - **RCVTIME**: 2007.323 14:01:01.100000 -08:00

---

**Summary Report**

- **Database Name**  
  - **DD/Area Name**: F2O2P1  
  - **DSID**: F2O2P1  
  - **Data Set Type**: Full Function  
  - **Status**: No errors encountered

---

**Data Set I/O Report**

- **Recover to point**: 2007.323 14:01:01.100000 -08:00

---

The major report sections are described here:

**Invocation Parameters**

This section of the report shows the options that VERIFY used for its processing.

**PROCESS**

This field indicates what kind of verification is done on the assets (LIST, ALLOC, or OPEN).

**SOURCE**

This field displays whether to verify primary or secondary image copy records. In the last report sample, the primary image copies are verified.

**RCVTIME**

This field displays the recovery time during which you want to verify assets for this recovery.

---

*Figure 8. Example of an IMS Database Recovery Facility: Extended Functions Verify Asset report*
TYPE  This field is the type of recovery, either TSR or PITR, that is requested.

Summary Report
This section of the report displays the databases that were selected for verification and the overall status of each database and DD name. The following list describes the column fields that are displayed:

Database Name
This column specifies the database name.

DD/Area Name
This column specifies the area name for fast path databases or the partition name for HALDB databases. This column is blank for full function databases.

DSID
This column specifies the data set identification number that is associated with this DD or AREA name.

Data Set Type
This column specifies the type of database: full-function, Fast Path, or HALDB.

Status
This column specifies the overall status of all of the assets that were verified.

Data Set I/O Report
This section of the report displays the individual status of each asset that has been processed by VERIFY. It begins by showing the Recover-To Point, which is the value that you specified on the VERIFY keyword. This report consists of three sections. Each section shows a different type of recovery asset:

• Image copy data sets
• Change accumulation data sets
• Log data sets

Image Copy Data Sets

Image Copy Data Set Name
This column specifies the name of the image copy data set for each database data set that was verified.

Volume Serial
This column specifies the volume serial number of the image copy data set for each database data set that was verified.

IC Type
This column specifies the type of image copy for each database data set that was verified. Valid values for this field are shown in the following table:

<table>
<thead>
<tr>
<th>IC Type listed</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>STD</td>
<td>Standard Image Copy</td>
</tr>
<tr>
<td>IC2</td>
<td>Image Copy 2</td>
</tr>
<tr>
<td>IC EXT</td>
<td>Extended Image Copy</td>
</tr>
<tr>
<td>HISAM UNL</td>
<td>HISAM Unload</td>
</tr>
</tbody>
</table>

Table 6. Valid values for IC Type in the Verify Asset report
Status of Allocate/Open
This column displays the return code from the ALLOC or OPEN process. If the process was successful, the return code is zero.

Change Accumulation Data Sets

Change Accum Data Set Name
This column specifies the name of the change accumulation data set for each database data set that was verified.

Volume Serial
This column specifies the volume serial number of the change accumulation data set for each database data set that was verified.

Status of Allocate/Open
This column shows the return code from the ALLOC or OPEN process. If the process was successful, the return code will be zero.

Log Data Sets

Log Data Set Name
This column specifies the name of the log data set for each database data set that was verified.

Volume Serial
This column specifies the volume serial number of the log data set for each database data set that was verified.

IMS SYSID
This column shows the SYSID that is associated with the IMS that created the log.

Status of Allocate/Open
This column shows the return code from the ALLOC or OPEN process. If the process was successful, the return code is zero.
Chapter 9. RECON cleanup (RCU) function

The RECON cleanup function (RCU) prepares the RECON data sets for recovery at a disaster recovery (DR) site in IBM IMS Recovery Solution Pack for z/OS: IMS Database Recovery Facility: Extended Functions.

Topics:
- “RCU overview”
- “Preparing for site recovery” on page 80
- “Clean up time” on page 81
- “RECON records that are updated” on page 81
- “Special considerations for using RCU” on page 83
- “RCU input” on page 84
- “RCU usage example” on page 84
- “RCU output” on page 85
- “RCU return and reason codes” on page 89

RCU overview

A copy of the RECON must be made and modified to match the data sets that exist at the recovery site to perform valid recoveries. RCU will create a clean copy of the RECON data sets at a particular point in time for recovery.

In general, a recovery site contains all the data that is needed to restore IMS databases to a known active site time. The following data is needed for recovery:
- Image copy data sets taken at the active site
- IMS log data from the active site (logs, change accumulation data sets, or both)
- A RECON that accurately reflects the activities that have taken place at the active site

Image copies

Database recovery works at the database data set (DBDS) level; each DBDS needs to be restored individually and starts with an image copy. Restoring an image copy creates an exact copy of the DBDS to a known time. Image copies can be concurrent (also called fuzzy) or non-concurrent. Non-concurrent image copies are usually called batch image copies. When a concurrent image copy is used, log data from before the time that the image copy was taken is usually needed. Batch image copies require only the log data that was created after the image copy was taken. Most image copy utilities take a primary and secondary copy (two copies in one pass). Often it is the secondary copy that is shipped to a recovery site. Information about image copy data sets is normally retained by DBRC.

After a DBDS has been restored by using an image copy, you must apply all of the changes that have been made since the image copy was taken. IMS log data sets are read to determine what changes have been made.
**IMS log data**

IMS keeps a log of all of the changes that are made to databases. This log is written to data sets that are called online log data sets (OLDS), system log data sets (SLDS), and recovery log data sets (RLDS). A SLDS contains all of the log data that was written by IMS. An RLDS is a subset of the SLDS and contains only the log data that is needed for database recovery. Information about log data sets is maintained by DBRC. Like image copies, there can be primary and secondary log data sets. Often the secondary copy is shipped to a recovery site. No IBM recovery utility uses an OLDS as input.

A change accumulation data set also contains log data that is usable by a recovery utility. The Change accumulation (CA) utility reads log data sets and reformats the data to a form that is more efficient for recovery utilities. The CA utility does not create a secondary copy. Information about CA data sets is kept by DBRC.

**RECON data sets**

RECON data sets must exist at the recovery site. Often, a copy of the RECON data sets that are taken at the active site are shipped to the recovery site. One method of making a copy of the RECON data sets is by using the DBRC BACKUP.RECON command.

---

**Preparing for site recovery**

Before a role change takes place, the databases must be restored to a predetermined time that is called the cleanup time. Information in the RECON most likely needs to be changed to conform to this time. RECON records are either deleted or changed to remove information that is later than the selected cleanup time.

RCU leaves the RECON in a real-time situation, as it would be in at cleanup time.

The typical tasks that are performed to prepare the RECON data sets are:

- Closing open PRILOG records
- Deleting PRIOLD records
- Deleting SECOLD records
- Closing open PRISLD records
- Closing open SECSLD records
- Deleting SUBSYS records
- Updating or deleting ALLOC records
- Updating or deleting LOGALL records
- Deleting Image copy records that are past the RECON cleanup time
- Deleting CA records that are past the RECON cleanup time
- Marking CA runs invalid if log data spans past the RECON cleanup time
- Marking DBs as Recov Needed if they are affected by RCU
- Incrementing Recov Needed Counts if they marked for recovery
- Adjusting ICUSED (image copy was used) when one or more image copies are deleted
- Adjusting CAUSED (change accumulation was used) when one or more change accumulations are deleted
- Deleting Batch Backout Needed records
Optionally marking primary or secondary log records and IC records in ERROR
Optionally deleting disabled partitions
Providing detailed reports with applicable warning messages in case manual intervention is required

RCU does not verify the physical existence of any of the data sets that are identified in the RECON.

Clean up time

Before preparing the RECON for site recovery, a clean up time must be determined.

The cleanup time is input to RCU, and a copy of the active RECON data sets are modified to make them conform to this time. All indications of IMS activity equal to or later than the cleanup time are removed from the RECON. The RECON is not modified to appear that all activity ended normally at the cleanup time. Rather, it will appear as if time simply stopped.

Attention: RCU will cause data to be lost after the clean up time. It is not intended to be used to create a mirror-image of the active site. RCU is intended to create a clean copy of the RECON data sets at a particular point in time.

RECON records that are updated

The RCU function makes updates to several of the RECON records.

The RCU function takes various actions for all of the records in the RECON.

Database (DB) records:

DB records (including AREA AAUTH) are never deleted. Any authorizations for the database are removed. Actions on other records might cause the DB record to be updated. All DB records that are changed in any way are reported.

Partition records:

All partitions that are marked as Disabled are deleted from the RECON by a DBRC DELETE.PART command if the DELDISABLEPART(Y) keyword is specified. If specified, The DBRC DELETE.PART command deletes all of the records that are associated with a partition. The partition that is deleted is reported, but not all of the deleted associated records.

Database data set (DBDS) records:

DBDS records (including AREA RECOV) are never deleted. All EEQEs are removed, and the record is marked as needing recovery. The Recovery needed count in the DB record is incremented. Actions on other records might cause the DBDS record to be updated. All DBDS records that are changed are reported.

Allocation (ALLOC) records:

All ALLOC records with an allocation time equal to or greater than the cleanup time are deleted. For each ALLOC record that is deleted, the associated LOGALL record is updated. If the last ALLOC is removed from a LOGALL record, the
LOGALL record is deleted. ALLOC records that are open before the cleanup time (ALLOC records that do not have a deallocation time) remain open so no deallocation time is added. When an ALLOC record is deleted, the DBDS or AREA RECOV record is marked as needing recovery (if not already marked as such), and the Recov Needed count in the DB record is incremented.

**Image copy (IC1 and IC2) records:**

Image copy records with a runtime that is equal to or greater than the cleanup time are deleted. Deleting an IC record also causes image copy counts (ICUSED) that are maintained in the DBDS record to be updated, but ICAVAIL is not adjusted. RCU reports DBDSs that have no image copy.

**REORG records:**

REORG records with a runtime that is equal to or greater than the cleanup time are deleted. Deleting a REORG record does not affect the IC Needed flag. The IC Needed flag might be on or off. If a REORG record is not deleted and there are no later image copies, this condition is reported.

**Note:** Special recovery consideration should be given to the associated records for Primary or Secondary Indexes, ILEs, and External logicals since they might not have been changed by the deletion of the Primary database's REORG record.

**RECOV records:**

RECOV records with a runtime that is equal to or greater than the cleanup time are deleted. If the recovery is a timestamp recovery (TSR), the DBDS is marked as needing recovery and the Recov needed count in the DB record is incremented. Both the DBDS or AREA record and the DB record are reported as having been changed. If the recovery is a TSR and the recover-to time is earlier than the cleanup time, this condition is reported.

**Tip:** Special recovery consideration should be given to the associated records for Primary or Secondary Indexes, ILEs, and External logicals because they might not have been changed by the deletion of the Primary database's RECOV record.

**Change accumulation (CA) records:**

CA execution records that include logs that extend past the cleanup time are deleted from the RECON by a DBRC DELETE.CA command. CA STOP TIME is used; not the RUN TIME.

The CA count in the CAGRP record that is used is adjusted accordingly. If the CAGROUP is REUSE, CA execution records that are deleted are not made available. CA execution records that are deleted are reported. CA Group records that might be changed by the deletion of execution records are not reported.

**Subsystem (SSYS) records:**

All subsystem records are deleted from the RECON and reported.
Log records (PRILOG, SECLOG, PRIOLDS, and SECOLDS):

All log records with a start time that is equal to or greater than the cleanup time are deleted along with their associated LOGALL record. Log data set entries in these log records with a start time equal to or greater than the cleanup time are deleted from the record. If a log data set entry is deleted, the stop time of the log is set to zero. The log data set count is adjusted accordingly. A log record with no data set entries can exist when the OLDS has not been archived. Both log records are deleted and their associated LOGALL records that are deleted are reported.

Log allocation (LOGALL) records:

LOGALL records are affected only by changes to other records.

Primary and secondary online log data sets (PRIOLDS and SECOLDS):

All PRIOLDS and SECOLDS records are deleted from the RECON and reported.

Backout records:

All BACKOUT records are deleted from the RECON and reported.

Special considerations for using RCU

Some conditions preclude running RCU and must be corrected before using the function. Other conditions require special consideration after RCU completes. In any case, RCU will either deliver an error message or report this information.

Interim log records:
Interim log records are temporary, they are created during the use of the IMS Log Recovery utility, and they remain in the RECON only while the utility is running. Therefore, you must remove these interim log records either by running the Log Recovery utility to completion or by using the DBRC DELETE.LOG command. RCU will terminate with an error message that indicates this condition.

OLR: The IMS Online Reorganization utility (OLR) might have been in a suspended or active state. If OLR is running at the same time as the cleanup time, a report will be generated for those databases that were being reorganized. Manual intervention is required to determine the state of the active or inactive data sets before any recovery of these databases occurs. RCU will report this information.

RECOV records that do not have a subsequent IC record:
RCU will detect if a RECOV record exists without a subsequent image copy. RCU will report this information.

REORG records that do not have a subsequent IC record:
RCU will detect if a REORG record exists without a subsequent image copy. RCU will report this information.

Databases that do not have any image copy records:
RCU will detect if a DBDS record does not have at least one associated IC record. Image copies are required for recovery. RCU will report this information.
Primary and secondary image copy records:
If you specify that RCU is to mark the primary image copies in error and no corresponding secondary image copy record exists, RCU will report this information.

Primary and secondary log records:
If you specify that RCU is to mark the primary log records in error and no corresponding secondary log record exists, RCU will report this information.

RCU input

Run the RCU function by issuing the RECONCOPY supporting function keyword and the RCU function keyword.

RCU function syntax
RCU ( TIME(timestamp)
DELDISABLEDPART(YES | Y | NO | N)
SOURCE(PRI | SEC | PRIIC | SECIC)
)

Parameters

TIME( timestamp )
Use the TIME parameter to specify the cleanup time. The timestamp variable has no default. The TIME parameter is required. The format of the time format specification can be found in the "Timestamp format" on page 30 section.

DELDISABLEDPART( Y | YES | N | NO )
Use the optional DELDISABLEDPART parameter to specify whether to delete partitions that have been marked as disabled. The default is YES.

SOURCE(PRI | SEC | PRIIC | SECIC)
Use the optional SOURCE parameter to specify whether to use only primary or secondary log data sets and image copy data sets for subsequent recovery.

PRI Use this subparameter to indicate that primary logs and primary image copies only are to be used for recovery. If secondary logs and secondary image copy data sets exist, RCU marks them in error.
SEC Use this subparameter to indicate that secondary logs and secondary image copies only are to be used for recovery. If secondary logs and secondary image copy data sets exist, RCU marks the primary data sets in error.
PRIIC Use this subparameter to indicate that primary image copies only are to be used for recovery. If secondary image copy data sets exist, RCU marks them in error.
SECIC Use this subparameter to indicate that secondary image copies only are to be used for recovery. If a secondary image copy data sets exists, RCU marks the primary data set in error.

RCU usage example

The RCU function can be used to clean up a copy of the active RECON data sets to prepare them for a disaster recovery site.
In the following example, the active RECONs are copied to DRSITE.IMS1.C1 / C2 / C3 data set names, and if the RECON copy data sets already exist, they are reused. Otherwise, the RECON copy data sets are allocated on a non-SMS volume with 300 primary and 10 secondary cylinders of space.

```
RECONCOPY (DSNPREF( DRSITE.IMS1 )
   REUSE( YES )
   SPACE(CYL(300 10))
   DATACLASS(NOSMS)
)
RCU ( TIME ('07.291 06:30:01.1') )
```

In the preceding example, RCU cleans up the copied RECON data sets to the specified time.

---

**RCU output**

The output of the RCU function is written to the RPTRCU DD.

If you did not specify the RPTRCU DD in the JCL, it is dynamically allocated to SYSOUT=*. Report output can be written to SYSOUT or to a data set. The characteristics of the output are RECFM=FBA,LRECL=133. Report output can also be written to the IMS Tools Base for z/OS: IMS Tools Knowledge Base by specifying the REPORT(ITKBOUT(YES)) keyword and parameter.

The default is REPORT(ITKBOUT(YES)).

The RCU report consists of two main report sections. The following figure shows an example of the RCU report.
IMS DATABASE FACILITY EXTENDED FUNCTIONS V1.1.0 (5655-V86)

--- Invocation Parameters:

- SOURCE: NONE
- DELDISABEPART: YES

--- Log Record Cleanup Report - Log Records Deleted

<table>
<thead>
<tr>
<th>SSID Name</th>
<th>Log Start Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>IMZ2</td>
<td>2007.287 20:11:14.500000 -07:00</td>
</tr>
<tr>
<td>IMZ1</td>
<td>2007.287 20:11:17.100000 -07:00</td>
</tr>
<tr>
<td>JCLBTCH1</td>
<td>2007.287 20:26:47.700000 -07:00</td>
</tr>
<tr>
<td>JCLBTCH2</td>
<td>2007.287 20:26:53.200000 -07:00</td>
</tr>
</tbody>
</table>

- Record Type: Log # Records Deleted: 4
- Record Type: Log # Records Modified: 0

--- Online Log Data Set Cleanup Report - OLDS Records Deleted

<table>
<thead>
<tr>
<th>SSID Name</th>
<th>Primary/Secondary</th>
</tr>
</thead>
<tbody>
<tr>
<td>IMZ1</td>
<td>Primary</td>
</tr>
<tr>
<td>IMZ2</td>
<td>Primary</td>
</tr>
</tbody>
</table>

- Record Type: OLDS # Records Deleted: 2

--- Subsystem Cleanup Report - SSYS Records Deleted

<table>
<thead>
<tr>
<th>Subsys Name</th>
<th>Subsys Type</th>
<th>Subsys Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>IMZ1</td>
<td>Online</td>
<td>Normal</td>
</tr>
<tr>
<td>IMZ2</td>
<td>Online</td>
<td>Normal</td>
</tr>
</tbody>
</table>

- Record Type: Subsystem # Records Deleted: 2

--- Backout Record Cleanup Report - Backout Records Deleted

- Record Type: Backout # Records Deleted: 0

--- Change Accum Execution Cleanup Report - CAs Deleted

- Record Type: CA Execution # Records Deleted: 0

--- Database/DEDB Cleanup Report - Database(s) Modified

<table>
<thead>
<tr>
<th>DBD/DEDB Name</th>
<th>DDN/Area Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>CUSTD1</td>
<td></td>
</tr>
<tr>
<td>PCUST21</td>
<td></td>
</tr>
<tr>
<td>PCUST22</td>
<td></td>
</tr>
<tr>
<td>PCUST42</td>
<td></td>
</tr>
</tbody>
</table>

- Record Type: Database # Records Modified: 4

--- Database Data Set Cleanup Report - DBDSs Modified

<table>
<thead>
<tr>
<th>DBD/DEDB Name</th>
<th>DDN/Area Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>CUSTD1</td>
<td></td>
</tr>
<tr>
<td>PCUST21</td>
<td>PCUST21A</td>
</tr>
<tr>
<td>PCUST22</td>
<td>PCUST22M</td>
</tr>
<tr>
<td>PCUST42</td>
<td>PCUST42A</td>
</tr>
<tr>
<td>PCUST42</td>
<td>PCUST42M</td>
</tr>
</tbody>
</table>

- Record Type: Database Data Sets # Records Modified: 6

Figure 9. Example of an Extended Functions RECON cleanup report (part 1 of 2)
Figure 10. Example of an Extended Functions RECON cleanup report (part 2 of 2)
The RCU report contains the following sections:

**Invocation Parameters**
This section of the report shows the options that RCU used for its processing.

- **TIME**  This field indicates the cleanup time.
- **SOURCE**  This field displays whether to mark primary or secondary log or image copy records. In the previous example report, the SOURCE keyword was not specified, so NONE was displayed (which means that neither primaries or secondaries are marked in error).

- **DELDISABLEPART**  This field displays whether any disabled partitions are deleted. In the previous example report, all disabled partitions would have been deleted.

The next section of the report describes the following groups of information:

**Log Record Cleanup Report - Log Records Deleted**
This section of the report shows information about the specific log records that were deleted or modified. The details shown are the SSID name and the Log Start Time of each log record. In addition, a total number of log records that were deleted and log records that were modified are reported.

**Online Log Data Set Cleanup Report - OLDS Records Deleted**
This section of the report shows information about the specific log records that were deleted or modified. The details that are listed are the SSID name and the Log Start Time of each log record. In addition, a total number of log records that were deleted and log records that were modified are reported.

**Subsystem Cleanup Report - SSYS Records Deleted**
This section of the report shows information about the specific subsystem (SSYS) records that were deleted. The details that are listed are the Subsystem name, subsystem type (online or batch), and the subsystem status. In addition, a total number of subsystem records deleted are reported.

**Backout Record Cleanup Report - Backout Records Deleted**
This section of the report shows the total number of Backout records that were deleted.

**Change Accum Execution Cleanup Report - CAs Deleted**
This section of the report shows the total number of Change Accumulation (CA) records that were deleted.

**Database/DEDB Cleanup Report - Database(s) Modified**
This section of the report shows the DBD or DEDB name of each DB record that was modified. In addition, a total number of database records that were modified is reported.

**Database Data Set Cleanup Report - DBDSs Modified**
This section of the report shows the DBD or DEDB name and the DDN or area name of each DBDS record that was modified. In addition, a total number of database data set records that were modified is reported.

**Alloc Record Cleanup Report - ALLOC Records Deleted**
This section of the report shows the DBD or DEDB name, the DDN or area
name, and the allocation time of all allocation (ALLOC) records that were
deleted. In addition, a total number of allocation records that were deleted
is reported.

**Image Copy Cleanup Report - IC Records Deleted**
This section of the report shows the total number of image copy (IC)
records that were deleted.

**DB Recovery Cleanup Report - RECOV Records Deleted**
This section of the report shows the DBD or DEDB name, the DDN or area
name, and the recovery time of all RECOV records that were deleted. In
addition, a total number of RECOV records that were deleted is reported.

**Reorganization Cleanup Report - REORG Records Deleted**
This section of the report shows the DBD or DEDB name, the DDN or area
name, and the reorganization time of all of the REORG records that were
deleted. In addition, a total number of REORG records that were deleted is
reported.

**Disabled Partition Cleanup Report - Partition Records Deleted**
This section of the report shows the total number of HALDB disabled
partitions that were deleted.

**No Image Copy Cleanup Report - DBDSs with No Image Copy**
This section of the report shows the DBD/DEDB name and the DDN/Area
Name of all databases that have no image copies recorded in the RECONs.
The database is not recoverable without at least one image copy. A total
number of DBDSs that do not have IC records is reported.

**No IC After REORG Cleanup Report - DBDSs With No IC**
This section of the report shows the DBD or DEDB name, the DDN or area
name, and the REORG run time of all databases that have no image copies
recorded after a REORG record. The database might or might not have had
the ICNEEDED flag on, depending on whether RCU deleted the image
copy record or not. A database must have an image copy taken after a
REORG. A total number of DBDSs that do not have IC records after a
REORG is reported.

---

**RCU return and reason codes**

The RCU issues these return and reason codes on the IRO0102E message.

<table>
<thead>
<tr>
<th>Module</th>
<th>Return code (hex)</th>
<th>Reason code (hex)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IROERC10</td>
<td>4</td>
<td>C3100001</td>
<td>The warning is contained in the report. DBDS or area without an image copy. Time stamp recovery spans cleanup time.</td>
</tr>
<tr>
<td>IROERC70</td>
<td>8</td>
<td>C3700001</td>
<td>Non zero return code from a call to DBRC DELETE.BKO command.</td>
</tr>
<tr>
<td>IROERC40</td>
<td>12</td>
<td>C3400001</td>
<td>DBRC had an error deleting an image copy record. A DSPnnnn1 message is issued.</td>
</tr>
<tr>
<td>IROERC80</td>
<td>12</td>
<td>C3800001</td>
<td>DBRC had an error deleting a partition record. A DSPnnnn1 message is issued.</td>
</tr>
<tr>
<td>Module</td>
<td>Return code (hex)</td>
<td>Reason code (hex)</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>------------------</td>
<td>------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>IROERC10</td>
<td>16</td>
<td>C3100001</td>
<td>RECON open failed.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>C3100002</td>
<td>RECON I/O error on sequential locate of interim logs and disabled partitions.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>C3100003</td>
<td>RECON I/O error on reading BACKOUT records.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>C3100004</td>
<td>RECON I/O error on reading SUBSYS records.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>C3100005</td>
<td>RECON I/O error on reading a log record while processing an ALLOC record.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>C3100006</td>
<td>RECON I/O error on changing an image copy record.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>C3100007</td>
<td>RECON I/O error on locating an image copy after REORG.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>C3100008</td>
<td>RECON I/O error deleting a REORG record.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>C3100009</td>
<td>RECON I/O error deleting a RECOV record.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>C310000A</td>
<td>RECON I/O error updating a DB record in NewDBD.</td>
</tr>
<tr>
<td>IROERC50</td>
<td>16</td>
<td>C3500001</td>
<td>RECON I/O error on first locate.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>C3500002</td>
<td>RECON I/O error on sequential locate.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>C3500003</td>
<td>A call to DBRC returned a non zero return code.</td>
</tr>
<tr>
<td>IROERC60</td>
<td>16</td>
<td>C3600001</td>
<td>RECON I/O error on first OLDS locate.</td>
</tr>
</tbody>
</table>

**Related information**

[“IRO0102E” on page 115](#)
Part 4. IMS Database Recovery Facility job generation

You can use the Recovery Point Identification ISPF interface (RPID ISPF interface) either with or without the RPID report to automatically generate IMS Database Recovery Facility jobs. Before you generate a job, you must first specify database components and recovery times. You can then customize that job to fit the needs of your environment.

Topics:
- IMS Database Recovery Facility job generation database component and recovery time selection
- IMS Database Recovery Facility job customization and generation
Chapter 10. IMS Database Recovery Facility job generation
database component and recovery time selection

Before you generate IMS Database Recovery Facility: Extended Functions jobs, you must specify the database components to be recovered and the times for recovery. To specify database components and recovery times, start the RPID ISPF interface either with or without the Recovery Point ID (RPID) report.

Using the RPID ISPF interface with the RPID report allows you to select database components and their recovery times directly from the RPID report. By using the RPID report you can specify database components and recovery times directly from the RPID report, which means that you do not have to independently verify database components and recovery times.

Alternatively, you can use the RPID ISPF interface without the RPID report. Specifying components and times without the report can save you time because you do not have to navigate through the RPID report to select database components and recovery times. However, you do have to manually verify the database components and recovery times.

Topics:
- “Methods for database component and recovery time selection”
- “Specifying database components and recovery times by using the RPID report” on page 94
- “Specifying database components and recovery times without the RPID report” on page 99

Methods for database component and recovery time selection

The RPID ISPF interface provides two methods for selecting database components and recovery times. You can use the RPID ISPF interface either with or without the RPID report.

When you use the interface with the RPID report, you select database components and recovery times from the RPID report based on common recovery timespans. Using the RPID ISPF interface with the RPID report is recommended because the report determines ideal recovery times for you.

You can also use the RPID ISPF interface without the RPID report. This method requires you to manually specify database components and recovery times. Specifying components and times without the report can save you time because you do not have to navigate through the RPID report to select database components and recovery times. However, you are responsible for verifying that the database components and recovery times are valid from IMS Database Recovery Control.
Specifying database components and recovery times by using the RPID report

When you use the Recovery Point ID (RPID) ISPF interface to generate IMS Database Recovery Facility jobs, you must specify the database components that you want to recover and the recovery time. Using the RPID ISPF interface with the RPID report allows you to specify database components and recovery times directly from the RPID report.

Because the RPID report lists validated recovery time spans for one or more databases, you do not have to independently verify database components and recovery times.

Before you begin, complete the following prerequisite tasks:

- Install the Extended Functions ISPF interface and allocate the ISPF libraries required for the RPID ISPF interface.
- Run the RPID function to generate an RPID report. The report must be in an EDIT or VIEW session.
- To recover a group of database components to a common recovery point, make sure that a common time span exists for all of the database components. If the database components do not have a common time span, you can generate one by using the Recovery Point Create function.

To select the database components and recovery times by using the RPID ISPF interface with the RPID report:

1. Start the RPID ISPF interface with the RPID report by issuing the IROTXEL command on the command line of an RPID report:

   Command ===> irotexl

   After you start the RPID ISPF interface, the initial RPID ISPF interface panel is displayed, as shown in the following figure:

   Figure 11. The initial RPID ISPF interface panel with the RPID report

   The RPID ISPF interface contains help panels that provide more detailed information about the interface and its options. You can access help panels by pressing the PF1 key at any time.

2. Select Common Time Span processing.
3. In the Common Time Spans panel, select a common time span to limit the recovery to a specific time span.

A common time span helps you recover a group of database components to a common recovery point. If you do not select a common time span, generated control statements will not contain an RCVTIME parameter, and any subsequent option that relies on a common time span cannot be selected.

For examples, the following figure shows that the 07.338 11:10:19.946777+0000 -- 07.364 14:15:40.100000+0000 common time span was selected after S was entered to the left of the selection. The selection is now indicated by *Recvr.

```
Row 1 to 2 of 2
Command ==> ________________________________________________ Scroll ==> CSR
Select/deselect time with "S" and press ENTER. Press END after selection.

----- Span Start Time ----- ------ Span End Time ------
- 07.314 14:15:40.100000+0000 -- 07.338 10:54:27.712196+0000
  Nov 10 2007 Dec 04 2007
- 07.338 11:10:19.946777+0000 -- 07.364 14:15:40.100000+0000
  Dec 04 2007 Dec 30 2007
*Recvr Dec 04 2007 Dec 30 2007

***************************************************************************** Bottom of data *****************************************************************************
```

Figure 12. Common Time Spans - Selected

4. Exit the Common Time Spans panel and select **Database component processing** from the IMS Recovery Point ID Report panel.

5. In the Database Component Processing panel, enter a forward slash (/) to the left of the components that you want to recover.

A list of processing options is displayed that allows you to specify the recovery options for the selected component. For example, you can specify the time to which components are to be recovered, exclude specific components for recovery, or view more information about each component.

**Attention:** To avoid data integrity exposure, do not specify individual ddnames for recovery. If you want to specify individual ddnames, make sure that you are familiar with dname recovery and its implications.

The following figure shows that the partitions POHIDKB and POHIDKC are selected for recovery and that the full recovery time of dname POHIDKCB is selected as the recovery time. All other components are excluded from recovery.
When you select a recovery time, the database component that you selected the recovery time from is highlighted. Any parents and children of that database component are also highlighted. The preceding figure uses bold to represent highlights.

If you have specified a recovery time, the time is shown in the Recovery basis field. The Recover column contains recovery status indicators that define how JCL will be generated for each component.

6. After you have selected the components that you want to recover, exit the Database Component Processing panel.

7. Optional: To review a summary of your recovery selections, select View selected recovery basis from the IMS Recovery Point ID Report panel.

8. Select Generate IMS Database Recovery Facility JCL to complete the selection of database components and recovery times.

The IMS Database Recovery Facility Job Generation panel is displayed, which allows you to customize parameters, customize control statements, and add additional JCL before generating an IMS Database Recovery Facility job.
Related concepts
Chapter 11, “IMS Database Recovery Facility job customization and generation,” on page 101

Related tasks
“Specifying database components and recovery times without the RPID report” on page 99

Related reference
“Recovery status indicators” on page 98

RPID ISPF interface initialization errors
If the RPID report fails validation when you attempt to start the RPID ISPF interface with the RPID report, an error message is issued and the RPID ISPF interface does not start.

The following sections describe initialization errors that might cause the RPID report validation to fail.

Physical file characteristics
Files with a data width of up to 256 bytes (LRECL=256 for FB, or LRECL=260 for VB) are supported. Files that exceed this size cause the RPID interface to stop and issue the following ISPF error message.

  Short message: Unsupported LRECL
  Long message: File being processed has Data_Width=nnn. Data_Width greater than 256 is not supported.

Make sure that the pre-allocated data set in which you save the RPID report is set to a data width of up to 256 bytes (LRECL=256 for FB, or LRECL=260 for VB), and then generate another RPID report. If the problem persists, contact IBM Software Support.

Title, section, and page headings
The RPID ISPF interface processes and presents composite information that is extracted from the RPID report sections with the following title and headers.

IMS DATABASE FACILITY EXTENDED FUNCTIONS RECOVERY POINT ID REPORT

  -> Invocation Parameters:
  --> Recovery Time Spans Common to All Entries in the DBLIST:
  --> Recovery Time Spans for Individual Entries in the DBLIST:
  --> Recovery Points (Utility End Times) for Individual Entries in the DBLIST:

If the title or headings cannot be located in the report that is being processed, the RPID interface stops, and an ISPF error message is displayed. To avoid this error, rerun the RPID function to generate another RPID report, and do not delete any titles or headers from the report.

Common time spans
The RPID ISPF interface examines the common time spans that are defined in the RPID report. If no common time spans are contained in the report, the following panel is displayed.
If you want to generate a common time span, use the Recovery Point Creation function.

**Related concepts**

"RPCR overview" on page 55

**Recovery status indicators**

Recovery status indicators show which database components are selected for recovery. The recovery status indicators are shown only in the Database Component Processing panel of the RPID ISPF interface with the RPID report.

The following table describes each recovery status indicator from the Recover column of the Database Component Processing panel.

*Table 7. Definitions of recovery status indicators*

<table>
<thead>
<tr>
<th>Status indicator</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add-Dbase</td>
<td>This database is selected for recovery. A DRF ADD statement is generated for the database’s recovery.</td>
</tr>
<tr>
<td>Add-Area</td>
<td>This FastPath area is selected for recovery. A DRF ADD statement is generated for the area’s recovery.</td>
</tr>
<tr>
<td>Add-Part</td>
<td>This HALDB partition is selected for recovery. A DRF ADD statement is generated for the partition’s recovery.</td>
</tr>
<tr>
<td>Add-DD</td>
<td>This ddname is selected for recovery. A DRF ADD statement is generated for the ddname’s recovery.</td>
</tr>
<tr>
<td>Add-Implied</td>
<td>This status occurs only for HALDB OLR partner ddnames when their primary ddnames have been selected for recovery and marked with Add-DD. DRF ADD control statements are not generated for ddnames that are marked Add-Implied. The recovery of these ddnames is implied by the recovery of their primary ddnames.</td>
</tr>
<tr>
<td>Excl-Dbase</td>
<td>This database is excluded from recovery. No ADD statements are generated for the database.</td>
</tr>
<tr>
<td>Excl-Area</td>
<td>This FastPath area is excluded from recovery. No ADD statement is generated for the area.</td>
</tr>
<tr>
<td>Excl-Part</td>
<td>This HALDB partition is excluded from recovery. No ADD statements are generated for the partition.</td>
</tr>
<tr>
<td>Excl-DD</td>
<td>This ddname is excluded from recovery. No ADD statements are generated for the ddname.</td>
</tr>
</tbody>
</table>
Table 7. Definitions of recovery status indicators (continued)

<table>
<thead>
<tr>
<th>Status indicator</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>blank ADD</td>
<td>Statements are not generated for components for which the Recover column is blank. Recovery of these components is incorporated into the recovery or exclusion of their database, area, or partition. For example, if a partition is indicated as Add-Part, the ddnames within the partition are recovered by the recovery of the partition. The Recover column for these ddnames is blank. The Recover column for the *DBLIST entry is always blank.</td>
</tr>
</tbody>
</table>

Specifying database components and recovery times without the RPID report

If you want to specify database components and recovery times that you have already verified, use the RPID ISPF interface without the RPID report. An ISPF panel is displayed in which you can specify any number and combination of database components to be recovered and the recovery time of those components.

Because using the RPID ISPF interface without the RPID report will not use database components and time spans that have been previously verified by the RPID function, you are responsible for validating recovery time spans for all database components from IMS Database Recovery Control.

Before you can use the RPID ISPF interface, you must install the Extended Functions ISPF interface.

To specify database components and recovery times:
1. Start the RPID ISPF interface without RPID report processing by issuing the TSO %IROTREXL command on the ISPF command line:
   Command ===> tso %irotrexl
   After starting the interface, the Define Recovery Components and Time panel is displayed, in which you can specify database components and recovery times.

   **Figure 15. Initial RPID ISPF interface without the RPID report**

   2. In the Database field, enter the database name to be recovered.
   3. In the DB Type field, select the correct database type.
The selection of data types are shown next to the **DB Type** field.

4. In the **Area/Part** field, enter the component of the database that you want to recover.

5. In the **DDname** field, specify the individual ddname for recovery.

   **Attention:** To avoid data integrity exposure, do not specify individual ddnames for recovery. If you want to specify individual ddnames, make sure that you are familiar with ddname recovery and its implications.

6. Optional: In the **RcvTime** field, you can specify the recovery time of the components that you want recovered.

   You can specify any time format that is acceptable in IMS Database Recovery Facility.

   If you specify a time, you must also identify the time as either TSR or PITR in the **Type** field. If you do not specify a time, the RPID ISPF interface generates a full recovery of the components.

7. Press **Enter** to complete your entry.

8. Enter more components to be recovered.

9. After you have specified all of the entries, press **Exit** to exit the Define Recovery Components and Time panel.

   When you exit the Define Recovery Components and Time panel, the IMS Database Recovery Facility Job Generation panel is displayed. The job generation panel allows you to customize parameters, customize control statements, and add additional JCL before generating a job.

The following figure shows a full-function database (MYFFDATA) and three partitions of a HALDB database (MYHAL) that have all been entered for IMS Database Recovery Facility job generation. A fourth MYHAL partition will be added when **Enter** is pressed.

---

Figure 16. Define Recovery Components and Time panel

**Related concepts**

- Chapter 11, “IMS Database Recovery Facility job customization and generation,” on page 101

**Related tasks**

- “Specifying database components and recovery times by using the RPID report” on page 94
Chapter 11. IMS Database Recovery Facility job customization and generation

After you have selected database components for recovery, you can customize your job by maintaining parameter sets, adding additional JCL to your job, or by specifying other options in IMS Database Recovery Facility: Extended Functions.

The job generation panel is displayed after you have chosen database components and recovery times from the RPID ISPF interface with either the RPID report or without the RPID report. Job customization helps you generate jobs that are specific to your environment. If you do not want to customize your job, select **Generate DRF job** to generate a generic job.

**Important:** You must supply a data set that contains the JCL that is required to execute the recovery utility. The job generation process creates only the ADD statements that are needed for the input to IMS Database Recovery Facility. These input control statements that are generated by the job generation process are appended to the JCL that you supply.

The following figure shows an example of the IMS Database Recovery Facility Job Generation panel. Your recovery selections are shown after the list of options.

![DRF Job Generation Panel](image-url)

**Figure 17. Main IMS Database Recovery Facility job generation option display menu**
The ISPF interface contains help panels that provide more detailed information about the interface and its options. You can access help panels by pressing the PF1 key at any time.

**IMS Database Recovery Facility job generation options**

The DRF Job Generation panel provides the following options six option that you can use to customize and generate IMS Database Recovery Facility jobs.

1. Use the Maintain Parameter Sets option to maintain groups of parameters that might be associated with IMS Database Recovery Facility ADD and START control statement. You can define separate parameter sets for Index Builder, Image Copy, Pointer Checker, and START.

2. Use the Specify ADD Parameters and Utilities option to customize each generated IMS Database Recovery Facility ADD control statement. For example, you can set individual ADD parameters, include utilities, and propagate parameter settings among ADD statements.

3. Use the Assign START Parameters option to customize the generated IMS Database Recovery Facility START control statement by specifying START parameters.

4. Use the Supply DRF Environmental Control Statements option to specify or override parameters that are contained in the FRXDRFxx control member.

5. Use the Supply Execution JCL option to include additional JCL that is required for executing the IMS Database Recovery Facility job.

6. Use the Generate DRF job option to produce output that is based on your recovery selections and customization specifications.

**Database component and recovery time re-selection**

You can exit the IMS Database Recovery Facility job generation process, re-select the set of database components to be recovered, and then reenter the IMS Database Recovery Facility job generation process. This ability to exit and reenter the job generation process allows you to modify the set of ADD control statements without starting over.

All parameters, except those that are related to ADD statements, are maintained when you exit and reenter the IMS Database Recovery Facility job generation process. Because exiting and reentering the IMS Database Recovery Facility job generation process can invalidate ADD statement parameter associations, all ADD statement parameters are discarded when you exit IMS Database Recovery Facility job generation.

**Generated jobs**

Generated jobs incorporate your database component selections, time selection, and parameter specifications.

After you generate a job, the RPID ISPF interface initiates an ISPF/PDF edit session with a temporary data set that contains the generated statements. The interface does not save these statements to a permanent data set or submit the generated job. You use the editor’s facilities to perform these functions.

An example of generated output is shown in the following figure. When multiple database components have the same attributes and have been assigned the same parameters, they are grouped into single ADD control statement.
If IMS Database Recovery Facility environment control statements were specified, they are added to the generated IMS Database Recovery Facility control statement. You must specify the IMS Database Recovery Facility JCL statements to form a complete job. Otherwise, the input control statements will be generated with no execution JCL.
**EXECUTE FRXJCLDF PROC FOR DATABASE RECOVERY FACILITY (DRF)**

**JCLLIB ORDER=(IMSTESTG.PROCLIB)
EXEC FRXJCLDF,RGN=0M,
SOUT='*'
VCAT=VCTAOV,
DELTA01='IMSTESTG.DELTALIB',
DELTA02='IMSTESTG.DELTALIB',
DELTA03='IMSTESTG.DELTALIB',
RESLIB='IMISL.PO1RTS16.DRF.SFXRESL',
NODE=191RTS16,
DRFMBR=01,
BPECFG=BPEDRFCG,
DRFPROC=DRFS,
GSGNAME=,
Pplexname=*
**

**Selected Recovery Time:**
**Event........ Full-Recovery**
**Time stamp.... 07.338 11:12:50.200000+0000 Com: Yes**
**Database...... DBOHIDK5**
**Area/Part..... POHIDKC**
**DDname........ POHIDKCB**

**Fast-Path Recover Area(s)**
IC:SOME PARMS
IC:IC.SOME PARMS

**ADD AREA(DEDDB01 DD01A00, -
DEDBJN24 DB2A000, -
DEDBJN24 DB2A001, -
DEDBJN24 DB2A239) -
DBATRB(DBDSL(001),DBDSN(003))**

**IC(COMP(Y),**
**COMPRN(FABJCMP3),**
**DSN TYPE (BASIC),**
**EXPDT(990000),**
**ICBUF(15),**
**ICCAT(Y),**
**ICHLQ(TEST.IC.PRIMARY, -
TEST.IC.SECOND),**
**ICNMRULE(N),**
**MGMTCLAS(STANDARD),**
**SPACE(CYL,50,75,RLSE),**
**STORCLAS(STORE#IC)**

*Figure 18. Sample Generated Output (Part 1 of 2)*
Figure 19. Sample Generated Output (Part 2 of 2)
Related tasks

“Specifying database components and recovery times by using the RPID report” on page 94

“Specifying database components and recovery times without the RPID report” on page 99
Part 5. Troubleshooting

Troubleshooting provides you with technical references to help you troubleshoot and diagnose IMS Database Recovery Facility: Extended Functions problems.

Topics:
- IMS Database Recovery Facility: Extended Functions messages and codes
- IMS Database Recovery Facility: Extended Functions user abend codes
Chapter 12. IMS Database Recovery Facility: Extended Functions messages

This reference section provides detailed information about IMS Database Recovery Facility: Extended Functions messages.

Extended Functions messages have the following format:

IROnnnx text

Where:

IRO
Indicates that the message was issued by Extended Functions.

nnnx
Is the message identification number.

x
Indicates the severity of the message as follows:

A  indicates that operator intervention is required before processing can continue.
E  indicates that the job step is about to terminate abnormally.
I  indicates that the message is for information only.
W  indicates that the message is a warning to alert you to a possible error condition.

Message Variables

In the message text, there can be lowercase variables (for example, xxx...). The variables represent values when the message appears such as:

- Data in a data set
- A return code
- An error code

Message Documentation

In addition to message number and message text, information for each message includes the following:

Explanation:
The Explanation section explains what the message text means, why it occurred, and what its variable entry fields are (if any).

System Action:
The System Action section explains what the system will do next.

User Response:
The User Response section describes whether a response is necessary, what the appropriate response is, and how the response will effect the system or program.

IRO0002I tcbname TCB INITIALIZATION COMPLETE

Explanation: This message indicates that one of the IMS Database Recovery Facility: Extended Functions TCBs has successfully initialized. tcbname identifies the specific TCB as one of the following values:

- REM  Main TCB
- TDI  DSPAPI TCB
- TOI  IMS Tools Online System Interface TCB
- CSV  Common Services TCB
- DRC  DBRC TCB
IRO0003I • IRO0006E

PRC Function Processing TCB

System action: Processing continues.

System programmer response: None

Module: IROMICSV, IROMIDRC, IROMIPRC, IROMIREM, IROMITDI, IROMITOI

IRO0003I  tcbname TCB TERMINATION COMPLETE

Explanation: This message indicates that one of the IMS Database Recovery Facility: Extended Functions TCBs has successfully terminated. tcbname identifies the specific TCB as one of the following values:

REM Main TCB
TDI DSPA TCB
TOI IMS Tools Online System Interface TCB
CSV Common Services TCB
DRC DBRC TCB
PRC Function Processing TCB

System action: Processing continues.

System programmer response: None.

Module: IROMTCSV, IROMTDRC, IROMTPRC, IROMTREM, IROMTTDI, IROMTTOI

IRO0004E  RECONID LOCATE PROCESSING FAILURE, REASON: reason

Explanation: This message indicates that an attempt to locate the RECONID member from the IMS Tools Base for z/OS: IMS Tools Knowledge Base has failed. The reason variable identifies the cause of the failure as one of the following values:

UNABLE TO LOCATE RECONID OR RECON1 DSN
The RECONID() keyword was not used to specify the RECONID and the RECON1 data set name could not be located. Either specify a valid RECONID() value or ensure that the RECON1 data set is allocated in the JCL or, the RECON1 dynamic allocation member can be located in the IMSDALIB DD or the JOBLIB/STEPLIB.

RECONID INTERFACE INIT FAILURE
A failure occurred attempting to initialize the RECONID interface. Ensure that the IMS Tools Base for z/OS: IMS Tools Knowledge Base has been initialized and is available.

RECONID RECORD NOT FOUND
A RECONID record matching either the RECONID() value or the RECON1 data set name could not be located in the IMS Tools Base for z/OS: IMS Tools Knowledge Base. Ensure that the correct value is specified and that the RECONID member exists in the IMS Tools Base for z/OS: IMS Tools Knowledge Base.

RECONID INTERFACE READ FAILURE
A failure occurred while attempting to read the RECONID record. Ensure that the ITKBSRVR that is used in the IMS Database Recovery Facility: Extended Functions job has been brought up before starting IMS Database Recovery Facility: Extended Functions. Ensure that the IMS Tools Base for z/OS: IMS Tools Knowledge Base server has a defined IMS Database Recovery Facility: Extended Functions product ID (PRODUCTID=DE) and that the IMS Tools Base for z/OS: IMS Tools Knowledge Base server repositories have at least one defined IMS Database Recovery Facility: Extended Functions RECONID. Refer to your installation jobs provided to install IMS Tools Base for z/OS: IMS Tools Knowledge Base for IMS Database Recovery Facility: Extended Functions.

RECONID DSN MISMATCH
The RECON data set names that are specified in the JCL do not match the RECON data set names that are specified in the RECONID record. Messages IRO0013E and IRO0014E are also issued indicating the data set names that are in error.

For more information on the reason codes, see the IMS System Programming API Reference.

System action: Processing terminates.

System programmer response: Determine the reason for the failure and ensure the correct RECONID record can be located. See this IMS Database Recovery Facility: Extended Functions User's Guide section about RECONID processing for details on how the RECONID record is located.

Module: IROMDRV0

IRO0005E  LOAD FAILED FOR MODULE module

Explanation: This message indicates that an attempt to load module module failed.

System action: Processing terminates.

System programmer response: Ensure the correct load libraries are available through JOBLIB, STEPLIB, or LPALIB.

Module: IROMINI0, IROMSDRC

IRO0006E  MISSING IMSREIN DD

Explanation: This message indicates that the IMSREIN DD statement, which is used to specify runtime parameters, could not be located.
System action: Processing terminates.

System programmer response: Ensure the IMSREIN DD is specified in the JCL.

Module: IROMPARS

**IRO0008E**  process PROCESS FAILED

Explanation: This message indicates that an error occurred during processing for the specified process. Valid values for process are:

- **LOCRECON**
  An attempt to locate the RECON data sets has failed. Make sure the RECON1, RECON2 and RECON3 data set names are available either through JCL or dynamic allocation.

- **RECONCOPY**
  The RECONCOPY process has failed. Review the messages printed to the IREDIAG DD statement for errors relating to this process.

- **ALLOCORECON**
  An error occurred attempting to dynamically allocate the RECON data sets.

- **INITDRC**
  Initialization for the DRC TCB has failed.

- **INITTDI**
  Initialization for the TDI TCB has failed.

- **INITTOI**
  Initialization for the TOI TCB has failed.

- **TERMDRC**
  Termination for the DRC TCB has failed.

- **TERMTDI**
  Termination for the TDI TCB has failed.

- **TERMTOI**
  Termination for the TOI TCB has failed.

System action: Processing terminates.

System programmer response: Check the joblog for additional messages relating to the error. Contact IBM if necessary.

Module: IROMDRV0

**IRO0010E** MISSING OR INVALID KEYWORD
PARA. SPECIFIED - keyword

Explanation: This message indicates that the keyword specified in the message is either invalid or is required and was not specified.

System action: Processing terminates.

System programmer response: Review the input control cards to ensure they are correctly specified.

Module: IROMINI0

**IRO0012E** DYNAMIC ALLOCATION FOR DD
dname FAILED

Explanation: This message indicates that an attempt to dynamically allocate the specified ddname has failed.

System action: Processing terminates.

System programmer response: Inspect the joblog for additional messages indicating the reason for the failure. If the specified ddname is allocated in the JCL, ensure that the attributes are correct.

Module: IROMINI0, IROMDRV0

**IRO0013E** RECON DATA SET NAME MISMATCH
FOR RECON DD ddname

Explanation: This message indicates that there is a mismatch between a RECON data set name specified in the JCL and in the RECONID member or between the JCL and the active IMS system. ddname indicates either RECON1, RECON2 or RECON3. Either IRO0014E or IRO0015E will follow.

System action: Processing terminates.

System programmer response: Ensure that the correct RECON data set names are specified in the JCL and that they match the RECON data set names specified in the RECONID member and the active IMS system.

Module: IROMDRV0, IROCVRC0

**IRO0014E** JCL=dsn1 RECONID=dsn2

Explanation: This message follows message IRO0013E when there is a mismatch between a RECON data set name specified in the JCL and in the RECONID member.

dsn1 specifies the data set name specified in the JCL, and dsn2 specifies the data set name specified in the RECONID member.

System action: Processing terminates.

System programmer response: Ensure that the correct RECON data set names are specified in the JCL and that they match the RECON data set names specified in the RECONID member and the active IMS system.

Module: IROMDRV0

**IRO0015E** IMSID=imsid, JCL=dsn1, IMS=dsn2

Explanation: This message follows message IRO0013E when there is a mismatch between a RECON data set name specified in the JCL and the active IMS system. imsid specifies the IMSID on which the mismatch occurred, dsn1 specifies the data set name specified in the JCL, and dsn2 specifies the data set name allocated to the IMS system.

System action: Processing terminates.

System programmer response: Make sure the correct RECON data set names are specified in the JCL and
that they match the RECON data set names used by
the online IMS system.

**Explanation:** This message is issued whenever IMS Database Recovery Facility: Extended Functions is unable to verify that the RECON data sets allocated to the IMS Database Recovery Facility: Extended Functions job are the same as those allocated to the online IMS systems. IMS Database Recovery Facility: Extended Functions issues an /RML DBRC='RECON STATUS' command to all active IMS systems in the TOSI XCF group. This output is then parsed to obtain the RECON data set names. This message is received whenever IMS Database Recovery Facility: Extended Functions is unable to locate the RECON data set names in the command output. For more information on the reason codes, see the *IMS System Programming API Reference.*

**System action:** Processing terminates.

**System programmer response:** Make sure the correct RECON data set names are specified in the JCL and that they match the RECON data set names used by the online IMS system. If this problem persists, specify VERIFYRCN(NO) in your IMSREIN input and contact IBM support.

**Module:** IROCVRC0

---

**Explanation:** This is a separator line for displaying parameter output to the joblog.

**System action:** Processing continues.

**System programmer response:** None.

**Module:** IROMDRV0

---

**Explanation:** This is a title line for displaying the RECONID member configuration to the joblog.

**System action:** Processing continues.

**System programmer response:** None.

**Module:** IROMDRV0

---

**Explanation:** This message indicates that an invalid application work element (AWE) was detected by one of the servers.

**System action:** The request represented by the AWE is rejected and processing continues.

**System programmer response:** This is an internal error. Contact IBM support.

**Module:** IROMSCSV, IROMSPRC, IROMSREM, IROMSDRC, IROMSTDI, IROMSTOI

---

**Explanation:** This message indicates that an attempt to acquire storage using the specified service has failed.

**System action:** Processing terminates.

**System programmer response:** Increase the storage region size for the job and resubmit the job.

**Module:** IROMSREM, IROMDRV0, IROMSTOI

---

**Explanation:** This message is used to display a configuration keyword and value to the joblog.

**System action:** Processing continues.

**System programmer response:** None.

**Module:** IROMDRV0

---

**Explanation:** This message indicates that the specified service has failed. The return and reason code associated with this failure are provided. For more information on the return and reason codes, see the *IMS System Programming API Reference.*

**System action:** Processing terminates.

**System programmer response:** This is an internal error. Contact IBM software support.
Module:  IROMDRV0, IROMIDRC, IROMINI0, IROMIPRC, IROMSTOI, IROMSREM

IRO0050E  RECONCOPY SERVICE FAILURE
RC=return code, RSN=reason code, FLAG=flag
Explanation:  This message indicates that an error was encountered while processing the RECONCOPY service. The return code, reason code, and flag (diagnostic flag) are returned in this message. For more information on the return and reason codes, see the IMS System Programming API Reference.
System action:  Processing terminates.
System programmer response:  Contact IBM with the information from the message.
Module:  IROMDRV0

IRO0051I  RECONCOPY COMPLETE COPY1=dsn
Explanation:  This message indicates that the RECONCOPY service has successfully created COPY1 of the RECON data set name specified by dsn.
System action:  Processing continues.
System programmer response:  None.
Module:  IROMDRV0

IRO0052I  RECONCOPY COMPLETE COPY2=dsn
Explanation:  This message indicates that the RECONCOPY service has successfully created COPY2 of the RECON data set name specified by dsn.
System action:  Processing continues.
System programmer response:  None.
Module:  IROMDRV0

IRO0053I  RECONCOPY COMPLETE COPY3=dsn
Explanation:  This message indicates that the RECONCOPY service has successfully created COPY3 of the RECON data set name specified by dsn.
System action:  Processing continues.
System programmer response:  None
Module:  IROMDRV0

IRO0054E  RECONCOPY FAILED, REASON: reason
Explanation:  This message indicates that the RECON copy service has failed due to the indicated reason. The possible reasons for the failure are:
PARAMETER ADDRESS IS ZERO
This is an internal error. Contact IBM support.
INVALID PARAMETER BLOCK
This is an internal error. Contact IBM support.
INVALID MACRO VERSION
This is an internal error. Contact IBM support.
RECON COPY DELETE FAILED
The delete for the existing RECON copies failed. Inspect the information in the IREDIAG output DD for further details.
OUTPUT DD SPECIFIED BUT NOT FOUND
This is an internal error. Contact IBM support.
INPUT RECON1 DSN MISSING OR INVALID
The RECON1 data set name is either missing or invalid. Inspect the information in the IREDIAG output DD for further details.
INPUT RECON2 DSN MISSING OR INVALID
The RECON2 data set name is either missing or invalid. Inspect the information in the IREDIAG output DD for further details.
INPUT RECON3 DSN MISSING OR INVALID
The RECON3 data set name is either missing or invalid. Inspect the information in the IREDIAG output DD for further details.
OUTPUT COPY DSN INVALID
The output copy data set name is invalid. Inspect the information in the IREDIAG output DD for further details.
DSNPREF GREATER THAN MAX LENGTH
The length specified for the DSNPREF() value exceeds the maximum allowable length of 38 characters. Reduce the number of characters in the data set prefix.
INVALID DSNPREF VARIABLE
The resolved data set prefix contains invalid characters. Correct the DSNPREF() value and resubmit the job.
COPY EXISTS BUT REUSE(NO) SPECIFIED
The RECON copy data set already exists and REUSE(NO) was specified. Either delete the existing RECON copy data set, specify a different DSNPREF() value, or specify REUSE(YES) and resubmit the job.
DELETE FAILED FOR RECON COPY
The delete for the existing RECON copies failed. Inspect the information in the IREDIAG output DD for further details.
ALLOCATE FAILED FOR RECON COPY
The allocate for the RECON copy failed. Inspect the information in the IREDIAG output DD for further details.
IDCAMS REPRO FAILED
The IDCAMS REPRO of the RECON data set into the RECON copy failed. Inspect the information in the IREDIAG output DD for further details.
DYNALLOC FOR RECON1 FAILED
Dynamic allocation for the RECON1 data set has failed. Inspect the information in the IREDIAG output DD for further details.

DYNALLOC FOR RECON2 FAILED
Dynamic allocation for the RECON2 data set has failed. Inspect the information in the IREDIAG output DD for further details.

DYNALLOC FOR RECON3 FAILED
Dynamic allocation for the RECON3 data set has failed. Inspect the information in the IREDIAG output DD for further details.

DYNALLOC FOR BACKUP1 FAILED
Dynamic allocation for the BACKUP1 data set has failed. Inspect the information in the IREDIAG output DD for further details.

DYNALLOC FOR SYSPRINT FAILED
Dynamic allocation for the SYSPRINT output DD has failed. Inspect the information in the IREDIAG output DD for further details.

DYNALLOC FOR SYSIN FAILED
Dynamic allocation for the SYSIN input DD has failed. Inspect the information in the IREDIAG output DD for further details.

DSPURXRT BACKUP.RECON FAILED
The BACKUP.RECON command issued using DSPURXRT has failed. Inspect the information in the IREDIAG output DD for further details.

System action: Processing terminates.

System programmer response: Take the appropriate action based on the reason and resubmit the job.

Module: IROMDRV0

RECONLOCATE FAILED, REASON:
reason RC=return code

Explanation: This message indicates that an attempt to locate the RECON data set names specified in the JCL or through dynamic allocation failed due to reason. The return code is also specified. For more information on the return and reason codes, see the IMS System Programming API Reference.

LOCATE JFCB FAILED - SWAREQ
The SWAREQ macro failed to locate the JFCB. This is an internal error.

LOAD FOR RECON1 FROM IMSDALIB FAILED
The RECON1 dynamic allocation member could not be loaded. Ensure that the RECON1 member is in this library.

System action: Processing terminates.

System programmer response: If the failure is due to a load failure for RECON1 from IMSDALIB, ensure that the RECON1 member is present in the library. Otherwise, contact IBM support.

Module: IROMDRV0

ALLOC|DEALLOC for ddname dsname SUCCESSFUL

Explanation: This message indicates that a RECON data set, identified by ddname and dsname, was successfully allocated or deallocated.

System action: Processing continues.

System programmer response: None.

Module: IROMDRV0

ALLOC|DEALLOC for ddname dsname FAILED, RC=return code, RS=reason code

Explanation: This message indicates that an attempt to allocate or reallocate the specified RECON data set failed with return code and reason code. For more information on the return and reason codes, see the IMS System Programming API Reference.

System action: Processing terminates.

System programmer response: Ensure that the specified RECON data set exists and is available.

Module: IROMDRV0

RECON1 DFSMDA MEMBER FOUND IN IMSDALIB|STEPLIB, DSN=dsn

Explanation: This message indicates that the RECON1 dynamic allocation member was found in either STEPLIB or IMSDALIB. The RECON1 data set name is specified with dsn.
**IRO0064I**  RECON1|RECON2|RECON3 ALLOCATED IN JCL, DSN=dsn

**Explanation:** This message indicates that specified RECON data set was specified through JCL. The RECON data set name specified by dsn.

**System action:** Processing continues.

**System programmer response:** None.

**Module:** IROMDRV0

**IRO0095W**  DLBLST() WARNING, reason

**Explanation:** This message indicates a warning condition in the specification for the DLBLST() keyword. The reason indicates the problem.

**System action:** Processing continues.

**System programmer response:** Correct the error in the DLBLST().

**Module:** IROMDRV0

**IRO0096W**  NO FUNCTIONS SPECIFIED IN IMSREIN DD

**Explanation:** This message is issued when no valid IMS Database Recovery Facility: Extended Functions functions were specified in the IMSREIN DD input.

**System action:** Processing continues.

**System programmer response:** Correct any function control card errors.

**Module:** IROMDRV0

**IRO0100I**  INITIALIZING FUNCTION function

**Explanation:** This message indicates that processing for the specified function is beginning. Valid values for function are:

- **HCHECK**  Health Check function
- **IMSCMD**  IMS Command function
- **RCU**  RECON Cleanup function
- **RPCR**  Recovery Point Create function
- **RPID**  Recovery Point Identification function
- **VERIFY**  Verify Recovery Assets function

**System action:** Processing continues.

**IRO0102E**  FUNCTION function PROCESS COMPLETED WITH RC=return code, RSN=reason code

**Explanation:** This message indicates that the specified function completed the specified process with return code and reason code. For more information on the return and reason codes, see the IMS System Programming API Reference.

Valid values for function are:

- **HCHECK**
- **IMSCMD**
- **RCU**  RECON Cleanup function
- **RPCR**  Recovery Point Create function
- **RPID**  Recovery Point Identification function
- **VERIFY**  Verify Recovery Assets function

Valid values for process are:

- **INIT**  Function initialization
- **EXEC**  Function execution
- **TERM**  Function termination

Valid values for the rc are:

- 0  Successful
- 4  Warning encountered
- 8  Error, function failed
- 16  Severe error encountered

Valid values for rc, by function are:

- **HCHECK**
  - 01 – INVALID RANGE() SPECIFICATION
  - 02 – INVALID RANGE() BEGIN TIME
  - 03 – INVALID RANGE() END TIME
  - 04 – END TIME LESS THAN BEGIN TIME
IRO0103E  IRO0104E

05 – MULTIPLE TIME PARAMETERS SPECIFIED
06 – HOURS() GREATER THAN MAXIMUM
07 – DAYS() GREATER THAN MAXIMUM
08 – INVALID DAYS() SPECIFIED
09 – HOURS() SPECIFIED

IMSCMD
01 – INVALID IMS COMMAND
02 – USER NOT AUTHORIZED FOR COMMAND
04 – NO ACTIVE IMS SYSTEMS
05 – ONE OR MORE SYSTEMS NOT ACTIVE
08 – QUERY ACTIVE SYSTEMS FAILED
12 – IMS COMMAND FAILED
16 – IMS COMMAND FAILED

RCU
01 – INVALID TIME SPECIFIED

RPCR
01 – DBLIST() REQUIRED BUT NOT SPECIFIED
02 – IROPAPI INIT FAILED
03 – INVALID WAITTIME() SPECIFIED
04 – IROPAPI CONNECT FAILED
05 – IROPAPI CREATE FAILED
06 – IROPAPI DISCONNECT FAILED

RPID
01 – INVALID RANGE() SPECIFICATION
02 – INVALID RANGE() BEGIN TIME
03 – INVALID RANGE() END TIME
04 – END TIME LESS THAN BEGIN TIME
05 – DBLIST() REQUIRED BUT NOT SPECIFIED

VERIFY
01 – TIME FORMAT INVALID
02 – INVALID TIME SPECIFIED
03 – DBLIST() REQUIRED BUT NOT SPECIFIED

System action: Processing continues.

System programmer response: If the return code is non-zero, check the joblog for additional messages that indicate the error. After IMS systems are up, ensure that the proper TOSI initialization complete messages are displayed with the correct XCFGROUP name in all IMS control regions that share the same RECONs as the IMS Database Recovery Facility: Extended Functions job. Also, ensure that the IMS Tools Base for z/OS: IMS Tools Knowledge Base RECON ID (TOIXCF field) matches the XCFGROUP=name referenced in the FOIsidP members for all IMS subsystems sharing the same RECONs.

Module: IROMDRV0, IROBDIST, IROERCUT, IROHHCKT, IROIICMT, IROPFRCT, IRORRPIT, IROVVFYT

IRO0103E  ERROR IN FUNCTION function
REASON: reason

Explanation: This message indicates that an error occurred during the processing of the specified function. Valid values for function are:

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>HCHECK</td>
<td>Health Check function</td>
</tr>
<tr>
<td>IMSCMD</td>
<td>IMS Command function</td>
</tr>
<tr>
<td>RCU</td>
<td>RECON Cleanup function</td>
</tr>
<tr>
<td>RPCR</td>
<td>Recovery Point Create function</td>
</tr>
<tr>
<td>RPID</td>
<td>Recovery Point Identity function</td>
</tr>
<tr>
<td>VERIFY</td>
<td>Verify Recovery Assets function</td>
</tr>
</tbody>
</table>

Valid values for reason are:

<table>
<thead>
<tr>
<th>Reason</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DBLIST() IS REQUIRED</td>
<td>The specified function requires a list of databases to be processed using the DBLIST() keyword. Ensure that the DBLIST command (and keywords) is specified before the RPID, VERIFY, HCHECK, or RPCR command. For more information on the reason codes, see the IMS System Programming API Reference.</td>
</tr>
</tbody>
</table>

System action: Processing for the specified function terminates. Other functions may continue to process.

System programmer response: Specify the list of databases to be processed using the DBLIST() keyword.

Module: IROMDRV0

IRO0104E  INVALID COMMAND INPUT: reason

Explanation: This message indicates that an error occurred while pre-processing command input. Valid values for reason are:

<table>
<thead>
<tr>
<th>Reason</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>RCU SPECIFIED MULTIPLE TIMES</td>
<td>The RCU function was specified more than once in the IMSREIN control cards. Only a single RCU function can be processed for each job execution.</td>
</tr>
<tr>
<td>RECONCOPY() NOT ALLOWED WITH RPCR()</td>
<td>The RPCR function was specified along with the RECONCOPY service. RPCR is mutually exclusive with RECONCOPY.</td>
</tr>
</tbody>
</table>
RECONCOPY() REQUIRED FOR RCU()

The RCU function was specified without specifying the RECONCOPY service. RCU requires that RECONCOPY be specified. Ensure that the RECONCOPY command (and keywords) is specified before the RCU command.

System action: Processing terminates.
System programmer response: Correct the error and resubmit the job.
Module: IROMDRV0

IRO0201E TOI INIT|TERM SERVICE FAILURE
RC=return code, RSN=reason code

Explanation: This message indicates that The IMS Tools Online System Interface (TOSI) initialization or termination service has failed with the specified return code and reason code. For more information on the return and reason codes, see the IMS System Programming API Reference.

System action: Processing terminates.
System programmer response: Ensure that the TOSI is correctly installed.
Module: IROMSTOI

IRO0201I TOSI INTERFACE INITIALIZATION COMPLETE

Explanation: This message indicates that The IMS Tools Online System Interface (TOSI) initialization process has successfully completed.

System action: Processing continues.
System programmer response: None.
Module: IROMSTOI

IRO0203I TOSI INTERFACE TERMINATION COMPLETE

Explanation: This message indicates that The IMS Tools Online System Interface (TOSI) termination process has successfully completed.

System action: Termination process continues.
System programmer response: None.
Module: IROMSTOI

IRO0251E INIT|TERM SERVICE FAILURE
RC=return code, RSN=reason code

Explanation: This message indicates that IMS Database Recovery Facility: Extended Functions DBRC Interface (DRC) initialization or termination service has failed with the specified return code and reason code. For more information on the return and reason codes, see the IMS System Programming API Reference.

System action: Processing terminates.
System programmer response: Ensure that the IMS Tools Online System Interface is correctly installed.
Module: IROMSDRC

IRO0252I DRC INTERFACE INITIALIZATION COMPLETE

Explanation: This message indicates that IMS Database Recovery Facility: Extended Functions DBRC Interface (DRC) initialization process has successfully completed.

System action: Processing continues.
System programmer response: None.
Module: IROMSDRC

IRO0253I DRC INTERFACE TERMINATION COMPLETE

Explanation: This message indicates that IMS Database Recovery Facility: Extended Functions DBRC Interface (DRC) termination process has successfully completed.

System action: Termination process continues.
System programmer response: None.
Module: IROMSDRC

IRO0301E TDI INIT|TERM SERVICE FAILURE
RC=return code, RSN=reason code

Explanation: This message indicates that IMS Database Recovery Facility: Extended Functions DSPAPI Interface (TDI) initialization or termination service has failed with the specified return code and reason code. For more information on the return and reason codes, see the IMS System Programming API Reference.

System action: Processing terminates.
System programmer response: Ensure that the IMS Tools Online System Interface is correctly installed.
Module: IROMSTDI, IROMDBL0

IRO0302I TDI INTERFACE INITIALIZATION COMPLETE

Explanation: This message indicates that IMS Database Recovery Facility: Extended Functions DSPAPI Interface (TDI) initialization process has successfully completed.

System action: Processing continues.
System programmer response: None.
Module: IROMSTDI, IROMDBL0

IRO0302I TDI INTERFACE TERMINATION COMPLETE

Explanation: This message indicates that IMS Database Recovery Facility: Extended Functions DSPAPI Interface (TDI) termination process has successfully completed.

System action: Termination process continues.
System programmer response: None.
Module: IROMSTDI, IROMDBL0
IRO0303I  TDI INTERFACE TERMINATION COMPLETE
Explanation: This message indicates that IMS Database Recovery Facility: Extended Functions DSPAPI Interface (TDI) termination process has successfully completed.
System action: Termination process continues.
System programmer response: None.
Module: IROMSTDI, IROMDBL0

IRO0304E  TDI ERROR AT location reason
Explanation: This message indicates that IMS Database Recovery Facility: Extended Functions DSPAPI Interface (TDI) encountered an error during processing.
System action: Termination process continues.
System programmer response: Contact IBM software support.
Module: IROMSTDI, IROMDBL0

IRO0400E  REPORT WRITER ERROR, REASON: reason
Explanation: This message indicates that an error occurred during report writer processing. The reason for the error is specified in the message. These are internal errors.
System action: Processing terminates.
System programmer response: Contact IBM software support.
Module: IROCRPW0

IRO0401E  ITKB ERROR, FUNCTION=function, RC=return code, RSN=reason code
Explanation: This message indicates that an error occurred during report writer processing while accessing the IMS Tools Base for z/OS: IMS Tools Knowledge Base. The function, return code, and reason code are specified in the message. These are internal errors. For more information on the return and reason codes, see the IMS System Programming API Reference.
System action: Processing terminates.
System programmer response: Correct the error based on the reason and resubmit the job.
Module: IROCRID0

IRO0405E  RECONID INTERFACE ERROR, FUNCTION=function, RC=return code, RSN=reason code
Explanation: This message indicates that an error occurred during the RECONID interface processing. The function, return code, and reason code are displayed. These are internal errors. For more information on the return and reason codes, see the IMS System Programming API Reference.
System action: Processing terminates.
System programmer response: Contact IBM software support.
Module: IROCRID0

IRO0406E  RECONID INTERFACE ERROR, REASON: reason
Explanation: This message indicates that an error has occurred attempting to access the RECONID record. This interface uses the IMS Tools Base for z/OS: IMS Tools Knowledge Base services. The possible reasons for the error are:

ITKB SERVER IS NOT ACTIVE
The IMS Tools Base for z/OS: IMS Tools Knowledge Base server, specified by the ITKBSERV() parameter, is not active. Ensure that the ITKBSRVR used in the IMS Database Recovery Facility: Extended Functions job has been brought up before executing IMS Database Recovery Facility: Extended Functions.

For more information on the reason codes, see the IMS System Programming API Reference.
System action: Processing terminates.
System programmer response: Correct the error based on the reason and resubmit the job.
Module: IROCRID0

IRO0426E  ERROR IN KEYWORD: RANGE|TIME REASON: reason
Explanation: This message indicates that an invalid date or timestamp was detected in the RANGE or TIME keyword. For more information on the reason codes, see the IMS System Programming API Reference.
System action: Processing terminates.
System programmer response: Correct the keyword and resubmit the job.
Module: IROCRPG0
IRO0427I  RANGE|TIME  time

Explanation: This message indicates the time value specified by the RANGE or TIME keyword.

System action: Processing continues.

System programmer response: None.

Module: IROCRNG0

IRO0900I  SHUTDOWN SCHEDULED

Explanation: This message indicates that IMS Database Recovery Facility: Extended Functions is beginning termination processing.

System action: Termination process continues.

System programmer response: None.

Module: IROMSREM

IRO1401E  VERIFY PARAMETER ERROR IN KEYWORD: keyword  REASON: reason

Explanation: This message indicates that an invalid keyword was detected during the Verify function processing. The reason for the error is displayed. For more information on the reason codes, see the IMS System Programming API Reference.

System action: Processing for the Verify function terminates.

System programmer response: Correct the error and rerun the job.

Module: IROVVFYI

IRO1402E  VERIFY COMMAND ERROR, REASON: reason

Explanation: This message indicates that an error was detected in the VERIFY command processing. The reason for the error is displayed.

System action: Processing for the Verify function terminates.

System programmer response: Correct the error and resubmit the job.

Module: IROVVFYI

IRO1403E  LOAD FAILED FOR MODULE module, RC=return code

Explanation: This message indicates that an attempt to load module module failed with return code return code. For more information on the return codes, see the IMS System Programming API Reference.

System action: Processing terminates.

System programmer response: For an explanation of the return and reason code, refer to the interpreting DYNALLOC return codes information in the z/OS MVS Programming: Authorized Assembler Services Reference, Vol 1. Take the appropriate action indicated by the return and reason codes. Ensure that the listed module can be loaded from the JOBLIB.

Module: IROVVFYP

IRO1450I  DYNAMIC ALLOCATION FAILED FOR IC/CA/LOG, RC=return code, RSN=reason code, DSN=data set name

Explanation: The VERIFY function of IMS Database Recovery Facility: Extended Functions failed to allocate an IC, a log, or a change accumulation data set indicated by data set name.


Reason Code: The Reason Code consists of two parts. The first four characters are the contents of error reason code (S99ERROR). The second four characters are the information reason code set (S99INFO).

Data set name: The name of the data set that is to be allocated.

For more information on the return and reason codes, see the IMS System Programming API Reference.

System action: Processing continues for the remaining members in the IMS Database Recovery Facility: Extended Functions verify list.

System programmer response: For an explanation of the return and reason code, refer to the interpreting DYNALLOC return codes information in the z/OS MVS Programming: Authorized Assembler Services Reference, Vol 1. Take the appropriate action indicated by the return code and reason code.

Module: IROVALL0, IROVIDY0

IRO1451I  DYNAMIC DEALLOCATION FAILED FOR IC/CA/LOG, RC=return code, RSN=reason code, DSN=data set name

Explanation: This message indicates that the VERIFY function of IMS Database Recovery Facility: Extended Functions failed to deallocate an IC, a log, or a change accumulation data set indicated by data set name.


Reason Code: The Reason Code consists of two parts. The first four characters are the contents of error DYNALLOC return codes information in the z/OS MVS Programming: Authorized Assembler Services Reference, Vol 1. Take the appropriate action indicated by the return code and reason code.

Module: IROVVFYP
reason code (S99ERROR). The second four characters are the information reason code set (S99INFO).

Data set name:
   The name of the data set that is to be deallocated.

For more information on the return and reason codes, see the IMS System Programming API Reference.

System action:  Processing continues for the remaining members in the IMS Database Recovery Facility: Extended Functions verify list.

System programmer response:  For an explanation of the return and reason code, refer to the interpreting DYNALLOC return codes information in the z/OS MVS Programming: Authorized Assembler Services Reference, Vol 1. Take the appropriate action indicated by the return and reason codes.

Module:  IROVALL0, IROVIDY0

IRO1452I  REPORT PROCESSING FAILED: reason

Explanation:  This message indicates that generation of the VERIFY reports failed for the indicated internal reasons.
   • Invalid report input control block.
   • Null dataset input control block.
   • Report initialization failed.
   • Time conversion routine failed.
   • Load of data conversion module failed.

System action:  Processing continues. However, the generated report might be incomplete.

System programmer response:  All of the above error situations are not user correctable. Contact the IBM Service Center and provide the return code information, if applicable.

Module:  IROVALL0, IROVVFYP

IRO1453I  NULL INPUT CONTROL BLOCK ENTERED FOR ALLOCATION

Explanation:  This message indicates that the VERIFY function is not able to allocate the data set because the address of control block that needs to be allocated is zero.

System action:  Processing continues for the remaining members in the IMS Database Recovery Facility: Extended Functions Verify report list. However, the generated report might be incomplete.

System programmer response:  Check the input control block for allocation.

Module:  IROVALL0

IRO1454I  UNABLE TO GET storage type FOR OPEN, RC=return code

Explanation:  This message indicates that a request to get storage could not be satisfied. In the message, storage type identifies what the storage request was for. For more information on the return codes, see the IMS System Programming API Reference.

System action:  Deallocates the data set.

System programmer response:  Check the storage size and correct the reason for the failure.

Module:  IROVALL0

IRO1455I  OPEN FAILED FOR IC/CA/LOG, RC=return code, DSN=data set name

Explanation:  This message indicates that the VERIFY function of IMS Database Recovery Facility: Extended Functions failed to open an IC, a log, or a change accumulation data set indicated by data set name. For more information on the return codes, see the IMS System Programming API Reference.

System action:  Processing continues for the remaining members in the Verify list.

System programmer response:  For additional information on the return code, see the z/OS MVS Programming: Authorized Assembler Services Guide.

Module:  IROVALL0

IRO1456I  BAD DCB BLOCK SIZE FOR OPEN, RC=return code

Explanation:  This message indicates that IMS Database Recovery Facility: Extended Functions is unable to set values for obtaining fixed storage. Register 14 contains the return code from the set storage service. For more information on the return codes, see the IMS System Programming API Reference.

System action:  Processing continues for the remaining members in the Verify list.

System programmer response:  Check block size.

Module:  IROVALL0

IRO1501E  HCHECK PARAMETER ERROR IN KEYWORD: keyword REASON: reason

Explanation:  This message indicates that a required parameter was not coded or a duplicate keyword was found during HCHECK control card syntax validation. The reason variable indicates one of the following values:
   • HOURS(), DAYS(), RANGE() parameter required.
   • NUMBER0 parameter required.
   • Duplicate instance of keyword found.
For more information on the reason codes, see the IMS System Programming API Reference.

**System action:** Processing terminates with a return code 16.
**System programmer response:** Correct the job control statements and resubmit the job.

**Module:** IROHHCKV

---

**IRO1502E**  
**HCHECK PARAMETER ERROR:**

**parameter**

**Explanation:** This message indicates that an unsupported or invalid parameter was found during HCHECK control card validation. The parameter indicates the keyword or parameter in error.

**System action:** Processing terminates with a return code 16.
**System programmer response:** Correct the control card syntax and resubmit the job.

**Module:** IROHHCKV

---

**IRO1504E**  
**DBRC function FAILED, RC=return code, RSN=reason code**

**Explanation:** This message indicates that a DSPAPI call to DBRC failed for the function specified in the message. The return code and reason code are documented in the IMS Vx Database Recovery Control (DBRC) Guide and Reference publications within the DBRC Application Programming Interfaces.

**System action:** Processing terminates with a return code 16.
**System programmer response:** Correct the control card syntax and resubmit the job.

**Module:** IROHHCKV

---

**IRO2000E**  
**RCU PARAMETER ERROR IN KEYWORD: keyword REASON: reason**

**Explanation:** This message indicates that an invalid keyword was detected during the RECON Cleanup function processing. The reason for the error is displayed. For more information on the return, see the IMS System Programming API Reference.

**System action:** Processing for the RCU function terminates.
**System programmer response:** Correct the error and resubmit the job.

**Module:** IROERCU1

---

**IRO2001E**  
**LOAD FAILED FOR MODULE modname RC=nnnn**

**Explanation:** This message indicates that RCU encountered an error loading the module modname. The z/OS LOAD macro completed with return code nnnn.

**System action:** RCU terminates and clean up is not performed.
**System programmer response:** Ensure the IMS reslib is included in the STEPLIB.

**Module:** IROERCUP

---

**IRO2002E**  
**RCU CAN NOT COMPLETE - INTERIM LOG RECORDS EXIST**

**Explanation:** This message indicates that RCU has determined that there are interim log records in the RECON to be cleaned up that will remain after clean up completes. Interim log records are created when running the log recovery utility and are removed when the utility completes. Because RCU may remove information necessary to complete log recovery, clean up is not performed. The interim log records are reported in the RCU Clean Up report.

**System action:** RCU completes with return code 4, reason code C3100002. Clean up is not performed.
**System programmer response:** Determine if these interim log records are needed. Either complete the log recovery utility or remove the interim log records using the DBRC DELETE.LOG command.

**Module:** IROERC00

---

---
IRO2004E RECORD POINTS TO NON-EXISTENT LOG RECORD FOR DB=dbname DD=ddname

Explanation: The RECON cleanup (RCU) function has found an allocation record for DB dbname, DD ddname that points to a nonexistent log record.

System action: The RCU function terminates.

User response: Review the job output and look for the associated IRO2005E messages that indicate the allocation and start time of the nonexistent log record. Remove the nonexistent log record and rerun the RCU function.

IRO2005E type TIME=time

Explanation: This message indicates the allocation record or log start time of the nonexistent log record that is shown in message IRO2004E. The type indicates:

• The time shown is the allocation record time.
• The time shown is the log start time.

System action: The RECON cleanup (RCU) function terminates.

User response: Review the job output and look for the associated IRO2004E and IRO2005E messages which indicate the allocation and start time of the nonexistent log record. Remove the nonexistent log record and rerun the RCU function.

IRO3001E IMS COMMAND ERROR: reason

Explanation: An invalid IMS command was specified.

System action: Processing for the command terminates.

System programmer response: Correct the command and resubmit the job.

Module: IROICMP

IRO3002E IMS COMMAND FAILED WITH RC=return code, RSN=reason code, ERROR=error

Explanation: This message indicates that an error occurred attempting to process the IMS command. The return code and reason code from the IMS Tools Online System Interface component as well as the error location are displayed. For more information on the return and reason codes, see the IMS System Programming API Reference.

System action: Processing for the command terminates.

System programmer response: Determine the reason for the error by looking up the return and reason codes in the IMS Tools Online System Interface messages and codes and correct the error. After IMS systems are up, ensure that the proper TOSI initialization complete messages are displayed with the correct XCFGROUP name in all IMS control regions that share the same RECONs as the IMS Database Recovery Facility: Extended Functions job. Also, ensure that the IMS Tools Base for z/OS: IMS Tools Knowledge Base RECONID (TOIXCF field) matches the XCFGROUP=name referenced in the FOIssidP members for all IMS subsystems sharing the same RECONs.

Module: IROICMP

IRO3003I IMS COMMAND NOT ISSUED, IMS SYSTEM insid IS NOT ACTIVE

Explanation: This message indicates that a request to run an IMS command against IMS system insid failed because the system was either not active or not a part of the IMS Tools Online System Interface XCF group.

System action: The command is not issued.

System programmer response: Ensure the correct IMS insid was specified. If so, ensure that the correct XCF group was specified for issuing IMS commands.

Module: IROICMP

IRO3005W SAF AUTH SERVICE FAILURE SAF-RC=saf-rc, RACF-RC=racf-rc, RACF-RSN=racf-rsn

Explanation: This message indicates that authorization for an IMS command failed with the specified SAF and RACF return and reason codes.

System action: The IMS command is rejected.

System programmer response: The return and reason codes shown can be found in the MVS Auth Assembler Services Guide in the RACROUTE REQUEST=AUTH section.

Module: IROICMP

IRO3006W IMS COMMAND SECURITY FAILED, USERID=userid, COMMAND=command

Explanation: This message indicates that the specified userid is not authorized to issue the specified IMS command.

System action: The IMS command is rejected.

System programmer response: Contact the security administrator to provide access for the userid to the IMS command.

Module: IROICMP
**IRO3007E**  IMS COMMAND NOT SUPPORTED, COMMAND=command

**Explanation**: This message indicates that the specified IMS command is not supported by the IMS Database Recovery Facility: Extended Functions IMS Command interface.

**System action**: The IMS command is rejected.

**System programmer response**: None.

**Module**: IROICMI

---

**IRO7001E**  RPID PARAMETER ERROR IN KEYWORD: keyword REASON: reason

**Explanation**: This message indicates that An error was found in a keyword of the RPID control statement.

**keyword** identifies the specific keyword in error:

**RANGE**

Keyword RANGE

**reason** identifies the reason for the error in keyword:

**REQUIRED PARAMETER**

RANGE is a required keyword parameter.

**INVALID FORMAT**

RANGE keyword has invalid format.

For more information on the reason codes, see the IMS System Programming API Reference.

**System action**: Processing continues with return code 8 and reason code 0.

**System programmer response**: None.

**Module**: IRORRPII

---

**IRO7002E**  RPID COMMAND ERROR, REASON: reason

**Explanation**: This message indicates that an error was found in the RPID command control statement. For more information on the reason codes, see the IMS System Programming API Reference.

**reason** identifies the reason for the error in the command:

**(NODBLST)**

There was no database list to process based on the DBLIST command.

**System action**: Processing continues with return code 8 and reason code 0.

**System programmer response**: None.

**Module**: IRORRPII

---

**IRO3007E**  IRO7006E

---

**IRO7003E**  RPID CALL TO DSPAPI FAILED, FUNCTION: function, RC= return code, RSN= reason code

**Explanation**: This message indicates that Recovery Point Identification has made a call to the DBRC API DSPAPI and that call has resulted in a failing return code. For more information on the return and reason codes, see the IMS System Programming API Reference.

**function** identifies the specific DSPAPI function call values:

**STARTDBRC**

START the DSPAPI API

**QUERYDB**

Database Query

**System action**: Processing continues with return code 8 and reason code 0.

**System programmer response**: None.

**Module**: IRORRPII

---

**IRO7004I**  RPID FOUND NO IN COMMON RECOVERY TIME SPANS FOR THE DBLIST

**Explanation**: This message indicates that Recovery Point Identification found no Recovery Time Spans in common for all the databases specified on the DBLIST control card read from IMSREIN.

**System action**: Processing continues and ends normally with a return code 4 and reason 0.

**System programmer response**: None.

**Module**: IRORRPII

---

**IRO7005E**  RPID QUERY LOG FAILED FOR DB=dbname DD=ddname

**Explanation**: The Recovery Point ID (RPID) function found an allocation record for the database dbname, DD ddname that points to a nonexistent log record.

**System action**: The RPID function terminates.

**User response**: Review the job output and look for the associated IRO7006E messages that indicate the allocation and start time of the nonexistent log record. Remove the nonexistent log record and rerun the RPID function.

**IRO7006E**  type TIME=time

**Explanation**: This message indicates the allocation record or log start time of the nonexistent log record that is shown in message IRO7005E. The **type** indicates:

- The time shown is the allocation record time.
- The time shown is the log start time.
The Recovery Point ID (RPID) function terminates.

User response: Review the job output and look for the associated IRO7005E and IRO7006E messages which indicate the allocation and start time of the nonexistent log record. Remove the nonexistent log record and rerun the RPID function.

IRO3008E  IMS COMMAND SECURITY NOT SET, IMS COMMAND INTERFACE IS NOT ACTIVE

Explanation: This message indicates that the IMS Database Recovery Facility: Extended Functions IMS command interface is not active because the security information has not been specified in the RECONID record.

System action: All IMS commands are rejected.

System programmer response: Update the RECONID record and specify the desired IMS command security settings.

Module: IROICMI

IRO7301E  RPCR phase FAILED, ERROR: error

Explanation: This message indicates that an error was detected in the Recovery Point Create command processing during the phase listed. The reason for the error is displayed. Valid values for error are:

<table>
<thead>
<tr>
<th>Reason code</th>
<th>Error Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>00000032</td>
<td>INVALID TOKEN FOR FUNCTION</td>
</tr>
<tr>
<td>00000033</td>
<td>TOKEN NOT FOR SAME TCB</td>
</tr>
<tr>
<td>00000034</td>
<td>UNABLE TO JOIN XCF GROUP</td>
</tr>
<tr>
<td>00000064</td>
<td>TARGET SYSTEM NOT AVAILABLE</td>
</tr>
<tr>
<td>00000065</td>
<td>UNABLE TO OBTAIN RECON INFO</td>
</tr>
</tbody>
</table>

When the return code is 04, 08 or 10, the reason for the error will be shown in the error field. The table above shows the reason code and the associated error value. When the return code is 14, then an error was encountered and internal trace entries were created to assist in determining the cause for the error. In this case, additional messages will also be issued indicating the error that occurred.

The following are valid values for error:

Table 8. Valid values for error

<table>
<thead>
<tr>
<th>Reason code</th>
<th>Error Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>00000032</td>
<td>INVALID TOKEN FOR FUNCTION</td>
</tr>
<tr>
<td>00000033</td>
<td>TOKEN NOT FOR SAME TCB</td>
</tr>
<tr>
<td>00000034</td>
<td>UNABLE TO JOIN XCF GROUP</td>
</tr>
<tr>
<td>00000064</td>
<td>TARGET SYSTEM NOT AVAILABLE</td>
</tr>
<tr>
<td>00000065</td>
<td>UNABLE TO OBTAIN RECON INFO</td>
</tr>
</tbody>
</table>

Failed: RPCR encountered an error which has prevented RPCR from completing successfully. This is specified when RC=8.

Error: RPCR encountered a severe error which has prevented RPCR from completing successfully. This is specified when RC=10.

Trace: RPCR encountered an error situation and has created trace entries in the TRACE report which can be used to diagnose the problem. This is specified when RC=14.

When the return code is 04, 08 or 10, the reason for the error will be shown in the error field. The table above shows the reason code and the associated error value. When the return code is 14, then an error was encountered and internal trace entries were created to assist in determining the cause for the error. In this case, additional messages will also be issued indicating the error that occurred.

The following are valid values for error:

Table 8. Valid values for error

<table>
<thead>
<tr>
<th>Reason code</th>
<th>Error Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>00000032</td>
<td>INVALID TOKEN FOR FUNCTION</td>
</tr>
<tr>
<td>00000033</td>
<td>TOKEN NOT FOR SAME TCB</td>
</tr>
<tr>
<td>00000034</td>
<td>UNABLE TO JOIN XCF GROUP</td>
</tr>
<tr>
<td>00000064</td>
<td>TARGET SYSTEM NOT AVAILABLE</td>
</tr>
<tr>
<td>00000065</td>
<td>UNABLE TO OBTAIN RECON INFO</td>
</tr>
</tbody>
</table>

Failed: RPCR encountered an error which has prevented RPCR from completing successfully. This is specified when RC=8.

Error: RPCR encountered a severe error which has prevented RPCR from completing successfully. This is specified when RC=10.

Trace: RPCR encountered an error situation and has created trace entries in the TRACE report which can be used to diagnose the problem. This is specified when RC=14.
<table>
<thead>
<tr>
<th>Reason code</th>
<th>Reason</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>00000066</td>
<td>UNABLE TO GET ONLINE DB STATUS</td>
<td>RPCR was unable to access the RECON data sets to obtain database information. Review the joblog for additional messages indicating the specific error.</td>
</tr>
<tr>
<td>0000067</td>
<td>UNABLE TO SWITCH LOG</td>
<td>RPCR was unable to complete the /SWI OLDS command processing. Review the joblog and the IMS messages for additional messages indicating the specific error.</td>
</tr>
<tr>
<td>0000068</td>
<td>UNABLE TO DBR A DB</td>
<td>RPCR was unable to DBR one or more databases. Review the joblog and the IMS messages for additional messages indicating the specific error.</td>
</tr>
<tr>
<td>0000069</td>
<td>UNABLE TO DBD A DB</td>
<td>RPCR was unable to DBD one or more databases. Review the joblog and the IMS messages for additional messages indicating the specific error.</td>
</tr>
<tr>
<td>000006A</td>
<td>UNABLE TO START DB</td>
<td>RPCR was unable to start one or more databases. Review the joblog and the IMS messages for additional messages indicating the specific error.</td>
</tr>
<tr>
<td>000006B</td>
<td>OPTIONS CONFLICT WITH DB TYPE</td>
<td>The RPCR options specified conflict with database type. Contact IBM software support.</td>
</tr>
<tr>
<td>000006C</td>
<td>GETMAIN REQUEST FAILED</td>
<td>Unable to acquire storage. Ensure that the region size for the Extended Functions job is adequate.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Reason code</th>
<th>Error</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>000006D</td>
<td>NAME TOKEN SERVICES FAILED</td>
<td>An error was encountered using MVS name token services. Contact IBM software support.</td>
</tr>
<tr>
<td>000006E</td>
<td>UNABLE TO LOAD PROGRAM</td>
<td>RPCR was unable to load a required program. Ensure that Extended Functions has been correctly installed and all required load modules are include in the JOBLIB or STEPLIB.</td>
</tr>
<tr>
<td>000006F</td>
<td>DYNAMIC ALLOCATION FAILED</td>
<td>Dynamic allocation failed for one or more database data sets. Review the joblog for details on the specific data set. This normally indicates that some job or user outside of IMS has the data set in use.</td>
</tr>
<tr>
<td>0000070</td>
<td>UNABLE TO LOCATE SUBTASK SVCS</td>
<td>Unable to locate subtask services. Contact IBM software support.</td>
</tr>
<tr>
<td>0000071</td>
<td>SVCS NOT AVAILABLE FOR SUBTASK</td>
<td>Subtask services are not available. Contact IBM software support.</td>
</tr>
<tr>
<td>0000072</td>
<td>ERROR STARTING SERVICE SUBTASK</td>
<td>Unable to start the subtask services. Contact IBM software support.</td>
</tr>
<tr>
<td>0000073</td>
<td>IMS ENVIRONMENTAL ERROR</td>
<td>RPCR encountered an error within the IMS environment. Contact IBM software support.</td>
</tr>
<tr>
<td>0000074</td>
<td>NOT APF AUTHORIZED</td>
<td>The RPCR program is not running in an APF authorized environment. Ensure that the load libraries in the JOBLIB or STEPLIB are APF authorized.</td>
</tr>
<tr>
<td>Reason code</td>
<td>Error</td>
<td>Description</td>
</tr>
<tr>
<td>-------------</td>
<td>------------------------</td>
<td>-----------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>00000075</td>
<td>MISSING IMS DD</td>
<td>The specific RPCR options specified require the IMS DD to be specified. Add the appropriate IMS DD concatenation.</td>
</tr>
<tr>
<td>00000076</td>
<td>INCORRECT DSNAME FOR RECON</td>
<td>The data set name associated with the RECON being accessed does not match the data set name passed to RPCR. Ensure that the same RECON data sets are accessed by the IMS Database Recovery Facility: Extended Functions job and the online IMS system.</td>
</tr>
<tr>
<td>00000077</td>
<td>DB IN USE FOR TIMEOUT DURATION</td>
<td>One or more databases were in use for the timeout duration and RPCR failed. If PARTIAL(Y) was specified, some databases may have succeeded. If PARTIAL(N) was specified, then RPCR failed.</td>
</tr>
<tr>
<td>00000078</td>
<td>DB NOT PROCESSED, PARTIAL(N)</td>
<td>One or more databases were in use for the timeout duration and RPCR failed. If PARTIAL(Y) was specified, some databases may have succeeded. If PARTIAL(N) was specified, then RPCR failed.</td>
</tr>
<tr>
<td>00000096</td>
<td>DB IS NOT REGISTERED</td>
<td>One or more databases passed to RPCR were not registered to DBRC. Ensure all databases are registered.</td>
</tr>
<tr>
<td>000000C8</td>
<td>DB TYPE IS UNSUPPORTED</td>
<td>The database type is either unsupported by RPCR or cannot be determined.</td>
</tr>
<tr>
<td>000000C9</td>
<td>UNABLE TO LOCATE DBD</td>
<td>RPCR could not locate the DBD in the IMS DD concatenation. Ensure the correct blocks are located in the IMS DD concatenation.</td>
</tr>
<tr>
<td>000000CA</td>
<td>UNABLE TO LOCATE PART INFO</td>
<td>RPCR could not locate information for a HALDB partition. Ensure the correct blocks are located in the IMS DD concatenation.</td>
</tr>
<tr>
<td>000000CB</td>
<td>DUPLICATE DBD/PART IN DBLIST</td>
<td>A duplicate database or partition was passed to RPCR in the DBLIST. Contact IBM.</td>
</tr>
<tr>
<td>000000CC</td>
<td>ALLOCATED AFTER DBR</td>
<td>RPCR has determined that one or more databases processed was allocated after it was DBR'd. The RPCR process must fail because it cannot ensure that a correct recovery point was created.</td>
</tr>
<tr>
<td>000000CD</td>
<td>CMDTYPE(DBD) NOT VALID FOR DB</td>
<td>This is an internal error. Contact IBM software support.</td>
</tr>
<tr>
<td>000000CE</td>
<td>DB TYPE NOT PROVIDED</td>
<td>This is an internal error. Contact IBM software support.</td>
</tr>
<tr>
<td>000000CF</td>
<td>DB DSLIST NOT PROVIDED</td>
<td>This is an internal error. Contact IBM software support.</td>
</tr>
<tr>
<td>000000D0</td>
<td>DB IS CURRENTLY BEING USED</td>
<td>One or more databases were in use for the timeout duration and failed RPCR.</td>
</tr>
</tbody>
</table>

**System action:** If the status is WARNING, then RPCR processing may have worked for some databases and failed for others. Otherwise, RPCR processing terminates.

**System programmer response:** Review the RPTRPCR
report and other messages in the joblog to determine the outcome of RPCR.

Module: IOPRPPCI, IOPRPACP

---

**IRO7304E**  
RPCR PARAMETER ERROR IN  
**KEYWORD:** *keyword*, **REASON:** *reason*

**Explanation:** This message indicates that an error was detected in the specified keyword on the RPCR command. Currently, valid combinations of *keyword* and *reason* include:

**KEYWORD:** WAITTIME, **REASON:** VALUE EXCEEDS MAXIMUM ALLOWED

- The value specified for WAITTIME exceeds the maximum allowed value. Refer to the documentation on the RPCR function for valid values.

For more information on the reason codes, see the *IMS System Programming API Reference.*

**System action:** The function is rejected.

**System programmer response:** Correct the value for the specified keyword and resubmit the job.

Module: IOPRPPCI

---

**IRO7305E**  
RPCR FAILED FOR ALL DATABASES,  
**REASON:** *reason*

**Explanation:** This message indicates that the RPCR function has failed to create a recovery point for all databases specified in the DBLIST(). The reasons for this failure include:

**DB IN USE FOR TIMEOUT DURATION**
- Either all databases specified in the DBLIST() were in use for the entire WAITTIME() duration or PARTIAL(NO) was specified and one or more of the databases were in use.

For more information on the reason codes, see the *IMS System Programming API Reference.*

**System action:** The function is rejected.

**System programmer response:** Correct the problem associated with the indicated reason and resubmit the job.

Module: IOPRPACP

---

**IRO7320E**  
APPLICATION ABEND *code*

**Explanation:** This message is issued by the RPCR BMP pause interface when the application program being paused abends. Normally, the abend code will be a U3303 associated with the pseudo-abend when the database is unavailable.

**System action:** If the abend code is 3303, the
Chapter 13. Extended Functions abend codes

This reference section provides detailed information about IMS Database Recovery Facility: Extended Functions abend codes.

For each abend code, the following information is provided where applicable:

**Explanation:**  
The Explanation section explains what the abend code means, why it occurred, and what its variable entry fields are (if any)

**System Action:**  
The System Action section explains what the system will do next

**User Response:**  
The User Response section describes whether a response is necessary, what the appropriate response is, and how the response will effect the system or program

<table>
<thead>
<tr>
<th>Code</th>
<th>Explanation</th>
<th>System action</th>
<th>User response</th>
</tr>
</thead>
<tbody>
<tr>
<td>0200</td>
<td>An error occurred in the use of a BPE service. The abend subcode describes the nature of the error:</td>
<td>The IMS Database Recovery Facility: Extended Functions address space is abnormally terminated.</td>
<td>Retain the diagnostic information and contact IBM Software support.</td>
</tr>
<tr>
<td>X'01'</td>
<td>AWE get failed</td>
<td></td>
<td></td>
</tr>
<tr>
<td>X'05'</td>
<td>AWE enqueue failed</td>
<td></td>
<td></td>
</tr>
<tr>
<td>X'0A'</td>
<td>BPETIMER initialization failed</td>
<td></td>
<td></td>
</tr>
<tr>
<td>X'0F'</td>
<td>BPETIMER cancel failed</td>
<td></td>
<td></td>
</tr>
<tr>
<td>X'14'</td>
<td>BPEATTCH failed</td>
<td></td>
<td></td>
</tr>
<tr>
<td>X'19'</td>
<td>BPELTCB failed</td>
<td></td>
<td></td>
</tr>
<tr>
<td>X'1E'</td>
<td>BPELOADC failed</td>
<td></td>
<td></td>
</tr>
<tr>
<td>X'23'</td>
<td>BPEPOST failed</td>
<td></td>
<td></td>
</tr>
<tr>
<td>X'64'</td>
<td>BPETERM failed</td>
<td></td>
<td></td>
</tr>
<tr>
<td>X'69'</td>
<td>BPELAGET failed</td>
<td></td>
<td></td>
</tr>
<tr>
<td>X'6E'</td>
<td>BPELAREL failed</td>
<td></td>
<td></td>
</tr>
<tr>
<td>X'73'</td>
<td>BPESPRNT failed</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Code</th>
<th>Explanation</th>
<th>System action</th>
<th>User response</th>
</tr>
</thead>
<tbody>
<tr>
<td>0030</td>
<td>An error occurred during the use of an internal IMS Database Recovery Facility: Extended Functions service. The abend subcode describes the nature of the error:</td>
<td>The IMS Database Recovery Facility: Extended Functions address space is abnormally terminated.</td>
<td>Retain the diagnostic information and contact IBM Software support.</td>
</tr>
<tr>
<td>X'01'</td>
<td>Internal logic error</td>
<td></td>
<td></td>
</tr>
<tr>
<td>X'02'</td>
<td>Open for DD failed</td>
<td></td>
<td></td>
</tr>
<tr>
<td>X'03'</td>
<td>CPOOL failed</td>
<td></td>
<td></td>
</tr>
<tr>
<td>X'04'</td>
<td>TCB initialization failed</td>
<td></td>
<td></td>
</tr>
<tr>
<td>X'05'</td>
<td>TCB termination failed</td>
<td></td>
<td></td>
</tr>
<tr>
<td>X'06'</td>
<td>Dynamic allocation failed</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Extended Functions address space is abnormally terminated.

**User response:** Additional error messages are issued before the abend. Review these messages and take the appropriate action. If the problems persists, retain the diagnostic information and contact IBM Software support.

**0060**

**Explanation:** An error occurred during an attempt to acquire storage. The abend subcode describes the nature of the error:

X'01'  Error acquiring SABL control block

**System action:** The IMS Database Recovery Facility: Extended Functions address space is abnormally terminated.

**User response:** Increase the REGION= size on the job and run the job again. If the problem persists, retain the diagnostic information and contact IBM Software support.

**0070**

**Explanation:** An error occurred during IMS Tools Online System Interface processing. The abend subcode describes the nature of the error:

X'01'  FUNC=INIT failed
X'02'  FUNC=CONNECT failed
X'03'  FUNC=REQUEST failed
X'04'  FUNC=RESPONSE failed
X'05'  FUNC=TERM failed
X'06'  FUNC=WRITE failed
X'07'  CPOOL failed
X'08'  FUNC=NEWPAGE failed
X'09'  FUNC=PRTHEAD failed
X'0A'  FUNC=REPSEP failed

**System action:** The IMS Database Recovery Facility: Extended Functions address space is abnormally terminated.

**User response:** Additional error messages are issued before the abend. Review these messages and take the appropriate action. If the problem persists, retain the diagnostic information and contact IBM Software support.

**0080**

**Explanation:** An error occurred during DBRC Interface processing. The abend subcode describes the nature of the error:

X'01'  FUNC=INIT failed
X'02'  FUNC=TERM failed
X'03'  DSPAPI FUNC=INIT failed
X'04'  DSPAPI FUNC=TERM failed
X'05'  DSPAPI FUNC=QUERY failed
X'06'  DSPAPI logic error

**System action:** The IMS Database Recovery Facility: Extended Functions address space is abnormally terminated.

**User response:** Additional error messages are issued before the abend. Review these messages and take the appropriate action. If the problem persists, retain the diagnostic information and contact IBM Software support.
Part 6. Reference: Base Primitive Environment (BPE)

The topics in this section provide you with technical references for the Base Primitive Environment (BPE).

Topics:
• BPE messages and codes
Chapter 14. BPE messages and codes

This section describes the messages and user abend codes that are issued by Base Primitive Environment (BPE) in IBM IMS Recovery Solution Pack for z/OS: IMS Database Recovery Facility: Extended Functions.

Topics:
- “BPE messages”
- “BPE service abend codes” on page 149
- “BPE user abend codes” on page 158

BPE messages

This topic describes the messages from Base Primitive Environment (BPE).

This topic contains General-Use Programming Interface information.

BPE0000I displayoutput

Explanation: Message BPE0000I is the message number associated with output generated by the DISPLAY command verb for BPE resources.

In the message text, displayoutput is output text from the display verb. Typically, when a display command is issued against a resource, the output will consist of header lines, and then one or more data lines containing information about the resources being displayed.

System action: None.

System programmer response: None.

Module: BPECTRA0, BPECURF0, BPEHT100

BPE0001E BPE INITIALIZATION ERROR IN MODULE module

details

Explanation: An error occurred in early BPE initialization. This is a two-line message. Each line begins with the message number.

In the message text:
module The module detecting the error

details A one-line explanation of the type of error that was detected

version The hexadecimal invalid version number from the definition module

The details line of the BPE0001E message further explains the error detected, and can be one of the following values:

• MODULE module HAS AN INVALID VERSION NUMBER version

A BPE definition module had an invalid version number. BPE uses version numbers as a consistency check to ensure that the definition modules being loaded match the version of BPE that is running.

In the message text:
module The BPE definition module in which the invalid version was detected

version The hexadecimal invalid version number from the definition module

• MODULE module IS LINKED REENTRANT - IT MUST BE NON-REENTRANT

A BPE definition module was loaded into key zero storage. BPE definition modules are modules that contain data (control blocks), and thus must be loaded into storage that is in the same storage key in which BPE is running. The most common cause for this error message is linking a definition module as reentrant, which loads it into key zero storage. BPE definition modules should be bound as non-reentrant.

In the message text:
module The BPE definition module that was loaded in key zero storage

• UNKNOWN ERROR, MODULE RC=rc

An internal error occurred that BPE does not recognize.

In the message text:
rc The return code from the failing module

• ERROR LOADING MODULE module BPELOAD RC=rc

Load failed for a module.

In the message text:
module The name of a module that could not be loaded

© Copyright IBM Corp. 2007, 2011
BPE0001E

- **ERROR LOADING MODULES** module... BPELOAD RC=rc
  Load failed for several modules.
  In the message text:
  
  module - The name of the first of several modules that could not be loaded
  rc - The return code from the BPE load service, BPELOAD

- **UNABLE TO CREATE** threadtype THREAD, BPETHDCR RC=rc
  A request to create a BPE thread (internal unit of work) failed.
  In the message text:
  
  threadtype - A 4-character name of the thread type that could not be created
  rc - The return code from the thread create service, BPETHDCR

- **MODULE module IS NOT A VALID type DEFINITION MODULE**
  BPE encountered an error with an internal BPE definition module. BPE uses definition modules to construct its run environment. If a definition module is not correct, BPE cannot build the appropriate environment.
  In the message text:
  
  module - The name of a BPE definition module that is in error
  type - The type of definition module in error

- **UNABLE TO GET NECESSARY STORAGE, BPEGTM RC=rc**
  BPE could not obtain the required storage.
  In the message text:
  
  rc - The return code from the BPE GETMAIN service, BPEGTM

- **UNABLE TO GET STORAGE FOR** blocktype BLOCK, BPECBGET RC=rc
  BPE could not obtain storage for a required control block.
  In the message text:
  
  blocktype - A 4-character name of an internal BPE control block that could not be obtained
  rc - The return code from the BPE control block get service, BPECBGET

- **ERROR READING PROCLIB DATA SET, BPERDPDS RC=rc**
  BPE could not read a PROCLIB DD data set. This message follows message BPE0002E, which provides further details on the error.
  In the message text:
  
  rc - The return code from the lower level initialization module that encountered the error.

- **INVALID CALLABLE SERVICE CODE** code IN MODULE module
  BPE detected an invalid user exit callable service code in the indicated module. BPE components that
run with BPE can define callable services that user exits can use. Each callable service has a callable service code used to request the service. This error indicates that there is a definition error in the specified module.

In the message text:
- **code**: The callable service code in error, in hexadecimal.
- **module**: The name of the callable service module that had the invalid callable service code defined.

**System action:** IMS abend 3400 with subcode 05 follows this message. The address space terminates.

**System programmer response:** If the error described in this message is caused by environmental conditions (for example, insufficient storage or modules missing from IMS.SDFSRESL), correct the indicated problem and restart the address space. Otherwise, save any dump and SYSLOG information and contact the IBM Support Center.

**Module:** BPEAWI00, BPECBI00, BPECMDI0, BPEDSI00, BPEHTI00, BPEINITI0, BPEMSGI0, BPEPCFG0, BPERV1I0, BPETRI00, BPEUXI00

---

**BPE0002E**  **ERROR READING ddname MEMBER member**

**Explanation:** An error occurred trying to read a partitioned data set member. This is a two-line message. Each line begins with the message number.

In the message text:
- **ddname**: The DD name of the data set being read.
- **member**: The member name of the data set being read.
- **details**: A one-line explanation of the type of error detected.

The details line of the BPE0002E message further explains the error detected, and can be one of the following values:
- **OPEN FAILED FOR DATA SET**
  The BPE PDS read service could not open the data set for reading.
- **DATA SET RECORDS ARE NOT FIXED FORMAT**
  The data set specified records that were not in fixed format. The BPE PDS read service requires fixed-format data sets.
- **MEMBER NOT FOUND IN DATA SET**
  The indicated member was not in the partitioned data set.
- **BSAM READ FAILED READING MEMBER**
  An error occurred during the reading of the data set member.

---

**BPE0003E**  **AN ERROR OCCURRED PARSING description**

**AT LINE** line, CHARACTER char

**Failing Text:** "text"

**details**

**Explanation:** An error occurred in the BPE parsing service. This is a four-line message. Each line begins with the message number.

In the message text:
- **description**: A text description of what was being parsed.
- **line**: The line number of the data where the error occurred. For data that is not line-oriented, the line number is omitted.
- **char**: The position of the character on the line where the error was detected. For data that is not line-oriented, char identifies the position of the character within the input data.
- **text**: Up to 16 characters of the text where the error was detected.
- **details**: A one-line explanation of the type of error detected.

The details line of the BPE0003E message further explains the error detected and can be one of the following values:
- **INVALID KEYWORD DETECTED**
  The parser found an unknown keyword in the input data.
- **UNKNOWN POSITIONAL PARAMETER**
  The parser found a positional parameter in the input data when one was not expected.
- **"=" ENCOUNTERED WHEN ";" EXPECTED**
  The parser found an equal sign in the input data when a left parenthesis was expected.
- **EARLY END OF INPUT DATA**
  The input data ended before all levels of parentheses were closed.
- **KEYWORD ENCOUNTERED WHEN VALUE EXPECTED**
  The input data contained a keyword when the parser expected a value.
- **NUMERIC VALUE OUTSIDE OF LEGAL RANGE**

---

Chapter 14. BPE messages and codes  135
A numeric value was outside the allowed range for the parameter.

- **DECIMAL NUMBER CONTAINED NONDECIMAL DIGITS**
  A decimal number contained a non-decimal character.

- **HEXADECIMAL NUMBER CONTAINED NONHEX DIGITS**
  A hexadecimal number contained a non-hexadecimal character.

- **UNKNOWN KEYWORD VALUE DETECTED**
  The parser found a parameter that could be one of a set of keyword values. The parameter was not one of the values in the set.

- **DUPPLICATE KEYWORD PARAMETER DETECTED**
  The parser found an unrepeatable keyword more than once in the input data.

- **A REQUIRED PARAMETER WAS OMITTED**
  A required parameter was not found in the input data.

- **CHARACTER VALUE WAS TOO LONG FOR PARAMETER**
  The character value specified was too long for the parameter field.

- **A REQUIRED VALUE WAS OMITTED**
  A keyword was coded without specifying a value. A value is required.

- **UNEXPECTED RIGHT PARENTHESIS ENCOUNTERED**
  The parentheses in the input data did not match. There were too many closing (right) parentheses.

- **PARSER DIRECTIVE IMBEDDED IN A SUBLIST**
  The parser encountered a parser directive while processing a sublist, that is parser input data enclosed in parentheses. A parser directive is parser metadata, and is delimited by angle brackets (< and >); for example, <SECTION=section_name>. Parser directives are not allowed within parser data sublists.

- **INVALID PARSER DIRECTIVE VALUE**
  The right hand side of a parser directive was not valid. Parser directives must be of the form <directive=value>. The value must be a single word, and you must include the closing angle bracket (>).

- **PARSER DIRECTIVE ENCOUNTERED WHEN VALUE EXPECTED**
  The parser encountered a parser directive when it was expecting the right hand side value of a keyword=value parameter.

**System action:** The BPE parsing service does not take any further action after issuing this message. The caller of the service can provide additional diagnostic messages or terminate the address space.

**System programmer response:** Correct the error as indicated in the BPE0003E message.

**Module:** BPEPARS0
The owning IMS RE component of the TCB that abended.

The 4-character TCB type of the TCB that abended.

Whether the abend is a system abend or a user abend. S is for a system abend; U is for a user abend.

The abend code. For system abends, this is a 3-character hexadecimal code. For user abends, this is a 4-character decimal code.

The value in register 15 at the time of abend. For some abends, this is the abend subcode.

The 4-character name of the BPE thread (internal unit of work) that was running when the abend occurred. The THD=ffff text is only printed for the BPE0006I message issued for the first TCB that abends. It is omitted for BPE0006I messages issued for subsequent TCBs that are abended as a result of the initial abend. THD=ffff is also omitted on the first BPE0006I message if BPE cannot determine the type of thread that abended.

Diagnostic data. ll is the value of the four-byte field SDWAFLGS from the SDWA passed to the BPE ESTAE recovery routine, in hex. Several of the flag bits in this word control IMS BPE ESTAE processing. The word is provided as diagnostic data to assist with problem determination in the BPE ESTAE. Note that this field is not displayed when the ESTAE is not passed an SDWA from the operating system.

Additional diagnostic data. mm is the value of the one-byte field SDWACMPF from the SDWA passed to the BPE ESTAE recovery routine, in hex.

This indicates whether the system is attempting to recover from the abend or not. If the system is trying to recover, the character string "(RETRYING)" is appended at the end of the message at the position indicated by retrystatus. If the abend is not being retried, there is no retrystatus data -- the message ends with DIAG=llllllllllmm.

The abending module's name from its module ID, if it can be determined.

The entry point address of the abending module, if it can be determined.

The PSW (program status word) contents at the time of abend.

The offset within the abending module in which the abend occurred, if it can be determined.

The contents of the registers at the time of abend.

System action: If the abend is being retried (retrystatus is "(RETRYING)" in the first line of the message), the system attempts to recover from the abend and the address space continues to function.

If the abend is not being retried, the action taken after this message depends on whether the abending TCB is considered a critical TCB to the address space. If it is not critical, the TCB is terminated, but the address space continues to run. If it is critical, the address space is terminated abnormally.

System programmer response: Save any dump and SYSLOG information and contact the IBM Support Center.

Module: BPEINIT0, BPESYES0

BPE0007I component BEGINNING PHASE 1 OF SHUTDOWN
Explanation: BPE is beginning the first phase of shutting down the address space.

In the message text:

component

The name of the IMS RE component being shut down.

System action: The address space enters the first phase of termination, in which all of the IMS RE component TCBs are terminated.

Module: BPESYTR0

BPE0008I component BEGINNING PHASE 2 OF SHUTDOWN
Explanation: BPE is beginning the second phase of shutting down the address space.

In the message text:

component

The name of the IMS RE component being shut down.

System action: The address space enters the second phase of termination, in which all of the BPE system TCBs are terminated.

Module: BPESYTR0

BPE0009I component SHUTDOWN COMPLETE
Explanation: Shutdown of the address space completed. BPE is returning to z/OS.

In the message text:

component

The name of the IMS RE component being shut down.

System action: The address space terminates normally.

Module: BPESYTR0
**BPE0010I • BPE0014E**

**BPE0010I  PSW AND REGISTERS AT ABEND ARE NOT AVAILABLE**

**Explanation:** An abend occurred and the BPE system ESTAE routine received control with no SDWA available. The ESTAE is unable to provide diagnostic information normally obtained from the SDWA, such as PSW (program status word) and register contents at abend.

**System action:** BPE continues to process the abend with limited capability because of the lack of the SDWA.

**Module:** BPESYES0

**BPE0011E  ABEND IN BPE SYSTEM ESTAE ROUTINE (BPESYES0)**

**Explanation:** An abend occurred in the BPE system ESTAE module itself while it was processing a prior abend.

**System action:** This message should be followed by a z/OS symptom dump on the z/OS console. Additionally, a SYS1.LOGREC entry is generated for the abend. The BPE ESTAE module attempts to recover from the abend and continues processing the original abend.

**Module:** BPESYES0

**BPE0012E  BPE ETXR UNABLE TO FIND block BLOCK FOR TCB AT address**

**Explanation:** The BPE end of task exit routine (ETXR) was called when a task control block (TCB) in the address space terminated. The routine tried to process the task termination, but was not able to locate a required BPE control block. This is probably caused by internal control block errors or overlays within the address space.

**In the message text:**
- **block** The name of the BPE control block that could not be found
- **address** The address of the terminating TCB

**System action:** BPE attempts to clean up the TCB. However, without the required control blocks, the cleanup might not be successful, and other TCBs in the address space might not terminate.

**System programmer response:** If the address space appears hung (is not processing or does not terminate), cancel the address space with a dump, and contact the IBM Support Center.

**Module:** BPESYET0

**BPE0013E  VERSION MISMATCH BETWEEN BPE AND component**

**BPE MODULE VERSION IS bver.brel.bptrel**

**component WAS ASSEMBLED AT BPE VERSION pver.prel.pptrel**

**Explanation:** The BPE version on which the identified IMS RE component was built does not match the version of the BPE modules that were loaded. This can occur, for example, if the starting IMS RE component was assembled at one BPE version and the IMS.SDFSRESL contained BPE modules at a different BPE level.

This is a three-line message. Each line begins with the message number.

In the message text:
- **component** An up-to-4 character name of the IMS RE component being started
- **bver** The version number of BPE modules
- **brel** The release number of BPE modules
- **bptrel** The point-release number of BPE modules
- **pver** The version number of BPE macros at which the IMS RE component was assembled
- **prel** The release number of BPE macros at which the IMS RE component was assembled
- **pptrel** The point-release number of BPE macros at which the IMS RE component was assembled

**System action:** BPE abends during early initialization with abend 3400, subcode X’09’.

**System programmer response:** Ensure that the IMS RE component being started is using the correct version of the IMS.SDFSRESL because this problem is probably caused by an IMS.SDFSRESL mismatch.

**Module:** BPEINIT0

**BPE0014E  ABEND IN RECOVERY ROUTINE**

**Explanation:** A BPE recovery routine (BRR) that was intended to provide recovery for abends in a section of code itself encountered an abend. (A BRR is an internal recovery routine established by either BPE or the IMS RE component using BPE to protect a functional area. BRRs run in an MVS ESTAE environment and attempt to recover from abends that occur in mainline code.)

**System action:** The BPE ESTAE routine treats this abend as if the recovery routine indicated that it could not recover, and continues abend processing. Usually, this results in the abnormal termination of the address space; however, if there were other BRRs established when the abend occurred, recovery of the original abend still might occur.

**System programmer response:** Save any dump and SYSLOG information, and contact the IBM Support Center.
Module: BPESYES0

BPE0015I UNKNOWN component EXIT TYPE type IN EXIT LIST PROCLIB MEMBER member Ignored

Explanation: While processing a user exit list PROCLIB member, BPE encountered an EXITDEF statement that specified a user exit type that was not defined to BPE. The exit definition for the indicated type is ignored.

In the message text:

- **component**: The name of the owning IMS RE component of the user exit list member that was being processed.
- **type**: The up-to-8 character exit type name that was undefined.
- **member**: The user exit list PROCLIB member name.

**System action**: The user exit definition is ignored. Processing of the user exit list PROCLIB member continues.

**System programmer response**: Examine the indicated user exit list PROCLIB member and correct the EXITDEF statement for the indicated user exit type.

Module: BPEUXRF0

BPE0016I ERROR LOADING component type EXIT module (service RC=rc)

Explanation: While processing a PROCLIB member for a user exit list, BPE was unable to load a user exit that was specified on an EXITDEF statement.

In the message text:

- **component**: The name of the owning IMS RE component of the user exit list member that was being processed.
- **type**: The up-to-8 character exit type name of the exit that could not be loaded.
- **module**: The load module name of the exit that could not be loaded.
- **service**: The name of the failing service that prevented the exit from being loaded. The most likely failing service is BLDL, which is the z/OS service that BPE uses to locate the user exit module to load. This usually means that BLDL could not find the indicated exit module in the STEPLIB or JOBLIB data set for the job.
- **rc**: The 4-digit hexadecimal return code from the failing service.

**System action**: The user exit module is ignored. Processing of the PROCLIB member continues. If this error occurs in the initial PROCLIB member processing during early address space initialization, BPE abends with abend U3400, subcode 5. If this error occurs during REFRESH USEREXIT command processing, the command is ignored, and no changes are made to the user exit environment.

**System programmer response**: Examine the indicated PROCLIB member for the user exit list and correct the EXITDEF statement for the indicated user exit type and module. If the BPE0016I message was issued during address space initialization, restart the address space. If the message was issued in response to a REFRESH USEREXIT command, reissue the command.

Module: BPEUXRF0

BPE0017I MULTIPLE type STATEMENTS IN member PROCLIB MEMBER - LAST WILL BE USED

Explanation: While processing a PROCLIB member, BPE encountered multiple statements of a specific type when only one was expected. BPE uses the last statement of the duplicated type; the others are ignored.

In the message text:

- **type**: A short description of the type of statement that was duplicated.
- **member**: The PROCLIB member name.

**System action**: Processing continues.

**System programmer response**: No response is required. However, you might want to examine the indicated PROCLIB member to ensure that the values specified on the final statement are correct.

Module: BPETRI00, BPEUXRF0

BPE0018I DUPLICATE EXIT module SPECIFIED FOR component type EXIT - IGNORED

Explanation: While processing a PROCLIB member for a user exit list, BPE found the same user exit module specified more than once in a single exit list (EXITS parameter) on an EXITDEF statement. Only the first instance of the module is in effect; all subsequent specifications of the module are ignored.

In the message text:

- **module**: The name of the user exit module that was specified more than once.
- **component**: The name of the owning IMS RE component of the user exit list member that was being processed.
- **type**: The up-to-8 character exit type name of the exit being processed.

**System action**: The duplicate user exits after the first occurrence are ignored. Processing of the user exit PROCLIB continues.

**System programmer response**: No response is required. However, you might want to examine the EXITDEF statement to insure that the exits specified are listed in the correct order for your installation.
exits are called in the order they are listed on the EXITS statement.

**Module:** BPEUXRF0

---

**BPE0019E** component type USER EXIT MODULE
module ABEND code

**Explanation:** An abend occurred while a user exit module was in control. In control means that BPE gave control to the exit. However, the abend does not have to be in the exit module itself. This message is also issued if the exit called another module, which then terminated abnormally.

In the message text:

- **component** The name of the owning IMS RE component of the user exit type being called.
- **type** The up-to-8 character exit type name of the exit that abended.
- **module** The load module name of the exit that abended.
- **code** The abend code. For system abends, the format of code is $xxx, where xxx is the 3-digit abend code in hexadecimal. For user abends, the format of code is $ddddd, where dddd is the 4-digit abend code in decimal.

**System action:** BPE protects all user exits with a recovery routine. When a user exit abends, BPE attempts to recover from the abend and continue processing. BPE's recovery actions include the following processes:

- The first time a specific user exit module abends after it is loaded, BPE issues an SDUMP to dump the address space for the abend. For abends after the first abend, BPE does not dump the address space again; however, it does generate a SYS1.LOGREC entry for each abend occurrence.
- If the number of abends for a specific user exit module reaches or exceeds the abend limit value for the exit's user exit type, no further calls are made to the exit until it is refreshed. The abend limit for an exit type is specified by the ABLIM parameter on the EXITDEF statement.

After a user exit abends, BPE attempts to call any other exit modules that were specified after the failing exit in the EXITS exit list.

**System programmer response:** Examine the SDUMP, the SYS1.LOGREC entries, or both to determine the problem with the user exit. Correct the problem with the exit, rebind the exit to the job's libraries, and issue the REFRESH USEREXIT command. The REFRESH USEREXIT command loads a copy of the corrected exit and causes BPE to resume calling the exit module whenever exits of its type are invoked.

**Module:** BPEUXCL0

---

**BPE0020E** ABEND LIMIT REACHED FOR component type USER EXIT MODULE
module

**Explanation:** The indicated user exit module reached the abend limit for the exit type as specified in the ABLIM parameter on the EXITDEF statement.

In the message text:

- **component** The name of the owning IMS RE component of the user exit type being called (for example, BPE, CQS, OM, RM, or SCI).
- **type** The up-to-8 character exit type name of the exit that terminated abnormally.
- **module** The load module name of the exit that terminated abnormally.

**System action:** No further calls are made to the indicated user exit module until the next refresh of the exit. Refreshing the user exit resets the abend count for the exit to zero, allowing it to be called again.

**System programmer response:** Examine the SDUMP, the SYS1.LOGREC entries, or both to determine the problem with the user exit. Correct the problem with the exit, rebind the exit to the job's libraries, and issue the REFRESH USEREXIT command. The REFRESH USEREXIT command loads a copy of the corrected exit and causes BPE to resume calling the exit module whenever exits of its type are invoked.

**Module:** BPEUXCL0

---

**BPE0021E** ABEND code IN BPE SVC INIT
MODULE BPESVC10, PSW=psw1 psw2

**Explanation:** An abend occurred while module BPESVC10 was in control. Module BPESVC10 is the module that initializes the BPE SVC routine (an internal SVC used by BPE and other IMS RE components). BPESVC10 processing is protected by an internal ESTAE, which attempts to retry from the abend and clean up any global resources (common storage, MVS Enqueues) that BPESVC10 obtained. Message BPE0021E is issued to alert the operator that an abend occurred.

In the message text:

- **code** The abend code. For system abends, the format of code is $xxx, where xxx is the 3-digit abend code in hexadecimal. For user abends, the format of code is $ddddd, where dddd is the 4-digit abend code in decimal.
- **psw1** The first word of the PSW (program status word) at abend.
- **psw2** The second word of the PSW at abend.

**System action:** BPE collects diagnostic data about the abend, and then resumes running in a cleanup routine within BPESVC10. This routine attempts to release any global resources that BPESVC10 obtained as a part of its processing. The routine returns to the BPESVC10's caller, which can choose to continue processing or to terminate abnormally.

**Module:** BPEUXCL0
The first time that BPESVC10 abends, its ESTAE creates an SDUMP of the address space, and causes a record to be written to the SYS1.LOGREC data set to document the abend. If BPESVC10 abends a second time or more, its ESTAE does not create another SDUMP. However, it writes a record to SYS1.LOGREC.

**System programmer response:** Save any dump, SYSLOG, and SYS1.LOGREC information and contact the IBM Support Center.

**Module:** BPESVC10

---

**BPE0022E**  
**ABEND command IN BPE SVC PROCESSING, PSW=psw1 psw2**

**Explanation:** An abend occurred during BPE SVC processing. The BPE SVC module (BPESVC00) establishes an ESTAE to protect its processing. This ESTAE attempts to retry from the abend and clean up any global resources (common storage, MVS Enqueues) that BPESVC00 obtained. Message BPE0022E is issued to alert the operator that an abend occurred.

In the message text:
- **code**: The abend code. For system abends, the format of code is Sxxx, where xxx is the 3-digit abend code in hexadecimal. For user abends, the format of code is Udddd, where dddd is the 4-digit abend code in decimal.
- **psw1**: The first word of the PSW (program status word) at abend.
- **psw2**: The second word of the PSW at abend.

**System action:** BPE collects diagnostic data about the abend, and then resumes running in a cleanup routine within BPESVC00. This routine attempts to release any global resources that BPESVC00 obtained as a part of its processing, and returns to the caller of BPESVC00, which can choose to continue processing or to terminate abnormally.

The first time that BPESVC00 abends, its ESTAE creates an SDUMP of the address space, and causes a record to be written to the SYS1.LOGREC data set to document the abend. If BPESVC00 abends a second time or more for the same SVC call, its ESTAE does not create another SDUMP. However, it writes a record to SYS1.LOGREC.

**System programmer response:** Save any dump, SYSLOG, and SYS1.LOGREC information and contact the IBM Support Center.

**Module:** BPESVC00

---

**BPE0023I**  
**command COMMAND REJECTED**

**Explanation:** A command was issued, but could not be processed. The command is rejected.

Depending on the IMS RE component, this message might not be issued when a command is rejected. The IMS RE component using BPE might choose to issue its own message rejecting the command, and might request that BPE not issue the BPE0023I message.

In the message text:
- **command**: The command that was rejected. If the command that was entered was the MVS STOP command (P jobname), MVS STOP displays as the command. For all other commands, the command verb and resource type are displayed (for example, UPDATE TRACETABLE or REFRESH USEREXIT).

**System action:** The command is not processed.

**Module:** BPEMOD00, BPECMD10

---

**BPE0024E**  
**command COMMAND FAILED**

**Explanation:** The processing for a command failed.

Depending on the IMS RE component, this message might not be issued when a command fails. The IMS RE component using BPE might choose to issue its own message about a command failure, and might request that BPE not issue the BPE0024E message.

In the message text:
- **command**: The command that failed. If the command that was entered was the MVS STOP command (P jobname), MVS STOP displays as the command. For all other commands, the command verb and resource type are displayed (for example, UPDATE TRACETABLE or REFRESH USEREXIT).

**System action:** The command failed.

**Module:** BPEMOD00, BPECMM1D0

---

**BPE0025I**  
**STOP OF component IS IN PROGRESS**

**Explanation:** An MVS STOP command was issued for an IMS RE component that is running with BPE. BPE is processing the stop request.

Depending on the IMS RE component, this message might not be issued when the address space is stopped. The IMS RE component using BPE might choose to issue its own shutdown messages and might request that BPE not issue the BPE0025I message.

In the message text:
- **component**: The name of the IMS RE component that is being stopped.

**System action:** BPE initiates a shutdown of the address space.

**Module:** BPEMOD00
BPE0026E CLEANUP FAILURE RC=rc RSN=rsn

Component

Explanation: An error occurred during BPE resource cleanup processing. Some resources might not be properly cleaned up.

BPE establishes a resource manager routine to clean up global resources when an address space using BPE services terminates. If the resource manager cannot clean up a particular resource, it issues a BPE0026E message.

In the message text:
rc The return code, if applicable, from the failing service. This code might help to identify the cause of the failure.
rsn The reason code, if applicable, from the failing service.

Component A short text string that identifies the component or resource that could not be cleaned up. Possible components are:

- BPESVC: Cleanup failed for the BPE SVC service. The return code and reason code in the message are from the BPE SVC EOMCLEANUP function. This error indicates that some of the SVC functions registered by the terminating address space might not have been properly deregistered.
- ALESERV: A call to the MVS ALESERV service to obtain the current address space's STOKEN failed. The return code in the message is the return code from the ALESERV macro; the reason code is always zero.

System action: BPE resource cleanup processing continues with the next resource, and the address space terminates.

System programmer response: Depending on the cause of the cleanup failure, BPE might have created an SDUMP. If this is the case, save the dump, SYSLOG, and SYS1.LOGREC information and contact the IBM Support Center. If BPE did not create an SDUMP, obtain the return code, reason code, and component from the BPE0026E message and contact the IBM Support Center with this information.

Module: BPERSM00

BPE0028I SDUMP FAILED FOR abend ABEND, RC=rc, RSN=rsn

Explanation: BPE issued an SDUMP call to z/OS to produce a dump of the address space after an abend, but the SDUMP was not successful.

In the message text:
abend The abend code for which the storage dump was created. For system abends, the format of code is Sxxx, where xxx is the 3-digit abend code in hexadecimal. For user abends, the format of code is Udddd, where dddd is the 4-digit abend code in decimal.
rc The return code from the MVS SDUMP macro.
rsn The reason code from the MVS SDUMP macro.

System action: The SDUMP is skipped.

System programmer response: Use the return and reason codes from the MVS SDUMP macro to determine the cause of the SDUMP failure. These return and reason codes are documented in z/OS MVS Programming: Authorized Assembler Services Reference, Volume 3 (SA22-7611). If appropriate, correct the cause of the failure so that future dumps are not lost.

Module: BPEINIT0, BPERSM00, BPESVC10, BPESVC00, BPESYES0

In the message text:
code The abend code. For system abends, the format of code is Sxxx, where xxx is the 3-digit abend code in hexadecimal. For user abends, the format of code is Udddd, where dddd is the 4-digit abend code in decimal.
psw1 The first word of the PSW (program status word) at abend.
psw2 The second word of the PSW at abend.
BPE0029I DAE SUPPRESSED DUMP FOR abend ABEND

**Explanation:** BPE issued an SDUMP call to z/OS to produce a dump of the address space after an abend, but the SDUMP was suppressed by z/OS dump analysis and elimination (DAE).

BPE recovery routines gather symptom string data related to an abend, and provide this data to z/OS when an SDUMP is requested. If DAE is enabled, z/OS suppresses duplicate dumps, for example, dumps that have symptom strings identical to previously captured dumps. DAE is controlled through the MVS ADYSETxx PARMLIB member and the MVS SET DAE command. For details on specifying DAE options, see z/OS MVS Initialization and Tuning Guide (SA22-7591).

In the message text:

*abend* The abend code for which the storage dump was created. For system abends, the format of code is Sxxx, where xxx is the 3-digit abend code in hexadecimal. For user abends, the format of code is Udddd, where dddd is the 4-digit abend code in decimal.

**System action:** The SDUMP is skipped. Note that a BPE-generated dump is suppressed if its symptom string matches a previous dump, and if the current DAE setting in ADYSETxx is either SUPPRESS OR SUPPRESSALL.

**System programmer response:** None.

**Module:** BPEINIT0, BPERSM00, BPESVC10, BPESVC00, BPESYES0

---

BPE0030I displayheader

**Explanation:** BPE0030I is the message number associated with the header line or lines generated by the DISPLAY command for BPE resources.

In the message text, displayheader is one or more lines of header information associated with the output from a DISPLAY command.

For more information see the DISPLAY command.

---

BPE0031E verb COMMAND IS INVALID

or

verb resourcetype COMMAND IS INVALID

**Explanation:** A command was issued that BPE did not recognize. Even if you entered a valid command verb, this message might be generated if you omitted a required resource type or specified a resource type to which the verb does not apply. For example, if you entered:

```
DISPLAY NAME(DISP)
```

you would receive the error message:

BPE0031E DISPLAY COMMAND IS INVALID

This error occurs because the DISPLAY command requires a resource type on which it is to operate. Similarly, if you entered:

```
REFRESH TRACETABLE NAME(AWE)
```

you would receive the error message:

BPE0031E REFRESH TRACETABLE COMMAND IS INVALID

This error occurs because the REFRESH command applies to the USEREXIT resource type, not the TRACETABLE resource type.

In the message text:

*verb* Is the command verb from the command that was issued.

*resourcetype* Is the resource type from the command that was issued, if present.

**System action:** The command is ignored.

**System programmer response:** Reenter the command with the correct verb, or verb and resource type.

**Module:** BPECMD00, BPECMD10

---

BPE0032I verb COMMAND COMPLETED

or

verb resourcetype COMMAND COMPLETED

**Explanation:** A command completed processing. This message is issued in two cases:

- For commands that generate a variable number of lines of output, such as DISPLAY commands. Message BPE0032I is issued to indicate the end of the command output.
- For commands that generate no other messages. Message BPE0032I is issued to provide feedback that command processing is complete.

Message BPE0032I is not issued when a command generates a fixed number of lines of output, such as a single response line or error message line.

In the message text:

*verb* Is the command verb from the command that was issued.

*resourcetype* Is the resource type from the command that was issued, if present.

**System action:** None.

**System programmer response:** None.

**Module:** BPECMD10
BPE0033E · BPE0037E

BPE0033E  MISSING COMMAND VERB

Explanation: A command was entered that is missing a command verb, which is the first word of the command string. You receive this message only if you enter a command that consists of only keyword(value) parameters. For example, if you enter NAME(DISP) OWNER(BPE), the BPE0033E message is issued. This command does not contain a non-keyword parameter to act as the command verb.

System action: The command is ignored.

System programmer response: Reenter the command with the proper command verb.

Module: BPECMD00

BPE0034E  NO MATCH FOUND FOR resource_type WITH NAME name

Explanation: A name or wild card pattern on the NAME parameter of a command did not match any instances of the requested resource type. For example, the command:

```
DISPLAY TRACETABLE NAME(ZZZZ,QQ*)
```

generates the following response:

```
BPE0034E NO MATCH FOUND FOR TRACETABLE WITH NAME "ZZZZ"
BPE0034E NO MATCH FOUND FOR TRACETABLE WITH NAME "QQ*"
```

In the message text:

- resource_type is the resource type specified in the command.
- name is the name or wild card pattern specified on the NAME parameter that did not match any instance of the resource type.

System action: If some of the names listed on the NAME parameter of the command matched existing resource instances, the command processes against those instances. If none of the names listed on the NAME parameter matched existing resource instances, the command is ignored.

System programmer response: Reenter the command with the proper resource type names.

Module: BPECTRA0, BPECURF0

BPE0035E  INVALID VALUE "value" FOR KEYWORD keyword

Explanation: The value specified on the indicated keyword was not valid. For example, the command:

```
UPDATE TRACETABLE NAME(DISP) OWNER(ABCD) LEVEL(GROUND)
```

would generate the following response:

```
BPE0035E INVALID VALUE "ABCD" FOR KEYWORD OWNER
BPE0035E INVALID VALUE "GROUND" FOR KEYWORD LEVEL
```

In the message text:

- value is the keyword parameter value that is in error.
- keyword is the keyword whose value is in error.

System action: The command is ignored.

System programmer response: Reenter the command with the proper value on the indicated keyword.

Module: BPECTRA0, BPECURF0

BPE0036E  INTERNAL COMMAND PROCESSING ERROR, service Rc=rc

Explanation: An internal error in BPE occurred while processing the command. For details on the failure, refer to the service return code, as indicated by service and rc, in "BPE service return codes" in IMS Version 10: Messages and Codes Reference, Volume 4: IMS Component Codes.

In the message text:

- service is the name of the failing service that prevented the command from processing.
- rc is the return code from the failing service call.

System action: The command is ignored.

System programmer response: Some internal errors that result in message BPE0036E could be due to temporary conditions, such as a temporary lack of storage. Other errors indicate possible logic errors within product code. Reenter the command and see if the error reoccurs. If it does, issue a console dump of the address space immediately after entering the command and contact the IBM support center. To ensure that trace information necessary to the resolution of the problem is available, you can turn on the following BPE traces at LEVEL(HIGH): CMD, SSRV, STG, DISP.

Module: BPECMD00, BPECMD10, BPECTRA0, BPECURF0

BPE0037E  UNABLE TO START BPE: details

Explanation: The BPE address space startup module, BPEIN100, encountered an error that prevented a BPE address space from starting. In the message text:

- details is a description of the problem that prevented the address space from starting.

- details is one of the following situations:

  NOT APF AUTHORIZED
  The address space being started is not APF-authorized. All BPE and IMS RE component modules must be in data sets that are APF-authorized. In addition, module BPEIN100 must be bound with authorization code 1:
  SETCODE AC(1)

  NOT RUNNING IN KEY 7
  Module BPEIN100 received control in a key other than key 7. BPE must run in key 7. Ensure that module
BPEINI00 is correctly added to MVS PPT (the program properties table).

**BPEINIT= PARAM MISSING OR INVALID**

Either module BPEINI00 could not find the BPEINIT= keyword on the JCL startup parameters (PARM=), or the specified module name was missing or invalid. The BPEINIT= parameter is required and identifies the name of the BPE startup parameter module. The parameter module defines the type of IMS RE component address space to start.

**ERROR LOADING BPEINIT= MODULE**

The module specified by the BPEINIT= keyword on the JCL startup parameters (PARM=) could not be loaded. In the message text:

- **module** The name of the module specified in the BPEINIT= keyword that could not be loaded.

**BPEINIT= MODULE**

**LINKED AS REENTRANT**

The module specified by the BPEINIT= keyword on the JCL startup parameters (PARM=) was loaded into key 0 storage; this situation probably occurred due to being bound as reentrant. This module must be bound as non-reentrant. In the message text:

- **module** The name of the module specified in the BPEINIT= keyword that was linked as reentrant.

**MISSING OR INVALID PARMS IN BPEINIT= MODULE**

The BPEINI00 module could not find a valid startup parameter data area (UCDB block) in the module specified by the BPEINIT= keyword on the JCL startup parameters. In the message text:

- **module** The name of the module specified in the BPEINIT= keyword.

**UNABLE TO GET STORAGE FOR COPY OF MVS PARAMETERS**

The BPEINI00 module could not get storage for a copy of the z/OS input parameters.

---

**UNKNOWN ERROR**

The BPEINI00 module encountered an unknown error.

**System action:** The starting address space is abnormally terminated with BPE abend 3403.

**Operator response:** If the error described in message BPE0037E is due to environmental conditions (for example, incorrect authorization or invalid parameters), correct the indicated problem and restart the address space. Otherwise, save any job log and system log information and contact your IBM Support Center.

**Module:** BPEINI00

**BPE0038E BPE STACK MANAGER INTERNAL ERROR**

**Explanation:** The BPE stack storage manager was unable to obtain storage for a new stack block.

**System action:** The BPE stack storage manager issues an SDUMP the first time this condition arises to gather diagnostic data about the problem. The stack manager then suspends the caller until a stack block is eventually freed by another thread.

**System programmer response:** This message is issued when storage is not available to satisfy an internal request for stack storage within the address space. BPE suspends the unit of work that is requesting additional stack storage until storage becomes available. However, this suspension could lead to reduced function and performance in the product that is running in the address space. You should increase the available region size for the address space and shut it down and restart it as soon as possible. If you continue to get this message, save the SDUMP produced by BPE and contact the IBM Support Center.

**Module:** BPESTKM0

**BPE0039E BPE STACK MANAGER MULTIPLE INTERNAL ERRORS**

**Explanation:** The BPE stack storage manager encountered multiple cases where it could not obtain storage for a new stack block.

**System action:** The BPE stack storage manager issues message BPE0038E the first time it is not able to get a new stack block when one is needed. Subsequent failures do not generate additional BPE0038E messages. However, if the stack manager repeatedly cannot get storage, it will eventually issue message BPE0039E, indicating that the storage problem is likely a chronic one, and that action should be taken quickly to resolve the storage shortage. In addition to issuing the BPE0039E message, BPE also creates a second SDUMP after the repeated failures to capture additional diagnostics.

**System programmer response:** This message is issued
by the BPE stack manager after repeated failures to obtain storage. It is unlikely that the address space will function well at this point. You should increase the available region size for the address space and immediately shut it down and restart it. If you continue to get this message, save the SDUMP produced by BPE and contact the IBM Support Center.

Module: BPESTKM0

BPE0040E DYN type FAILED FOR dstype, RC=rc/rsn

Explanation: Dynamic allocation or unallocation of a data set failed. In the message text:
- type The type of service that failed.
  - ALLOC for dynamic allocation
  - UNALLOC for dynamic unallocation
- dstype The data set type or DD name of the data set that BPE could not allocate or deallocate.
- rc The return code from the MVS DYNALLOC call.
- rsn The reason code from the MVS DYNALLOC call.
- dsn The name of the data set that BPE was trying to allocate or deallocate.

System action: The BPE dynamic allocation service returns to its caller. The caller might take some additional action based on the failure, such as issue an additional message or an abend.

System programmer response: Analyze the state of the data set that failed dynamic allocation or unallocation. The data set must be cataloged. For information on the dynamic allocation or unallocation return codes and reason codes, see z/OS MVS Programming: Authorized Assembler Services Guide (SA22-7608).

Module: BPEDYA00

BPE0041E UNABLE TO ALLOCATE REQUESTED STORAGE

Explanation: The BPE storage allocation service, BPEGETM, could not allocate the storage requested by its caller. This message will only be issued the first time that BPEGETM could not obtain storage.

System action: BPEGETM returns an error return code to its caller.

System programmer response: The storage shortage which lead to this message being issued might be due to the region size for the address space being set too small. Increase the region size by changing the REGION= parameter on the address space JCL, and restart the address space.

If the problem persists, save any dumps and contact the IBM Support Center. If no dumps were produced, create a console dump of the address space that is causing the BPE0041E message to be issued.

Module: BPESTG00

BPE0042E ERROR INITIALIZING BPE LIMITED FUNCTION SERVICES: details

Explanation: The initialization of BPE limited function services (LFS) failed. BPE LFS is used in certain address spaces (such as the IMS control region) that do not otherwise use BPE services. If BPE is unable to successfully initialize the LFS environment, it issues message BPE0042E to document the error.

In the message text:
- details The type of error that was detected.
  The details portion of the BPE0042E message further explains the error that was detected, and the explanation can be one of the following values:
  - UNABLE TO GET STORAGE
    A STORAGE OBTAIN call failed to get storage necessary to initialize the BPE LFS environment.
  - UNABLE TO LOAD BPEZLFS0
    BPE could not load the main BPE LFS module, BPEZLFS0.
  - INVALID BPEZLFS0 VECTOR TABLE
    The vector table at the beginning of module BPEZLFS0 was invalid. This table is used to locate all of the functional routines contained within the BPEZLFS0 load module. When processing the table, BPE detected an invalid entry.
  - UNABLE TO CREATE NAME-TOKEN
    BPE attempted to create an z/OS name-token for use later by BPE LFS. The create call failed.
  - UNKNOWN ERROR
    An unexpected error occurred during BPE LFS initialization.

System action: BPE LFS initialization fails, and a non-zero return code is passed back to the caller. BPE LFS services are not available. When the caller is the IMS control region, IMS initialization fails. IMS will issue message DFS2930I, followed by an abend 0071.

System programmer response: If the error described in this message is caused by environmental conditions (for example, insufficient storage or modules missing from IMS.SDFSRESL), correct the indicated problem and restart the address space. Otherwise, save any dump and SYSLOG information, and contact the IBM support center.

Module: BPEZINI0
**Explanation:** An error occurred during external trace processing.

In the message text:
- `module` The module detecting the error.
- `details` The details line further explains the error detected. The `details` variable can be one of the following values:
  - **UNABLE TO OBTAIN** `type` STORAGE
    - Contact your IMS system programmer.
  - **UNABLE TO ENQUEUE AN AWE TO START EXTERNAL TRACE**
    - Contact your IMS system programmer.
  - **UNABLE TO ENQUEUE AN AWE TO STOP EXTERNAL TRACE**
    - Contact your IMS system programmer.
  - **EXTTRACE PARM NOT DEFINED IN BPE CONFIG PARM MBR**
    - Define EXTTRACE parameter in the BPE configuration parameter member (BPECFG=) and retry bringing up the address space.
  - **UNIT() SPECIFIED WITHOUT VOLSER() ON EXTTRACE PARAMETER**
    - Modify EXTTRACE parameter in the BPE configuration parameter member (BPECFG=) to include both UNIT() and VOLSER().
  - **VOLSER() SPECIFIED WITHOUT UNIT() ON EXTTRACE PARAMETER**
    - Modify EXTTRACE parameter in the BPE configuration parameter member (BPECFG=) to include both UNIT() and VOLSER().
  - **STORCLAS() AND UNIT/VOL SPECIFIED ON EXTTRACE PARAMETER**
    - Modify EXTTRACE parameter in the BPE configuration parameter member (BPECFG=) to include either UNIT()/VOLSER() combination or STORCLAS().
  - **SPACEUNIT() AND AVGREC() SPECIFIED ON EXTTRACE PARAMETER**
    - Modify EXTTRACE parameter in the BPE configuration parameter member (BPECFG=) to include either SPACEUNIT or AVGREC.
  - **UNABLE TO LOAD DYNAMIC ALLOCATION MODULE**
    - Contact your IMS system programmer.
  - **UNABLE TO ALLOCATE EXTERNAL TRACE DATA SET**
    - Contact your IMS system programmer.

**System action:** BPE external trace functionality is no longer available.
System programmer response: Messages that further explain the reason for the error might preceed this message. Refer to the job log output from the failed job for explanatory messages.

If you cannot determine and correct an environmental cause for the problem, save any storage dump and SYSLOG information. Then contact IBM Software Support.

Module: BPETRBG0, BPETRFO0, BPETROC0

---

**Explanation:** An error occurred during external trace processing and external trace functionality is no longer active. This message is also issued if the external trace function is stopped by using a BPE UPDATE TRTAB command.

System action: BPE external trace functionality is no longer available.

System programmer response: Messages that further explain the reason for the error might preceed this message. Refer to the job log output from the failed job for explanatory messages.

Issue UPD TRTABLE command to restart the external trace functionality.

If you cannot determine and correct an environmental cause for the problem, save any storage dump and SYSLOG information, and contact IBM Software Support.

Module: BPETRFO0, BPEXXTF0

---

**Explanation:** An address space using BPE could not be started because BPE required a system facility that was not available. Facilities that prevent BPE from starting can include hardware (including not having a particular hardware level or facility) and software (including not having a required software component installed or at the correct level). Information in the message text is as follows:

- **ver.rel** The BPE internal version and release levels. For example, 1.6 is the internal BPE version and release for IMS Version 10.
- **facility** The facility that BPE needed but was not available. The facility variable can be one of the following values:
  - **Z/ARCHITECTURE MODE** BPE 1.6 and above can only run on processors running in z/Architecture® mode. ESA and below is no longer supported. You must either re-IPL your z/OS system in z/Architecture mode, or you must move the address space you are trying to start to a different machine that is running in z/Architecture mode.
  - **64-BIT VIRTUAL SUPPORT** You can only run BPE 1.6 and above under a z/OS release that has 64-bit virtual storage support. If you receive this message, then you are running BPE under a non-supported z/OS level that does not allow 64-bit virtual storage. You must upgrade your z/OS to the minimum level for the current release of BPE.

System action: The address space terminates with a 3400 abend.

System programmer response: Make the facility indicated available on the machine on which you run the BPE address space, or else run the BPE address space on a different machine that has the indicated facility.

Module: BPEINIT0

---

**Explanation:** A BPE external trace data set, indicated by **dsn**, was opened or closed.

System action: An external trace data set is opened if the current data set becomes full, or if the external trace functionality is started. An external data set is closed when it becomes full or when the external trace functionality is stopped.

System programmer response: None.

Module: BPETROC0

---

**Explanation:** An error occurred when processing the OPTION(REREAD) option of the UPDATE TRACETABLE command. The UPDATE command is not processed.

The **details** of the message provides more information about the error that was detected, and **details** can be one of the following values:

- **UNABLE TO GET NECESSARY STORAGE** BPE was not able to obtain the storage required to reprocess the BPE configuration PROCLIB member when trying to obtain any updated value for the EXTRACE parameter.
- **ERROR READING BPE CONFIGURATION PROCLIB MEMBER member** An error occurred reading the BPE configuration PROCLIB member. This message is preceeded by message BPE0002E, which provides details about the error. In the **details**
text, *member* identifies the name of the BPE configuration PROCLIB member that could not be read.

**ERROR PARSING BPE CONFIGURATION PROCLIB MEMBER**

An error occurred parsing the BPE configuration PROCLIB member. If this is due to a syntactical error within the data being parsed, then this message is preceded by message BPE0003E describing the error. In the *details text*, *membername* is the name of the BPE configuration PROCLIB member that could not be parsed.

**UNKNOWN ERROR rc FROM PARSING MODULE BPEPCF10**

Module BPEPCF10 returned an unexpected return code to the UPDATE TRACETABLE command processor. BPEPCF10 is the module that reads and parses the BPE configuration PROCLIB member. In the *details text*, *rc* is the unknown return code from module BPEPCF10.

**LOAD FAILED FOR BPEPCF10, BPELOADC RC=rc**

Module BPEPCF10 could not be loaded. BPEPCF10 is the module that reads and parses the BPE configuration PROCLIB member. In the *details text*, *rc* is the return code from the BPELOADC service used to load and call BPEPCF10.

**BPE CONFIG PROCLIB MEMBER NAME WAS NOT CODED ON STARTUP PARMS**

The BPE configuration PROCLIB member could not be reread because no member name was specified on the startup parameters for the address space. You must include the BPECFG= parameter in the job parameters and must specify a valid BPE configuration PROCLIB member name for the OPTION(REREAD) parameter to be usable on the UPDATE TRACETABLE command.

**PREVIOUS UPDATE COMMAND REREAD PROCESSING IN PROGRESS**

A previous UPDATE TRACETABLE command specified OPTION(REREAD). The reprocessing of the EXTTRACE statement for this previous UPDATE command has not yet completed. A second UPDATE TRACETABLE command with OPTION(REREAD) cannot be processed until the first command’s processing has completed.

**NO MATCHING EXTTRACE STATEMENT FOUND FOR ADDRESS SPACE**

The BPE configuration PROCLIB member did not contain an EXTTRACE statement that applies to this address space. In order to apply, an EXTTRACE statement must either contain the COMP= parameter specifying the component type for the current address space, or else must have no COMP= parameter specified (so that the EXTTRACE statement applies to all address spaces).

**System action:** The UPDATE TRACETABLE command fails, and no change is made to the trace table settings.

**System programmer response:** For errors that are correctable (for example: parsing errors, incorrect parameters, incorrectly named PROCLIB members), correct the error and reissue the UPDATE TRACETABLE command.

Some internal errors that result in message BPE0047E could be due to transient conditions, such as a temporary lack of storage. Enter the command again and check whether the error reoccurs. If it does, issue a console dump of the address space immediately after entering the command, and contact the IBM Software Support. To ensure that trace information necessary to the resolution of the problem is available, you can turn on the following BPE traces at LEVEL(HIGH): CMD, SSERV, STG, DISP

**Module:** BPECTRA0

---

### BPE service abend codes

This section identifies and explains user abend codes issued by Base Primitive Environment (BPE).

**BPEATTCH**

**Explanation:** Attach a TCB.

The following is a list of return codes and their explanations:

**Return code**

**Explanation**

*X'00000004'*

The TCB index passed to BPEATTCH was not a valid index.

*X'00000008'*

There is no TCB type of the requested type defined to BPE.

*X'0000000C'*

A new TCB of the indicated type could not be attached because there were no available entries for it in the BPE TCB table. The maximum number of TCBs of this type is already attached.

*X'00000010'*

The z/OS ATTACH for the new TCB failed.
**BPEAWSRV • BPEBPCRE**

X'00000014'
- The caller is a non-thread running under the parent TCB of the TCB that was requested to be attached. This is not a valid environment for the BPEATTCH call.

X'00000020'
- Internal processing error: The enqueue of an AWE to the TCB attach processor for the requested TCB type failed.

X'00000024'
- Internal processing error: The BPEWAIT for a response from the TCB attach processor for the requested TCB failed.

X'00000028'
- Internal processing error: The routine called by BPEATTCH enqueued an AWE to the TCB attach processor for the requested TCB; however, the DDB address it saved in the AWE was invalid, and the attach processor rejected the attach request.

X'0000002C'
- Internal processing error: The routine called by BPEATTCH enqueued an AWE to the TCB attach processor for the requested TCB; however, the TCB attach processor rejected the AWE because it contained an invalid function code.

X'00000030'
- Internal processing error: BPEATTCH was unable to obtain a DQCB (Dispatcher Queue Control Block) for the new TCB -- the BPECBGET call for the DQCB failed.

X'000000F8'
- The requested function is not present in the BPE system at run time. This is likely due to a macro/module mismatch.

X'000000FC'
- The parameter list version generated by the macro is not supported by the called service routine. This is likely due to a macro/module mismatch.

**Source:** BPE codes

---

**BPEAWSRV**

**Explanation:** The following is a list of return codes and their explanations:

<table>
<thead>
<tr>
<th>Return code</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'00000004'</td>
<td>The CREATE request for an AWE server was successful for at least one, but not all, of the requested AWE servers.</td>
</tr>
<tr>
<td>X'00000008'</td>
<td>Unable to create a thread for an AWE server. The Thread CREATE service, BPETHDCR, failed.</td>
</tr>
<tr>
<td>X'0000000C'</td>
<td>Unable to obtain an AQCB block for an AWE server being created.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Return code</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'00000010'</td>
<td>The BPE TCB token, that passed the TCBTKN parameter for an AWE server CREATE request, was not a valid token.</td>
</tr>
<tr>
<td>X'00000020'</td>
<td>AQHE passed to create routine was not a valid AQHE. This is likely an internal processing error.</td>
</tr>
<tr>
<td>X'00000024'</td>
<td>BPEAWSRV could not determine the TCB under which it should create the server. The TCB is determined by the value of TCBTKN on this macro, or by the value of the TCBTYPE on the BPEAWDEF macro when the queue server is defined. If neither of these parameters are coded, then the server create function does not know under what TCB to create the server thread.</td>
</tr>
<tr>
<td>X'00000028'</td>
<td>If AWEIDX was specified, the value passed was not a valid AWE server index. If TYPE was specified, the type indicated was not a valid AWE server type according to the AWE server definition module. For BPE AWE servers, macro BPEAWIX and module BPEAQHT0 are probably out of sync. For user-product AWE servers, the user-product type definition macro and module are probably out of sync.</td>
</tr>
<tr>
<td>X'0000002C'</td>
<td>The NUMTHDS parameter specified an invalid number of threads. It must specify a number from 1 to 255.</td>
</tr>
<tr>
<td>X'000000F8'</td>
<td>The requested function is not present in the BPE system at run time. This is likely due to a macro/module mismatch.</td>
</tr>
<tr>
<td>X'000000FC'</td>
<td>The parameter list version generated by the macro is not supported by the called service routine. This is likely due to a macro/module mismatch.</td>
</tr>
</tbody>
</table>

**Source:** BPE codes

---

**BPEBPCRE**

**Explanation:** A problem occurred when creating a buffer pool.

The following is a list of return codes and their explanations:

<table>
<thead>
<tr>
<th>Return code</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'00000004'</td>
<td>The value specified on the INCREMENT parameter was zero or negative.</td>
</tr>
<tr>
<td>X'00000008'</td>
<td>The value specified on the MINBUF parameter was zero or negative.</td>
</tr>
</tbody>
</table>
BPEBPGET • BPECBGET

Return Code

Explanation

X’00000004’

No buffer could be obtained because the pool is at its upper limit.

X’00000008’

No buffer could be obtained because a BPEGETM call for buffer storage failed.

X’00000020’

The value specified on the SIZE parameter was zero or negative.

X’00000024’

The value specified on the PREFIX was not between 0 and 256.

X’00000028’

A nonzero value was specified on the PREFIX parameter, but the buffer pool was not created with FORMAT=PREFIX.

X’0000002C’

The address of the word to receive the returned buffer address (BUFFERPTR) was not valid (it was in the first 4K of storage).

X’00000030’

The token passed to BPEBPGET was not a valid token for any BPE-managed buffer pool.

X’00000034’

A page-fixed buffer with a length greater than 100 pages was requested. BPEBPGET limits page-fixed buffers to 100 pages (409,600 bytes) or fewer.

X’000000F8’

The requested function is not present in the BPE system at run time. This is likely due to a macro/module mismatch.

X’000000FC’

The parameter list version generated by the macro is not supported by the called service routine. This is likely due to a macro/module mismatch.

Source: BPE codes

BPECBGET

Explanation: A problem occurred getting a control block.

The following is a list of return codes and their explanations:

Return Code

Explanation

X’00000004’

A bad CBTE address was passed to the CB get routine. The get routine validates that the CBTE address it is passed has the character string “CBTE” as the first word of the block. If it does not, this return code is passed back to the caller. This is likely an internal systems error of some kind.

X’00000008’

Storage was unavailable to satisfy the request.
The requested function is not present in the BPE system at run time. This is likely due to a macro/module mismatch.

The parameter list version generated by the macro is not supported by the called service routine. This is likely due to a macro/module mismatch.

Source: BPE codes

BPECMSRV

Explanation: A problem occurred when submitting a command for processing.

The following is a list of return codes and their explanations:

<table>
<thead>
<tr>
<th>Return code</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'000000020'</td>
<td>The command was rejected by the command action routine and was not processed.</td>
</tr>
<tr>
<td>X'000000024'</td>
<td>The command failed.</td>
</tr>
<tr>
<td>X'000000028'</td>
<td>Unknown command.</td>
</tr>
<tr>
<td>X'00000002C'</td>
<td>Missing command verb.</td>
</tr>
<tr>
<td>X'000000040'</td>
<td>Bad command pointer. The address of the command passed to BPECMSRV was zero.</td>
</tr>
<tr>
<td>X'000000044'</td>
<td>Bad command length. The length of the command passed to BPECMSRV was zero.</td>
</tr>
<tr>
<td>X'000000048'</td>
<td>Bad BPEWPRNT exit parameter list length. The length of an exit parameter list passed to BPECMSRV was zero or negative.</td>
</tr>
<tr>
<td>X'00000004C'</td>
<td>The BPECMSRV caller was not running as a BPE thread.</td>
</tr>
<tr>
<td>X'000000060'</td>
<td>BPE was unable to get the necessary working storage to process the command.</td>
</tr>
<tr>
<td>X'000000064'</td>
<td>BPE was unable to create a new thread to process the command.</td>
</tr>
<tr>
<td>X'000000068'</td>
<td>BPE could not process the command because a BPEPOST call to start the command processing thread failed.</td>
</tr>
<tr>
<td>X'00000006C'</td>
<td>BPE could not process the command because a BPELAGET call for a latch to serialize the command failed.</td>
</tr>
<tr>
<td>X'000000070'</td>
<td>The command action module that processed the command returned an undefined return code to BPE. The result of the command processing is not known.</td>
</tr>
</tbody>
</table>

Source: BPE codes

BPEGETM

Explanation: A problem occurred when issuing a GETMAIN storage command.

The following is a list of return codes and their explanations:

<table>
<thead>
<tr>
<th>Return code</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'00000004'</td>
<td>An invalid or unsupported subpool was specified.</td>
</tr>
<tr>
<td>X'00000008'</td>
<td>A zero (or negative) length was requested.</td>
</tr>
<tr>
<td>X'0000000C'</td>
<td>Unable to obtain the requested storage (MVS GETMAIN failed).</td>
</tr>
<tr>
<td>X'000000F8'</td>
<td>The requested function is not present in the BPE system at run time. This is likely due to a macro/module mismatch.</td>
</tr>
<tr>
<td>X'000000FC'</td>
<td>The parameter list version generated by the macro is not supported by the called service routine. This is likely due to a macro/module mismatch.</td>
</tr>
</tbody>
</table>

Source: BPE codes

BPELAGET

Explanation: A problem occurred when getting a latch.

The following is a list of return codes and their explanations:

<table>
<thead>
<tr>
<th>Return code</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'00000004'</td>
<td>The latch was not obtained because it was owned in exclusive mode by another caller (WAIT=NO only).</td>
</tr>
<tr>
<td>X'00000008'</td>
<td>The latch was not obtained because it was owned in shared mode by one or more callers (WAIT=NO only).</td>
</tr>
<tr>
<td>X'0000000C'</td>
<td>A request for a latch in shared mode was not...</td>
</tr>
</tbody>
</table>
granted because there was a waiter for the latch in exclusive mode (WAIT\(=\)NO only).

X'00000010' The latch was successfully obtained in exclusive mode, and was transferred from another thread to the current thread.

X'00000020' The latch was not obtained because the caller already owns the latch in exclusive mode.

X'00000024' The latch was not obtained because the latch services module was unable to WAIT the caller. This is usually due to an error in the caller's environment: either the caller is not in TCB mode, or the caller's ECB is already WAITing.

X'00000028' A request was not granted for a latch in shared mode because the latch had been transferred to the caller's ECB by another thread; for this reason the request must be in exclusive mode.

X'000000F8' The requested function is not present in the BPE system at run time. This is likely due to a macro or module mismatch.

X'000000FC' The parameter list version generated by the macro is not supported by the called service routine. This is likely due to a macro or module mismatch.

Source: BPE codes

---

BPELOADC

Explanation: A problem occurred when attempting to load, call, or delete a module.

The following is a list of return codes and their explanations:

Return code Explanation

X'00000004' For a list-form load (MODLIST), at least one of the modules in the list had an error.

X'00000008' Specified module was not found.

X'00000010' BDL for module failed. This error is most likely due to an internal error with BDL, such as an out-of-storage condition.

X'00000014' Load of module failed. The module was found in the library, but LOAD returned a nonzero return code.

X'00000028' TCB=JSTP required an AWE to be queued to JSTP TCB for LOAD processing, and the AWE enqueue failed.

X'00000030' Internal failure in load processing.

X'000000F8' The requested function is not present in the BPE system at run time. This is likely due to a macro/module mismatch.

Source: BPE codes

---

BPELOADERL

Explanation: A problem occurred when getting a control block.

The following is a list of return codes and their explanations:

Return code Explanation

X'00000004' The latch was not released because it is owned in exclusive mode by the caller.

X'00000008' The latch was not released because it is owned in exclusive mode by another caller.

X'0000000C' The latch was not released because it is owned in shared mode by one or more other callers.

X'00000010' The latch was not released because it is not owned.

X'00000014' The latch was not transferred because the ECB, to which the latch was to be transferred, was waiting to get the latch in shared mode.

X'00000018' The latch was released (MODE=EX), but a protocol error may have occurred. The latch had been transferred to the caller's ECB from a different ECB, but the caller had never accepted the latch.

X'000000F8' The requested function is not present in the BPE system at run time. This is likely due to a macro or module mismatch.

X'000000FC' The parameter list version generated by the macro is not supported by the called service routine. This is likely due to a macro or module mismatch.

Source: BPE codes
### BPEPARSE

The following is a list of return codes and their explanations:

<table>
<thead>
<tr>
<th>Return code</th>
<th>Explanation</th>
<th>X'00000004'</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>The LOAD and call of the module were successful, but the subsequent DELETE of the module failed. The return code that the module passed back in register 15 is in the word or register specified by MODRC.</td>
<td></td>
</tr>
<tr>
<td>X'00000008'</td>
<td>The module specified on EP or EPLOC could not be loaded. The contents of the word or register specified by MODRC is undefined.</td>
<td></td>
</tr>
<tr>
<td>X'000000F8'</td>
<td>The requested function is not present in the BPE system at run time. This is likely due to a macro/module mismatch.</td>
<td></td>
</tr>
<tr>
<td>X'000000FC'</td>
<td>The parameter list version generated by the macro is not supported by the called service routine. This is likely due to a macro/module mismatch.</td>
<td></td>
</tr>
</tbody>
</table>

**Source:** BPE codes

---

### BPEPARSE

**Explanation:** A problem occurred when parsing data.

The following is a list of return codes and their explanations:

<table>
<thead>
<tr>
<th>Return code</th>
<th>Explanation</th>
<th>X'00000004'</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>The parser definition grammar passed on PADEF was not a valid BPEPADEF grammar.</td>
<td></td>
</tr>
<tr>
<td>X'00000080'</td>
<td>The control block storage passed on CBSTG was not big enough to contain the control blocks that needed to be built to contain the parsed input data. Issue BPEPARSE again with a larger amount of CBSTG storage. Note that the data in the CBSTG area is incomplete, and should not be used.</td>
<td></td>
</tr>
<tr>
<td>X'0000000C'</td>
<td>The CBSTG address passed to the parsing service was zero.</td>
<td></td>
</tr>
<tr>
<td>X'00000010'</td>
<td>The input data address passed to the parsing service was zero.</td>
<td></td>
</tr>
<tr>
<td>X'00000014'</td>
<td>An internal error occurred in the parsing service.</td>
<td></td>
</tr>
<tr>
<td>X'00000040'</td>
<td>An invalid keyword was detected in the input data.</td>
<td></td>
</tr>
<tr>
<td>X'00000044'</td>
<td>An unknown positional parameter was encountered in the input.</td>
<td></td>
</tr>
</tbody>
</table>

---

Some specific return codes and their explanations are as follows:

- **X'00000048'**
  - A keyword parameter was specified with only an equal sign (KEYWORD=value), but the keyword was defined as having a sublist of values. Sublists can only be specified in parenthesis. An equal sign can only be used by itself (i.e., with no parentheses) if a keyword has a single value.

- **X'0000004C'**
  - The input ended before all of a sublist or keyword had been parsed.

- **X'00000050'**
  - A keyword was encountered (KEYWORD or KEYWORD=) when a value was expected.

- **X'00000054'**
  - An input number being parsed was out of the range allowed for its output field length. For decimal numbers, the numbers must be less than or equal to 255, 65535, 16777215, and 2147483647 for 1, 2, 3, and 4 byte fields, respectively. For hexadecimal numbers, the number may not have digits more than 2 times the number of bytes in the output field.

- **X'00000058'**
  - A parameter value defined as decimal contained non-decimal digits.

- **X'0000005C'**
  - A parameter value defined as hexadecimal contained non-hexadecimal digits.

- **X'00000060'**
  - A parameter defined with REQUIRED=YES on BPEPADEF was not found in the input data (omitted).

- **X'00000064'**
  - A parser directive was found within a sublist.

- **X'00000068'**
  - The parser encountered an invalid parser directive.

- **X'00000070'**
  - A keyword was coded specifying a null value. A non-null value is required.

- **X'00000074'**
  - An unexpected right parenthesis was encountered in the input data.

- **X'00000078'**
  - A parser directive was found within a sublist.

- **X'0000007C'**
  - The parser encountered an invalid parser directive.

- **X'00000080'**
  - The parser encountered an invalid parser directive value.
BPERDPDS • BPESVC

X’00000084’
The parser encountered a parser directive, but it expected the value of a `keyword=` parameter.

X’000000F8’
The requested function is not present in the BPE system at run time. This is likely due to a macro/module mismatch.

X’000000FC’
The parameter list version generated by the macro is not supported by the called service routine. This is likely due to a macro/module mismatch.

Source: BPE codes

BPERDPDS

Explanation: A problem occurred when reading a PDS member.

The following is a list of return codes and their explanations:

Return code

<table>
<thead>
<tr>
<th>Code</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>X’00000004’</td>
<td>The member name address passed to the read PDS routine was 0.</td>
</tr>
<tr>
<td>X’00000008’</td>
<td>The output buffer pointer address passed to the read PDS routine was 0.</td>
</tr>
<tr>
<td>X’0000000C’</td>
<td>The output buffer length word address passed to the read PDS routine was 0.</td>
</tr>
<tr>
<td>X’00000010’</td>
<td>The value specified on the <code>RIGHTMAR</code> parameter was either negative or not less than the actual LRECL of the data set.</td>
</tr>
<tr>
<td>X’00000014’</td>
<td>The BPE SVC module was unable to get a SYSTEM level ENQ on the resource SYSZBPE SVCNFHTB, which is required in order to serialize the named function hash table. R0 = the return code from the ENQ macro.</td>
</tr>
<tr>
<td>X’00000018’</td>
<td>An abend occurred in the BPE SVC module BPESVC00. R0 contains the abend code.</td>
</tr>
<tr>
<td>X’0000001C’</td>
<td>The address for the name of the named function was zero (REGISTER, DEREgISTER, CALL).</td>
</tr>
<tr>
<td>X’00000020’</td>
<td>The BPE SVC function requested requires the caller to be authorized (supervisor state and key 0-7), and the caller was not authorized.</td>
</tr>
<tr>
<td>X’00000024’</td>
<td>A BSAM READ for the specified member failed.</td>
</tr>
<tr>
<td>X’00000028’</td>
<td>The data set specified had an unsupported RECFM (for example, RECFM=U or RECFM=VBS).</td>
</tr>
<tr>
<td>X’00000030’</td>
<td>The BPE SVC function could not establish a recovery environment using the MVS ESTAE macro. R0 = return code from ESTAE.</td>
</tr>
<tr>
<td>X’00000034’</td>
<td>The BPE SVC function was unable to get GETMAIN storage for an internal work area. R0 = return code from GETMAIN.</td>
</tr>
<tr>
<td>X’00000038’</td>
<td>The PDS specified was not in fixed format.</td>
</tr>
<tr>
<td>X’0000003C’</td>
<td>The member specified on the <code>MEMBER</code> parameter was not in the PDS.</td>
</tr>
<tr>
<td>X’00000040’</td>
<td>An invalid SVC function code was passed to the BPE SVC routine in register 0. R0 contains the invalid function code.</td>
</tr>
<tr>
<td>X’00000044’</td>
<td>The PSW (program status word) key indicated for the named function being registered was not in the range 0-7. Named function routines must run in a system key (REGISTER).</td>
</tr>
</tbody>
</table>

BPESVC

Explanation: The Base Primitive Environment supervisor call (BPE SVC) encountered a problem.

The following is a list of return codes and their explanations:

Return code

<table>
<thead>
<tr>
<th>Code</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>X’00000004’</td>
<td>The BPE SVC routine was unable to GETMAIN storage for an internal work area. R0 = return code from GETMAIN.</td>
</tr>
<tr>
<td>X’00000008’</td>
<td>The BPE SVC routine could not establish a recovery environment using the MVS ESTAE macro. R0 = return code from ESTAE.</td>
</tr>
<tr>
<td>X’0000000C’</td>
<td>The BPE SVC routine could not locate its main control block (the BPESVCA block). This is likely due to errors initializing the BPE SVC routine or other BPE system services.</td>
</tr>
<tr>
<td>X’00000010’</td>
<td>An invalid SVC function code was passed to the BPE SVC routine in register 0. R0 contains the invalid function code.</td>
</tr>
<tr>
<td>X’00000014’</td>
<td>The BPE SVC function requested requires the caller to be authorized (supervisor state and key 0-7), and the caller was not authorized.</td>
</tr>
<tr>
<td>X’00000018’</td>
<td>An abend occurred in the BPE SVC module BPESVC00. R0 contains the abend code.</td>
</tr>
<tr>
<td>X’0000001C’</td>
<td>The address for the name of the named function was zero (REGISTER, DEREgISTER, CALL).</td>
</tr>
<tr>
<td>X’00000020’</td>
<td>The BPE SVC module was unable to get a SYSTEM level ENQ on the resource SYSZBPE SVCNFHTB, which is required in order to serialize the named function hash table. R0 = the return code from the ENQ macro.</td>
</tr>
<tr>
<td>X’00000024’</td>
<td>The named function indicated could not be found (QUERY, REGISTER, DEREgISTER, CALL).</td>
</tr>
<tr>
<td>X’00000028’</td>
<td>The BPE SVC is not installed.</td>
</tr>
<tr>
<td>X’00000030’</td>
<td>The PSW (program status word) key indicated for the named function being registered was not in the range 0-7. Named function routines must run in a system key (REGISTER).</td>
</tr>
</tbody>
</table>
BPESVC10 • BPETCBSW

X'00000044'
The BPE SVC routine could not obtain storage for an SVC Function Block (SVCF) for the function being registered. R0 = return code from GETMAIN (REGISTER).

X'00000048'
The named function being registered is already registered and the REPLACE parameter did not allow it to be replaced (REGISTER).

X'00000060'
The named function that was requested to be called requires the caller to be authorized (supervisor state, key 0-7), and the caller was not authorized (CALL).

X'00000064'
The BPE SVC routine could not obtain work area storage for the named function routine to be called. R0 = return code from GETMAIN (CALL).

X'00000080'
The STOKEN address passed to the EOMCLEANUP function was zero (EOMCLEANUP).

X'000000FC'
The parameter list version generated by the macro is not supported by the called service routine. This is likely due to a macro/module mismatch.

Source: BPE codes

BPETCBSW

Explanation: A problem occurred during switch execution to a new TCB.

The following is a list of return codes and their explanations:

Return code Explanation
X'00000000' The thread was successfully switched.
X'00000004' An invalid TCB token was passed on TCBTKN.
X'00000008' An invalid TCB index was passed on TCBIDX.
X'0000000C' An invalid TCBT address was passed.
X'00000010' The caller is not running under a BPE thread.
X'00000014' The TCB specified by TCBTKN (or TCBTKNPtr) does not exist.
X'00000018' A non-pool TCB type with multiple TCBs was specified on either TCBTYPE or TCBIDX.
X'000000F8' The requested function is not present in the BPE system at run time. This situation is likely due to a macro or module mismatch.
X'000000FC' The parameter list generated by the macro is
not supported by the called service routine. This situation is likely due to a macro or module mismatch.

**Source:** BPE codes

### BPETHDCR

**Explanation:** A problem occurred when creating a thread.

The following is a list of return codes and their explanations:

<table>
<thead>
<tr>
<th>Return code</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'00000004'</td>
<td>Create failed: An invalid BPE TCB token was passed on the TCBTKN parameter.</td>
</tr>
<tr>
<td>X'00000008'</td>
<td>Create failed: A bad TCB index value was passed on the TCBIDX parameter.</td>
</tr>
<tr>
<td>X'0000000C'</td>
<td>Create failed: A zero routine address was passed on the ROUTINE parameter.</td>
</tr>
<tr>
<td>X'00000010'</td>
<td>Create failed: A bad TCB table entry address was passed into the thread create routine. The BPETHDCR macro determines the TCBT address based on whichever of the TCBIDX, TCBTKN, or TCBTYPE parameters is specified. Ensure that this parameter is correctly coded.</td>
</tr>
<tr>
<td>X'00000014'</td>
<td>Create failed: Unable to get storage for a Thread Control Block (THCB) for the thread.</td>
</tr>
<tr>
<td>X'00000018'</td>
<td>Create failed: Unable to get stack storage for the thread.</td>
</tr>
<tr>
<td>X'0000001C'</td>
<td>Create failed: The initial POST of the thread failed.</td>
</tr>
<tr>
<td>X'00000020'</td>
<td>Create failed: Unable to load the requested thread module.</td>
</tr>
<tr>
<td>X'00000028'</td>
<td>The requested function is not present in the BPE system at run time. This is likely due to a macro/module mismatch.</td>
</tr>
<tr>
<td>X'000000FC'</td>
<td>The parameter list version generated by the macro is not supported by the called service routine. This is likely due to a macro/module mismatch.</td>
</tr>
</tbody>
</table>

**Source:** BPE codes

### BPETIMER

**Explanation:** A problem occurred when issuing a timer request.

The following is a list of return codes and their explanations:

<table>
<thead>
<tr>
<th>Return code</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'00000004'</td>
<td>FUNC=TIMER request was canceled.</td>
</tr>
<tr>
<td>X'00000008'</td>
<td>The request was canceled because the timer service has been shutdown.</td>
</tr>
<tr>
<td>X'0000000C'</td>
<td>FUNC=CANCEL request failed because no outstanding timer request with a matching token could be found.</td>
</tr>
<tr>
<td>X'00000010'</td>
<td>The timer AWE enqueued to the timer service contained an invalid function code.</td>
</tr>
<tr>
<td>X'00000014'</td>
<td>The FUNC=TIMER request indicated that the expiration value was specified as TOD. The current version of timer services does not support TOD=.</td>
</tr>
<tr>
<td>X'00000018'</td>
<td>The DINTVL parameter contained a non-decimal digit.</td>
</tr>
<tr>
<td>X'00000020'</td>
<td>The minutes field in the DINTVL parameter was greater than 59. It must be between 0 and 59.</td>
</tr>
<tr>
<td>X'00000024'</td>
<td>The seconds field in the DINTVL parameter was greater than 59. It must be between 0 and 59.</td>
</tr>
<tr>
<td>X'00000028'</td>
<td>The FUNC=TIMER request could not be processed because the timer service could not get storage for a timer queue element (TQUE) for the request.</td>
</tr>
<tr>
<td>X'0000002C'</td>
<td>The BPETIMER request failed: MVS STIMERM service returned a nonzero return code.</td>
</tr>
<tr>
<td>X'00000080'</td>
<td>Unable to obtain an AWE from control block services (BPECBGET) to initiate timer request. Register 0 contains the return code from BPECBGET. No timer request was issued.</td>
</tr>
<tr>
<td>X'00000084'</td>
<td>Unable to enqueue initiate timer request. AWE enqueue to timer server failed. R0 contains the return code from BPEAWSRV FUNC=ENQ. Note that the state of the AWE is indeterminate. It is probably on an AWE queue, but it may or may not be processed.</td>
</tr>
<tr>
<td>X'00000088'</td>
<td>The requested function is not present in the BPE system at run time. This is likely due to a macro/module mismatch.</td>
</tr>
<tr>
<td>X'000000FC'</td>
<td>The parameter list version generated by the macro is not supported by the called service routine. This is likely due to a macro/module mismatch.</td>
</tr>
</tbody>
</table>
BPETRSRV • 3400

Explanation: A problem occurred when requesting trace services.

The following is a list of return codes and their explanations:

<table>
<thead>
<tr>
<th>Return code</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'00000004'</td>
<td>There is no trace table of the requested type defined to BPE.</td>
</tr>
<tr>
<td>X'00000008'</td>
<td>Input trace token was not valid or was not for a trace table of the indicated type. (FUNC=DELETE, GETUDATA, SETUDATA)</td>
</tr>
<tr>
<td>X'0000000C'</td>
<td>The trace table specified was not defined as a trace type that supports multiple trace tables. The CREATE and DELETE functions may be issued only for trace types that were defined as multiple types. See BPETRDEF and BPETRIX for further information. (FUNC=CREATE and DELETE)</td>
</tr>
<tr>
<td>X'00000010'</td>
<td>The user data address passed on USERDATA was invalid (less than 00001000). (FUNC=GETUDATA and SETUDATA)</td>
</tr>
<tr>
<td>X'00000014'</td>
<td>The trace table specified does not support user data (UDATALEN was not specified or specified zero on the trace table's BPETRDEF definition). (FUNC=GETUDATA and SETUDATA)</td>
</tr>
<tr>
<td>X'00000030'</td>
<td>Unable to get the storage required for a trace table header entry for the new trace table. (FUNC=CREATE)</td>
</tr>
<tr>
<td>X'00000034'</td>
<td>Unable to get the storage required for the new trace table. (FUNC=CREATE)</td>
</tr>
<tr>
<td>X'000000F8'</td>
<td>The requested function is not present in the BPE system at run time. This is likely due to a macro/module mismatch.</td>
</tr>
<tr>
<td>X'000000FC'</td>
<td>The parameter list version generated by the macro is not supported by the called service routine. This is likely due to a macro/module mismatch.</td>
</tr>
</tbody>
</table>

Source: BPE codes

BPEUXRF0

Explanation: A problem occurred when loading user exits.

The following is a list of return codes and their explanations:

<table>
<thead>
<tr>
<th>Return code</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'00000001'</td>
<td>Couldn't read BPE PROCLIB (Rsn code = BPERDPDS RC)</td>
</tr>
<tr>
<td>X'00000002'</td>
<td>Couldn't read user-product PROCLIB (Rsn code = BPERDPDS RC)</td>
</tr>
<tr>
<td>X'00000003'</td>
<td>BPEGETM failed for BPE parse data area (Rsn code = BPEGETM RC)</td>
</tr>
<tr>
<td>X'00000004'</td>
<td>BPEGETM failed for BPE parse data area (Rsn code = BPEGETM RC)</td>
</tr>
<tr>
<td>X'00000005'</td>
<td>BPEPARSE failed for BPE exit list PROCLIB member (Rsn code = BPEPARSE RC)</td>
</tr>
<tr>
<td>X'00000006'</td>
<td>BPEPARSE failed for user prod exit list PROCLIB member (Rsn code = BPEPARSE RC)</td>
</tr>
<tr>
<td>X'00000010'</td>
<td>BPECBGET failed for a UXB block (Rsn code = BPECBGET RC)</td>
</tr>
<tr>
<td>X'00000011'</td>
<td>BPELAGET failed for global user exit latch (UXRF_Lhdr) (Rsn Code = BPELAGET RC)</td>
</tr>
<tr>
<td>X'00000012'</td>
<td>User exit refresh failed due to abend in refresh processing (Rsn code = abend code)</td>
</tr>
<tr>
<td>X'00000013'</td>
<td>BPECBGET failed for a static work area for a user exit module (Rsn code = BPECBGET RC)</td>
</tr>
<tr>
<td>X'00000014'</td>
<td>BPEGETM failed for storage for a user exit module (Rsn code = BPEGETM RC)</td>
</tr>
<tr>
<td>X'00000015'</td>
<td>BLDL failed for a user exit module (Rsn code = BLDL RC)</td>
</tr>
<tr>
<td>X'00000016'</td>
<td>LOAD failed for a user exit module (Rsn code = LOAD RC)</td>
</tr>
</tbody>
</table>

Source: BPE codes

BPE user abend codes

This section identifies and explains user abend codes issued by Base Primitive Environment (BPE).

3400

Explanation: An error during early BPE initialization prevented BPE services from starting. Register 15 at the time of abend contains a subcode that can be used to determine the reason for the abend.
The following is a list of subcodes and their explanations:

<table>
<thead>
<tr>
<th>Subcode</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'01'</td>
<td>BPEINIT0 could not establish an ESTAE recovery routine.</td>
</tr>
<tr>
<td>X'02'</td>
<td>BPE stack storage services initialization failed.</td>
</tr>
<tr>
<td>X'03'</td>
<td>A request to obtain a stack block for early BPE initialization failed.</td>
</tr>
<tr>
<td>X'04'</td>
<td>A lower-level service initialization module could not be loaded.</td>
</tr>
<tr>
<td>X'05'</td>
<td>A lower-level service initialization module could not complete the initialization of its service.</td>
</tr>
<tr>
<td>X'06'</td>
<td>An error occurred while parsing the BPE configuration PROCLIB member specified on the BPECFG= parameter in the z/OS startup parameters.</td>
</tr>
<tr>
<td>X'07'</td>
<td>The IMS RE component using BPE services detected a problem during its early initialization and indicated to BPE that it should abend.</td>
</tr>
<tr>
<td>X'08'</td>
<td>An error occurred creating a z/OS name or token for the IMS RE component address space.</td>
</tr>
<tr>
<td>X'09'</td>
<td>A mismatch occurred between the BPE version used by the IMS RE component and the version of the BPE modules.</td>
</tr>
<tr>
<td>X'0A'</td>
<td>The BPE SVC routine (BPESVC00) could not be established.</td>
</tr>
<tr>
<td>X'0B'</td>
<td>The BPE resource manager (BPERSM00) could not be established.</td>
</tr>
<tr>
<td>X'0C'</td>
<td>The BPE address space was started on a machine that was not running in z/Architecture mode. BPE requires z/Architecture mode.</td>
</tr>
<tr>
<td>X'0D'</td>
<td>The BPE address space was started on a z/OS release that does not support 64-bit virtual storage services. BPE requires 64-bit storage services.</td>
</tr>
</tbody>
</table>

**System action:** The address space is abnormally terminated.

**Operator response:** Take the appropriate action based on the subcode you received.

For subcodes X'01' through X'05' and X'07' through X'0B': The problem is one of the following values:
- An environmental error, such as missing modules in IMS.SDFSRESL or a storage shortage in the address space.
- An internal error within BPE or the component using BPE.

Messages that further explain the reason for the error might precede this abend. Refer to the job log output from the failed job for explanatory messages.

If you cannot determine and correct an environmental cause for the problem, save any dump and SYSLOG information, and contact the IBM Support Center.

For subcode X'06': Although this abend can result from internal or environmental errors, its most probable cause is a problem with the BPE configuration PROCLIB member. If BPE cannot find or read the member, abend 3400 follows message BPE0002E, which indicates the problem. Correct the problem and restart the IMS RE component address space.

If there is an error parsing the member, this abend follows message BPE0003E, which indicates where in the member the error occurred. Correct the configuration member and restart the IMS RE component address space.

For subcode X'09': This abend occurs when the IMS RE component modules in your STEPLIB data sets are at different versions. This abend follows message BPE0013E, which indicates the different levels of the BPE modules. Ensure that you are running the correct level of BPE modules.

**Source:** BPE codes

**Module:** BPEINIT0

---

### 3401

**Explanation:** An error during the initialization of a new TCB in the IMS RE component address space prevented the TCB from starting. Register 15 at the time of abend contains a subcode that can be used to determine the reason for the abend.

The following is a list of subcodes and their explanations:

<table>
<thead>
<tr>
<th>Subcode</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'01'</td>
<td>BPESYIN0 could not establish an ESTAE recovery routine.</td>
</tr>
<tr>
<td>X'02'</td>
<td>A request to obtain a stack block for the TCB initialization failed.</td>
</tr>
<tr>
<td>X'03'</td>
<td>BPE could not initialize the BPE internal dispatcher for the new TCB.</td>
</tr>
<tr>
<td>X'04'</td>
<td>A lower-level TCB initialization module could not complete initialization because of an error or unexpected condition.</td>
</tr>
<tr>
<td>X'05'</td>
<td>BPE could not create an internal server that runs under the new TCB.</td>
</tr>
<tr>
<td>X'06'</td>
<td>BPE could not load a lower-level TCB initialization module, and therefore could not complete the initialization of the new TCBs environment.</td>
</tr>
</tbody>
</table>

**System action:** If the TCB being created is not a required TCB, the IMS RE component address space continues to function; otherwise, the IMS RE component address space is abnormally terminated.

**Operator response:** The problem is one of the following values:
- An internal error within BPE.
3402 • 3403

- An environmental error, such as missing modules in IMS.SDFSRESL, or a storage shortage in the address space.

Messages that further explain the reason for the error might precede this abend. Refer to the job log output from the failed job for explanatory messages.

If you cannot determine and correct an environmental cause for the problem, save any dump and SYSLOG information, and contact the IBM Support Center.

Source: BPE codes
Module: BPESYIN0

3402
Explanation: An error occurred during the termination of a TCB in the IMS RE component address space. Register 15 at the time of abend contains a subcode that can be used to determine the reason for the abend.

The following is a list of subcodes and their explanations:

Subcode Explanation
X'01' A routine involved in the cleanup process for the terminating TCB encountered an error that prevented the cleanup from completing.
X'02' A routine involved in the cleanup process for the IMS RE component address space encountered an error that prevented the cleanup from completing.
X'03' BPE could not start required processes to terminate subtask TCBs under the TCB that was terminating.

System action: If the TCB being terminated is not a required TCB, the IMS RE component address space continues to function. If the TCB is required, or if the IMS RE component address space is in the process of being shut down, the IMS RE component address space is abnormally terminated.

Operator response: The problem is one of the following errors:
• An internal error within BPE.
• An environmental error, such as missing modules in IMS.SDFSRESL or a storage shortage in the address space.

Messages that further explain the reason for the error might precede this abend. Refer to the job log output from the failed job for explanatory messages.

If you cannot determine and correct an environmental cause for the problem, save any dump and SYSLOG information, and contact the IBM Support Center.

Source: BPE codes
Module: BPESYIN0

3403
Explanation: The BPE address space startup module BPEINI00 encountered an error that prevented a BPE address space from starting.

Abend 3403 is a standard abend issued by module BPEINI00 when it encounters a problem trying to start an IMS RE component address space using BPE services. At the time of the abend, register 15 contains a subcode that indicates the reason for the abend.

Register 2 contains the address of the z/OS parameters passed to BPEINI00. Abend 3403 is preceded by message BPE0037E, which further explains the problem.

The following is a list of subcodes and their explanations:

Subcode Explanation
X'01' The address space being started is not APF-authorized. All BPE and IMS RE component modules must be in data sets that are APF-authorized. In addition, module BPEINI00 must be bound with authorization code 1: SETCODE AC(1).
X'02' BPEINI00 was given control in a key other than 7. BPE must run in key 7. Ensure that BPEINI00 is correctly added to the z/OS PPT (program properties table).
X'03' Either BPEINI00 could not find the BPEINIT= keyword on the z/OS JCL startup parameters (PARM=), or the specified module name was missing or not valid. The BPEINIT= parameter is required. The parameter defines the name of the BPE startup parameter module to BPEINI00, which defines the type of IMS RE component address space to start.
X'04' The module specified by the BPEINIT= keyword on the z/OS JCL startup parameters (PARM=) could not be loaded.
X'05' The module specified by the BPEINIT= keyword on the z/OS JCL startup parameters (PARM=) was loaded into key 0 storage. This situation probably occurred due to being bound as reentrant. This module must be bound as non-reentrant.
X'06' BPEINI00 could not find a valid startup parameter data area (UCDB block) in the module specified by the BPEINIT= keyword on the z/OS JCL startup parameters.
X'07' BPEINI00 could not get storage for a copy of the z/OS input parameters.

System action: The address space terminates abnormally.

Operator response: If the error indicated by the subcode is due to environmental conditions (for example, incorrect authorization or invalid parameters), correct the indicated problem and restart the address space. Otherwise, save any joblog and syslog.
information and contact the IBM Support Center for assistance.

Source: BPE codes
Module: BPEINI00

4095

Explanation: BPE issues this abend once for each TCB in the address space when an unrecoverable abend occurs and it is necessary to abnormally terminate the address space.

System action: The IMS RE component address space is abnormally terminated.

Operator response: BPE issues this abend after one of the address space's TCBs has an unrecoverable abend. It is used to abnormally terminate the remaining TCBs; it is not itself part of the original problem. Examine the job log output for the first TCB to abend with a code other than 4095 in order to determine the cause of the subsequent 4095 abends.

Source: BPE codes
This information was developed for products and services offered in the U.S.A. IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program or service may be used. Any functionally equivalent product, program or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user’s responsibility to evaluate and verify the operation of any non-IBM product, program or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan Ltd.
1623-14, Shimotsuruma, Yamato-shi
Kanagawa 242-8502 Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information that has been exchanged, should contact:
IBM Corporation
J46A/G4
555 Bailey Avenue
San Jose, CA 95141-1003
U.S.A.

Such information may be available subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this information and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement, or any equivalent agreement between us.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify and distribute these sample programs in any form without payment to IBM, for purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs.

Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. These and other IBM trademarked terms are marked on their first occurrence in this information with the appropriate symbol (® or ™), indicating US registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries. A complete and current list of IBM trademarks is available on the Web at [http://www.ibm.com/legal/copytrade.shtml](http://www.ibm.com/legal/copytrade.shtml).

Linux is a trademark of Linus Torvalds in the United States, other countries, or both.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Other company, product, and service names may be trademarks or service marks of others.
### Index

**A**
- abend codes
  - BPE: 149, 158
  - IMS Recover Expert: 129
- accessibility features: 11
- architecture: 7

**B**
- backup solutions: 3
- Base Primitive Environment
  - See BPE
- benefits: 5
- BPE messages and codes: 133
- service abend codes: 149
- user abend codes: 158

**C**
- clean up time: 81
- codes
  - abend: 129
  - BPE
    - service abend: 149
    - user abend: 158
    - Extended Functions messages: 109
- COMMAND parameter: 41
- components
  - BPE architecture: 6
  - Extended Functions components: 6
  - functions: 6
  - ISPF interface: 6
  - main address space: 6
  - parser: 6
  - report writer: 6
- configuration
  - IMS Database Recovery Facility: Extended Functions: 21
  - members: 21
  - parameters: 22

**D**
- database component
  - methods for selecting: 93
- Database Component processing panel: 94
- database health checking: 45
- Database Recovery Facility
  - See DRF jobs
- DBD command: 56
- DBLIST processing: 24
- DBR command: 56
- disability: 11
- documentation: 11
- DRF job generation
  - RID ISPF interface: 93
  - without the RID report: 99
- DRF jobs
  - customization: 101
  - generation: 101

**E**
- examples
  - HCHECK usage: 51
  - IMSCMD usage: 42
  - RCU usage: 85
  - RPCR usage: 58
  - RID usage: 67
  - VERIFY usage: 75
- EXECs
  - JCL statement: 32

**F**
- features: 5
- formats
  - timestamp: 30
- functions
  - Health Checker
    - See HCHECK function
  - IMS Command
    - See IMSCMD function
  - RECON cleanup
    - See RCU function
  - Recovery Point Creation
    - See RPCR function
  - Recovery Point Identification
    - See RID function
  - supporting: 24
  - Verify Recovery Assets
    - See VERIFY function

**H**
- hardware requirements: 4
- HCHECK function
  - example usage: 51
  - input: 45
  - messages: 52
  - options: 45
  - output: 52
  - overview: 45
  - parameters
    - DAYS: 45
    - HOURS: 45
    - NUMBER: 45
    - RANGE: 45
    - RPTHCHK DD: 52
  - scenarios: 13
- Health Checker function
  - See HCHECK function

**I**
- image copy type: 75
- IMS Command function
  - See IMSCMD function
- IMSCMD function
  - example usage: 42
  - input: 41
  - output: 42
  - overview: 41
  - RPTIMSCM DD: 42
  - initialization errors
  - RID ISPF interface: 97
  - input
    - HCHECK
      - See HCHECK function
    - IMSCMD
      - See IMSCMD function
    - RCU
      - See RCU function
    - RPCR
      - See RPCR function
    - RID
      - See RID function
    - VERIFY
      - See VERIFY function
  - input DD statement: 32
  - installation
    - prerequisites: 4
    - interfaces
      - RID ISPF interface: 93

**J**
- JCL statements: 32
- job generation
  - See DRF job generation

**K**
- keyboard shortcuts: 11

**L**
- legal notices
  - notices: 163
  - trademarks: 164
- LookAt: vi

**M**
- message retrieval
  - LookAt: vi
- messages and codes
  - BPE: 133
  - Extended Functions: 109
  - methods for accessing: vi
solutions
  backup 3
  recovery 3
statements
  EXEC 32
  input DD 32
  output DD 32
summary of changes 12
summary report 34
supporting functions
  DBLIST 24
  RECONCOPY 27
  SWI OLDS command 56
  SYSOUT 34
  SYSTEMS parameter 41

T
  terminology 9
  TIME parameter 30
  time range 30
  time span recovery 65
  timestamps 30
  format 30
  trademarks 164
troubleshooting
  abend codes 129
  messages 109

U
  updates 11
  user abend codes
    See BPE

V
  VERIFY function 73
    example usage 75
    image copy type 75
    input 73
    output 75
    overview 73
    parameters
      RCVTYPE 73
      SOURCE 73
      TIME 73
      TYPE 73
      RPTVERIFY DD 75
    scenarios 16
Verify Recovery Assets
  See VERIFY function